PROJECT REPORT

Department of Optical and Electro-Optical Instrumentation

Instruments Research & Development Establishment, Dehradun

Submitted To:

Mr. Vaibhav Gupta

Submitted By:

Vaibhav Kumar Kapriyal and
Arundhati Shukla

DECLARATION

I, **Vaibhav Kumar Kapriyal**, student of B. Tech CSE VIIth Semester, Graphic Era Hill University, Dehradun, and **Arundhati Shukla**, student of B. Tech CSE VIth Semesters Institute of Science and Technology declare that the technical project work based on Cybersecurity and Image Processing has been carried out by us and submitting partial fulfilment of the course requirements for the award of degree in **Bachelor of Technology**. The matter embodied in this synopsis has not been submitted to any other university or institution for the award of any other degree or diploma.

Place: Dehradun

Date: 10 September 2023

ACKNOWLEDGMENT

We would like to express our profound gratitude to Department of Optical and Electro-Optical Instrumentation Instruments Research & Development Establishment, Dehradun for their contributions to the completion of our project titled.

We would like to express our special thanks to our mentor **Mr. Vaibhav Gupta** for his time and efforts he provided throughout the project. Your useful advice and suggestions were really helpful to us during the project's completion. In this aspect, we are eternally grateful to you.

At last, but not the least we greatly indebted to all other persons who directly or indirectly helped us during this course.

Vaibhav Kumar Kapriyal and
Arundhati Shukla

TABLE OF CONTENTS

Problem Statement

To develop a secure wireless transmission system to convert analog video signals from a thermal camera, encrypt the data, and transmit it wirelessly over a long-range using an optical sensor. The system must ensure real-time and reliable signal transmission for effective monitoring and surveillance, while ensuring the confidentiality and integrity of the video data through strong encryption measures.

Currently, surveillance operations heavily rely on wired transmission methods to transmit video signals from cameras to monitoring stations.

Limitations of current system:

- Limited Flexibility
- Higher Installation Costs
- Maintenance Challenges
- Vulnerability to Physical Damage
- Limited Range
- Signal Interference

Considering these disadvantages, the development of a secure and efficient wireless transmission system can address these challenges and provide a more flexible, scalable, and reliable solution for surveillance applications.

INTRODUCTION TO VARIOUS ASPECTS OF THE PROJECT

Different Mode of Communication between Devices:

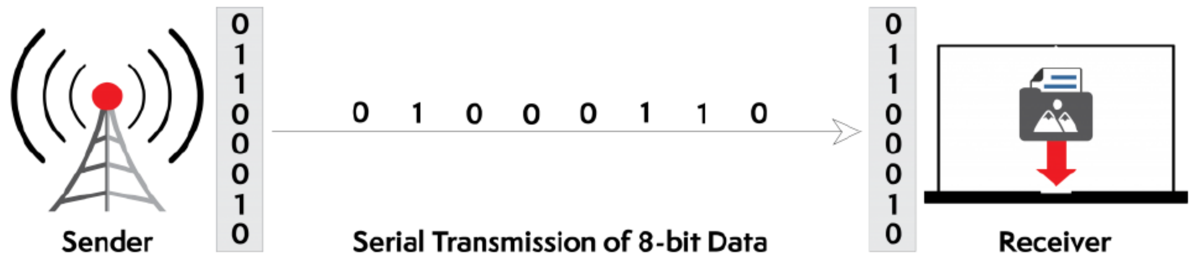
Simplex: The Simplex transmission mode is used in computing networks when there is a single or one-way flow of information from sender to receiver. In this mode of transmission, communication occurs only in one direction, i.e., the circuit is configured so that it is either transmit only or receive only.

Half Duplex: The half-duplex mode of transmission is used in computer networks when there is a way to flow information from sender to receiver but only one at a time. In this mode, the connected devices can transmit or receive the data but not simultaneously.

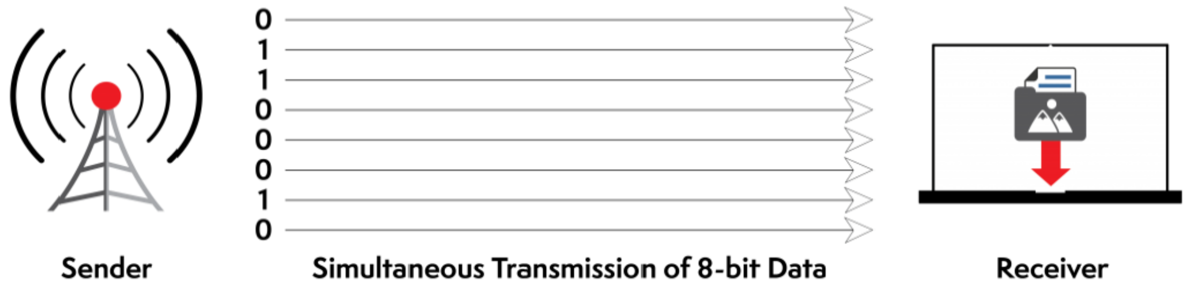
Full Duplex: The Full Duplex mode of transmission is used in computing networks when there is simultaneous information flow in both directions, from sender to receiver. In this mode of transmission, the channel capacity is shared between the two devices, and communication occurs in both directions across a communication link that requires two wires.

Data Transmission – Parallel vs Serial

Serial Transmission: When data is sent or received using serial data transmission, the data bits are organized in a specific order, since they can only be sent one after another. The order of the data bits is important as it dictates how the transmission is organized when it is received. It is viewed as a reliable data transmission method because a data bit is only sent if the previous data bit has already been received.



Parallel Transmission: When data is sent using parallel data transmission, multiple data bits are transmitted over multiple channels at the same time. This means that data can be sent much faster than using serial transmission methods.

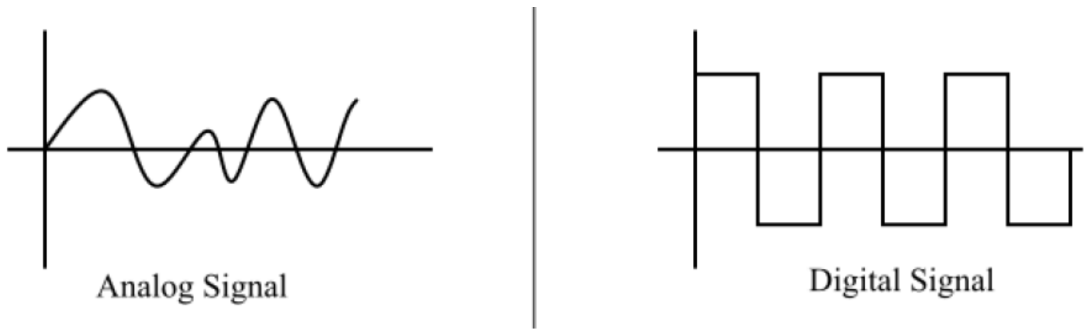


Difference between Analog and Digital Signal

Analog signal: A signal which is a continuous function of time and used to carry the information is known as an analog signal. An analog signal represents a quantity analogous to another quantity, for example, in case of an analog audio signal, the instantaneous value of signal voltage represents the pressure of the sound wave.
Analog signals utilize the properties of medium to convey the information. All the natural signals are the examples of analog signals. However, the analog signals are more susceptible to the electronic noise and distortion which can degrade the quality of the signal.

Digital Signal: A signal that is discrete function of time, i.e. which is not a continuous signal, is known as a digital signal. The digital signals are represented in the binary form and consist of different values of voltage at discrete instants of time.

Basically, a digital signal represents the data and information as a sequence of separate values at any given time. The digital signal can only take on one of a finite number of values.



What is bandwidth?

Bandwidth is the total range of frequency required to pass a specific signal that has been modulated to carry data without distortion or loss of data. The ideal bandwidth allows the signal to pass under conditions of maximum AM or FM adjustment. (Too narrow a bandwidth will result in loss of data. Too wide a bandwidth will pass excessive noise.)

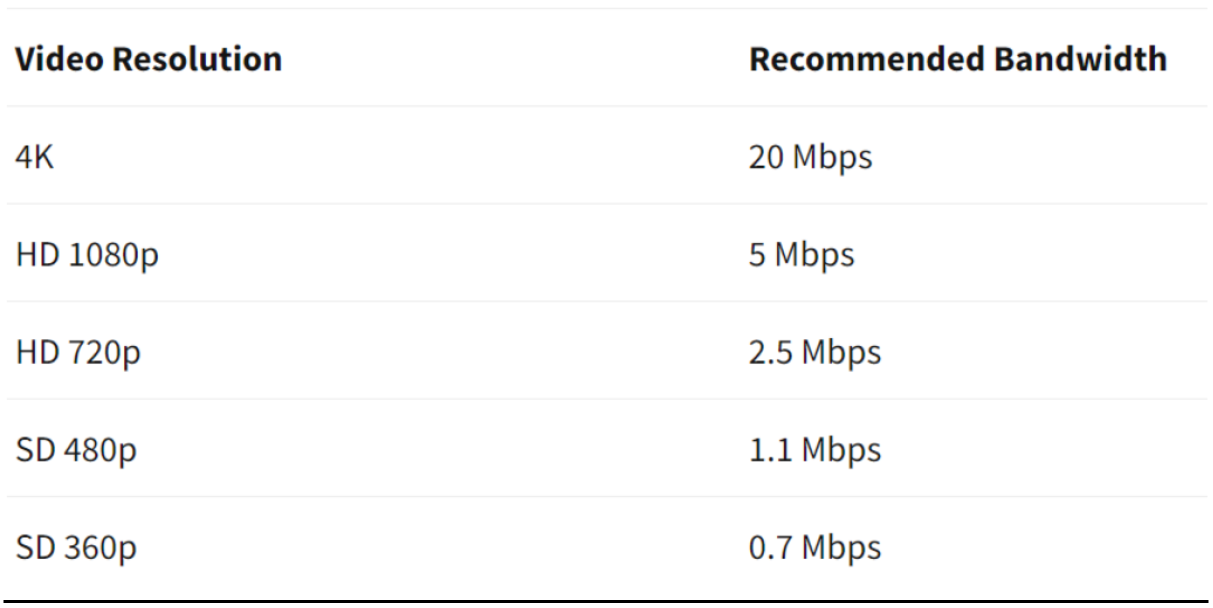
Transmitters and receivers have bandwidths. The "wider" the receiver's bandwidth is, the more information it can receive on different frequencies.

The term bandwidth is used metaphorically for the carrying ability of Internet carriers. For example, if you can receive information from the Internet over a slow modem, you get less information per second than if you were connected to a fast modem. Thus, you have "low bandwidth" and the Internet appears slower to you.

In the context of video streaming, the amount of information or data per unit of time that a transmission medium like an internet connection can handle is Video Bandwidth. In simple words, Bandwidth is the maximum data-carrying capacity of a transmission medium and measured in bits per second(bps), Megabits per second(Mbps) or Gigabits per second(Gbps). So, for example, if the bandwidth is 50 Mbps, a maximum of 50 Mb data can be transferred per second on the channel.

High bandwidth capabilities mean you will be able to achieve higher data transfer speed, download and upload files faster and stream HD content. In addition, having a higher bandwidth has several advantages like allowing more simultaneous visitors to your website, faster application performance and support for multiple concurrent sessions.

Minimum Bandwidth required



Video Resolution	Recommended Bandwidth
4K	20 Mbps
HD 1080p	5 Mbps
HD 720p	2.5 Mbps
SD 480p	1.1 Mbps
SD 360p	0.7 Mbps

Requirements

Hardware Requirements:

1.Thermal Camera (video Source)

2. BNC Cable

3. AV-to-HDMI Converter with BNC Inputs

4. Transmitter

5. Antenna

6. Receiver

7. Display Device

8. Power Supply

All components should be Military Standard components.

MIL GRADE Components(MIL-STD)

These components are those components that have to be of the highest quality, capable of performing under the toughest of conditions, and able to easily achieve interoperability with other parts. To reach these goals, the Department of Defence has created very specific standards known as MIL-STD.

WHAT IS MIL-STD?

Although engineers might throw around terms like MIL-DTL, or MIL-SPEC, MIL-STD is the official designation used to achieve standardization objections set by the DoD.

These standards cover the end characteristics of the product, as well as the processes, materials and security standards to be followed in their production. The five types of defense standards include:

- One: Interface standards
- Two: Design criteria standards
- Three: Manufacturing process standards
- Four: Standard practices
- Five: Test method standards

These defense standards are meant to ensure proper performance and logistical usefulness of military equipment. Although their origins can be traced back for decades, the pressing need for standardization became most apparent during WWII.

As electronics and PCB capabilities rapidly emerged and evolved, the standards had to quickly update to keep up. In 2011, the Defense Department instituted the Supply Chain Hardware Integrity for Electronics Defense (SHIELD) program which calls for improved measures to safeguard against tampering or component removal.

Knowing which standards are relevant enable ECMs to meet or exceed military grade electronic components quality expectations.

Although constantly changing to keep up with the latest PCB innovations, some standards from the Defense Standardization Program that are particularly applicable to PCB components include:

- MIL-STD-883: Test method standards for microcircuits
- MIL-STD-750-2: Test methods for semiconductor devices
- MIL-STD-202G: Test methods for standard electronic and electrical component parts

In general, the best mil-grade PCBs conform to specific parameters regarding dielectric thickness, annular ring for inner and outer layer, and drill to copper clearance.

The interface specifications for the BNC and many other connectors are referenced in MIL-STD-348.

THERMAL CAMERA

A thermal camera is a specialized device that detects and captures images based on the heat emitted by objects or living beings, enabling visibility in low-light conditions and finding temperature differences in various applications, such as security, industrial inspections, and medical diagnostics.

Properties :

- Night Vision
- Unaffected by Lighting Conditions
- Detection of Intruders
- Accurate Temperature Measurement
- Wide Area Coverage

BNC Connector

The BNC connector (initialism of "Bayonet Neill–Concelman") is a miniature quick connect/disconnect radio frequency connector used for coaxial cable. It is designed to maintain the same characteristic impedance of the cable, with 50 ohm and 75-ohm types being made. It is usually applied for video and radio frequency connections up to about 2 GHz and up to 500 volts. The connector has a twist to lock design with two lugs in the female portion of the connector engaging a slot in the shell of the male portion. The type was introduced on military radio equipment in the 1940s and has since become widely applied in radio systems, and is a common type of video connector. Similar radio-frequency connectors differ in dimensions and attachment features, and may allow for higher voltages, higher frequencies, or three-wire connections.

The interface specifications for the BNC and many other connectors are referenced in MIL-STD-348.

Uses: The BNC was originally designed for military use and has gained wide acceptance in video and RF applications to 2 GHz. BNC connectors are used with miniature-to-subminiature coaxial cable in radio, television, and other radio-frequency electronic equipment. They were commonly used for early computer networks, including ARC net, the IBM PC Network, and the 10BASE2 variant of Ethernet.

The BNC connector is used for signal connections such as:

- analog and serial digital interface video signals
- radio antennas
- aerospace electronics (avionics)
- nuclear instrumentation
- test equipment.

The BNC connector is used for analog composite video and digital video interconnects on commercial video devices.

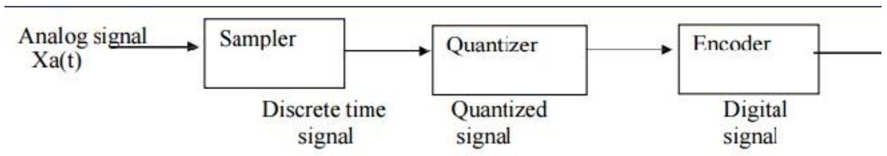
Type: BNC connectors are most commonly made in 50- and 75-ohm versions, matched for use with cables of the same characteristic impedance. The 50-ohm connectors are typically specified for use at frequencies up to 4 GHz and the 75-ohm version up to 2 GHz. Video (particularly HD video signals) and DS3 Telco central office applications primarily use 75-ohm BNC connectors, whereas 50-ohm connectors are used for data and RF. Many VHF receivers used 75-ohm antenna inputs, so they often used 75-ohm BNC connectors.

Analog to Digital Converter

In electronics, an analog-to-digital converter (ADC, A/D, or A-to-D) is a system that converts an analog signal, such as a sound picked up by a microphone or light entering a digital camera, into a digital signal. An ADC may also provide an isolated measurement such as an electronic device that converts an analog input voltage or current to a digital number representing the magnitude of the voltage or current. Typically, the digital output is a two's complement binary number that is proportional to the input, but there are other possibilities.

There are several ADC architectures. Due to the complexity and the need for precisely matched components, all but the most specialized ADCs are implemented as integrated circuits (ICs). These typically take the form of metal–oxide–semiconductor (MOS) mixed-signal integrated circuit chips that integrate both analog and digital circuits.

A digital-to-analog converter (DAC) performs the reverse function; it converts a digital signal into an analog signal.



TRANSMITTER

A transmitter is an electronic device or system that is responsible for sending or transmitting signals, data, or information over a communication channel or medium to a receiver.

Microwave transmitters operate at high frequencies (above 1 GHz) and are commonly used for long-range point-to-point video communication between fixed locations or relay stations, providing high data rates and low latency.

BANDWIDTH

Bandwidth is the total range of frequency required to pass a specific signal that has been modulated to carry data without distortion or loss of data. The ideal bandwidth allows the signal to pass under conditions of maximum AM or FM adjustment.

The bandwidth required for thermal camera video transmission can vary widely. As an example, let’s estimate the bandwidth for a thermal camera with the following specifications:

- Resolution: 640x480 pixels
- Frame Rate: 30 fps
- Color Depth: 8-bit
- Compression: H.264

Bandwidth required ≈ 73.73 Mbps

Software Requirement:

- Latest Version of Python installed
- Have pycharm installed on your computer
- We will need this library to develop a GUI(Graphical User Interface) to make and operate on the system that is user friendly and the user can use without any prior knowledge and easy to understand.

Device Specifications:

- Processor: 11th Gen Intel(R) Core(TM) i5-1135G7 @ 2.40GHz 2.42 GHz
- Installed RAM: 16.0 GB (15.8 GB usable)
- System type: 64-bit operating system, x64-based processor

Window Specifications:

- Edition Windows 10 Home Single Language
- Version 20H2
- OS build 19042.1469
- Experience Windows Feature Experience Pack 120.2212.3920.0

Libraries Used

- OpenCV(pip install cv2)
- NumPy(pip install numpy)
- Random(pip install random)
- Threading(pip install threading)
- OS(pip install os)

Methodology and Algorithm Used

WIRELESS VIDEO TRANSMISSION PROCESS:

- **VIDEO SOURCE:** The process begins with the video source, such as a thermal camera, which captures the video feed that needs to be transmitted.

- **ANALOG TODIGITALCONVERSION:** Analog to Digital Conversion i.e., video source captured via thermal camera is converted into a digital signal for transmission.

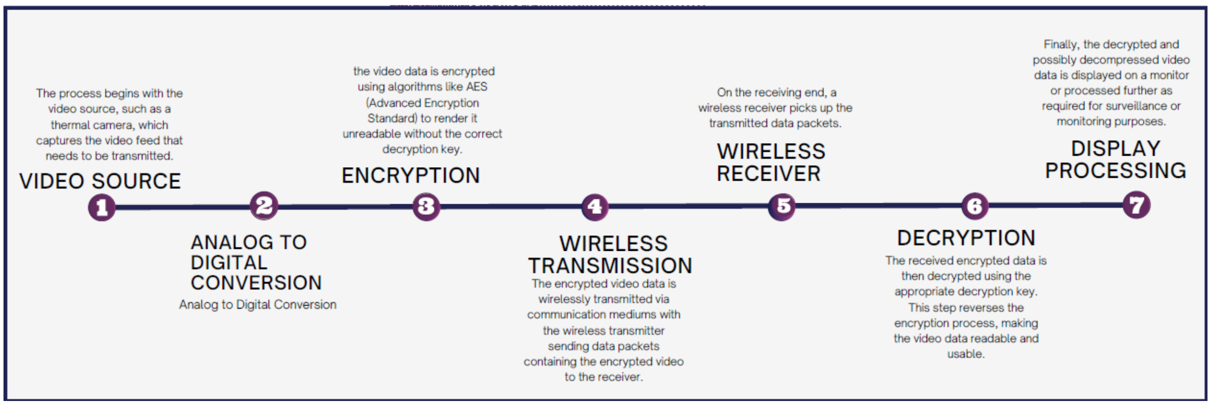
- **ENCRYPTION:** the video data is encrypted using algorithms like AES(Advanced Encryption Standard) to render it unreadable without the correct decryption key.

- **WIRELESS TRANSMISSION:** The encrypted video data is wirelessly transmitted via communication mediums with the wireless transmitter sending data packets containing the encrypted video to the receiver.

- **WIRELESS RECEIVER:** On the receiving end, a wireless receiver picks up the transmitted data packets.

- **DECRYPTION:** The received encrypted data is then decrypted using the appropriate decryption key. This step reverses the encryption process, making the video data readable and usable.

- **DISPLAY PROCESSING:** Finally, the decrypted and possibly decompressed video data is displayed on a monitor or processed further as required for surveillance or monitoring purposes.


The process begins with the video source, such as a thermal camera, which captures the video feed that needs to be transmitted.
VIDEO SOURCE
1
2
ANALOG TO DIGITAL CONVERSION
Analog to Digital Conversion
the video data is encrypted using algorithms like AES (Advanced Encryption Standard) to render it unreadable without the correct decryption key.
ENCRYPTION
3
4
WIRELESS TRANSMISSION
The encrypted video data is wirelessly transmitted via communication mediums with the wireless transmitter sending data packets containing the encrypted video to the receiver.
On the receiving end, a wireless receiver picks up the transmitted data packets.
WIRELESS RECEIVER
5
6
DECRYPTION
The received encrypted data is then decrypted using the appropriate decryption key. This step reverses the encryption process, making the video data readable and usable.
Finally, the decrypted and possibly decompressed video data is displayed on a monitor or processed further as required for surveillance or monitoring purposes.
DISPLAY PROCESSING
7

<u>HOW TO SECURE DATA DURING TRANSMISSION?</u>

ENCRYPTION/DECRYPTION: Encryption is the process of converting data into a coded form that can only be read or understood by authorized parties with the appropriate decryption key, while decryption is the reverse process of converting the encrypted data back into its original, readable form.

<u>VARIOUS ENCRYPTION PROTOCOLS</u>

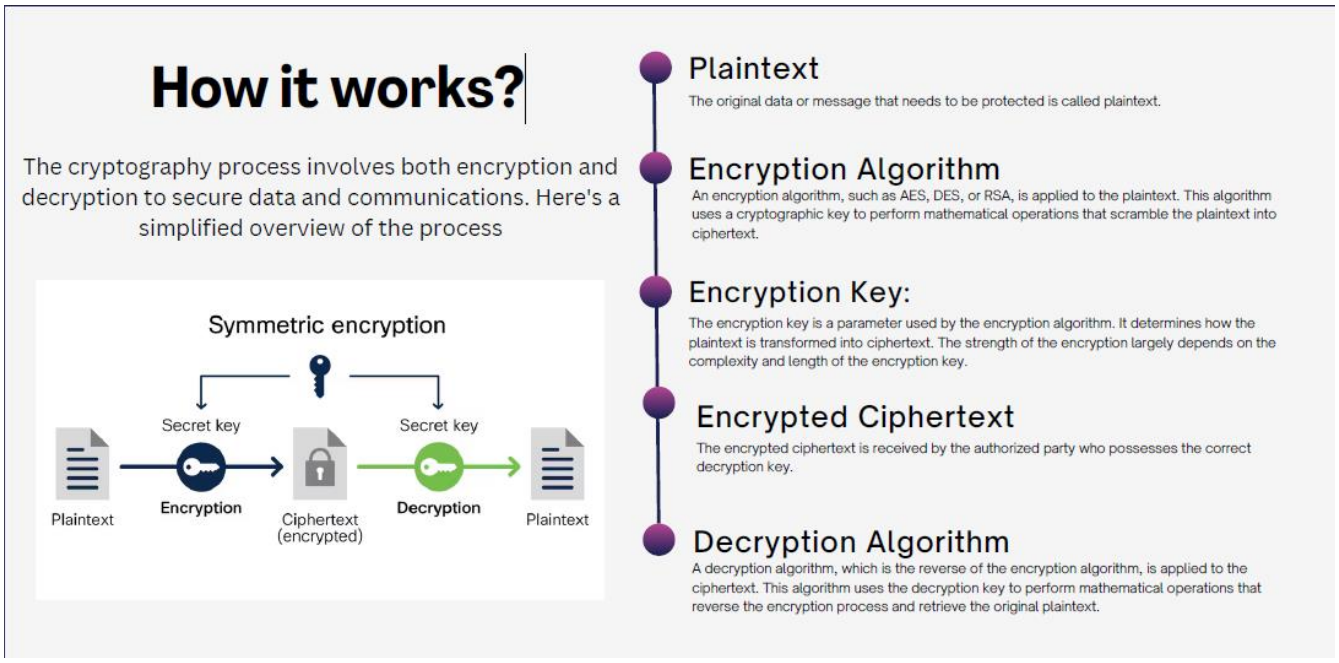
- **AES (Advanced Encryption Standard):** AES is a widely used symmetric encryption algorithm that ensures data confidentiality. It is commonly used in various applications, such as securing data at rest on storage devices and encrypting sensitive data during transmission.

- **RSA (Rivest–Shamir–Adleman):** RSA is a widely used asymmetric encryption algorithm used for secure key exchange and digital signatures. It allows secure communication without the need for sharing a secret key and is commonly used in secure online transactions and certificate-based authentication.

- **Triple DES (Data Encryption Standard):** Triple DES is a symmetric encryption algorithm that applies the DES encryption algorithm three times to increase security. It is commonly used in legacy systems and applications.

- **Diffie-Hellman Key Exchange:** Diffie-Hellman is a key exchange protocol that allows two parties to establish a shared secret key over an insecure communication channel. It is often used as a component in encryption protocols to securely negotiate session keys between communicating parties.

How it works?



How it works?

The cryptography process involves both encryption and decryption to secure data and communications. Here's a simplified overview of the process

Symmetric encryption
Secret key
Secret key
Plaintext
Encryption
Ciphertext (encrypted)
Decryption
Plaintext

Plaintext

The original data or message that needs to be protected is called plaintext.

Encryption Algorithm

An encryption algorithm, such as AES, DES, or RSA, is applied to the plaintext. This algorithm uses a cryptographic key to perform mathematical operations that scramble the plaintext into ciphertext.

Encryption Key:

The encryption key is a parameter used by the encryption algorithm. It determines how the plaintext is transformed into ciphertext. The strength of the encryption largely depends on the complexity and length of the encryption key.

Encrypted Ciphertext

The encrypted ciphertext is received by the authorized party who possesses the correct decryption key.

Decryption Algorithm

A decryption algorithm, which is the reverse of the encryption algorithm, is applied to the ciphertext. This algorithm uses the decryption key to perform mathematical operations that reverse the encryption process and retrieve the original plaintext.

HOW ARE WE ENCRYPTING:

The encryption technique we're using here involves breaking down the video frames into a grid of puzzle pieces, shuffling those pieces randomly, and then saving the order of shuffling as a permutation. This process creates an "encrypted" version of the video that appears as shuffled puzzle pieces. The decryption process then reverses the shuffling to reconstruct the original video.

- **Step 1: Capturing the video frame by frame**

```python
def encrypt_video(grid_size):
    cap = cv2.VideoCapture(0)  # 0 for default webcam, change to the appropriate device ID if needed

    frame_width = int(cap.get(3))
    frame_height = int(cap.get(4))
    fps = 30.0  # Assuming a constant 30 frames per second

    out = cv2.VideoWriter('encrypted_video.mp4', cv2.VideoWriter_fourcc(*'mp4v'), fps, (frame_width, frame_height))

    # Create an empty list to store the permutation
    permutation = []

    while cap.isOpened():
        ret, frame = cap.read()
        if not ret:
            break

        puzzle_grid = create_puzzle_grid(frame, grid_size)
        shuffled_grid, piece_permutation = shuffle_puzzle_grid(puzzle_grid)

        # Add the piece permutation to the overall permutation list
        permutation.extend(piece_permutation)

        # Calculate the size of each puzzle piece in the frame
        piece_width = frame_width // grid_size
        piece_height = frame_height // grid_size

        # Create a new frame to store the shuffled puzzle pieces
        encrypted_frame = np.zeros_like(frame)

        # Assemble the shuffled puzzle pieces back into the encrypted frame
        for i in range(grid_size):
            for j in range(grid_size):
                piece = shuffled_grid[i * grid_size + j]
                left = j * piece_width
                upper = i * piece_height
                encrypted_frame[upper:upper + piece_height, left:left + piece_width] = piece

        out.write(encrypted_frame)

        # Show the encrypted frame in a window (optional)
        cv2.imshow('Encrypted Video', encrypted_frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break
```

- **Step 2: Dividing each frame into a number of grids**

```python
def create_puzzle_grid(frame, grid_size):
    height, width = frame.shape[:2]

    # Calculate the size of each puzzle piece
    piece_width = width // grid_size
    piece_height = height // grid_size

    puzzle_grid = []

    # Split the frame into puzzle pieces and store them in the puzzle_grid list
    for i in range(grid_size):
        for j in range(grid_size):
            left = j * piece_width
            upper = i * piece_height
            right = left + piece_width
            lower = upper + piece_height
            piece = frame[upper:lower, left:right]
            puzzle_grid.append(piece)

    return puzzle_grid
```

- **Step 3: Shuffling those grids randomly**

```
def shuffle_puzzle_grid(puzzle_grid):
    # Shuffle the puzzle pieces and store their original positions in a dictionary
    shuffled_grid = puzzle_grid.copy()
    indices = list(range(len(puzzle_grid)))
    random.shuffle(indices)

    permutation = [0] * len(puzzle_grid)
    for i, idx in enumerate(indices):
        shuffled_grid[i] = puzzle_grid[idx]
        permutation[idx] = i

    return shuffled_grid, permutation
```

- **Step 4: Saving the random premutation into a file**

```
def read_permutation_file(filename, frame_size):
    try:
        with open(filename, 'r') as perm_file:
            permutation_str = perm_file.read()
            permutation_list = [int(index) for index in permutation_str.split(',')]
            num_frames = len(permutation_list) // frame_size
            permutation_chunks = [permutation_list[i * frame_size: (i + 1) * frame_size] for i in range(num_frames)]
            print("Permutation Chunks:", permutation_chunks)
            return permutation_chunks
    except Exception as e:
        print("Error reading permutation:", e)
        return None
```

- **Step 5: Displaying the Encrypted frame**

```
        # Assemble the shuffled puzzle pieces back into the encrypted frame
        for i in range(grid_size):
            for j in range(grid_size):
                piece = shuffled_grid[i * grid_size + j]
                left = j * piece_width
                upper = i * piece_height
                encrypted_frame[upper:upper + piece_height, left:left + piece_width] = piece

        out.write(encrypted_frame)

        # Show the encrypted frame in a window (optional)
        cv2.imshow('Encrypted Video', encrypted_frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # Save the permutation to a file
    with open('encrypted_video.mp4.permutation', 'w') as perm_file:
        perm_file.write(','.join(str(index) for index in permutation))
        print(permutation)
        print("wwwwwwwwwwwwwwwwwwwwwwwwwwww")
    cap.release()
    out.release()
    cv2.destroyAllWindows()
```

- **Step 6: re-shuffling and rearranging the encrypted frames**

```
def reverse_shuffle_puzzle_grid(shuffled_grid, permutation):
    grid_size = int(np.sqrt(len(shuffled_grid)))
    original_grid = [None] * len(shuffled_grid)

    for i, idx in enumerate(permutation):
        original_grid[idx] = shuffled_grid[i]


    # Reshape the original_grid back to the original puzzle_grid
    puzzle_grid = np.array(original_grid).reshape((grid_size, grid_size, *original_grid[0].shape))


    return puzzle_grid
```

```
        # Split the frame into puzzle pieces
        puzzle_grid = []
        # Split the encrypted image into puzzle pieces and store them in the encrypted_grid list
        for i in range(grid_size):
            for j in range(grid_size):
                left = j * piece_width
                upper = i * piece_height
                right = left + piece_width
                lower = upper + piece_height
                piece = frame[upper:lower, left:right]
                puzzle_grid.append(piece)
        # Create an empty frame to store the decrypted frame
        decrypted_frame = np.zeros_like(frame)


        # Get the permutation for the current frame
        permutation = permutation_chunks.pop(0)

        # Assemble the puzzle pieces back into the decrypted image using the original ordering
        for i in range(grid_size):
            for j in range(grid_size):
                piece = puzzle_grid[permutation[i * grid_size + j]]
                left = j * piece_width
                upper = i * piece_height
                decrypted_frame[upper:upper + piece_height, left:left + piece_width] = piece
```

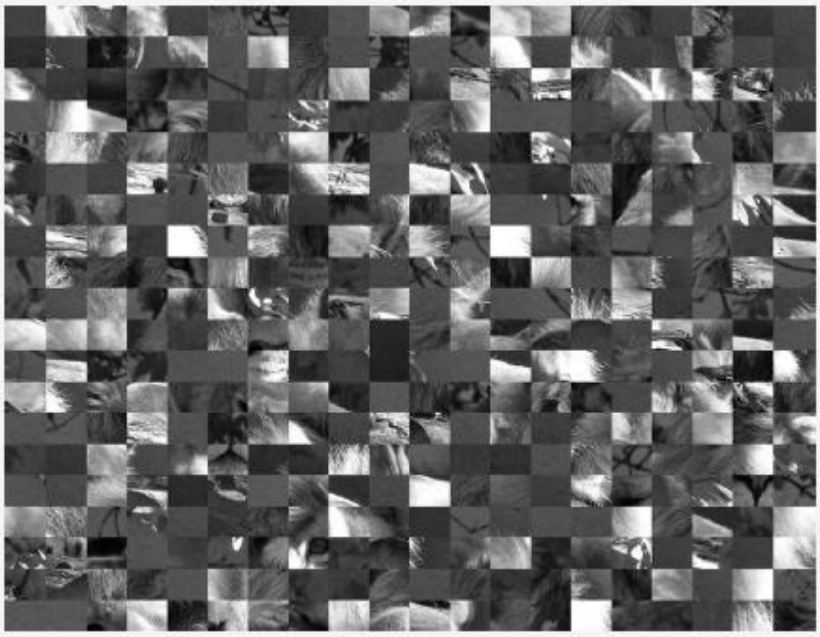
- **Step 7: Displaying decrypted frames**

```
        # For debugging, print the frame number and permutation
        print("Frame Number:", frame_number)
        print("Permutation for Frame:", permutation)

        frame_number += 1

        out.write(decrypted_frame)

        # Show the decrypted frame in a window (optional)
        cv2.imshow('Decrypted Video', decrypted_frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    cap.release()
    out.release()
    cv2.destroyAllWindows()
```

Outputs

Encryption:

The "encrypt_video" function takes an input video file, divides it into puzzle pieces, shuffles them randomly, and saves the shuffled pieces to a new video file called "encrypted_video.mp4." It also stores the permutation used for shuffling in a pickle file named "encrypted_video_permutation.pkl" for future decryption.



Decryption:

The "decrypt_video" function reads the "encrypted_video.mp4" file and loads the permutation from the pickle file. It then rearranges the puzzle pieces in the video frames based on the permutation and creates a new video file called "decrypted_video.mp4." The decrypted video should be similar to the original video.

Conclusion

In conclusion, The project aims to design a secure wireless transmission system for long-distance video transmission from a thermal camera. It converts analog video to digital, encrypts the data, and transmits it securely through a wireless network, ensuring confidentiality. At the receiver's end, the data is decrypted and displayed for real-time monitoring and surveillance applications.

- Wired to Wireless
- Hardwares involved in Wireless Transmission
- Military Grade
- Wireless Transmission Process
- Encryption Process
- Various encryption Protocols
- How are we encrypting