Development of a modified steganographic model of data transmission using IPv6 protocol

Olha Suprun[1,2,*,†], Oleksandr Provotar[2,†], Oleh Suprun[2,†], Olena Nechyporuk[1,†], Viktoriia Lukashenko[1,†] and Nataliia Zhuravel[1,†]

[1] National Aviation University, Liubomyra Huzara Ave. 1, Kyiv, 03058, Ukraine
[2] Taras Shevchenko National University of Kyiv, Volodymyrska Str., 64/13, Kyiv, 01601, Ukraine

Abstract

Steganography is the method to write hidden messages into different medium in such a way that no one but the sender and the intended recipient will suspect the existence of the message. Approaches to the implementation of network steganography and existing algorithms were investigated; theirs pros and cons are described and compared. A steganographic model for embedding information when transmitting data over IPv6 based on Diffie-Hellman protocol on elliptic curves and elliptic curve digital signature algorithm is proposed. Applications have been developed to demonstrate the operation of the model and the performance of the developed and existing systems have been analyzed. The proposed model showed much faster speed of encoding and decoding of hidden messages.

Keywords

steganography, data protection, network protocol, cybersecurity

1. Introduction

Cryptography and steganography are interactive methods for confidential communication and data transfer. Using cryptography or steganography alone is not enough to protect important data. Thus, to increase the level of information protection and preserve the secrecy and confidentiality of data, both methods are used together. Cryptography can be used where steganography is ineffective, and steganography can be used where cryptography is ineffective. Both methods protect in their own way, but adding multiple layers of protection is always considered a good practice when using a combination of these methods.

Steganography is the ability to write hidden messages on a medium in such a way that no one but the sender and the intended recipient will suspect the existence of the message. In most cases, steganographic programs use sound, image and video files as a medium for hiding data. According to [1, 2], steganography can be applied in digital watermarks to protect copyright in various digital audio, video and software objects. Hiding data at the network level, such as protocols, is relatively new, but at the same time it raises an important issue of network security. All information hiding methods that can be used to share secret data in computer networks can be combined under the general term of network steganography. Different from typical steganographic methods that use digital media as a medium to hide data, network steganography uses communication protocols, control fields and their basic predefined functionality.

CH&CMiGIN'24: Third International Conference on Cyber Hygiene & Conflict Management in Global Information Networks, January 24–27, 2024, Kyiv, Ukraine

* Corresponding author.
† These authors contributed equally.
olhasuprun@knu.ua (O. Suprun); a.i.provotar@gmail.com (O. Provotar); oleh.suprun@knu.ua (O. Suprun); olena.nechyporuk@npp.nau.edu.ua (O. Nechyporuk); viktoriia.lukashenko@npp.nau.edu.ua (V. Lukashenko); zhuravel.nata83@gmail.com (N. Zhuravel)
0000-0002-1196-5655 (O. Suprun); 0000-0002-6556-3264 (O. Provotar); 0000-0002-6243-3720 (O. Suprun); 0000-0001-8203-7998 (O. Nechyporuk); 0009-0009-0458-2590 (V. Lukashenko); 0000-0001-5962-318X (N. Zhuravel)

The main purpose of our research is to improve the reliability and stability of the process of data transmission over a steganographic channel in the IPv6 protocol. The developed model must use the latest encryption and data embedding algorithms to ensure the resistance of the steganographic model to attacks on the stegosystem. Combining the newest approach of cryptography and steganography will make it possible to develop a reliable network steganographic model, with the help of which it is possible to embed and transmit information to the recipient secretly from everyone.

2. Analysis of existing models in network steganography

2.1. Classification of network steganography

Typical network steganography techniques, or network covert channels, exploit certain properties of the communication medium in such a way as to transmit secret information over the medium without attracting the attention of anyone but the actors operating the covert channel. Network steganography is synonymous with hidden channels, divided into three broad categories (Figure 1):

1. Methods of modifying the header or payload of a network packet.
2. Methods of modifying the structure of packet flows.
3. Hybrid schemes.

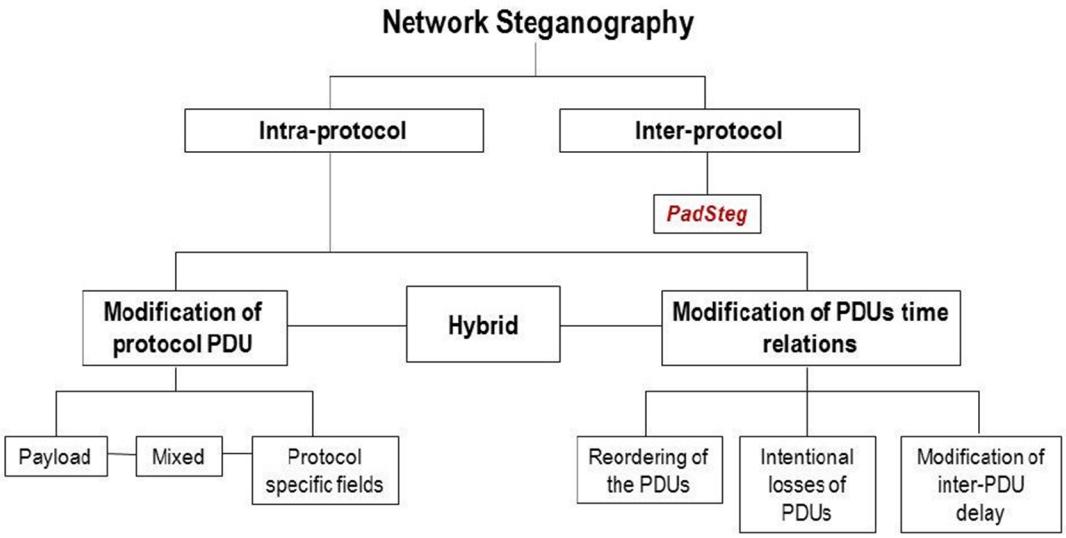

Figure 1: Example figure caption.

2.1.1. Methods of modifying the network packet header

In this method, data hiding is performed by modifying protocol-specific fields. For example, TCP, IP, or UDP headers are modified to embed secret messages, as noted in [3] and [4]. All steganographic methods according to this approach have a high steganographic capacity (the ability to hide a sufficiently large amount of data using a unit of media). Some steganographic techniques based on the application layer modify the packet payload. There is also a method that involves hiding data in both the header and the payload of the network packet, as mentioned in [3] and HICCUPS (Hidden Communication System for Corrupted Networks). This method provides high steganographic ability, but the implementation is more complicated than any other method. This requires reprogramming of the interface network cards. The disadvantages are an increase in the frequency of errors in the package.

2.1.2. Methods of modifying the structure of packet flows

Data coverage can also be accomplished by modifying network packet flows, as described in [5]. Some of the examples in this method are those that affect the sequence order of packets [6], those that change the delay between packets [7], and those that introduce intentional loss by missing sequence numbers at the sender [8]. The main problem with these schemes involves the synchronization between sender and receiver. Another disadvantage is that delays can affect the quality of the transmission.

2.1.3. Hybrid systems

In the hybrid system, the packet header and their time dependencies change. Audio packet loss steganography [4, 5] and relay steganography are some examples that fall under this scheme. Compared to other methods, this method has greater steganographic capabilities (the ability to modify the approach to hiding information due to combination). Another advantage of this method is that it is difficult to detect, that is, the method is resistant to steganalysis and attacks.

2.2. IPv6 steganography

Originally introduced in [3] and [4], IPv6 was designed to improve upon IPv4 in various areas, such as mobility, security, and addressing. However, the deployment of IPv6 is largely driven by its 128-bit address space, which allows it to recover from problems caused by the insufficient number of IPv4 addresses. Due to the slow deployment of IPv4, the two protocols are expected to co-exist for a long period, so proper transition mechanisms have been proposed. For IPv6-oriented network covert channels, references [8] and [9] show several steganographic techniques that embed data in the header or in additional extensions. To evaluate the feasibility of using IPv6 covert channels, 6 methods targeting the header shown in Figure 2 should be considered.

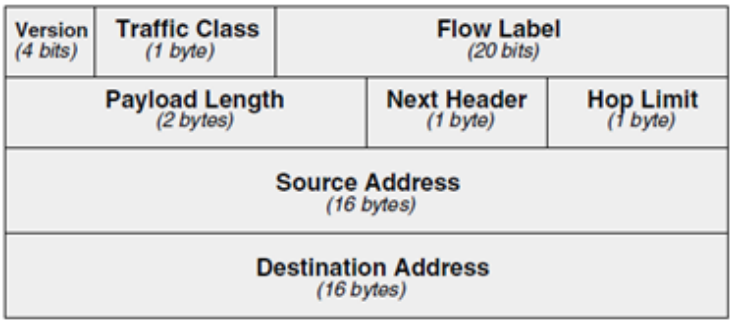

Figure 2: IPv6 header.

The fields used and the associated hiding mechanisms are described below.

- Traffic Class: This is an 8-bit field indicating the service expected from the network. The first 6 bits define the Differentiated Services Code Point (DSCP) and classify traffic according to quality criteria. The remaining 2 bits are used for Explicit Congestion Message (ECN) for end-to-end flow control. The information contained in the traffic class can be replaced with hidden data to establish a hidden channel with a bandwidth of 8 bits/packet.
- Flow Label: 20 bits long and helps network nodes to direct traffic along the most appropriate path [10]. In general, labels should be pseudo-random and future values should not be predictable. Intermediate nodes should not switch labels not to disrupt the flow.
- Payload Length: it defines the size of the data field of the datagram, which can be up to 65,536 bytes. Information can be hidden by manipulating the length of the payload to add arbitrary data to the payload. To avoid IPv6protocol misbehavior, the checksum must be properly updated to prevent packets from being dropped by intermediate nodes.

- Next Header: It defines the next header that is present in the payload of the packet. Typical values are 6 for TCP, 58 for ICMPv6, 17 for UDP, and 1 for ICMP. The information can be hidden by changing the following header to point to a "dummy" additional header containing the data. As before, an IPv6 datagram must be properly reconstructed before it is delivered to its destination.
- Hop Limit: it defines the maximum number of "hops", that is, nodes that a packet can pass. Since it is 8 bits long, the transition boundary can have up to 256 values. Data can be hidden by incrementing or decrementing the field value for successive packets.
- Source Address: contains the network address of the source. Hidden information is inserted by replacing some bits of the address with arbitrary data.

2.3. Conclusions

The network protocol stack has different layers containing different header fields for proper communication. These fields can be used as covert storage channels for secret communication. It is established that the steganogram (ie, the carrier with the embedded message) should not appear as an anomaly. For example, in the case of channels, the fields containing hidden data should not deviate too much from the average values, so as not to invalidate the hidden channel. Understanding the behavior of exposed traffic is also critical to developing appropriate detection techniques. After analyzing the possible options for using the fields of the IPv6 header, the Flow Label field was chosen as one of those that satisfies the conditions for creating a steganographic channel and does not lead to the detection of modified packets. In general, flow labels should be pseudo-random, and future values should not be predictable. Intermediate nodes should not switch labels so as not to disrupt the flow. It was found that the theoretical capacity of the steganographic channel is 20 bits/packet.

3. Design of the steganographic model of data transmission using the IPv6 protocol

After analyzing possible options for choosing a medium for embedding secret information, a part of the IPv6 header - Flow Label (flow label) was selected. The length of this field is 20 bits, which can create a hidden data channel with a bandwidth of 20 bits/packet. Also, in the general case, labels should be pseudo-random, and future values should not be predictable. This feature is suitable for creating a covert channel, since the label of the flow will contain with each packet a part of the secret message, it is not possible for a third party to predict the next value of the label. Intermediate nodes should not switch labels so as not to disrupt the flow.

3.1. Model with a chaotic coding method and RSA encryption

The approach used by Sandip Bobade, Rajeshawari Goudar, in a December 2014 publication in the International Journal of Engineering and Advanced Technology is shown in Figure 3.

A fifth-order low-overhead chaotic method algorithm with the following chaotic maps was used for coding: logistic chaotic map, improved logistic chaotic map, Chebyshev chaotic map. For encryption - RSA algorithm.

The advantages of this approach include:

- Coding speed, encryption algorithm. Much faster than the LME approach (see Figure 3).
- Resistance to attacks on the stegosystem. Due to the use of asymmetric RSA encryption, this model is more secure.

The disadvantages of this approach include:

- It is not possible to check the correct order of packets.
- The encryption data validation mechanism in the Flow Label field is not implemented.

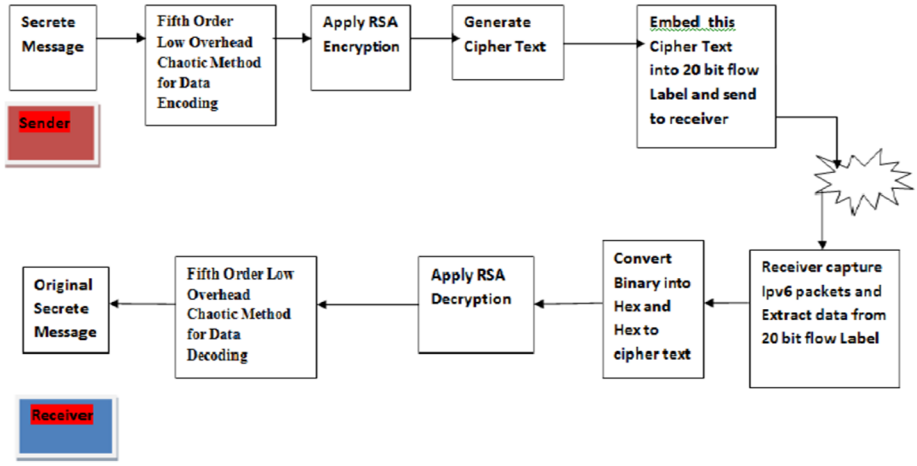

Figure 3: Steganographic model (authors Sandip Bobade, Rajeshawari Goudar).

3.2. Model with CBC-RC6 encryption algorithm and MAC authentication

The second approach used by Ra'ad A. Muhajjar, Farah A. Badr, published in the June 2017 International Journal of Engineering & Technology is shown in Figure 4.

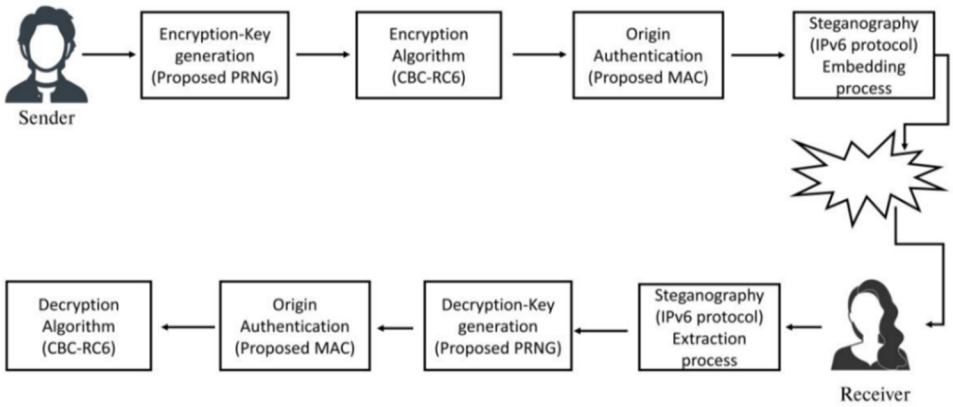

Figure 4: Steganographic model (authors Ra'ad A. Muhajjar, Farah A. Badr).

The proposed pseudo-random number generator was used to generate the key to be used in the encryption/decryption process. When the encryption/decryption key is generated, the CBC-RC6 encryption algorithm is implemented, then the proposed message authentication code is used to authenticate the source to calculate the MAC, after receiving the ciphertext and the MAC, both values are embedded in the IPv6 flow label field.

The characteristic feature and advantage of this approach is as follows: after encrypting the secret message and calculating the MAC, the message along with the MAC is hidden in the field of the flow label (Flow Label). The numbers (range of values) of stream labels vary from 1 to the hexadecimal number FFFFF. The 20 bits (ie 5 hex characters) of the field in each packet will be used as follows: when embedding data in the stream label field, the first 8 bits of the field will be used to identify the sequence of each packet, the next 8 bits will be used to hide the secret bits, and the last 4 bits will be used for MAC transmission. The bandwidth of the proposed channel will be 8 bits per packet. Although the proposal narrows the bandwidth from 20 bits per packet to 8 bits per packet, it will ensure the correct order of packets at the receiver(s).

3.3. The proposed approach to model design

Considering the shortcomings of the previous two approaches, it was proposed to build a steganographic model using the Diffie-Hellman protocol on elliptic curves for encryption (ECDH) and a public key algorithm for creating a digital signature.

Elliptic curve Diffie-Hellman is a key agreement protocol that allows two parties, each with an elliptic curve public and private key pair, to establish a shared secret over an open (secure) communication channel. This shared secret can be directly used as a key or to obtain another key. The key or derived key can be used to encrypt subsequent messages using a symmetric key cipher. It is a variant of the Diffie-Hellman protocol that uses elliptic curve cryptography.

ECDH is very similar to the classic DHKE (Diffie-Hellman Key Exchange) algorithm, but it uses ECC point multiplication instead of modular exponentiation.

It should be noted that ECDH does not provide authentication. Thus, the protocol is vulnerable to a "Man in the middle" attack. Therefore, the solution is to additionally use a digital signature algorithm such as ECDSA.

ECDSA (Elliptic Curve Digital Signature Algorithm) is a public key algorithm for creating a digital signature, a successor to the Digital Signature Algorithm (DSA). ECDSA was created when two mathematicians named Neil Koblitz and Victor S. Miller proposed the use of elliptic curves in cryptography. However, it took nearly two decades for the ECDSA algorithm to become standardized.

ECDSA is an asymmetric cryptographic algorithm built around elliptic curves and a basic function known as the "hatch function".

Advantages of ECDSA vs RSA:

- Like all asymmetric algorithms, ECDSA works in such a way that it is easy to compute in one direction but very difficult in the reverse. In the case of ECDSA, a number on the curve is multiplied by another number and therefore creates a point on the curve. Finding a new point is difficult, even if the starting point is known.
- Compared to RSA, ECDSA has been found to be more secure against modern hacking techniques due to its complexity. ECDSA provides the same level of security as RSA, but does so by using much shorter keys. Therefore, longer ECDSA keys take significantly longer to crack brute force attacks.
- Another big advantage ECDSA offers over RSA is the performance and scalability advantage. Because ECC provides optimal security with a shorter key length, it requires less network and computing power. This is great for devices with limited data storage and processing capabilities. In SSL / TLS certificates, the ECC algorithm reduces the time required to perform SSL / TLS handshakes and can help a website load faster.

As noted above, ECDSA requires much shorter key lengths to provide the same level of security as long RSA keys (Table 1).

Table 1
Comparison of Required Key Lengths for RSA and ECC

Data (In bits)	Required RSA key length	Required ECC key length
80	1024	160-223
112	2048	224-255
128	3072	256-383
192	7680	384-511
256	15360	512+

3.4. Model architecture

The proposed system is aimed at increasing the level of security by using a combination of two protection methods: cryptography and steganography. Because while steganography hides the existence of a message, cryptography encrypts the message itself. The ECDH protocol was used to encrypt the secret message. After generating the encrypted message and the initialization vector (IV), the data is signed using the ECDSA digital signature algorithm. After receiving the ciphertext, the initialization vector, the digital signature, the data is embedded in the label field of the IPv6 flow, in addition to the digital signature, which is sent in the payload of the IPv6 protocol.

At the first stage, the secret message entered by the sender is encrypted using the Diffie-Hellman protocol on elliptic curves (ECDH). As a result, we will receive an encrypted message and an initialization vector (IV), which will be transmitted together with the finished message.

In the second stage, the encrypted message and the initialization vector are processed by the ECDSA digital signature algorithm, which will allow the recipient to validate the data and be sure that the message is intact and has not been altered / damaged. As a result, we will receive a digital signature, which is transferred to the recipient according to the protocol.

At the third stage, the process of steganography takes place - the encrypted message together with the initialization vector are embedded in the Flow Label field (the process of creating a secret steganographic communication channel takes place). The stream label is 20 bits in size and ranges from 0 to 0xFFFFF. The following rules are defined, according to which the process of embedding information in the medium takes place:

1. For packets #0 – 10 (these packets transmit the initialization vector for decryption):

- The first 8 bits (1 byte) are used to indicate the sequence number of the packet. This ensures that the recipient will process the packets in the correct order.
- The last 12 bits (1.5 bytes) are used to embed the initialization vector (IV). The vector is necessary for decrypting data using the ECDH protocol.

2. For packet No. 16 (embedding the last byte of the initialization vector and beginning of embedding secret data):

- The first 8 bits (1 byte) are used to indicate the sequence number of the packet. This ensures that the recipient will process the packets in the correct order.
- The next 8 bits (1 byte) are used to embed the initialization vector (IV).
- The last 4 bits are used to embed the encrypted message.

3. For packages No. 12-N, N<256 (embedding secret data):

- The first 8 bits (1 byte) are used to indicate the sequence number of the packet. This ensures that the recipient will process the packets in the correct order.
- The last 12 bits are used to embed the encrypted message.

According to this approach, the bandwidth of the steganographic channel is 12 bits/packet. It should be noted that the initialization vector (IV) is always 16 bytes long and this size does not depend on the input data. The maximum amount of data that can be transmitted over a covert channel is 256 bytes, 16 of which are always the initialization vector, the rest are user-encrypted data.

Visually, the third stage of the model can be depicted as shown in Figure 5.

At the fourth stage, created and modified packets are sent over the network to the recipient.

At the fifth stage, packets and all values from the Flow Label of the packet header are received. It should be noted that the receiver processes incoming packets according to the rules described in the third step.

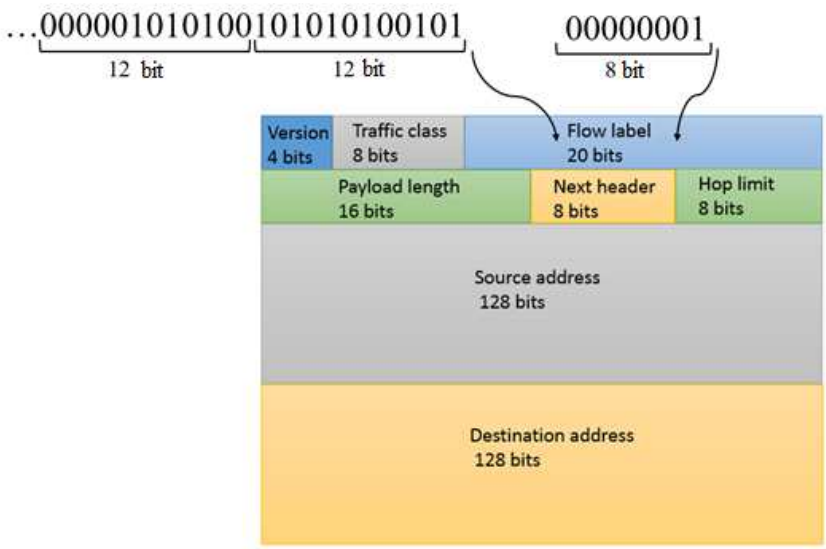

Figure 5: Embedding information in a covert channel.

At the sixth stage, there is a process of validation of the received data according to the ECDSA algorithm. If the validation is not successful, the packets are rejected and the further process is stopped. If the validation is successful, the process moves to the next stage of decryption.

At the last, seventh stage, the data decryption process takes place according to the Diffie-Hellman protocol on elliptic curves (ECDH). After successfully decrypting the data, the recipient has the initial message as a result.

The developed model has significant advantages compared to existing implementations:

- Encryption is provided by one of the most resistant and difficult to break asymmetric encryption methods, namely the Diffie-Hellman protocol on elliptic curves (ECDH) as opposed to RSA and CBC-RC6 in other models.
- The latest ECDSA digital signature method is provided to ensure the integrity of the data and verify the correctness and integrity of the received data by the recipient. Only one of the models discussed above was checked for integrity Increased bandwidth of the steganographic channel - up to 12 bits/packet.
- The process of transferring data to the recipient is much faster and the number of required carriers is reduced (packets).

4. Software implementation of the model and analysis of the results

4.1. Brief description of the used technologies

WPF technology from Microsoft based on .NET 5 was used to create a software product that implements a locally developed steganographic model. WPF, which stands for Windows Presentation Foundation, is a development platform and subsystem of .NET 5. WPF is used to create Windows client applications that run on the Windows operating system. WPF uses XAML as the interface language and C# as the programming language. WPF was introduced as part of NET Framework 3.0 as a Windows library for creating Windows client applications and the next generation of Windows Forms. The current version of WPF is 5.0.

WPF is a mechanism responsible for creating, displaying and managing user interfaces, documents, images, movies and media in Windows 7 and later operating systems. WPF is a set of libraries that have all the features you need to create, run, run, and manage Windows client applications.

XAML is a new descriptive programming language developed by Microsoft for writing user interfaces for next-generation managed applications. XAML is used to create user interfaces for

Windows and mobile applications that use Windows Presentation Foundation (WPF), UWP, and Xamarin forms.

The purpose of XAML is simple - to create user interfaces using a markup language that looks like XML. XAML uses the XML format for elements and attributes. Each element in XAML represents an object that is an instance of a type. The scope of a type (class, enumeration, etc.) is determined by the namespace that is physically located in the assembly (DLL) of the library. NET.

WinPcap technology is used to physically create and send packets between hosts.

WinPcap is a standard tool that provides access to connections between network layers (connection and selection between two host systems) in Windows environments. It allows the capture and forwarding of network packets that bypass the protocol stack, including kernel-level packet filtering, a network statistics engine, and support for remote packet capture.

WinPcap has a driver that extends the operating system to provide low-level network access. It also has a library that provides easy access to low-level networking layers. This library has a Windows version of the popular UNIX libpcap API. It should be noted that WinPcap is a packet capture and filtering mechanism for many open source tools and commercial networks. Some of these tools, such as Wireshark, Nmap or Snort, are widely used in network management.

Accordingly, in order to use this technology in the .NET environment, the source code of the Pcap.Net library [11] was taken and modified for use in the developed steganographic model. Pcap.Net is a .NET wrapper for WinPcap written in C++/CLI and C# that includes almost all WinPcap features and includes a packet interpretation framework.

4.2. Software system architecture

Two desktop applications have been developed:

- Program for sending packages (sender's side).
- Program for receiving packages (recipient side).

It should be noted that to simulate the recipient's side, a developed program - a sniffer for receiving all sent packets - was connected to the local host. A packet analyzer, or packet sniffer, is software or hardware that can intercept and track packets as they travel over a network. In this way, IT professionals and cybercriminals can effectively inspect the contents of files and messages transmitted to, from, or within the network [12, 13].

Packet sniffers can be used in two modes: filtered and unfiltered. Filtered packet reading means that the analyzers will look for certain data and will capture or copy only those packets that contain that data. Reading unfiltered packets means that all packets are captured and/or copied, regardless of the data they contain. They may also collect a wide range of information, including what websites a particular user visits, what they browse, the destinations and content of any emails or messages they send, and any files that they download

The sniffer was developed as a console application on the .NET 5 platform using the Pcap.Net library and WinPcap technology. This software has the ability to listen to all network interfaces that exist on the local machine.

For example, a virtual network adapter vEthernet (Default switch) was selected, through which packets were sent to the default gateway. The sniffer listened and received packets passing through the vEthernet (Default switch) - Default Gateway path.

In general, the following software components were implemented:

- Subsystem for encryption and decryption according to the EDCH protocol.
- A subsystem for creating a digital signature and its validation according to the ECDSA.
- WPF application as a packet sender.
- A WPF application as a receiver of packets (together with a sniffer).

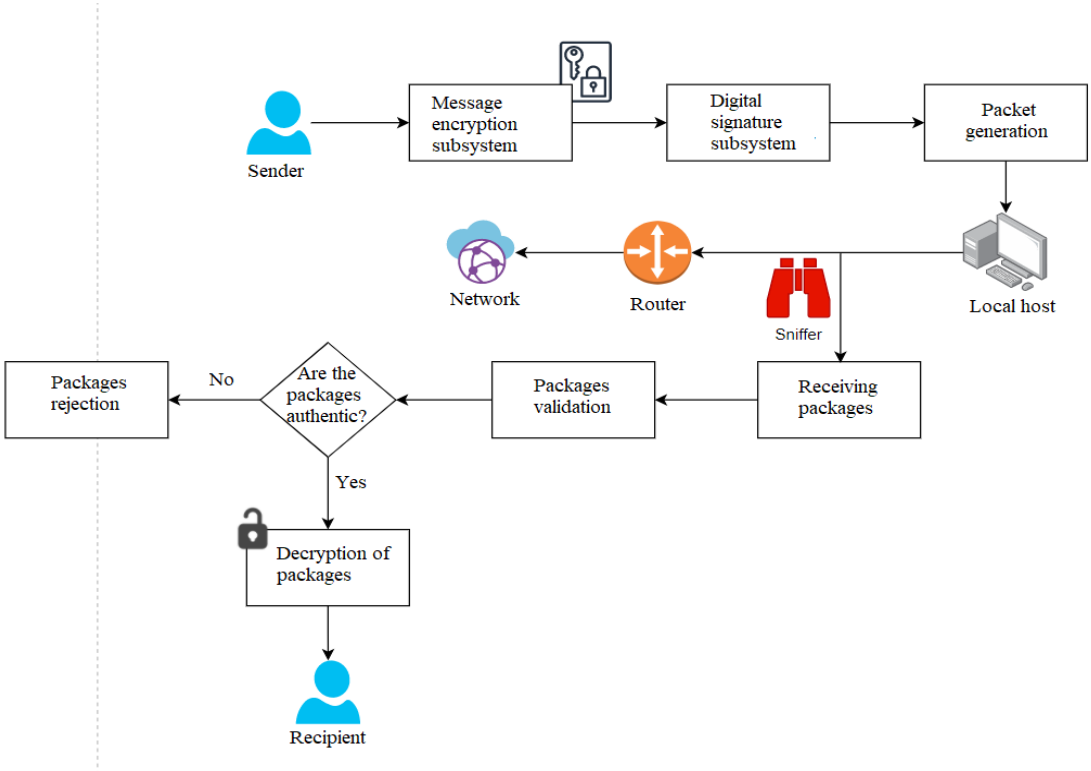

Figure 6: The architecture of the implemented steganographic model.

4.3. Analysis of the obtained results

Research of the stages of encryption, application of a digital signature and validation of a digital signature, and decryption was conducted.

The speed of execution of these stages was compared with the previous implementation of this model. The research was conducted on a computer with the following technical characteristics: Intel Core i7-8565U, 16 GB RAM, 1 TB SSD. For each of the test cases, the average value was obtained from a sample of 100 tests. The following results were obtained as shown in Table 2 and Table 3.

Table 2
Comparison of the Execution Speed on the Sender's Side

№	Size of input data (bytes)	The proposed model (s)	Preliminary implementation(s)	Percentage change (+%)
1	1	0.6131	1.3738	55.4
2	10	0.7850	1.4625	46.3
3	20	0.8583	1.6025	46.4
4	30	0.9515	1.6133	41.0
5	40	0.9881	1.6624	40.6
6	50	1.0977	1.9955	45.0
7	100	1.3016	2.4513	46.9
8	150	1.3501	2.9941	54.9
9	200	1.5104	3.2851	54.0
10	250	1.6238	3.8189	57.5

After analyzing the obtained data, it can be concluded that at the stage of the sender, the proposed model works on average 41.1% faster than the previous implementation.

Table 3
Comparison of the execution speed on the receiver's side

№	Size of input data (bytes)	The proposed model (s)	Preliminary implementation(s)	Percentage change (+%)
1	1	0.6208	1.1523	46.1
2	10	0.7042	1.2288	42.7
3	20	0.816	1.2978	37.1
4	30	0.8506	1.3201	35.6
5	40	0.8874	1.3965	36.5
6	50	0.9239	1.5852	41.7
7	100	1.0672	1.8229	41.5
8	150	1.1092	1.9079	41.9
9	200	1.1735	2.0841	43.7
10	250	1.3055	2.3428	44.3

5. Conclusions

The existing approaches to the construction of a steganographic model of data transmission over a hidden channel in the IPv6 protocol were analyzed and it was concluded that among the proposed ones there is none that would offer simultaneous processing of data with stable algorithms and verification of the integrity of packets on the recipient side.

Developed WPF applications (for the Windows operating system) on the .NET 5 platform to demonstrate the operation of the steganographic model using the C# 9.0 programming language and the XAML markup language.

It is established that the proposed approach significantly increases the reliability and stability of the model due to the use of the Diffie-Hellman protocol on elliptic curves (ECDH) for encryption and the ECDSA digital signature method. The proposed model provides verification of the integrity and intactness of packages on the recipient's side. Only in one of the previously considered models was a check of data integrity (with the help of MAC - message authentication code).

The bandwidth of the steganographic channel has been increased - up to 12 bits/packet. This made it possible to significantly speed up the process of transferring data to the recipient and reduce the number of necessary media (packages).

It was proven that the use of the ECDH protocol together with the ECDSA algorithm significantly increases the performance of the system in comparison with the previous existing implementation of the model using the fifth-order chaotic method and the asymmetric RSA encryption algorithm. On the sender's side, the speed of processing packets increased by 48.8% on average, while on the recipient's side - by 41.1%, respectively. The use of this model increases the security of data transmission over a hidden channel and enables the recipient to validate packets. Thus, this model can be used to provide secure covert communication in real network systems.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] R. A. Muhajjar, F, Badr, Secure data communications using cryptography and IPv6 steganography, International Journal of Engineering & Technology 7 (4.19) (2018) 624–628.

[2] S. Bobade, R. Goudar, Secure data communication using protocol steganography in IPv6, International Journal of Engineering & Advanced Technology 4(2) (2014) 104–109.

[3] W. Mazurczyk, K. Powójski, L. Caviglione, IPv6 covert channels in the wild, in: Proceedings of the Third Central European Cybersecurity Conference (CECC '19), Munich, Germany, pp. 1–6. doi: 10.1145/3360664.3360674.
[4] S. Fei, C. Zhe, A method for low-overhead secure network coding, Appl. Math. Inf. Sci. 7(5) (2013) 1699–1703.
[5] K. Kurin, O. Yudin, O. Suprun, O. Suprun, O. Provotar, O. Yudin, Visual data coding algorithms for the problem of steganographic information protection, in: Proceedings of 4th International Conference on Advanced Trends in Information Theory (ATIT), IEEE, Kyiv, Ukraine, 2022, pp. 290–294, doi: 10.1109/ATIT58178.2022.10024189.
[6] E. Cauich, R. G. Cárdenas, R. Watanabe, Data hiding in identification and offset IP fields, In: F.F. Ramos, V. Larios Rosillo, H. Unger (Eds.), Advanced Distributed Systems. ISSADS 2005, volume 3563 of Lecture Notes in Computer Science, Springer, Berlin, 2005, pp. 118–125. doi: 10.1007/11533962_11.
[7] M. Ivasenko, O. Suprun, O. Suprun, Information transmission protection using linguistic steganography with arithmetic encoding and decoding approach, in: Proceedings of IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT), IEEE, Kyiv, Ukraine, 2021, pp. 174–178. doi: 10.1109/ATIT54053.2021.9678855.
[8] S. Gianvecchio, H. Wang, Detecting covert timing channels: An entropy-based approach, in: Proceedings of the 14th ACM conference on Computer and communications security (CCS'07), Alexandria, Virginia, USA, 2007, pp. 307–316. doi: 10.1145/1315245.1315284.
[9] N. Singh, J. Bhardwaj, G. Raghav, Network steganography and its techniques: A survey, International Journal of Computer Applications 174(2) (2017) 8–14.
[10] S. Popereshnyak, O. Suprun, O. Suprun, T. Wieckowski, Intrusion detection method based on the sensory traps system, in: Proceedings of XIV-th International Conference on Perspective Technologies and Methods in MEMS Design (MEMSTECH), IEEE, Lviv, Ukraine, 2018, pp. 122–126. doi: 10.1109/MEMSTECH.2018.8365716.
[11] K. Szczypiorski, Steganography in TCP/IP networks.state of the artand a proposal of a new system – HICCUPS, in: Proceedings of Institute of Telecommunications' seminar, Warsaw University of Technology, Poland, 2003. URL: http://krzysiek.tele.pw.edu.pl/pdf/steg-seminar2003.pdf.
[12] M. Zaliskyi, R. Odarchenko, S. Gnatyuk, Y. Petrova, A. Chaplits, Method of traffic monitoring for DDoS attacks detection in e-health systems and networks, CEUR Workshop Proceedings 2255 (2018) 193–204. URL: https://ceur-ws.org/Vol-2255/paper18.pdf.
[13] J.S. Al-Azzeh, M. Al Hadidi, R.S. Odarchenko, S. Gnatyuk, Z. Shevchuk, Z. Hu, Analysis of self-similar traffic models in computer networks, International Review on Modelling and Simulations 10(5) (2017) 328–336. doi: 10.15866/iremos.v10i5.12009.