Journal of Information Hiding and Multimedia Signal Processing ©2021 ISSN 2073-4212
Ubiquitous International Volume **12**, Number **2**, June **2021**

An Information Hiding Scheme for Live P2P Streaming Based on the R^2 Protocol

Mohamed Amine BELHAMRA

Department of Computer Science
Mohammed V University in Rabat, Faculty of Sciences
Laboratory of Mathematics, Computer Science, Applications and Information Security
BP 1014 RP, Rabat 10000, Morocco
mabelhamra@gmail.com

El Mamoun SOUIDI

Department of Computer Science
Mohammed V University in Rabat, Faculty of Sciences
Laboratory of Mathematics, Computer Science, Applications and Information Security
BP 1014 RP, Rabat 10000, Morocco
emsouidi@gmail.com

Received November 2020; revised February 2021

Abstract. *In this paper we propose a distortion-less information hiding scheme with maximum embedding capacity based on the P2P (peer-to-peer) streaming video protocol called R^2 (Random push with Random network coding). Specifically, we use the encoding vectors attached in the headers containing the coefficients used to encode the packets as our covert channel, in which we hide secret messages. We give the embedding capacities of the scheme and compare it with state-of-the-art schemes performances.*
Keywords: Network Steganography, P2P, Video Streaming Protocol, Random Network Coding, R^2 Protocol.

1. **Introduction.** Steganography is the art and science of hiding a secret message within an ordinary message (the cover-medium) in such a way that no one realizes there is a hidden message, apart the sender and the intended receiver. The term *Steganography* have its origin from the ancient Greek words *stegano* and *graphein*, where the former word means concealed or covered, and the latter means writing. A particular property over which steganographic methods are classified is the type of the cover-medium used in the process.

The principle of a steganographic scheme, is to hide a secret message in a cover and allow to extract it back. The distortion, the security and the embedding capacity are the most important factors in defining a steganographic scheme. The distortion is the ratio of the number of changed bits in the cover to the total number of cover bits, and a scheme is distortion-less means that the cover containing secret information is not distinguishable from itself after the embedding process. A secure scheme means that the secret message should be undetectable and no one, except the eligible recipient, should be able to extract it. Whereas, the embedding capacity refers to the quantity of secret information the scheme can hide with respect to the cover size.

Constructing steganographic schemes using communication protocols with both, the maximising embedding capacities and the minimising distortion has always interested the

researchers in the field. Many steganographic schemes have been proposed in the literature for hiding secrets in different types of media [27]. Among these schemes, the *LSB* technique [1] hides secret bit sequences in the *least significant bits* (LSB) of a bitmap image. This embedding technique causes an imperceptible distortion to the digital image, that is, without a rigorous comparison between the original and the cover images, it is difficult to see that something has changed in it. Further works such as [15] consider reversible data hiding schemes for multiple histogram selection. The $F5$ scheme [25] uses compressed JPEG images as a cover and it is the first implementation of a steganographic scheme based on codes. The $F5$ scheme permits to hide messages of length k in words of length $2^k - 1$. An explicit description of the relation between error correcting codes and steganography protocols is given in [11, 13]. One interesting scheme for digital watermarking using Quik Response (QR) codes is given in [16] and recently, steganography based on Redundant Residue Number System Codes (RRNS) codes was introduced in [12, 14].

In the following, we consider a specific class of steganographic schemes which is mainly based on protocol-related functions associated with the Open System Interconnect-Reference Model (OSI-RM) layers, referred to as Network Steganography (NS). The cover-mediums are defined in this case using the control data and timing properties of the transmission/user data. Many NS schemes have been studied in the literature. Hereafter we cite some well known proposals.

Starting from the presentation layer, NS techniques that embeds secret bits into user data such as voice samples, images, *etc.* have been heavily studied. The LSB scheme is one such technique which modifies the least significant bits (LSB) of digital images. Furthermore, a scheme that exploits the application layer by inserting secret bits into the HTTP's headers and tags is given in [21]. NS methods can also exploit the adjustment of the messages to the type of network or means of transport. The Retransmission Steganography (RSTEG) [10] technique for protocols with retransmission schemes is based on the successful acknowledgement of received TCP segments to intentionally invoke retransmission, then the retransmitted packet carries secret information in the payload field. Two additional approaches were proposed in [8] where one idea consists in hiding secret messages in reserved parts of the packet's headers, taking into consideration that communication protocols in general do not impose specific values for the unused/reserved parts. In particular, the authors in [8] have proposed the use of the IP header's Don't Fragment (DF) flag when the transmitted packets are smaller than the path's Maximum Transfer Unit (MTU). Another proposed scheme exploiting the unused fields of the Session Initiation Protocol (SIP) to hide information is given in [18].

Furthermore, some schemes exploiting the physical and data link layers of the OSI model are given in [20, 19]. The physical layer based method called WiPad (Wireless Padding) [20] is intended for IEEE 802.11 Orthogonal Frequency Division Multiplexing (OFDM) networks. The secret information is inserted into the padding of transmitted symbols. A data link layer based scheme for Wireless Local Area Networks (WLANs) called Hidden Communication System for Corrupted Networks [19], consists in using transmission frames with intentionally wrong checksums. That is, only terminals that are aware of the scheme read such frames and extract hidden data from payload field.

Moreover, the authors in [3] have first developed a scheme where more than one protocol is exploited, known as Padding Steganography (PadSteg). Such class of schemes is called inter-protocol steganography. A good classification of NS methods based on patterns is given in [24].

Researchers in the field always aim to propose new and modified steganographic protocols exhibiting enhanced properties (*i.e.*, higher embedding capacity, more robustness

against adversary attacks). Especially for new communication technologies emerging with the evolution of communicating mediums and terminals such as smart-phones [9].

The Random push with Random network coding protocol, known as R^2 as described in [23] is a live Peer-to-Peer streaming algorithm designed to incorporate random network coding with a randomized push algorithm. It is is designed to improve the performance of live streaming in terms of initial buffering delays, resilience to peer dynamics and bandwidth costs on dedicated streaming servers.

In this paper, we propose a distortion-less steganographic scheme with maximum embedding capacity using the R^2 protocol. Specifically, we use the random coefficients used for encoding the R^2 packets as our covert channel rather than picking randomly these coefficients. Furthermore, we discuss the embedding capacity of the scheme and compare it with other known steganographic protocols.

This paper is organized as follows : In Section 2, we recall some definitions and background about network steganography and the R^2 protocol. In Section 3, we describe the proposed steganographic scheme, we highlight its efficiency and embedding capacities, then compare it with some network steganographic schemes before concluding the paper in Section 4.

2. Definitions and Background.

In this section, we recall first some network steganography related definitions (for more details see [24]) then we give an overview of the R^2 video streaming protocol.

2.1. Network Steganography.

Network Steganography (NS) is a class of steganographic schemes which are based on functions of communication protocols in contemporary networks. The features constituting the base of NS are formulated as follows:

1. Some functions of the protocols are modified. The modifications may be:
   • Functions of the protocols introduced to correct the imperfectness of communication channels (errors, delays, *etc.*).
   • Functions of the protocols introduced to define the communication type (*e.g.* query/response, file transfer, *etc.*) and/or to adapt the form of messages to the transmission medium (*e.g.* fragmentation, segmentation, *etc.*).
2. The effects of these modifications are difficult to discover (*e.g.*, to seem resulting from the imperfectness of the communication network and/or protocols).

NS schemes are classified into storage, timing and hybrid methods based on how the secret messages are encoded into the carrier. Storage methods hide data by modifying packet's fields, while timing methods hide information in the timing of protocol packets. Hybrid techniques combines both the timing and storage methods.

2.2. P2P communication and Network Coding.

Peer-to-Peer (P2P) communication networks have been successfully used in live multimedia streaming applications over the Internet [28]. The essential advantage of live P2P streaming is to dramatically increase the number of peers a streaming session may sustain with several dedicated streaming servers. Intuitively, as participating peers contribute their upload bandwidth capacities to serve other peers in the same streaming session, the load on dedicated streaming servers is significantly mitigated. Therefore, as one of the most significant benefits, P2P streaming enjoys the advantage of scalability in live sessions, where upload capacities on streaming servers are no longer the bottleneck.

The fundamental principle of today's networks is that the sender forwards data and the information is processed only at the end nodes. Such principle considerably limits the information contained in a transmission and the number of devices benefiting from a transmission. This also increases the possibility of missing the throughput, delay and

quality of service requirements. Network coding (NC) is a relatively new paradigm that breaks this fundamental idea and advocates for combining several incoming packets at the sender or intermediate nodes of a network to generate outgoing packets [29]. NC related works are generally classified in two categories, Inter-flow NC and Intra-flow NC. In Inter-flow NC, packets belonging to different flows of data are combined while Intra-flow NC is based on the combination of packets from the same flow. Further works such as in [30] developed this idea by constructing linear codes. In linear NC, to retrieve the original packets at the devices, linear operations over a finite field are performed on the combined packets. Numerous works have demonstrated the ability of NC to achieve a high transmission efficiency compared to the traditional transmissions. For example, a randomized approach referred to as Random Linear Network Coding (RLNC) and achieving the maximum multicast capacity with high probability is given in [31, 32, 33, 34, 35]. The basic idea in RLNC is to operate over a large enough field so that even random choices of the encoding coefficients offer, with high probability, a valid solution.

2.3. **The R^2 Protocol.** P2P multimedia streaming protocols generally fall into two strategic categories. Wang and Li [22] have evaluated the effectiveness of applying network coding in P2P multimedia streaming, by replacing traditional block scheduling algorithms with a group network coding scheme in an experimental testbed. It has been shown that network coding provides some marginal benefits when the overall bandwidth supply barely exceeds the demand, or when peers are volatile with respect to their arrivals and departures. This motivates a complete redesign of P2P multimedia streaming with network coding. Indeed, Wang and Li have proposed the R^2 protocol.

In R^2, the streaming content is first divided into a sequence of equal-sized segments, and each segment is further divided into k equal-sized data blocks. The coding operation is only performed within each segment, but not across different segments to reduce the computational cost. Thus a downstream peer is able to decode segment s as long as it has received k linearly independent coded blocks. More importantly, even slow overlay connections may be utilized in R^2, which is generally impossible in *pull strategy* [26]. Peers periodically exchange information on segment availability with active neighbours (commonly referred to as buffer maps). Those peers that have a particular segment available for transmission are referred to as seeds of this segment. Each peer maintains a playback buffer that consists of segments to be played in the immediate future.

Each segment in R^2 is further divided into n blocks $[b_1, b_2, \cdots, b_n]$ where each b_i has a fixed number of k bytes (referred to as the block size). If the segment duration (for example, four seconds) and the streaming rate is predetermined, the block size k can be directly computed from n. Practically, each segment in R^2 is divided into 128 data blocks of 2 KB each. Good overall performance using this design has been observed and when the field size is $q = 2^8$, the overhead (around 6%) induced by the "header" is optimal.

The network coding is performed only within the segment. Given a peer p, the seed randomly chooses a segment whenever it produces one coded block, among all remaining segments that p has not completely received. The coded block is then sent to p without the need of any requests. Since all coded blocks are equally innovative, all seeds of p cooperatively serve the missing segments on p.

Before pushing coded blocks, an upstream peer should obtain precise knowledge of the missing segments on its downstream peers at any time. This requires participating peers in the system to exchange their buffer maps in a timely fashion. R^2 with its large segments (instead of small data blocks) buffer maps that indicate segment availability information, can afford "real time" exchanges of buffer maps. Whenever a peer has played back or completely received a segment, it sends a new buffer map to all its neighbours.

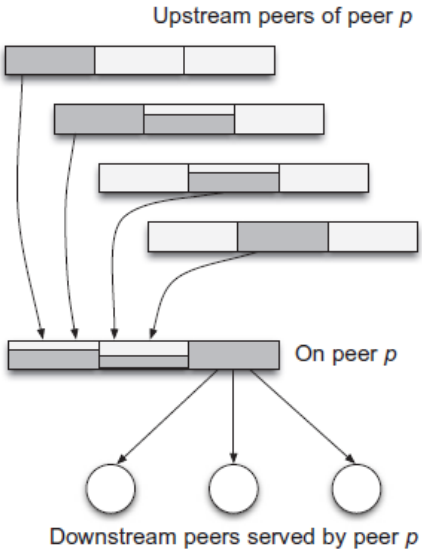

FIGURE 1. In the R^2 protocol, multiple upstream peers are able to perform push operations on coded blocks within a segment without any explicit coordination.

The R^2 protocol features synchronized playback as follows. When a peer joins or switches to a new streaming channel, it first retrieves buffer maps from its neighbours, along with information of the current segment being played back. To synchronize the playback buffer, the new peer only retrieves segments that are δ seconds after the current playback point, where δ corresponds to the initial buffering delay. The peer, then starts playback after precisely δ seconds have elapsed in real time, regardless of the status of the playback buffer.

Under synchronized playback, peers are able to help each other more effectively, since their playback buffers overlap as much as possible. As newly arrived peers request almost the same segments, they are able to help one another as soon as they receive a small number of coded blocks. This leads to a better utilization of their upload bandwidth, which in turn improves the scalability of the streaming system.

The design space of R^2 is flexible to accommodate more elaborate protocols designed for different purposes. For example, a peer in R^2 has the freedom to decide which segments to be pushed to which neighbours. Referred to as a push strategy, this decision may be made based on timing requirements to ensure smooth playback of urgent segments, on fairness issues to encourage cooperation and reduce free-riding, or on geography considerations to reduce traffic across different Internet Service Providers (ISPs).

When a segment is selected by a seed to code for its downstream peer p, the seed independently and randomly chooses a set of coding coefficients $[\alpha_1, \alpha_2, \ldots, \alpha_m]$ with $m \leq n$ in $\mathbb{F}_{2^8}$ for each coded block to be sent to p, where again, n is the number of blocks in each segment. The seed then randomly chooses m blocks $[b_1, b_2, \ldots, b_m]$ out of all the blocks in this segment that it has received so far (all the original blocks in the segment if the seed is a streaming server), and produces one coded block x of k bytes $\sum_{i=1}^{m} \alpha_i \times b_i$ (see Figure 2).

Note that in R^2, to make sure that coded blocks from one segment is not "spread too thin" in all the peers, a seed only sends a segment to a limited number of downstream peers at any given time, subject to an upper bound. To select such limit, the seed can randomly choose from all its downstream peers, or select those that historically had the

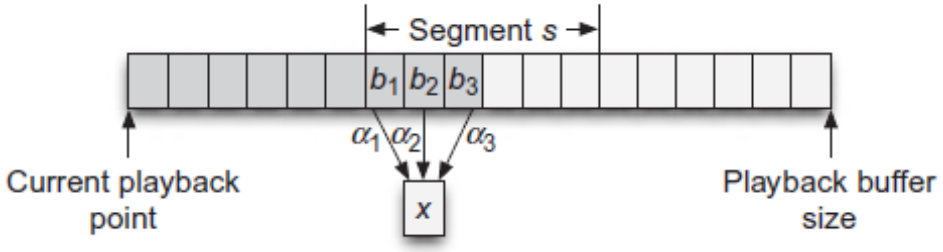

FIGURE 2. An example of coding on peer p, where peer p has received 3 coded blocks within the segment l and each segment consists of 6 blocks.

highest flow rate with the seed. The maximum number of downstream receivers should be linearly related to the upload capacity of a seed: The lower the upload-capacity is, the smaller is the number of active downstream receivers it should maintain.

3. The Proposed Steganographic Scheme using R^2.

3.1. Description of the Scheme. Hereafter, we consider an instance of the multimedia P2P streaming protocol R^2 and the best setting experienced in [23]. That is, the segments (representing 4 seconds of the playback) are divided into 128 blocks, each of 2048 bytes, while the encoding coefficients are of one byte, *i.e.*, picked from the field $\mathbb{F}_q$ where $q = 2^8$.

Now, we consider the sub-network of an R^2 streaming session as given in (3). Say the server s wants to send secret data to a downstream peer, say P_2 (see Figure 3).

To establish our communication channel, besides that the server s and the peer P_2 must use R^2 as their video streaming protocol, they need first realize the following steps:

- The server s must be a peer in a streaming session (seeder in our case) in the network.
- The peer P_2 joins the streaming session (newly arrived peer).
- The server s (seeder) uses its power to select the peer P_2 as one of its downstream peers.

Let us first consider the simplest case for one segment (128 blocks) where we use the setting given above, and $m = 128$ for an optimal scheme. The server s can hide a binary sequence $\mathcal{B}$ of size $|\mathcal{B}| \leq 128K$ bits in one segment upload. That is, the server s first cuts $\mathcal{B}$ into symbols $\langle \mathcal{B}_i \rangle_q$ for $i \in \{1, 2, \ldots, 128^2\}$, *i.e.*, $\lceil \frac{|\mathcal{B}|}{8} \rceil = 128^2$ blocks:

$$S = \{\langle \mathcal{B}_1 \rangle_q, \langle \mathcal{B}_2 \rangle_q, \ldots, \langle \mathcal{B}_{128^2} \rangle_q\} \tag{1}$$

If $|\mathcal{B}| < 128\ Kb$, the sender fills the last symbols by zero bits to adapt to the symbol size then picks uniformly and randomly the rest of the symbols as assured by the R^2 protocol.

The node s constructs the transfer matrix $T \in \mathcal{M}_{128}(\mathbb{F}_{2^8})$ such that

$$T = \mathcal{E}(S) = LU$$

were the embedding map $\mathcal{E}()$ is described in Algorithm 1, transforms an array of 128^2 symbols to its associated transfer matrix, and L and U are the lower and upper triangular matrices as in (2) and (3) respectively, which contain the secret symbols to send. That is, set $\mathfrak{b}_i = \langle \mathcal{B}_i \rangle_q$ for $i = 1, 2, ..., 128^2$. So :

$$L = \begin{pmatrix} 1 & & & \\ \mathfrak{b}_1 & 1 & & 0 \\ \mathfrak{b}_2 & \mathfrak{b}_{128} & 1 & \\ \vdots & \vdots & \vdots & \ddots \\ \mathfrak{b}_{127} & \mathfrak{b}_{253} & \cdots & \mathfrak{b}_{8128} \quad 1 \end{pmatrix} \tag{2}$$

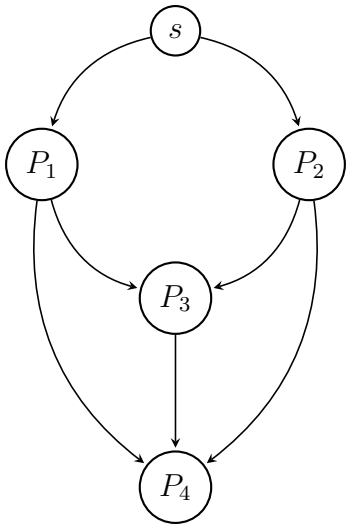

FIGURE 3. Our sub-network in the P2P streaming session in R^2.

and

$$U = \begin{pmatrix} \mathfrak{b}_{8129} & \mathfrak{b}_{8130} & \mathfrak{b}_{8131} & \dots & \mathfrak{b}_{8256} \\ & \mathfrak{b}_{8257} & \mathfrak{b}_{8258} & \dots & \mathfrak{b}_{8383} \\ & & \mathfrak{b}_{8384} & \dots & \vdots \\ & 0 & & \ddots & \vdots \\ & & & & \mathfrak{b}_{128^2} \end{pmatrix} \tag{3}$$

Then the constructed transfer matrix is $T = LU$ where all arithmetic operations are performed over $\mathbb{F}_q$. Note that we suppose the elements $\langle \mathcal{B}_i \rangle_q$ for $i \in \{1, 2, \dots, 128^2\}$ to be non zero in order to assure the non-zero determinant condition of NC and hence, the matrix T to be uniquely LU-factorisable, *i.e.*, $L_{i,j} = 1$ and $U_{i,j} \neq 0$ for $i = j$ ([36] Corollary 3.5.5). Otherwise, we can code the zero elements as a non used agreed upon character.

Algorithm 1: The map $\mathcal{E}$ transforms an array of 128^2 symbols to its associated transfer matrix.

Input: An array $S = [s_1, s_2, ..., s_{128^2}]$, where $s_i \in \mathbb{F}_{2^8}$ for $i = 1, 2, ..., 128^2$.
Output: Transfer matrix $T = L \times U$

```
int[][] L = I_128;                          // I_128 : identity matrix
int[][] U;
int k = 1, l = 1;
int i = 1, j;
while i ≤ 128 do
    for j = 0; j ≤ i; j++ do
        U[j][i] = S[k++];
        L[i][j] = S[k++];
    U[j][i] = S[k++];
    i++;
return L × U;
```

The node s sends the linear combinations of the blocks, *i.e.* $b_i = \sum_{j=1}^{128} \alpha_{ij} b_j$ for $i = 1, 2, \ldots, 128$, where α_{ij} are elements of T and b_i (*resp.* b_j) are the coded blocks (*resp.* original blocks) of our settings. Then s attaches the encoding vector in the header as assured by R^2.

The peer P_2 as stated previously, must be a downstream peer for s for the decoding process:

- P_2 waits until receiving the whole 128 innovative combinations exclusively from s, then reassembles the transfer matrix T.
- P_2 decomposes T as LU, then retrieves the array of secret blocks $S = \mathcal{R}(T)$ via Algorithm 2 where again, matrices L and U are the resulting LU decomposition.

Algorithm 2: The map $\mathcal{R}$ retrieves the array S of 128^2 symbols.

Input: Square transfer matrix T of size 128 and identity matrix I_{128}.
Output: Array of symbols S.

```
int k, i, , j, l = 0;
int[][] U, L;
U ← T;
L ← I_128;
/* The retrieval process is in parallel with the LU-factorisation.
   */
for k = 1; k ≤ 128; k++ do
    p ← U[k][k];
    for i = k + 1, i ≤ 128; i++ do
        q ← U[i][k];
        U[i][k] ← 0;
        L[i][k] ← q/p;
        S[l] ← L[i][k]; for j = k + 1, j ≤ 128; j++ do
            U[i][j] ← U[i][j] − U[k][j].q/p;
            S[(128(128−1))/2 + l] ← U[i][j];
            l++;
return S;
```

That is, the protocol for the whole process is the embedding and retrieval maps below $\mathcal{E}()$ and $\mathcal{R}()$ respectively defined by:

$$\begin{array}{ccc} \mathcal{E} : \mathbb{F}_q^{128^2} & \to & \mathcal{M}_{128}(\mathbb{F}_q) \\ S & \mapsto & LU \end{array} \tag{4}$$

and given by Algorithm 2:

$$\begin{array}{ccc} \mathcal{R} : \mathcal{M}_{128}(\mathbb{F}_q) & \to & \mathbb{F}_q^{128^2} \\ T & \mapsto & S \end{array} \tag{5}$$

Note that the triangular matrices obtained via LU-decomposition are kept to simplify Gaussian elimination in order to obtain the inverse of T for the decoding process. Hence, the complexity of the LU-decomposition does not figure out in the embedding capacity.

Now we consider the case of a streaming session where the sender s wants to send a binary sequence $\mathcal{B}$ of much bigger size $|\mathcal{B}|$. For optimisation purposes say $|\mathcal{B}| = \gamma^2$ for some integer $\gamma \geq 1$. Once again, s can fill the binary sequence with zero bits to adapt to the mentioned size.

The sender s first cuts $\mathcal{B}$ into symbols $\langle \mathcal{B}_i \rangle_q$ for $i \in \{1, 2, \ldots, \gamma^2\}$, *i.e.*, $\lceil \frac{|\mathcal{B}|}{8} \rceil = \gamma^2$ blocks:

$$S = \{\langle \mathcal{B}_1 \rangle_q, \langle \mathcal{B}_2 \rangle_q, \ldots, \langle \mathcal{B}_{\gamma^2} \rangle_q\} \tag{6}$$

Once the set S is prepared, the sender s organizes S into S_l and S_u of sizes $\frac{\gamma(\gamma-1)}{2}$ and $\frac{\gamma(\gamma+1)}{2}$ respectively, *i.e.*, :

$$S = \underbrace{\{\langle \mathcal{B}_1 \rangle_q, \langle \mathcal{B}_2 \rangle_q, \ldots, \langle \mathcal{B}_{\frac{\gamma(\gamma-1)}{2}} \rangle_q}_{S_l}, \underbrace{\langle \mathcal{B}_{\frac{\gamma(\gamma-1)}{2}+1} \rangle_q, \ldots, \langle \mathcal{B}_{\gamma^2} \rangle_q\}}_{S_u} \tag{7}$$

Then s constructs the unilateral lower and the upper matrices:

$$L = \begin{pmatrix} 1 & & & & \\ \mathfrak{b}_1 & 1 & & 0 & \\ \mathfrak{b}_2 & \mathfrak{b}_\gamma & 1 & & \\ \vdots & \vdots & \vdots & \ddots & \\ \mathfrak{b}_{\gamma-1} & \mathfrak{b}_{2\gamma-3} & \cdots & \mathfrak{b}_{\frac{\gamma(\gamma-1)}{2}} & 1 \end{pmatrix} \tag{8}$$

and

$$U = \begin{pmatrix} s_{\frac{\gamma(\gamma-1)}{2}+1} & s_{\frac{\gamma(\gamma-1)}{2}+2} & s_{\frac{\gamma(\gamma-1)}{2}+3} & \cdots & s_{\frac{\gamma(\gamma+1)}{2}} \\ & s_{\frac{\gamma(\gamma+1)}{2}+1} & s_{\frac{\gamma(\gamma+1)}{2}+2} & \cdots & s_{\frac{\gamma(\gamma+3)}{2}-1} \\ & & s_{\frac{\gamma(\gamma+3)}{2}} & \cdots & \vdots \\ & 0 & & \ddots & \vdots \\ & & & & s_{\gamma^2} \end{pmatrix} \tag{9}$$

Using L and U, s computes $T = L \times U$ of size γ and arranges it in an array A as

$$A = \{T_{1,1}, T_{1,2}, \ldots, T_{2,1}, T_{2,2}, \ldots, T_{\gamma,1}, T_{\gamma,2}, \ldots, T_{\gamma,\gamma}\} \tag{10}$$

with a "stand by" state awaiting the establishment of the communication channel.

Now, as stated before, s and P_2 are respectively the sender and a downstream peer of s in a video streaming channel session. Say that s wants to send an m coded blocks $[b_1, b_2, \ldots, b_m]$ with $1 \leq m < n$ in a given transmission phase, then:

- s inserts in the array A a key $K = |\mathcal{B}|$ which corresponds to the number of secret symbols in $\mathcal{B}$. We suppose that γ is a fixed and agreed up-on parameter fixed by s and P_2 (otherwise we can simply insert it in the key K).
- s takes the first m symbols in A as encoding coefficients for $\{b_1, b_2, \ldots, b_m\}$, *i.e.*, s uses the elements A_i respectively as encoding coefficients for b_i where $i = 1, 2, \ldots, m$, then s sends the coded blocks.
- As soon as the next m blocks to code are ready for s to send, s proceeds to the following m elements in A (*i.e.*, $\{A_{m_1+1}, A_{m_1+2}, \ldots, A_{m_2}\}$, in case of $m = m_2$) and encodes the m blocks as the previous steps.

The receiver on the other hand:

- With respect to the order discussed above, P_2 extracts from the header of each downloaded coded block the non-zero encoding coefficients before the decoding process, then saves the symbols in an array A_r.
- P_2 proceeds with the same manner with the following coded blocks until the number of extracted symbols reaches K.
- Finally, P_2 constructs the matrix T of size γ and LU-decomposes it with Algorithm 2 using a size of γ rather than 128.

3.2. **Embedding Capacity.** For the case of one segment streaming with $m = 128$, a sender s can hide in this scheme a maximum of 128^2 blocks each of $1KB$, in one segment uploading session, *i.e.* the embedding capacity in bits per segment (*i.e.*, $C_{e|s}$) and in bits per operation time (*i.e.*, $C_{e|o}$) respectively in this setting are

$$C_{e|s} = 128^2 \ sps = 128K \ bps. \tag{11}$$

$$C_{e|o} = 0,2 \ spo = 1,6 \ bpo. \tag{12}$$

where we denote by *bps* and *bpo*, bits per segments and bits per operation time respectively.

If the number of coded blocks m is less than the number of segment's blocks n (*i.e.*, $m < n$), we can obtain the embedding capacity by multiplying the result in (11) by $\frac{m}{n}$. That is, if we consider an average number of coded blocks $= \frac{n}{2}$ then $C_{e|s} = 56K \ bps$ and $C_{e|o} = 0,8 \ bpo$.

Remark 3.1. *To decrease the overhead of the packets in case $m = n$, the authors of R^2 proposed to attach the seed of the Random Number Generator (RNG) of the coding coefficients rather than the coefficients themselves in case the sender peer has all the n packets. To cope with this setting we consider that the sender uses always $2 \leq m < n$.*

3.3. **Example.** Consider the network in Figure 3. For simplicity we consider the sender to be the source s and the receiver the peer P_1. Now, say that $\mathcal{B}$ is a large sequence of bits such that $|\mathcal{B}| \leq 128Kb$, *i.e.*, in an optimal setting

$$\mathcal{B} = 0110101111101011 \cdots 10011010111 \tag{13}$$

such that $|\mathcal{B}| = 128Kb$.

Now, the sender organizes $\mathcal{B}$ into an array S of symbols $i \in \{1, 2, \ldots, \lceil \frac{|\mathcal{B}|}{8} \rceil\}$ of 8 bits (*i.e.*, into symbols in $\mathbb{F}_q$ where $q = 2^8$). As stated before, if $|\mathcal{B}|$ does not exceed the value of 128 symbols the sender simply fills the last symbol by zero bits to adapt to the symbol size then picks uniformly and randomly the rest of the symbols as assured by the R^2 protocol.

Hence, sender s constructs the triangular matrices L and U respectively as stated in (2) and (3), and as described in Algorithm 1 constructs the transfer matrix T:

$$T = \mathcal{E}(S) = L \times U \tag{14}$$

Then, s sends $m = 128$ packets $Y = T \times X$, where X is the vector of the 128 original packets of a given segment. The encoding vectors which are the rows of matrix T are attached to the headers as assured by R^2.

The receiver (*i.e.*, peer P_1 in Figure 3) waits until it receives all m packets from the sender s and rejects any other received packets from other peers in the network. When P_1 collects all the m packets, it decodes both the packets and the secret array S using LU-decomposition as described in Algorithm 2.

Remark 3.2. *In case the sender is another peer (e.g., P_2), then P_2 must wait until it receives all the* 128 *innovative packets, decodes them and then combines linearly $m = n-1$ original packets before sending them to the receiver, say P_3 in our example network.*

3.4. **Efficiency Comparison.** Since the segment is divided into 128 blocks and the block size is $1KB$. The embedding capacity is

$$C_{e|b} = 128^2 \ sps \approx 128M \ bps.$$

and

$$C_{e|o} = 0,2 \times 1024 \times 8 \ bpo \approx 1638,4 \ bpo.$$

Thus, for a 1 GHz single Core processor the embedding capacity in bits per second (bs^{-1}) is

$$C_{e|o} = 1638,4 \times 10^9 bs^{-1} \approx 1,6.10^3 Gbs^{-1}.$$

Furthermore, the scheme remains distortion-less since the original packets are fully recovered.

Hereafter we compare the secret channel capacity of our scheme with some well known network steganographic protocols. That is, we consider using the IEEE 802.11n technology. Hence, our comparison is based on the Orthogonal Frequency Division Multiplex (OFDM). The transmission time of a single OFDM symbol is given by $T_{sym} = T_u + T_{CP}$ where T_u is the useful part (*i.e.*, useful time) of the symbol and T_{CP} is the duration of the Guard Interval (GI). The GI is inserted between each pair of symbols as a delimiter to ensure that distinct transmissions do not interfere. In OFDM, the beginning of each symbol is preceded by a GI of size T_{CP} of the ending part of that symbol. That is, this process consumes one T_{CP} from the useful time T_u.

Furthermore, we use the most basic IEEE 802.11n coding scheme which uses BPSK modulation, code rate $\frac{1}{2}$ and 20 MHz channels. In this case $T_{sym} = 4\mu s$ and the Cyclic Prefix CP (the GI) has a transmission time $T_{CP} = 0,8\mu s$. There are $S_R = 250000$ OFDM symbols per second transmitted, each symbol carries 52 coded bits denoted by N_{CBPS}, and half of them are data bits referred to as N_{DBPS}. Then, the achieved network throughput is $6,5$ Mb/s. In the case of BPSK modulation, the useful part of the OFDM symbol (*i.e.*, $T_u = 3,2\mu s$) carries 52 bits.

Now, we consider a one sender/one receiver stable route with a 100% delivery rate. Additionally, since for each coded block of $1KB$ we need an overhead of $n = 128$ bytes (*i.e.*, a ratio of $\frac{128}{1024} = 0,125 = 12,5\%$), the capacity of the secret channel is then:

$$C = N_{CBPS} \times \frac{T_u - T_{CP}}{T_u} \times S_R \times 12,5\% \tag{15}$$

That is, $C = 1,162$ Mb/s ($3,068$ Mb/s RRNS-based and CP-scheme).

Furthermore, if the used canal has good propagation conditions, the OFDM cyclic prefix in this case is divided by two, *i.e.*, $T_{CP} = 0,4\mu s$ rather than $T_{CP} = 0,8\mu s$ which obviously divides the channel capacity of the CP-scheme by two for all modulations (BPSK, QPSK, 16-QAM and 64-QAM). That is, the embedding capacity for each modulation becomes:

- $C = 1,356$ Mb/s for BPSK.
- $C = 2,712$ Mb/s for QPSK.
- $C = 5,424$ Mb/s for 16-QAM.
- $C = 8,136$ Mb/s for 64-QAM.

In Table 1, we give a comparison with some well known NS schemes [17, 19, 20], where WLAN-/HW stands for Wireless Local Area Networks/ Hardware Wi-Fi based implementations, IEEE 802.11 specifies the set of Media Access Control (MAC) and physical layer protocols for implementing (WLAN) Wi-Fi communications, and SS for Spectrum Selection.

3.5. **Perspectives on Security.** Studying possible attacks on our technique is beyond the scope of this paper. Nevertheless, at a first glance, the only attacks that could break this steganographic scheme, since it profits of the random property of MORE, are statistical attacks, and precisely the two-samples Kolmogorov-Smirnov (KS) test [37].

Suppose $X = [X_1, X_2, ..., X_{\gamma^2}]$ to be a series of random variables with values $x_1, x_2, ..., x_{\gamma^2}$.

The two-samples KS test verifies the hypothesis that two samples are drawn from the same distribution. A low KS test statistic means that the distributions are similar, whereas a high KS test statistic means the distributions are different.

TABLE 1. Comparison with some steganographic scheme.

Channel Covert	Used carrier	Embedding capacity in bps
WLAN/HW	IEEE 802.11 FCF [7]	$16,8$
WLAN/HW	IEEE 802.11 [19]	$216K$
WLAN/HW	IEEE 802.11 Padding [20]	$1,1M$ for data frames, $0,44M$ for ACKs
Network/SS	VoIP stream payload [17]	$32K$
WLAN/HW	IEEE 802.11 R^2 The proposed scheme	For $\boldsymbol{T_{CP} = 0,8\mu s}$: 1,162 M (BSPK), 2,324 M (QPSK), 4,648 M (16-QAM) and 6,972 M (64-QAM). For $\boldsymbol{T_{CP} = 0,4\mu s}$: 1,356 M (BSPK), 2,712 M (QPSK), 5,424 M (16-QAM) and 8,136 M (64-QAM).

KS test is applicable to a variety of types of data with different distributions.

Let $F(x)$ be the empirical cumulative distribution function of X. The KS test statistic for two empirical distribution functions $F_1(x)$ and $F_2(x)$ is :

$$D_{KS} = sup_x|F_1(x) - F_2(x)| \tag{16}$$

where sup_x is the the least upper bound of the set of distances, and for $i = 1, 2.$:

$$F_i(x) = \frac{1}{\gamma^2}\sum_{i=1}^{\gamma^2} \uparrow_{x_i \leq x} . \tag{17}$$

Where we denote by $\uparrow_E$, the indicator function of the event E.

Hence, an attacker who knows the R^2 protocol and its header's structure, observes different samples of R^2's coded transmissions between the sender and the receiver, then collects their transfer matrices and tests them via KS will probably find a high statistic.

Note that the transfer matrix carrying the hidden information exists only between the sender and its downstream peers and hence, the adversary must be a downstream peer for the sender in order to perform his attack. That is, as soon as one of these peers uses coding operation for its downstream peers, the information is lost and hence, no attack or observation can be performed.

As stated above, the existence of hidden data can be detected, and the system confronted to passive and/or active statistical attacks. However, the high embedding capacity of the scheme allows to send m^2 secret symbols in each transmission of m coded blocks from a given peer to its downstream peers (for successful decoding) and hence, it is possible to counter the statistical attacks by using a non-uniformly agreed up on transmission phases to send the secret data.

4. **Conclusion.** In this paper, we propose a distortion-less information hiding scheme using the live P2P video streaming protocol R^2. We show how effective this scheme is in term of embedding capacity compared with some well known schemes in the literature. In a future work, we aim to analyse the security of the scheme and propose further techniques to counter both passive and active adversarial attacks.

Acknowledgment. The authors gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation of this paper.

REFERENCES

[1] W. Bender, D. Gruhl, N. Morimoto, A. Lu, *Techniques for data hiding.* IBM Systems Journal 35, pp.313–336, 1996.

[2] J. Bierbrauer, J.J. Fridrich, *Constructing good covering codes for applications in steganography*, Trans. Data Hiding and Multimedia Security 3, pp.1–22, 2008.

[3] Jankowski, B., Mazurczyk, W., Szczypiorski, K., Padsteg: introducing inter-protocol steganography. Telecommunication Systems 52, pp.1101–1111, 2013.

[4] M.A. Belhamra, E.M. Souidi, *A steganogaphic scheme for MAC-Independent Opportunistic Routing and Encoding (MORE) protocol*, in: Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, Volume 2, SECRYPT, Porto, Portugal, July 26-28, pp. 254-264, 2018.

[5] M.A. Belhamra, E.M. Souidi, *Steganogaphy using Mac-Independent Opportunistic Routing and Encoding (MORE) protocol based communications*, in: E-Business and Telecommunications. Springer. volume 1118 of Communications in Computer and Information Science. Chapter 4, 2019.

[6] S. Grabski and K. Szczypiorski, Steganography in OFDM symbols of fast IEEE 802.11n networks. In *2013 IEEE Symposium on Security and Privacy Workshops, San Francisco, CA, USA, May 23-24, 2013,*2013.

[7] C. Krätzer, J. Dittmann, A. Lang, T. Kühne, *WLAN steganography, a first practical review*, in: Proceedings of the 8th workshop on Multimedia & Security, MM&Sec 2006, Geneva, Switzerland,September 26-27, pp. 17–22, 2006.

[8] D. Kundur, K. Ahsan, *Practical internet steganography: data hiding in IP.* Proc. Texas wksp. security of information systems, 2003.

[9] W. Mazurczyk, L. Caviglione, *Steganography in modern smartphones and mitigation techniques.* IEEE Communications Surveys and Tutorials 17,pp.334–357, 2015.

[10] W. Mazurczyk, M. Smolarczyk, K. Szczypiorski,*Retransmission steganography and its detection.* Soft Comput. 15,pp.505–515, 2011.

[11] C. Munuera, *Steganography and error-correcting codes.* Signal Processing 87, pp.1528–1533, 2007.

[12] M.A. Belhamra, E.M. Souidi, *Introduction to steganography in RRNS based communications*, in: Proceedings of the 2nd International Conference on Networking, Information Systems & Security, ACM, pp. 21:1–21:7, 2019.

[13] W. Zhang and S. Li. *A coding problem in steganography. Des. Codes Cryptography*, 46(1):67–81, January 2008.

[14] M.A. Belhamra, E.M. Souidi, *Steganography over redundant residue number system codes*, Journal of Information Security and Applications 51, 102434, 2020.

[15] S. Weng, W. Tan, B. Ou, J. S. Pan, *Reversible data hiding method for multi-histogram point selection based on improved crisscross optimization algorithm*, Information Sciences, Volume 549, pp.13–33, 2021.

[16] J. S. Pan, X. X. Sun, S. C Chu, A. Abraham, B. Yan, *Digital watermarking with improved SMS applied for QR code*, Engineering Applications of Artificial Intelligence, Volume 97, 104049, 2021.

[17] W. Mazurczyk, P. Szaga, K. Szczypiorski,*Using transcoding for hidden communication in IP telephony.* Multimedia Tools Appl. 70, pp.2139–2165, 2014.

[18] W. Mazurczyk, K. Szczypiorski, *Covert channels in sip for VoIP signalling*, in: Jahankhani, H., Revett, K., Palmer-Brown, D. (Eds.), Global E-Security, Springer, Berlin, Heidelberg, vol 12, pp. 65–72, 2008.

[19] K. Szczypiorski, *Steganography in TCP/IP networks*, in State of the Art and a Proposal of a New System-HICCUPS, Institute of Telecommunications' seminar, Warsaw University of Technology, Poland, 2003.

[20] K. Szczypiorski, W. Mazurczyk, *Steganography in IEEE 802.11 OFDM symbols.* Security and Communication Networks 9, pp.118–129, 2016.

[21] M. Van Horenbeeck, *Deception on the network: thinking differently about covert channels*, 7th Australian Information Warfare and Security Conference, Edith Cowan University, Perth Western Australia, 4th - 5th December, 2006.

[22] Wang, M., Li, B., 2007a. Lava: A reality check of network coding in peer-to-peer live streaming, in: INFOCOM 2007. 26th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 6-12 May 2007, Anchorage, Alaska, USA, pp. 1082–1090, 2007.

[23] M. Wang, B. Li, *R2: random push with random network coding in live peer-to-peer streaming*. IEEE Journal on Selected Areas in Communications 25, pp. 1655–1666, 2007.
[24] S. Wendzel, S. Zander, B. Fechner, C. Herdin, Pattern based survey and categorization of network covert channel techniques. ACM Comput. Surv., 47, 50:1–50:26, 2015.
[25] A. Westfeld, *F5-A steganographic algorithm*, In Information Hiding, 4th International Workshop, IHW 2001, Pittsburgh, PA, USA, April 25-27, 2001, Proceedings, pp. 289–302, 2001.
[26] M. Zhang, Q. Zhang, L. Sun, S. Yang, *Understanding the power of pull-based streaming protocol: Can we do better?* IEEE Journal on Selected Areas in Communications, Volume 25, pp.1678–1694, 2007.
[27] E. Zieliska, W. Mazurczyk, K. Szczypiorski, *Trends in steganography*. Communications of the ACM 57, pp.86–95, 2014.
[28] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, *CoolStreaming/DONet: A Data-driven Overlay Network for Efficient Live Media Streaming*, in Proceedings of IEEE INFOCOM, 2005.
[29] R. Ahlswede. *Network coding*. In Probabilistic Methods and Distributed Information, Springer, pp.333–357, 2019.
[30] S. R. Li, R.W. Yeung, and N. Cai. *Linear network coding*. IEEE Trans. Information Theory, 49(2), pp. 371–381, 2003.
[31] T. Ho, R. Koetter, M. Médard, D. R. Karger, and M. Effros, *The benefits of coding over routing in a randomized setting*. IEEE Int. Symp. Information Theory, 1228459, 2003.
[32] A. Tassi, I. Chatzigeorgiou, and D. Vukobratovic, *Resource-allocation frameworks for network-coded layered multimedia multicast services*. IEEE J. Sel. Areas Commun., 33(2), pp.141–155, 2015.
[33] A. Tassi, C. Khirallah, D. Vukobratovic, F. Chiti, J. S. Thompson, and R. Fantacci, *Resource allocation strategies for network-coded video broadcasting services over lte-advanced*. IEEE Trans. Vehicular Technology, 64(5), pp.2186–2192, 2015.
[34] N. Thomos and P. Frossard, *Toward one symbol network coding vectors*. IEEE Communications Letters, 16(11), pp.1860–1863, 2012.
[35] D. Vukobratovic and V. Stankovic, *Unequal error protection random linear coding strategies for erasure channels*. IEEE Trans. Communications, 60(5), pp.1243–1252, 2012.
[36] R. A. Horn and C. R Johnson, *Topics in matrix analysis*. Cambridge University Press, 1994.
[37] A. Justel, D. Peña, R. Zamar, *A multivariate kolmogorov-smirnov test of goodness of fit*. Statistics & Probability Letters, **35**(3), pp.251–259, 1997.