Military Technical College
Kobry El-Kobbah,
Cairo, Egypt

8th International Conference
on Electrical Engineering
ICEENG 2012

New Steganographic Methods for Data Hiding in the Payloads of Marked IP Datagrams

By

Manal A. Shehab Noha O. Korany*

Abstract:

The payload of a datagram carries the user data which is required to be transmitted from the source to the destination. So to hide sensitive data in a datagram payload; the data should be first encrypted or hashed then embedded in the payload. Furthermore; the datagram source needs to mark the stego datagrams and mix them with normal ones, and the datagram receiver needs to check for the mark to identify the stego datagrams and excludes them away from the received stream, and then decodes the hidden data.

This paper suggests two steganographic methods for data hiding in the payloads of marked IP datagrams. The first suggested method uses an appropriate encryption algorithm and key to encrypt the plaintext blocks, then embeds the resulted ciphertext blocks in the payloads of marked IP packets. The second method hides data in the payloads of marked IP packets' fragments sets. In the second method; IP packet fragmentation is required as the fragment offset field of the fragment (except the first fragment) is used by an intelligent way to hash data blocks before embedding them in its corresponding fragment payload, encryption could be used as an option to encrypt the data before or after hashing it, the matter which provides different available scenarios for this method. The paper also briefly discusses the confidentiality effect of using the IPsec encryption with its different modes with each suggested steganographic method.

Keywords:

Steganographic, data hiding and mark.

---

* Electrical Engineering Department, Faculty of Engineering, Alexandria University - Egypt

1. Introduction:

Lucena et al. identified a network steganographic method for the IPv6 protocol based on fake fragments insertion. Two solutions were proposed to avoid including an inserted fragment in the reassembly process of the original IP packet. The first is based on inserting an invalid value in the IP ID field in the fragment extension header of the fake fragment to help the receiver from identifying and excluding this fragment in the reassembly. The second is inserting an overlapping fragment offset value which causes data overwritten during the reassembly. The main disadvantage of Lucena's method is being easy to detect because the warden can monitor all the fragments and detect abnormal behaviors of the fragments like having overlapping offsets or single unrelated fragments [1].

W. Mazurczyk and K. Szczypiorski developed a steganographic method that they refer to it by the F3 method in [1]. The method is an enhancement of Lucena's method. The F3 method uses legitimate fragments with embedded steganograms in their payloads for high steganographic bandwidth and hard detectability.

In this paper we will briefly describe the steganographic method F3, then we will improve the F3 method by suggesting two new stegnographic methods M2 and M3 for data hiding in the payloads of marked IP packets and marked IP packets' fragments sets respectively [2]. The IP ID of the stego datagrams in the M2 and the M3 methods should be marked by using the new steganographic marking method which was suggested in [2] & [3].

F3, M2 and M3 steganographic methods are not considered as covert channels because information hiding in the packet payload is outside the realm of the network covert channels and fit into the general field of the steganography as the payload is the default carrier of the transmitted data.

2. Mazurczyk's Steganographic Method (F3) for Data Hiding in the Payloads of Marked Legitimate Fragments:

2.1 Data Hiding Scheme of the F3 Steganographic Method:

In the F3 method [1]; the steganogram sender (SS) must be the source of the packet fragmentation. Instead of inserting user data into the payloads of the selected fragments, SS inserts steganograms. The main idea of this approach is properly marking the fragments which are used to carry the hidden data to help the steganogram receiver (SR)

in identifying and excluding them away by a way that not interfere them with the reassembly process.
Mazurczyk proposed a procedure to mark the stego fragments to be hard for a warden to detect. He assumed that the SS and the SR share a secret Steg-Key (SK). For each chosen fragment for the steganographic communication; the following hash function H is used to calculate an Identifying Sequence (IS):

IS = H (SK || Fragment Offset || Identification) (1)

Where the Fragment Offset and the Identification denote their values from the corresponding fields of the IP header of the stego fragment and '||' is the bit concatenation function.

For every used fragment in the hidden communication; the resulted IS would have a unique value due to having a unique pair of the fragment offset and IP ID fields. All the IS bits or only selected ones could be distributed across the payload of the stego-fragment in a predefined manner to mark it.

For each incoming fragment; SR calculates the IS based on the values of SK, IP ID and fragment offset of the fragment, then SR checks if the fragment payload carries a steganogram or a user data. If the verification is successful, then the rest of the payload is considered as hidden data and extracted by the SR which skips this fragment in the reassembly process of the original IP packet.

If the number of the used fragments which are used to hide data in each packet is N_F, then the F3 method usually uses the last N_F fragments to carry the hidden data in each packet to avoid having missed fragment offsets in the first or the middle of the packet's fragments set during the reassembly process.

Figure 1 illustrates an example for the F3 steganographic method. The IP packet with ID 345 and size of 1500 bytes is divided into three fragments Fr1=740, Fr2= 740 and Fr3=60 bytes respectively [1]. The last fragment Fr3 is used for steganographic purposes, so a steganogram is inserted together with the IS inside its payload. Values in Fragment Offset (FO) and IP ID fields remain the same as in other legitimate fragments. While reassembling the original packet, SR merges the payloads P1, P2 and omits the fragment Fr3 to use it only in extracting the steganogram from its payload.

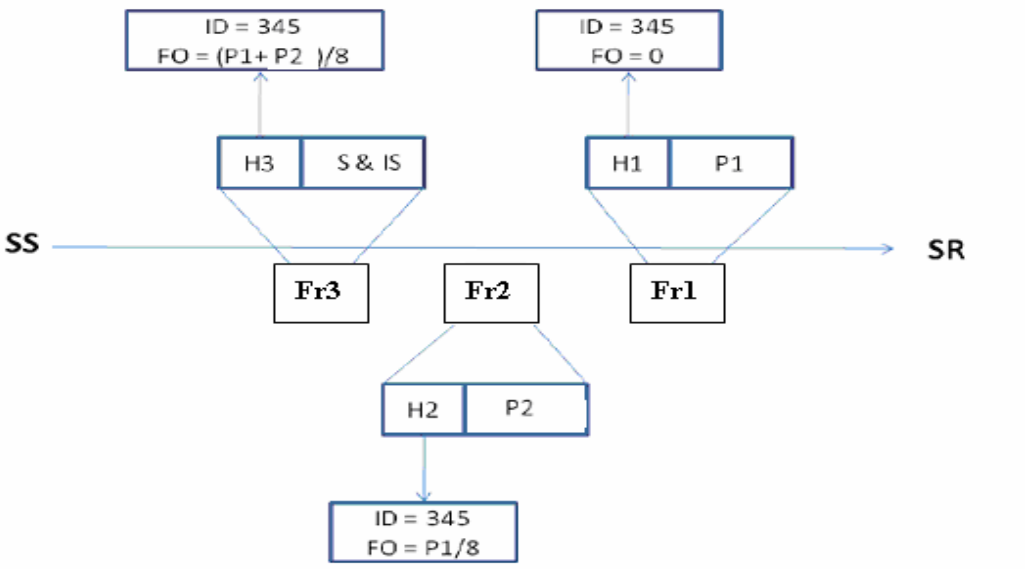


Figure (1): F3 steganographic method (H – Header, P – Payload, S- Steganogram)

2.2 Evaluation of the F3 Steganographic Method:

2.2.1 Advantages of the F3 Steganographic Method:

1- Method F3 uses legitimate fragments as hidden data carriers.

2- Steganographic bandwidth for the F3 method = $\Sigma_{NF} F_s$ [bits / packet] (2)
Where N_F is the number of fragments which are used to hide data in each packet and F_S is the size of the fragment payload after excluding the bits of the IS, regarding that the size of the last fragment of a packet usually differs than the other fragments.

2.2.2 Disadvantages of the F3 Steganographic Method:

1- Using legitimate fragment to carry the IS and the hidden data in its payload may need re-engineering processes for the packet's fragments set. This might consume the SS time and resources.

2- Skipping one or more fragments from a packet in its reassembly process may cause the following:
A- Detection for illogical reassembly process of fragments at the SR.
B- Consuming the performance and resources of the SR in remodeling the default reassembly process to skip the stego fragments and avoid waiting them.

2.2.3 Hidden Communication Scenario of the F3 Steganographic Method:

For the F3 steganographic method, SS must be the packet source and the source of packet fragmentation [1].

3. New Suggested Steganographic Method (M2) for Data Hiding in the Payloads of Marked IP Packets:

3.1 Data Hiding Scheme of the M2 Steganographic Method:

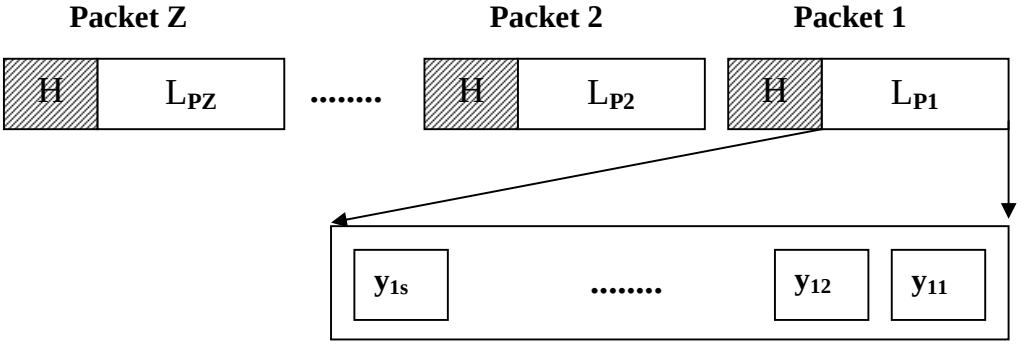
The M2 steganographic method could be applied to both versions of the IP packets. In the M2 method; the SS inserts a steganogram in the whole payload of the IP stego packet that could be marked by including a mark in its IP ID field using the new steganographic marking method which was suggested and described in [2] & [3]. For IPv6; inserting a pre-agreed mark within the IP ID requires packet fragmentation by the packet source to include the IPv6 fragment header extension. In M2; the SR checks for the marked IP packets, identifies them as stego packets, excludes them away from the received traffic and extracts the steganogram from its payload.

Figure 2 shows the distribution of an M (bits) covert message through the payloads of Z marked stego packets in the M2 Steganographic method regarding the following:

1- If each of the plaintext message and its corresponding ciphertext message has a size of M (bits) and the packet payload size of the packets which are used to hide the M (bits) message is L_P (bits) each, then the number of the stego packets which are required to carry the M (bits) covert message is $Z = M / L_P$ (3)

2- Dividing the M (bits) covert message to a number of (y-bit) blocks = M / y (4)

3- The number of (y-bit) blocks which would be encrypted and then embedded in the packet payload is S regarding that the chosen value y should divide L_P to avoid padding the last blocks. So; $S = L_P / y$ (5)



H Total Headers of a packet with a mark in the IP ID field of the IP header

Figure (2): Embedding an M (bits) encrypted message in the form of (y-bit) blocks within the payloads of Z marked packets

Assume that Alice is the SS and Bob is the SR. The general framework of the M2 steganographic method at the SS side is shown in figure 3, and it requires from Alice to do the following:

1- Generate D packets stream with a packet payload size = L_P (bits) each. The D stream randomly includes Z marked packets which carry the pre-agreed mark in their IP ID, where D > Z. The Z marked packets will be used to carry the M (bits) covert message, while the other (D- Z) packets will be used to carry a normal data (of any normal file).

2- Exclude the Z marked packets from the D packets stream.

3- Encode the M (bits) plaintext message as follows:

I- Divide the M (bits) plaintext message to (y-bit) blocks. We could use the character as a block data unit and divide the plaintext to (8-bit) blocks (i.e. y = 8, which is the 8-bit binary equivalent of the ASCII of the character).
II- Encrypt each (y-bit) block of the plaintext using a suitable pre-agreed encryption algorithm and key to produce the (y-bit) ciphertext block. The conditions of the used encryption algorithm in the M2 steganographic method are:
- It should be reversible (invertible) to allow the SR from retrieving the plaintext.
- It should convert the (y-bit) plaintext block to (y-bit) ciphertext block. If y = 8, we could use the modular exponential cipher with modulus 257 [2]. If y = 8 or 64, we could use the new suggested encryption method which was described in [2] & [4].

4- Concatenate each of the S sequenced (y-bit) ciphertext blocks within a larger (L_P-bit) block as was shown in figure 2.

5- Embed every (L_P- bit) encoded block according to its sequence into the payload of its corresponding packet through the Z packets to generate the Z stego packets as was shown in figure 2.

6- Embed the normal data in the form of (L_P-bit) blocks through the payloads of the (D - Z) normal packets regarding the data order.

7- Returning back the Z stego packets to their positions within the D packets stream. Then transmitting the D packets stream to the SR.

The general framework of the M2 steganographic method at the SR side is shown in figure 4, and it requires from Bob to do the following:

1- Receive the D packets stream.

2- Find the Z marked stego packets by checking for the pre-agreed mark in the IP ID field, then exclude them away from the D received packets.

3- Exclude the (L_P-bit) encoded data block from each payload of the sequenced Z stego marked packets.

4- Concatenate the excluded (L_P-bit) blocks from the Z packets by their ascending sequence to get the complete M (bits) ciphertext message.

5- Decode the M (bits) ciphertext message as follows:
   I- Divide the ciphertext message to (y-bit) blocks.
   II- Decrypt each (y-bit) block of the ciphertext using the pre-agreed decryption algorithm and key to produce the (y-bit) plaintext block.
   III- Concatenate the (y-bit) plaintext blocks in sequence to get the M (bits) plaintext message.

6- Exclude the normal (L_P-bit) data block from each payload of the sequenced D-Z normal packets.

7- Concatenate the normal (L_P-bit) data blocks by their ascending sequence to get the complete normal data.

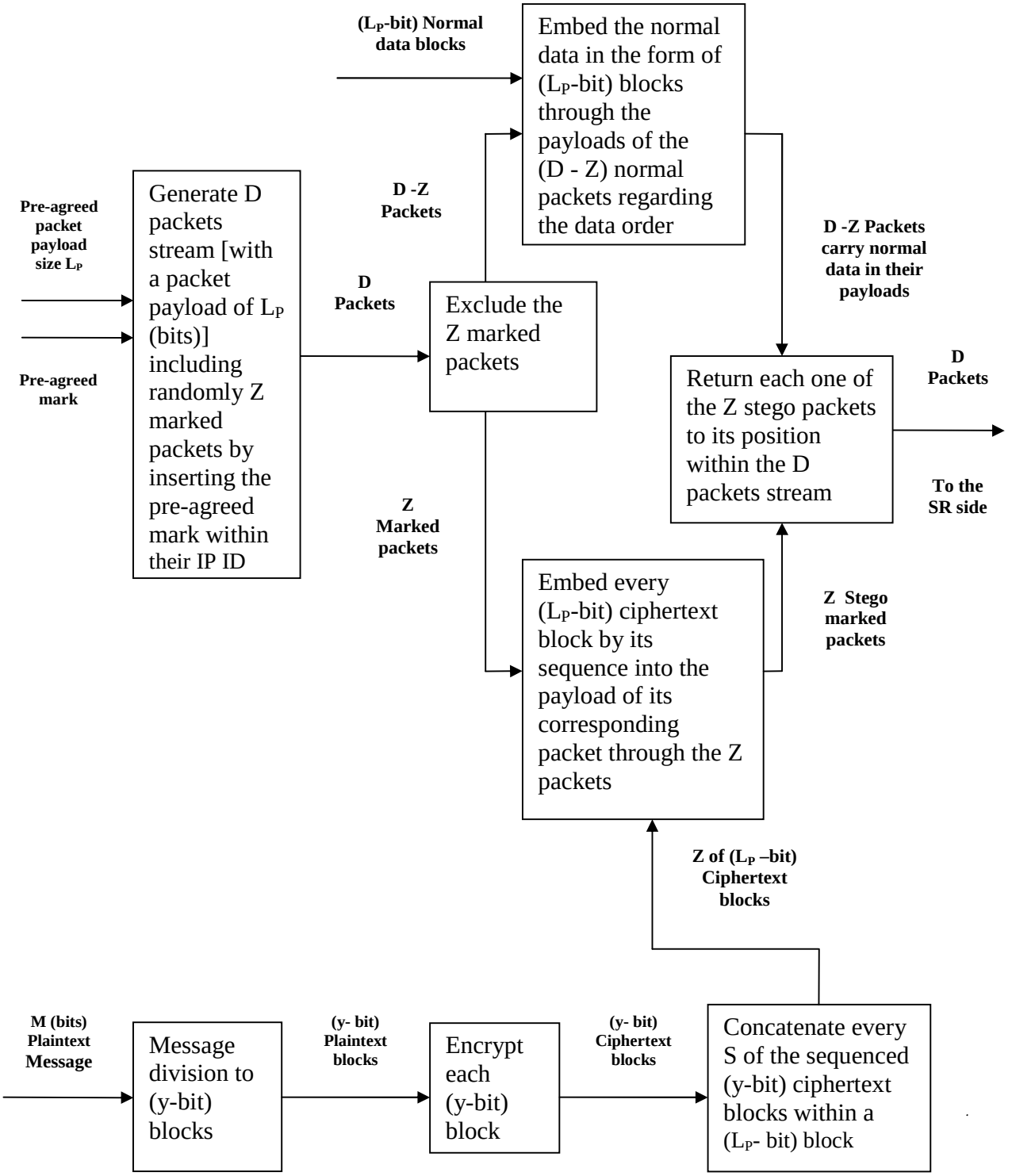


Figure (3): General framework of the M2 steganographic method at the steganogram sending side

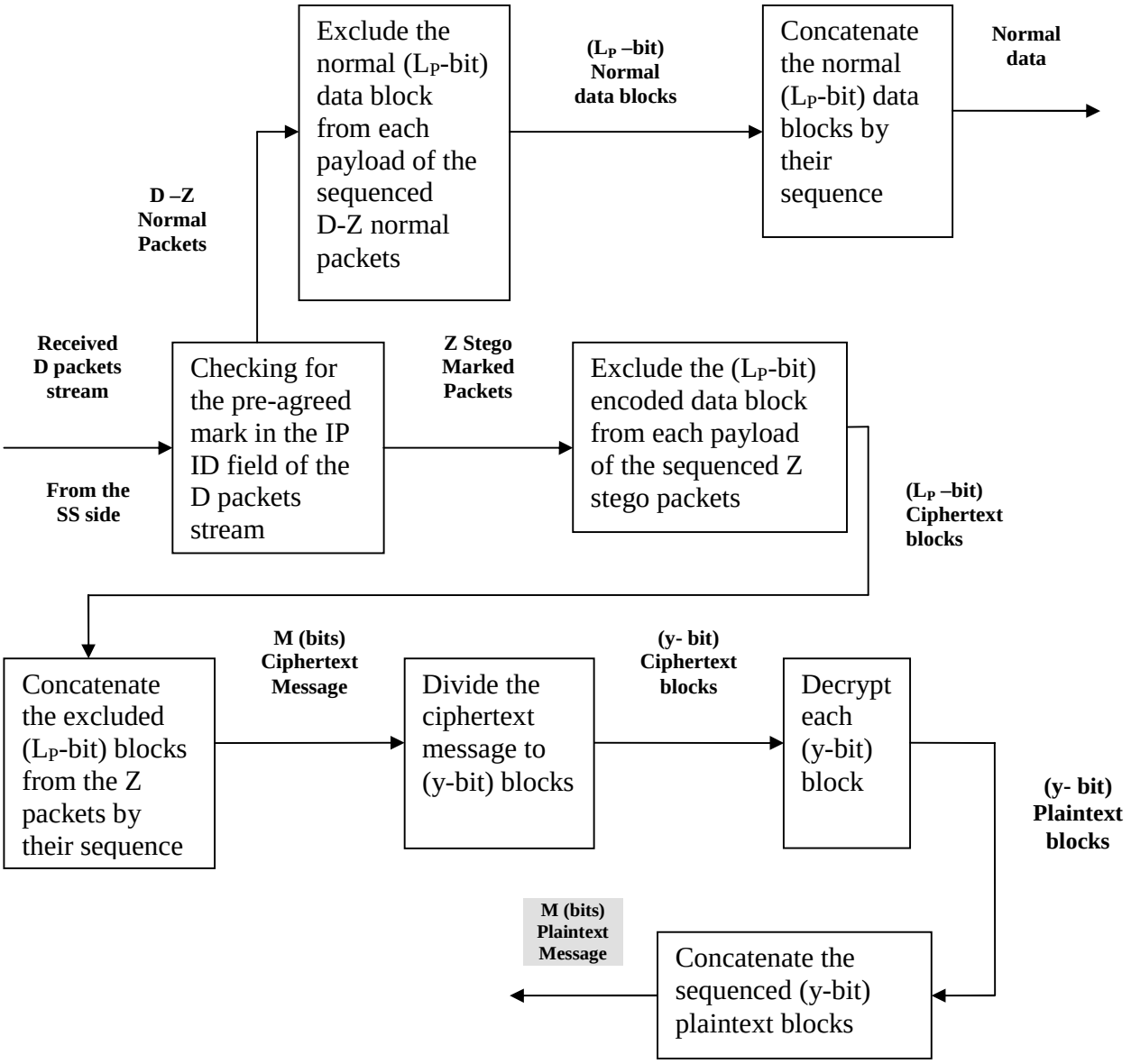


Figure (4): General framework of the M2 steganographic method at the steganogram receiving side

3.2 Evaluation of the M2 Steganographic Method:

3.2.1 Advantages of the M2 Steganographic Method:

1- The steganographic bandwidth of M2 is higher than F3, because using the whole packet payload as a carrier for the hidden data in M2 provides greater hidden size than using the payload of some fragments of the packet in F3. Also there's no consumption for the available hidden size per packet payload to embed the identifying mark, because the mark in M2 is embedded in the IP ID field. Therefore; the M2 steganographic bandwidth = L_P [bits / packet], where L_P is the payload size of the stego-packet (in bits).

2- The M2 steganographic method might be used with the IPv4 or the IPv6 packets.

3.2.2 Disadvantages of the M2 Steganographic Method:

Programming and storage buffers may be required in different stages of the M2 implementation at SS and SR to keep data blocks and packets in their correct sequence and to avoid the problems of having missing data, data disorder or out-of-order packets. If packets were being enforced at the receiving node without any buffering of the out-of-order packets, then the fragmented IP packet may never reach its destination, this is true especially for any tunneling mechanism of the IP packets that includes sequence checking on the reception (as the IPsec tunneling) [5]. Note that; packet ordering could be employed to introduce covert information as in Ashan's packet sorting/resorting steganographic technique [6] which utilizes the sequence number field of the Authentication Header (AH) or the Encapsulating Security Protocol (ESP) headers of the IPsec protocol to reveal hidden information.

3.2.3 Hidden Communication Scenario of the M2 Steganographic Method:

In the M2 steganographic method; the SS must be the packet source and the SR must be the packet receiver, because network routers and gateways are not designed to implement the framework steps of the M2 method in either the SS or the SR. Also unusual packet processing in intermediate nodes could attract the detection of a warden.

4. New Suggested Steganographic Method (M3) for Data Hiding in the Payloads of Marked IP Packets' Fragments Sets:

4.1 Data Hiding Scheme of the M3 Steganographic Method:

The M3 steganographic method hides data in the fragments' payloads of marked IP packets. For each fragment in the fragments set of a marked packet, 8 bits from the 13 (bits) of the fragment offset (FO) field are chosen to construct a hashing key to hash the (8-bit) plaintext or ciphertext blocks before embedding them in the payload of that fragment. There are many available scenarios for the M3 method depending on the type of the (8-bit) input block (B) to the used hashing algorithm, and the order of the decided steps of each scenario as using encryption in M3 is an option that could take place after or before hashing. M3 uses the steganographic marking method which was suggested in [2] & [3] to mark its IP stego packets so that the SR could identify them to exclude them away from the received stream.

The fragment offset has a different value for each fragment of a packet which provides a different (8-bit) hashing key for each fragment within the marked packet's fragments set. These keys act as a set which is repeated for every one of the Z marked packets' fragments sets, where Z is the number of required packets to carry a hidden message of size M (bits). The reason of hashing data in the form of (8-bit) blocks is because fragments other than the last fragment have a payload size which is multiple of 64 bits, and the total length of the IP datagram is presented in octets. Therefore; the size of the payload could be divided by 8 and the data in the payload could be presented in the form of (8-bit) data units.

The M3 method implies that the SS and the SR should agree about the mark which identifies the stego packets, the packet size, the MTU of the utilized fragmentation for which the SS is the source of fragmentation and the SR is the reassembly processor, the pre-defined manner of selecting the (8-bit) hashing key from the fragment offset, the used hashing algorithm, the decided steps of the used M3 scenario with their sequence order and the used encryption algorithm and key (if encryption was decided).

If the used packet in the M3 steganographic method has a payload size = L_P (bits) and the size of its total headers = H (Bytes), then the total packet size = $H + L_P / 8$ (Bytes). The packet would be fragmented according to the pre-agreed MTU to a predictable number = 'J' fragments. As one of the fragmentation necessities is the pre-agreed MTU should be smaller than the total packet size [i.e. $MTU < H + (L_P /8)$]. Each fragment in the fragment set except the last fragment would have a total length = MTU (Bytes) and a payload size = $L_F = 8$ (MTU- H) in (bits).

Assume that the fragment number i (Fr_i) has a 13-bit fragment offset field denoted as FO_i, where $1 \leq i \leq J$. Figure 5 shows the fragment set of a packet that has J fragments, where the FO_i value is measured by the 8-byte unit. The figure reflects the following fragmentation review:

- The first fragment Fr_1 has $FO_1 = 0$
- The second fragment Fr_2 (if it's not the last) has $FO_2 = (MTU - H) /8 = L_F/ 64$.
- The third fragment Fr_3 (if it's not the last) has $FO_3 = 2 [MTU - H] / 8= 2 L_F / 64$.

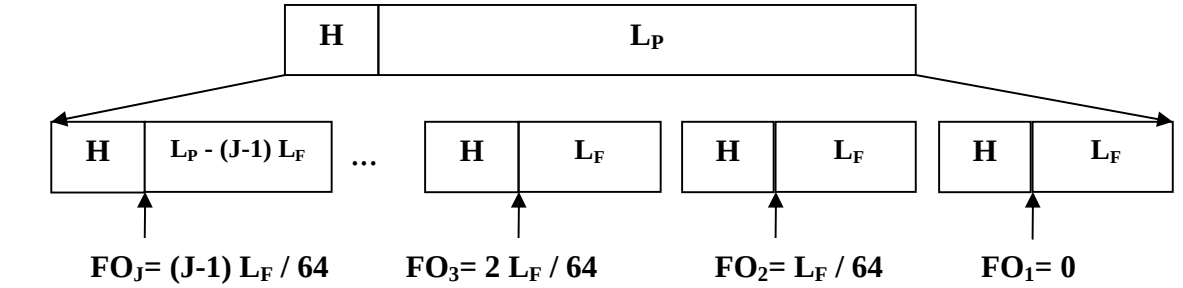
And so on till reaching the last fragment.

- The last fragment Fr_J has $FO_J = (J-1) [MTU – H] /8 = (J-1) L_F /64$;

The total length of the last fragment = packet size - FO_J

$= H + (L_P /8) – [(J-1) (MTU – H)] = H + (L_P /8) - [(J-1) (L_F /8)]$ in (Bytes).

The payload size of the last fragment = $[L_P – [(J-1) L_F]] / 8$ in (Bytes).



H Total Headers of a packet (in bytes)

L_P : Packet payload size in bits, **L_F** = 8 (MTU –H) in bits, where MTU in (bytes).
FO_i : Fragment Offset of the fragment number i presented by the 8-byte unit, where $1 \leq i \leq J$.

Figure (5): Fragmenting a packet of a size greater than the path MTU to a set of J fragments

If Alice and Bob pre-agreed about the used packet size, the MTU and the (8-bit) selection algorithm of the hashing keys, and if they simply choose the M3 steganographic scenario that doesn't include the encryption option, then the general framework of this scenario at the SS side is shown in figure 6 and it requires from Alice to do the following:

1- Generate the J (8-bit) hashing keys set by the knowledge of the pre-agreed packet size and MTU as follows:

I- Calculate the J fragments offsets set as $[0, L_F / 64, 2 L_F / 64, ..., (J-1) L_F / 64]$.

II- Select the 8 bits of the hashing key from the corresponding fragment offset according to the pre-agreed manner. The first fragment of any packet has a zero fragment offset, thus any 8 selected bits from it for the hashing key would introduce a zero key. To overcome this problem with the M3 scenario that doesn't include encryption to avoid embedding a clear plaintext in the payload of the first fragment; SS could use a pre-agreed (8-bit) key as an 8-bit map from the IP ID field of the first fragment to hash the (8-bit) plaintext blocks which would be embedded into its payload.

2- Encode the M (bits) plaintext as follows:

I- Divide the M (bits) plaintext message to Z blocks. Each block has a size of L_P (bits).
II- Divide each (L_P-bit) plaintext block to a set of followed J blocks, each of the followed J blocks would have a size = L_F (bits) except the last one (the J^{th} block) would have a size = L_P - [(J-1) L_F] (bits).
III- For every L_P block; present data in each block within its J blocks set as sequenced (8-bit) data units.
IV- For every L_P block; hash all the (8-bit) data units of each J block by its corresponding (8-bit) hashing key within the J hashing keys set.

3- Generate D packets stream (with a packet payload size = L_P (bits) each) including randomly Z marked packets which carry the pre-agreed mark in their IP ID as explained in [2] & [3], where D > Z. The Z marked packets would carry the M (bits) covert message, while the other (D- Z) packets would carry a normal data.

4- Fragment the D packets stream according to the pre-agreed MTU and packet size to sets of J fragments each.

5- Exclude the Z fragmented marked packets away from the D packets stream.

6- For each one of the Z fragmented marked packets; embed every block of the J encoded blocks set into the payload of its corresponding fragment within the J fragments set to generate the Z stego marked fragments' sets (regarding the sequence).

7- Embed normal data in the payloads of the (D - Z) packets' fragments (regarding the payload size of each fragment).

8- Return back the Z stego fragmented packets to their packets' positions with the (D- Z) fragmented packets to form the sequenced D fragmented packets stream, then send it to the SR.

The general framework of the M3 steganographic scenario that doesn't include the encryption option at the SR side is shown in figure 7 and it requires from Bob to do the following:

1- Generate the J (8-bit) hashing keys set as follows:

I- Calculate the J fragment offsets set as [0, L_F / 64, 2 L_F / 64, ... , (J-1) L_F / 64].
II- Select the 8 bits of the hashing key from the corresponding fragment offset according to the pre-agreed manner.

2- Receive the D fragmented packets stream of the pre-agreed packet size and MTU. Regarding that for the oversized IPv4 packets; more than one fragmentation could occur to the packet at its journey from the SS to the SR. Hence any reassembly due to smaller MTU than the pre-agreed MTU (i.e. other than the last reassembly) would occur before reaching the SR at the packet receiver. For the oversized IPv6 packets; packet fragmentation could occur by the packet source only and the reassembly could occur by the packet receiver only.

3- Reassemble the D fragmented packets stream and check for the Z stego marked packets, then exclude them away from the received D packets stream.

4- Exclude the (L_P-bit) encoded data blocks from the Z stego marked packets and use them to retrieve the M bits plaintext message as follows:

I- Divide each (L_P-bit) encoded block to a set of J blocks with a size of L_F (bits) each except the J^{th} block has a size = L_P - [(J-1) L_F] (bits).
II- For every L_P block; present data in each block within the J blocks set as sequenced (8-bit) data units.
III- For every L_P block; decode all the (8-bit) data units of certain J block by its corresponding (8-bit) hashing key within the J hashing keys set.
IV- Concatenate (in sequence) the resulted data blocks from decoding all the excluded data from all the J fragments of all the sequenced Z stego packets to get the complete M (bits) plaintext message.

5- Exclude the (L_P-bit) data block from each payload of the (D-Z) normal packets.

6- Concatenate the excluded (L_P-bit) normal data blocks by their sequence to get the complete normal data.

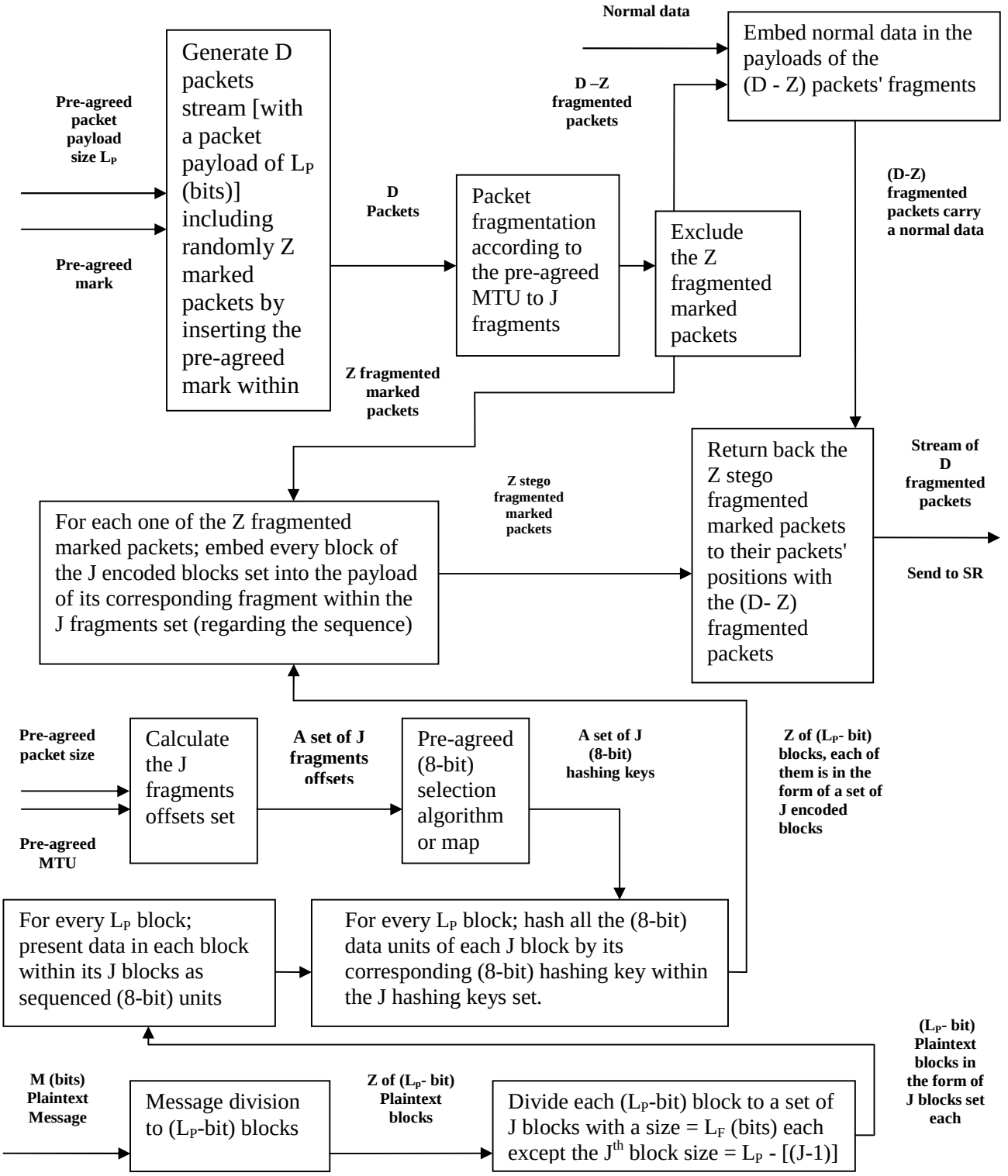


Figure (6): General framework of the M3 steganographic method (for the scenario that doesn't include the encryption option) at the steganogram sending side

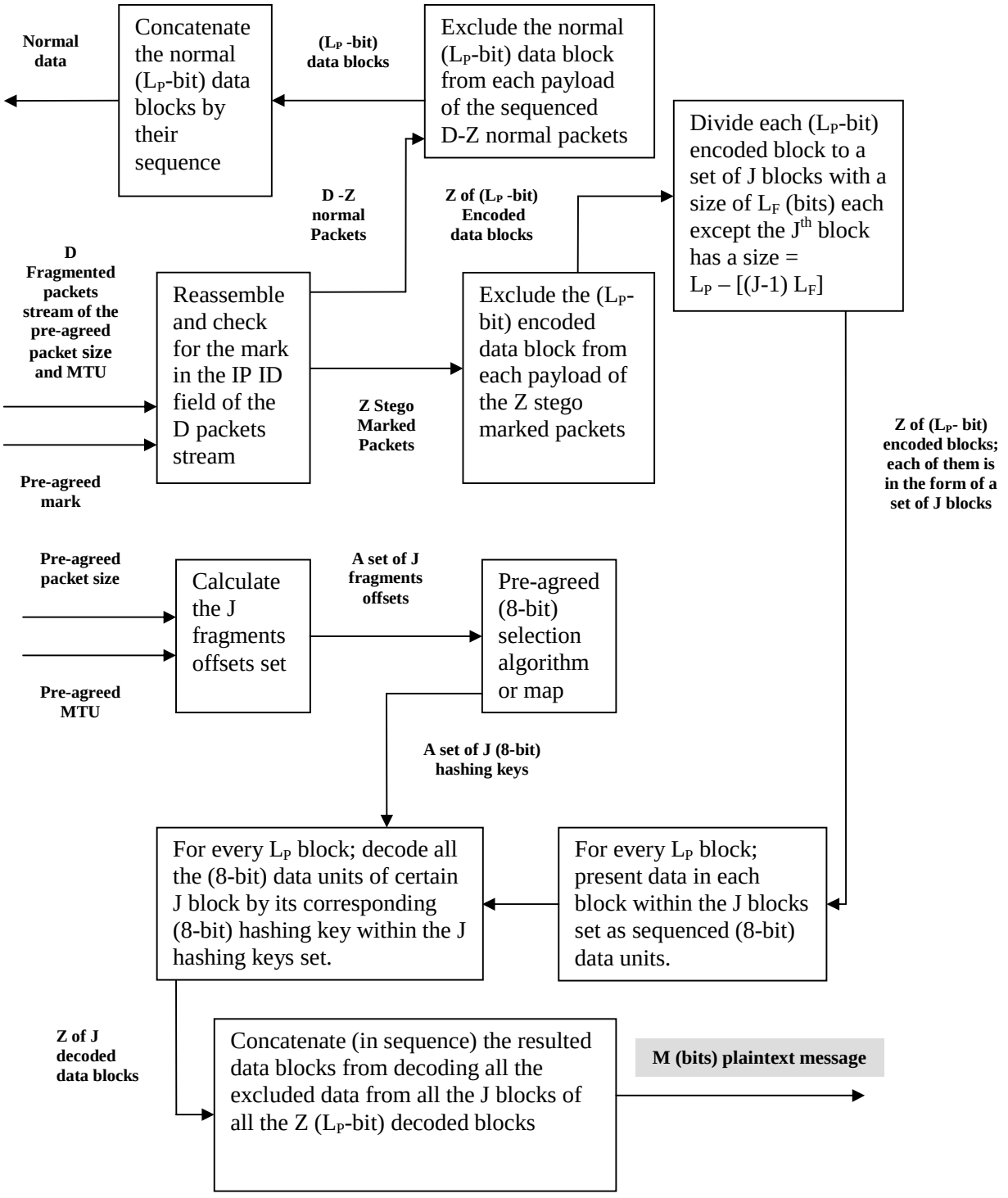


Figure (7): General framework of the M3 steganographic method (for the scenario that doesn't include the encryption option) at the steganogram receiving side

4.2 Conditions of the Used Hashing Algorithm with the M3 Steganographic Method:

The M3 steganographic method requires hashing an (8-bit) data block with a selected (8-bit) key from the fragment offset in, the conditions of the used hashing algorithm are:

1- The result must be an (8-bit) block.
2- The used hashing algorithm should be invertible (reversible) to allow the decoding process at the SR from retrieving the input (8-bit) data block.
3- The hashing function should be relatively easy to compute in order to make the hardware and software implementation practical.
4- There's no two different (8-bit) input blocks to the hash function return the same result, this is referred as the one-way property. The hash function has a strong collision resistance if it's computationally infeasible to find any two different inputs with the same output [7].

4.3 Suggested Scenarios for Selecting the Eight Bits of the Hashing Key from the Fragment Offset

Assume that the 13 bits of the fragment offset (FO) are denoted as $[f_{12},: :,f_1,f_0]$ and the 8 (bits) of the input block B to the hashing function are denoted as $[b_7, ,: : b_1 ,b_0]$. The following are some suggested scenarios for selecting the 8 bits of the hashing key from the fragment offset:

4.3.1 The First Suggested Scenario

1- Perform a respective bit logic operation such as the XOR operator to hash the first 8 bits of the fragment offset $[f_7,: :,f_1,f_0]$ with the (8-bit) input block B to get the first hashed code H' as $[h'_7,: :,h'_1,h'_0]$ so that;

$$H'_l = B_l \oplus FO_l ; \text{For } l = 0; 1;. . . ; 7$$

i.e.

$$[h'_7,: :,h'_1,h'_0] = [b_7, ,: : b_1 ,b_0] \oplus [f_7,: :,f_1,f_0]$$

The next step is optional to get a stronger hashed code.

2- Perform an XOR operation between the last 8 bits of the fragment offset $[f_{12},: :,f_6,f_5]$ and the first 8 bits of H' $[h'_7,: :,h'_1,h'_0]$ to get the hashed code H $[h_7,: :,h_1,h_0]$ which would be used in the M3 encoding process.

$$H_l = H'_l \oplus FO_l ; \text{For } l = 5; 6;. . . ; 12$$

$$[h_7,: :,h_1,h_0] = [h'_7,: :,h'_1,h'_0] \oplus [f_{12},: :,f_6,f_5]$$

4.3.2 The Second Suggested Scenario:

Use a pre-agreed key k' as a reference pointer key to point to the first selected bit of the fragment offset which is the bit number $x' \equiv k' \pmod{13} + 1$, where $1 \leq x' \leq 13$. Then select the x'^{th} bit with its 7 following bits in either the clock wise or the anti-clock wise direction according to what was previously agreed between the SS and the SR to construct the 8-bit hashing key from the fragment offset.

4.3.3 The Third Suggested Scenario

Many algorithms could be used to select the (8-bit) hashing key from the 13 bits of the fragment offset field like using any pre-defined eight bits map (that could include repeated bits) as $[f_{12}, f_{10}, f_8, f_6, f_4, f_2, f_1, f_0]$.

4.4 Evaluation of the M3 Steganographic Method:

4.4.1 Advantages of the M3 Steganographic Method:

The M3 method has the advantages of the M2 method which were mentioned in section 3.2.1. Also retrieving the plaintext message from the M3 steganographic method could be hard for a warden because hashing with a changeable (8-bit) key and optional encryption with another key may increase the message confidentiality.

4.4.2 Disadvantages of the M3 Steganographic Method:

1- Programming and storage buffers may be required in different implementation stages of the M3 method at the SS and the SR to keep data blocks and packets in sequence to avoid having missing data, data disorder or out-of-order packets problems.

2- The fragmentation/ reassembly processes affect the security of the hidden message due to using a selected (8-bit) hashing key revealed from the fragment offset. This may need some re-engineering processes.

3- The M3 steganographic method is more complicated and harder to implement than M2 because selecting the (8-bit) hashing key from the fragment offset bits, hashing and optional encryption with another key are used to encode the message which is required to be hidden.

4- The complication of processing the M3 method could show illogical behaviors at the SS and the SR which could attract the warden or the traffic analyzer attentions.

4.4.3 Hidden Communication Scenario of the M3 Steganographic Method:

In the M3 method, the packet source must be the SS and the source of the pre-agreed MTU fragmentation, and the packet receiver must be the SR and the processor of the last reassembly.

5. Using F3, M2 and M3 Steganographic Methods with IPsec:

To prevent any warden from checking or revealing data from the payload, IP ID and fragment offset fields of the IP datagram; the sending and the receiving sides could implement the IPsec encryption in tunnel mode through the network channel which communicates them when using F3, M2 and M3 steganographic methods.

5.1 IPsec Transport Mode Encryption:

IPsec encryption in transport mode encrypts only the packet payload and makes it invisible but both the IP ID and the fragment offset fields would be clear to any traffic analyzer. For the M2 steganographic method; only the embedded steganogram in the packet payload would be IPsec encrypted and unclear for wardens, while the IP ID field with the embedded mark code wouldn't be encrypted and so it would be clear for any man in the middle traffic analyzer.

The IPsec transport mode security association (SA) has been defined to not carry fragments, where the transport mode SA is always terminated at endpoints. If we assume that IPv4 fragmentation was applied after IPsec encryption (i.e. for a ciphertext packet) in the route to the destination, then IP fragments reassembly procedures at the IPsec receiver would not be able to distinguish between the pre-IPsec fragments and the fragments which were created after the IPsec processing. For IPv6; only the sender is allowed to fragment the packet. Due to the specific nature of the transport mode; the receiver would not attempt to reassemble the fragments until finishing the IPsec processing. This specification prohibits carriage of fragments on transport mode SAs for IPv6 traffic. So; transport mode SAs have been defined to not carry IPv4 nor IPv6 fragments [8].

Therefore; IPsec transport mode should not be used with the steganographic methods which require packet fragmentation as F3 and M3 methods. If the IPsec transport mode

is used with the M2 method, fragmentation shouldn't be allowed and so IPsec transport mode couldn't be used when M2 uses the IPv6 packets.

5.2 IPsec Tunnel Mode Encryption:

In the IPsec tunnel mode; any intermediate fragmentation for the IPv4 datagram after the IPsec encryption would use the outer IP header and it does not affect the inner IP header with its fragmentation data fields.

If a packet or its fragments are encapsulated by IPsec encryption in the tunnel mode, then the IP ID and the fragment offset of the inner IP header as well as the payload would be IPsec encrypted and hence they would not be clear to the man in the middle warden. If the entire IP stego datagram of the M2 or the M3 method was encrypted by the IPsec protocol in the tunnel mode, then any traffic warden monitors or captures the traffic can't read neither the IP ID field with its included mark code nor the embedded hidden data in the payload. For the M3 method; if the fragment offset which was used to generate the hashing key is that of the inner IP header, then it would not be invisible for a warden. Therefore, to provide more confidentiality with F3, M2 and M3 steganographic methods; IPsec tunnel mode encryption could be implemented between the network gateways of the SS and the SR.

6. Conclusions:

This paper suggests two new steganographic methods M2 and M3 for data hiding in the payload of a marked IP packet and a marked IP packet's fragments set respectively. The IP stego packets of both M2 and M3 methods could be marked by having a mark in their IP ID field. Both M2 and M3 methods have a high steganographic bandwidth equals the packet payload size and they may use legitimate IPv4 or IPv6 packets as hidden data.

For both M2 and M3 steganographic methods; the packet source must be the SS and the packet receiver must be the SR. In the M3 steganographic method, packet fragmentation is mandatory by the packet source. M3 is more complicated to implement than M2. It's difficult for a warden to retrieve the covert plaintext which was hidden by the M3 method, because it's difficult to know the used M3 scenario and the used hashing and encrypting keys and algorithms.

To reach more confidentiality with M2 and M3 steganographic methods; IPsec tunneling encryption could be implemented through the network channel which communicates the gateways of the SS and the SR. In this case; the entire IP datagram would be encrypted, so its payload, IP ID (that carries the mark sign) and even the

fragment offset field (which was used to generate the hashing key in the M3 method) would be IPsec encrypted to counter the traffic analysis.

References:

[1] Wojciech Mazurczyk, Krzysztof Szczypiorski, "Evaluation of Steganographic Methods for Oversized IP Packets", Telecommunication Systems: Modeling, Analysis, Design and Management, Volume 49: 3-4, ISSN: 1018-4864 (print version), ISSN: 1572-9451 (electronic version), Springer US, Journal no. 11235, March/April 2012. http://www.springerlink.com/content/20xn5j76r25002t5/
[2] Manal A. Shehab, "New Encryption and Steganographic Methods for Data Hiding in the IP Packets or Their Fragments", Master of Science in Electrical Engineering, Electrical Engineering Department, Faculty of Engineering, Alexandria University, Egypt, October 2011.
[3] Manal A. Shehab, Noha O. Korany, "New Steganographic Method for Marking IP Stego Datagrams Based on the IP ID Field", Submitted to: The 8th International Conference On Electrical Engineering ICEENG-8, May 2012.
[4] Manal A. Shehab, Noha O. Korany, "New Encryption Method Based on Using The Kharaghani Array of Order 8", Submitted to: The 8th International Conference On Electrical Engineering ICEENG-8, May 2012.
[5] Cisco Systems Inc., "Cisco VPN Services Port Adapter Configuration Guide, Chapter 5: Configuring IPsec VPN Fragmentation and MTU", Document OL-16406-01, November 2008.
www.cisco.com/en/US/docs/interfaces_modules/services_modules/vspa/configuration/guide/ivmvpnb.html
[6] Kamran Ahsan, "Covert Channel Analysis and Data Hiding in TCP/IP", Master of Science, Department of Electrical and Computer Engineering, University of Toronto 2002. http://www.springerlink.com/content/20xn5j76r25002t5/
[7] William Stallings, "Cryptography and Network Security: Principles and Practices" Third addition, Prentice Hall, ISBN 0130914290, August 2002.
[8] S. Kent & K. Seo," Security Architecture for the Internet Protocol ", IETF RFC 4301 December 2005. http://tools.ietf.org/pdf/rfc4301.pdf

Last access to web sides was in 14 March 2012.