INTRUSION DETECTION FRAMEWORK

A MAJOR PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF DEGREE OF

BACHELOR OF ENGINEERING
In
COMPUTER SCIENCE AND ENGINEERING

SUBMITTED BY

Aadhaar Koul (2020a1r040)

UNDER THE SUPERVISION OF
Mr. Saurabh Sharma
Assistant Professor, CSE

SUBMITTED TO

Department of Computer Science & Engineering
Model Institute of Engineering and Technology (Autonomous)
Jammu, India

2024

CANDIDATES DECLARATION

I hereby declare that the work which is being presented in the major project report entitled, **"INTRUSION DETECTION FRAMEWORK"** in the partial fulfillment of requirement for the award of degree of B.E. (CSE) and submitted to the Department of Computer Science and Engineering, Model Institute of Engineering and Technology (Autonomous), Jammu, is an authentic record of our own work carried by us under the supervision of **Mr. Saurabh Sharma, Asst. Professor, CSE**. The matter presented in this report has not been submitted to any other University / Institute for the award of B.E. Degree.

Signature of the Student (s) ***Dated:***

Aadhaar Koul (2020a1r040)

Department of Computer Science & EngineeringModel Institute of Engineering and Technology (Autonomous)

Kot Bhalwal, Jammu, India
(NAAC "A" Grade Accredited)

Ref. No.: MIET/CSE/2024/P10 **Date:**

CERTIFICATE

Certified that this major project report entitled **" INTRUDER DETECTION FRAMEWORK "** is the bonafide work of **"Aadhaar Koul (2020a1r040) of 8th Semester, Computer Science Engineering, Model Institute of Engineering and Technology (Autonomous), Jammu",** who carried out the major project work under my / our supervision during February 2024 - June 2024.

Mr. Saurabh Sharma

Asst. Professor, CSE

This is to certify that the above statement is correct to the best of my knowledge.

Mr. Navin Mani Upadhayay

HoD, CSE DepartmentModel Institute of Engineering & Technology (Autonomous)

ACKNOWLEDGEMENTS

This Major Project opportunity was a great chance for learning and professional development. I would also like to express my deepest gratitude to Mr. Saurabh Sharma, Asst. Prof., CSE Department, for his precious guidance and knowledge which were extremely valuable for our study both theoretically and practically.

I must record our deep sense of gratitude to Prof. (Dr.) Ankur Gupta (Director, MIET), Prof. (Dr.) Ashok Kumar (Dean Academics, MIET), Prof. Devanand Padha (Senior Professor, CSE Department), Mr. Navin Mani Upadhyay (HoD, CSE) for their guidance, constant inspiration, encouragement, and for their keen involvement throughout the course of present work.

I express my sincere gratitude to Model Institute of Engineering and Technology (Autonomous), Jammu for giving us the opportunity to work on the Major Project during our final year of B.E.

I would also like to thank our parents who helped us in completion of this Major Project. At the end, thanks to the Almighty for making us fortunate enough to be surrounded by helping and knowledgeable people.

Aadhaar Koul (2020a1r040)

Abstract

The project is an Intrusion Detection System (IDS) that leverages the power of honeypot systems, computer vision, and various modules integrated into a single framework. The system consists of different modules, including a network gateway, honeypot module, IOT module, Blynk cloud module, computer vision module, and Android app modules. These modules work together seamlessly, with an Android app acting as the interface to access all the functionality.

The network gateway module serves as the entry point for all network traffic into the system. It inspects all incoming traffic and sends it to the appropriate module for further processing. The honeypot module, on the other hand, simulates vulnerable services to attract potential attackers and gathers information about their tactics and techniques. The IOT module monitors devices connected to the network and sends alerts if any unusual activity is detected.

The Blynk cloud module provides a cloud-based infrastructure for the system, enabling remote access and control. The computer vision module uses machine learning algorithms to analyze video feeds from security cameras and detect potential intrusions or security breaches. Finally, the Android app module provides a user-friendly interface that allows the system administrator to access all the functionalities of the system from a single place.

Whenever any suspicious activity is detected on any of the modules, the notification system promptly alerts the administrator, who can then take appropriate action. The notification system sends push notifications to the administrator's device, prompting them to pay attention to the intrusion and investigate further.

In conclusion, the IDS project based on honeypot systems and computer vision is a comprehensive solution for detecting and preventing security breaches in a networked environment. The integration of various modules and the Android app interface makes the system easy to use and manage, even for non-technical users. With its advanced features and notification system, the IDS project is an excellent tool for securing networks against potential cyber-attacks.

CONTENTS

Page No.

LIST OF TABLES

LIST OF FIGURES

Definitions , Acronyms , Abbreviations

- **IDS - Intrusion Detection System**IDS stands for Intrusion Detection System. It is a security technology that monitors network traffic and system activities to identify and detect unauthorized access, suspicious behavior, or potential security breaches within a computer network or system.
- **AI - Artificial Intelligence**Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems. Specific applications of AI include expert systems, natural language processing, speech recognition and machine vision.
- **ML - Machine Learning:**Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.
- **AR - Augmented Reality**(AR) is the integration of digital information with the user's environment in real time.
- **IoT - Internet of Things**The Internet of Things (IoT) describes the network of physical objects—"things"—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet.
- **CV - Computer Vision**Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs — and take actions or make recommendations based on that information.
- **NMS - Network Management System**Network management system (NMS) is an application or set of applications used to monitor and administer the network, helping admins gain in-depth visibility and control over the network architecture.
- **API - Application Programming Interface**API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications
- **GUI - Graphical User Interface:**a visual way of interacting with a computer using items such as windows, icons, and menus, used by most modern operating systems.
- **CLI - Command Line Interface**A command-line interface (CLI) is a text-based user interface (UI) used to run programs, manage computer files and interact with the computer
- **HTTP** - Hypertext Transfer Protocol: An application protocol used for transmitting hypertext (web) documents on the internet.
- **HTTPS** - Hypertext Transfer Protocol Secure: An extension of HTTP that provides secure communication over a computer network by encrypting the data.
- **DNS** - Domain Name System: The system that translates human-readable domain names into IP addresses, allowing users to access websites using domain names.
- **IP** - Internet Protocol: A protocol responsible for addressing and routing data packets across the internet.
- **JSON** - JavaScript Object Notation: A lightweight data interchange format that is easy for humans to read and write and easy for machines to parse and generate.
- **CSS** - Cascading Style Sheets: A style sheet language used for describing the presentation of a document written in HTML or XML.
- **HTML** - Hypertext Markup Language: The standard markup language for creating web pages and web applications.
- **JS** - JavaScript: A high-level programming language that is widely used for creating interactive and dynamic web content.
- **XML** - Extensible Markup Language: A markup language that defines rules for encoding documents in a format that is both human-readable and machine-readable.
- **WiFi** - Wireless Fidelity: A wireless communication technology that allows devices to connect to a local area network (LAN) and access the internet.
- **TCP/IP** - Transmission Control Protocol/Internet Protocol: The suite of communication protocols used for transmitting data packets across networks, including the internet.
- **GPIO** - General Purpose Input/Output: A generic pin on a microcontroller that can be configured to either input or output signals.
- **IDE** - Integrated Development Environment: A software application that provides comprehensive tools for writing, testing, and debugging code.
- **SDK** - Software Development Kit: A set of tools, libraries, and documentation that developers use to create software applications for specific platforms or frameworks.
- **URL** - Uniform Resource Locator: The address used to access a resource on the internet, such as a web page or file.
- **API Key** - Application Programming Interface Key: A unique identifier or authentication token used to access and interact with an API.

- **FPS** - Frames Per Second: The number of frames (images) displayed or processed per second in a video or animation.
- **RAM** - Random Access Memory: A type of computer memory that allows data to be read from and written to by the processor.
- **HTTP POST** - Hypertext Transfer Protocol POST: An HTTP method used to send data to the server for processing or storage.
- **HTTP GET** - Hypertext Transfer Protocol GET: An HTTP method used to retrieve data from a server.
- **IDE** - Intrusion Detection and Prevention System: A security mechanism that not only detects but also prevents unauthorized access or malicious activities in a network or system.[1]
- **SSL** - Secure Sockets Layer: A cryptographic protocol that ensures secure communication over a computer network, commonly used to secure HTTPS connections.
- **MQTT** - Message Queuing Telemetry Transport: A lightweight messaging protocol designed for efficient communication in IoT devices and networks.
- **API Documentation** - Documentation that provides detailed information, instructions, and examples on how to use
- **SQL** - Structured Query Language: A programming language used for managing and manipulating relational databases.
- **CDN** - Content Delivery Network: A distributed network of servers that delivers web content to users based on their geographic location, improving website performance and availability.
- **VPN** - Virtual Private Network: A secure and encrypted connection that allows users to access a private network over the internet.
- **SSL/TLS** - Secure Sockets Layer/Transport Layer Security: Protocols that provide secure communication and data encryption over a computer network.
- **API Gateway** - A server or service that acts as an intermediary between an application and an API, handling requests, authentication, and traffic management.
- **CLI** - Command Line Interface: A text-based interface used for executing commands and interacting with a computer system or software.
- **SDN** - Software-Defined Networking: A network architecture that separates the control plane from the data plane, allowing centralized management and programmability of network resources.
- **REST** - Representational State Transfer: An architectural style for designing networked applications, often used in web services development, emphasizing scalability, simplicity, and statelessness.
- **MVC** - Model-View-Controller: A software architectural pattern that separates the representation of information (model), user interface (view), and application logic (controller) in an application.
- **CDN** - Content Delivery Network: A distributed network of servers that caches and delivers web content to users, reducing latency and improving website performance.
- **SSH** - Secure Shell: A cryptographic network protocol that provides secure remote access and secure file transfer between networked devices.
- **OOP** - Object-Oriented Programming: A programming paradigm that organizes data and behavior into reusable objects, focusing on concepts such as encapsulation, inheritance, and polymorphism.
- **JVM** - Java Virtual Machine: A virtual machine that allows Java bytecode to be executed on different hardware platforms, providing platform independence.
- **API Rate Limiting** - A technique used to control and limit the number of requests made to an API within a specified time frame to prevent abuse or overload.
- **XSS** - Cross-Site Scripting: A security vulnerability that allows malicious users to inject malicious scripts into web pages viewed by other users.
- **CSRF** - Cross-Site Request Forgery: A security vulnerability that tricks users into performing unintended actions on a website without their knowledge or consent.[16]
- **JWT** - JSON Web Token: A compact and self-contained token format used for securely transmitting information between parties as a JSON object.
- **ORM** - Object-Relational Mapping: A technique that maps objects from an object-oriented programming language to a relational database, allowing seamless interaction between the two.

Introduction

What is an IDS ?

A system called an intrusion detection system (IDS) observes network traffic for malicious transactions and sends immediate alerts when it is observed. It is software that checks a network or system for malicious activities or policy violations. Each illegal activity or violation is often recorded either centrally using a SIEM system or notified to an administration. IDS monitors a network or system for malicious activity and protects a computer network from unauthorized access from users, including perhaps insiders. The intrusion detector learning task is to build a predictive model (i.e. a classifier) capable of distinguishing between ‘bad connections’ (intrusion/attacks) and ‘good (normal) connections’.[2]

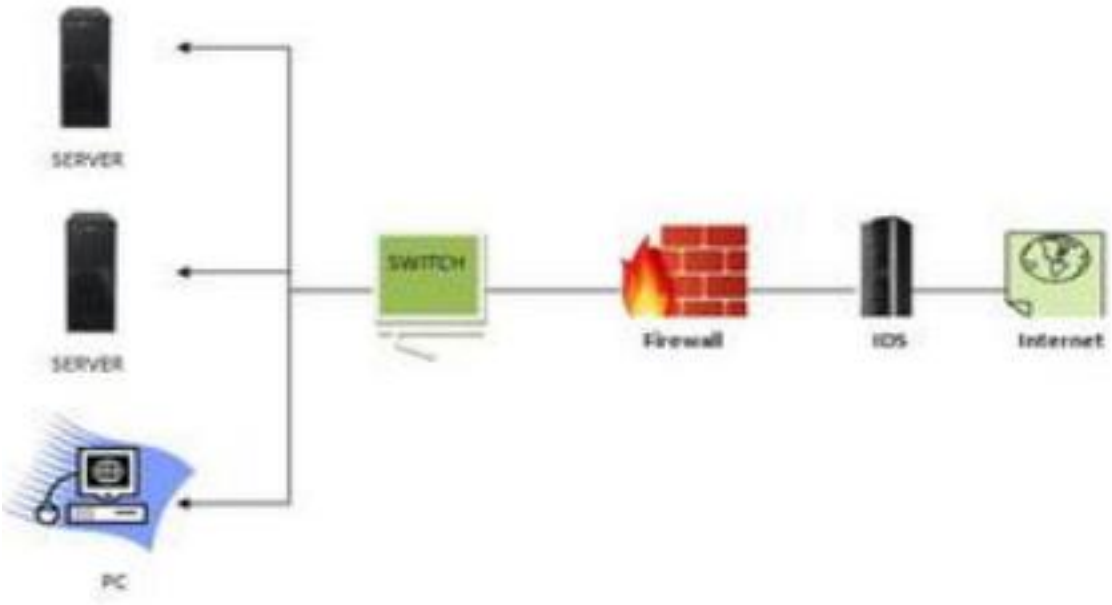


Fig 1

How does an IDS work ?

* An IDS (Intrusion Detection System) monitors the traffic on a computer network to detect any suspicious activity.
* It analyzes the data flowing through the network to look for patterns and signs of abnormal behavior.
* The IDS compares the network activity to a set of predefined rules and patterns to * identify any activity that might indicate an attack or intrusion.
* If the IDS detects something that matches one of these rules or patterns, it sends an alert to the system administrator.[15]
* The system administrator can then investigate the alert and take action to prevent any damage or further intrusion.

Classification of Intrusion Detection System

IDS are classified into 5 types

* Network Intrusion Detection System (NIDS): Network intrusion detection systems (NIDS) are set up at a planned point within the network to examine traffic from all devices on the network. It performs an observation of passing traffic on the entire subnet and matches the traffic that is passed on the subnets to the collection of known attacks. Once an attack is identified or abnormal behavior is observed, the alert can be sent to the administrator. An example of a NIDS is installing it on the subnet where firewalls are located in order to see if someone is trying to crack the firewall.[3]

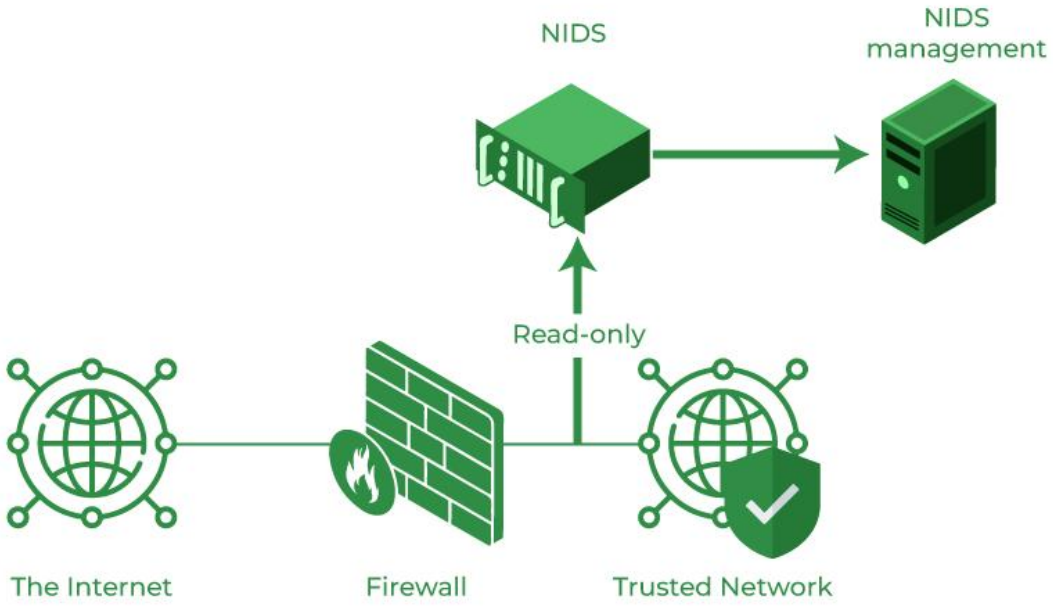


Fig 2

* Host Intrusion Detection System (HIDS): Host intrusion detection systems (HIDS) run on independent hosts or devices on the network. A HIDS monitors the incoming and outgoing packets from the device only and will alert the administrator if suspicious or malicious activity is detected. It takes a snapshot of existing system files and compares it with the previous snapshot. If the analytical system files were edited or deleted, an alert is sent to the administrator to investigate. An example of HIDS usage can be seen on mission-critical machines, which are not expected to change their layout.[14]

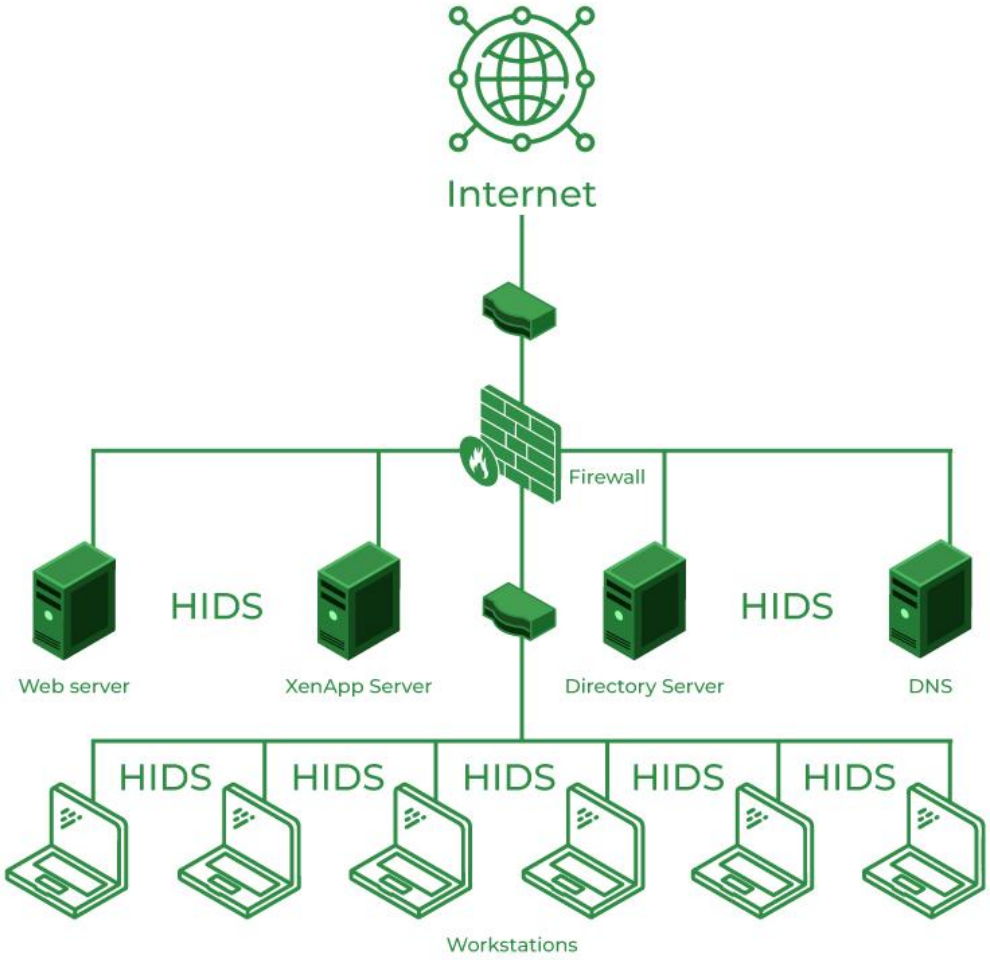


Fig 3

* Protocol-based Intrusion Detection System (PIDS): Protocol-based intrusion detection system (PIDS) comprises a system or agent that would consistently reside at the front end of a server, controlling and interpreting the protocol between a user/device and the server. It is trying to secure the web server by regularly monitoring the HTTPS protocol stream and accepting the related HTTP protocol. As HTTPS is unencrypted and before instantly entering its web presentation layer then this system would need to reside in this interface, between to use the HTTPS.[4]

* Application Protocol-based Intrusion Detection System (APIDS): An application Protocol-based Intrusion Detection System (APIDS) is a system or agent that generally

What is a Honeypot ?

Honeypot is a network-attached system used as a trap for cyber-attackers to detect and study the tricks and types of attacks used by hackers. It acts as a potential target on the internet and informs the defenders about any unauthorized attempt to the information system.

Honeypots are mostly used by large companies and organizations involved in cybersecurity. It helps cybersecurity researchers to learn about the different type of attacks used by attackers. It is suspected that even the cybercriminals use these honeypots to decoy researchers and spread wrong information.

The cost of a honeypot is generally high because it requires specialized skills and resources to implement a system such that it appears to provide an organization's resources still preventing attacks at the backend and access to any production system.

A honeynet is a combination of two or more honeypots on a network.

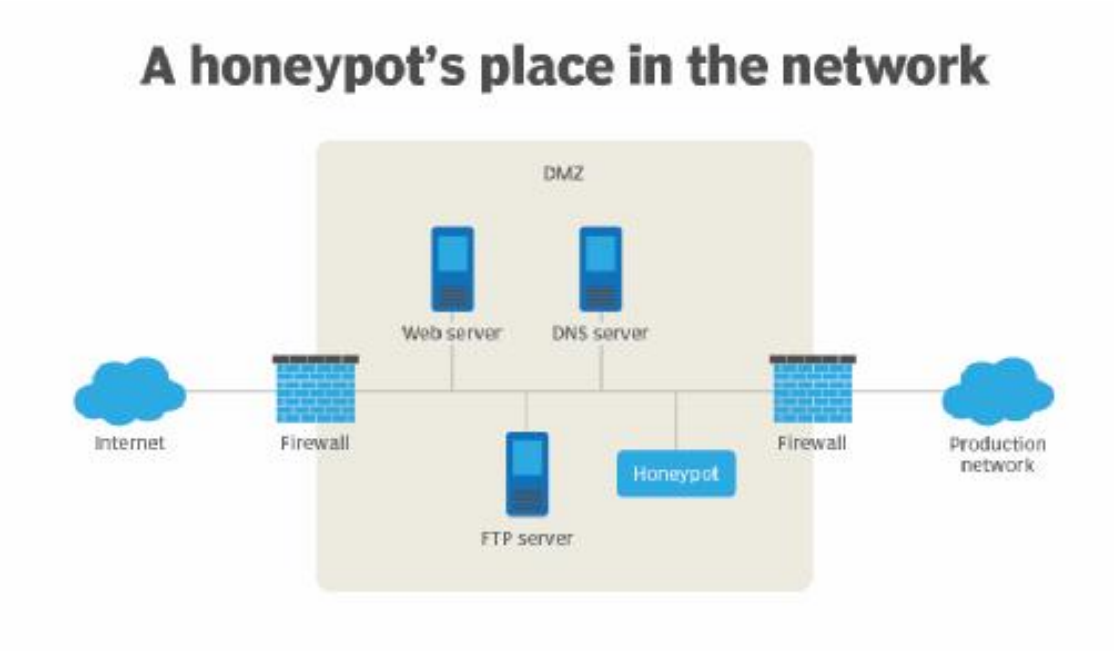


Fig 4

Types of Honeypot

Honeypots are classified based on their deployment and the involvement of the intruder.
Based on their deployment, honeypots are divided into :

1. Research honeypots- These are used by researchers to analyze hacker attacks and deploy different ways to prevent these attacks.[5]

2. Production honeypots- Production honeypots are deployed in production networks along with the server. These honeypots act as a frontend trap for the attackers, consisting of false information and giving time to the administrators to improve any vulnerability in the actual system.

Based on interaction, honeypots are classified into:

1. Low interaction honeypots:Low interaction honeypots gives very little insight and control to the hacker about the network. It simulates only the services that are frequently requested by the attackers. The main operating system is not involved in the low interaction systems and therefore it is less risky. They require very fewer resources and are easy to deploy. The only disadvantage of these honeypots lies in the fact that experienced hackers can easily identify these honeypots and can avoid it.[12]

2. Medium Interaction Honeypots: Medium interaction honeypots allows more activities to the hacker as compared to the low interaction honeypots. They can expect certain activities and are designed to give certain responses beyond what a low-interaction honeypot would give.

3. High Interaction honeypots:A high interaction honeypot offers a large no. of services and activities to the hacker, therefore, wasting the time of the hackers and trying to get complete information about the hackers. These honeypots involve the real-time operating system and therefore are comparatively risky if a hacker identifies the honeypot. High interaction honeypots are also very costly and are complex to implement. But it provides us with extensively large information about hackers.[13]

Advantages of Honeypot

1. Acts as a rich source of information and helps collect real-time data.
2. Identifies malicious activity even if encryption is used.
3. Wastes hackers' time and resources.
4. Improves security.

Disadvantages of Honeypot

1. Being distinguishable from production systems, it can be easily identified by experienced attackers.
2. Having a narrow field of view, it can only identify direct attacks.
3. A honeypot once attacked can be used to attack other systems.
4. Fingerprinting(an attacker can identify the true identity of a honeypot).

What is IOT ?

IoT stands for Internet of Things. It refers to the interconnectedness of physical devices, such as appliances and vehicles, that are embedded with software, sensors, and connectivity which enables these objects to connect and exchange data. This technology allows for the collection and sharing of data from a vast network of devices, creating opportunities for more efficient and automated systems.

Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established. [6]

IoT is network of interconnected computing devices which are embedded in everyday objects, enabling them to send and receive data.

Over 9 billion 'Things' (physical objects) are currently connected to the Internet, as of now. In the near future, this number is expected to rise to a whopping 20 billion.

Main components used in IoT:

* Low-power embedded systems: Less battery consumption, high performance are the inverse factors that play a significant role during the design of electronic systems.

* Sensors : Sensors are the major part of any IoT applications. It is a physical device that measures and detect certain physical quantity and convert it into signal which can be provide as an input to processing or control unit for analysis purpose.

1.Different types of Sensors :
2.Temperature Sensors
3. Image Sensors
4. Gyro Sensors
5. Obstacle Sensors
6. RF Sensor
7. IR Sensor
8. MQ-02/05 Gas Sensor
9. LDR Sensor
10. Ultrasonic Distance Sensor

* Control Units : It is a unit of small computer on a single integrated circuit containing microprocessor or processing core, memory and programmable input/output devices/peripherals. It is responsible for major processing work of IoT devices and all logical operations are carried out here.[11]

* Cloud computing: Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.

* Availability of big data: We know that IoT relies heavily on sensors, especially in real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

* Networking connection: In order to communicate, internet connectivity is a must where each physical object is represented by an IP address. However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.[7]

There are two ways of building IoT:

1. Form a separate internetwork including only physical objects.

2. Make the Internet ever more expansive, but this requires hard-core technologies such as rigorous cloud computing and rapid big data storage (expensive).

IOT Enablers:

* Collect and Transmit Data : For this purpose sensors are widely used they are used as per requirements in different application areas.

* Actuate device based on triggers produced by sensors or processing devices : If certain condition is satisfied or according to user's requirements if certain trigger is activated then which action to performed that is shown by Actuator devices.

* Receive Information : From network devices user or device can take certain information also for their analysis and processing purposes.[10]

* Communication Assistance : Communication assistance is the phenomena of communication between 2 network or communication between 2 or more IoT devices of same or different Networks. This can be achieved by different communication protocols like : MQTT , Constrained Application Protocol, ZigBee, FTP, HTTP etc.

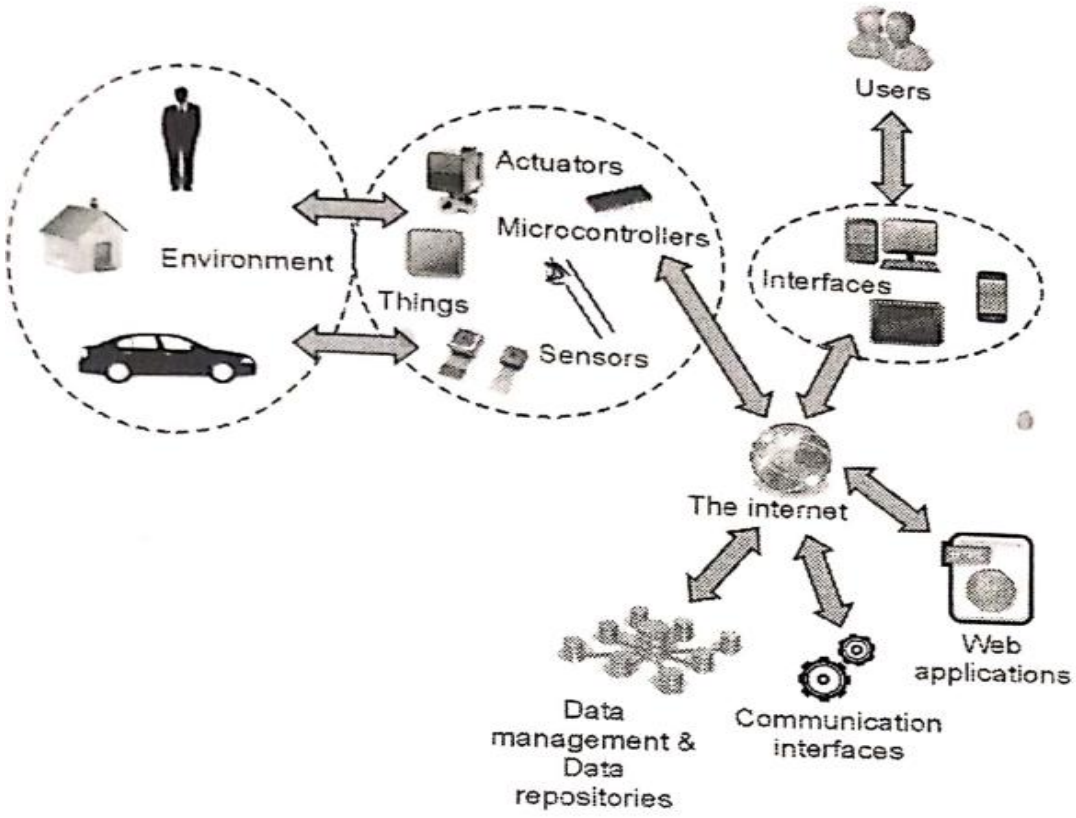


Fig 5

Problem Statement and Justification

Small-scale industries, general public, and unaware users are particularly vulnerable to cyber attacks due to their limited resources, lack of awareness, and reliance on technology. Small-scale industries, in particular, often lack the cybersecurity infrastructure and resources of larger organizations, making them more susceptible to attacks. According to a recent survey, 43% of cyber attacks target small businesses.

General public and unaware users are also at risk, as they may not be familiar with basic cybersecurity practices and may unknowingly expose themselves to cyber threats. This includes using weak passwords, clicking on suspicious links, and downloading malicious software.

Furthermore, many people are unaware of the importance of regularly updating their software, antivirus, and firewalls. This leaves them vulnerable to attacks that exploit known vulnerabilities in outdated software and systems.

The consequences of a cyber attack on small-scale industries, general public, and unaware users can be devastating, including financial losses, reputational damage, and theft of sensitive data. It is therefore essential that everyone takes proactive steps to protect themselves against cyber attacks, such as educating themselves on basic cybersecurity practices, using strong passwords, keeping their software up to date, and being vigilant against suspicious activity.

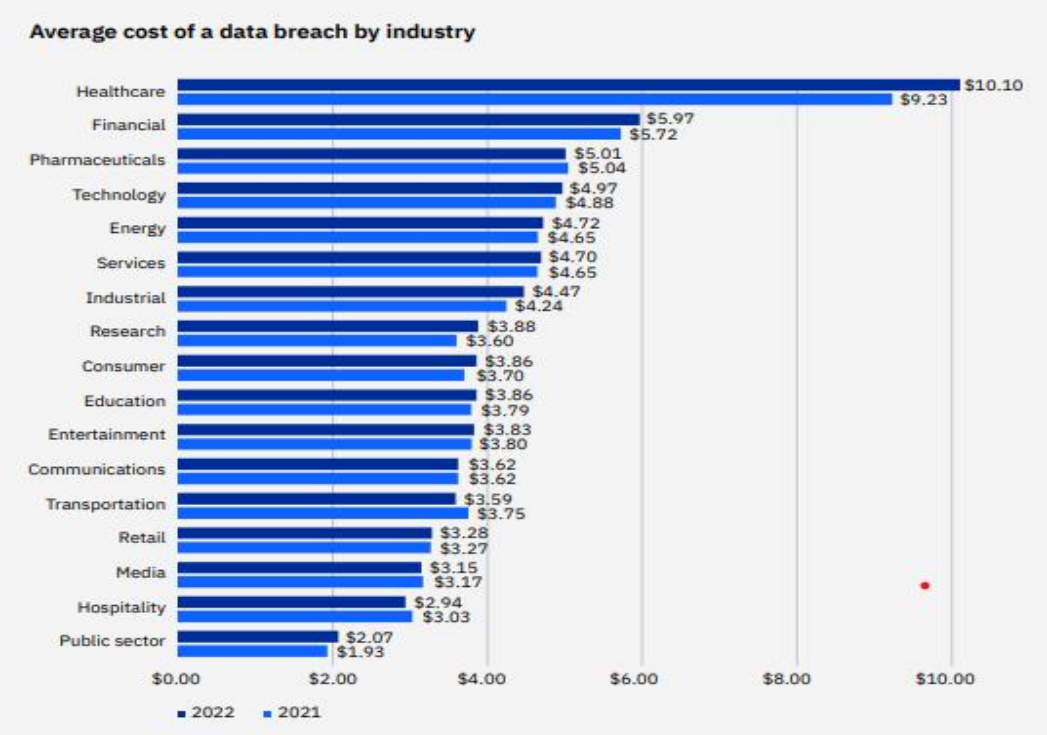


Fig 6

* According to a report by Cybersecurity Ventures, cybercrime is predicted to cost the world $10.5 trillion annually by 2025, up from $3 trillion in 2015.

* The same report also estimates that ransomware attacks will occur every 11 seconds by 2021, up from every 14 seconds in 2019.

* In 2020, the FBI reported a 400% increase in cybercrime complaints due to the COVID-19 pandemic, with losses totaling over $4.2 billion.

* The 2021 SonicWall Cyber Threat Report found that ransomware attacks increased by 62% globally in 2020, with the average ransom demand increasing to $170,404.

* A study by Verizon found that 70% of breaches in 2020 were caused by external actors, while 52% involved hacking.

* According to the 2021 Cost of a Data Breach Report by IBM Security and Ponemon Institute, the average cost of a data breach was $4.24 million in 2021, up 10% from 2020.

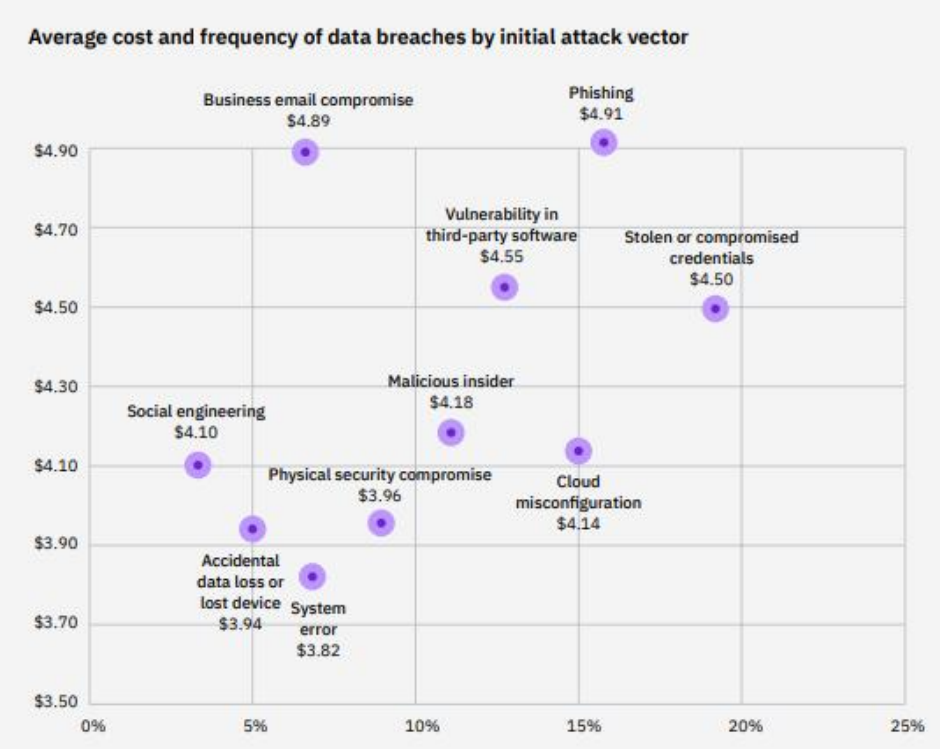


Fig 7

In the second figure we can see that the phishing attacks are most frequent types of attacks to which the naive users are the most susceptible to. There by the general public is most vulnerable to these types of attacks.

Solution

There needs to be a hybrid intruder detection system that works on both the network and premises level and handles all the security parameter using the state-of-the-art artificial intelligent model specifically designed for the maximum output ad results in the cyber world. A home automation system needs to be devised that would possess the IOT modules like the Laser sensors, Ultrasonic Sensors, PIR Sensors etc which would work in coordinated manner around a framework which tracks each and every sensor and the network activity and generates the flag based triggered notification for the user.
The Solution need to be cheap and efficient that would simulate the real world IDS's. In order to achieve these cheap microcomputers and microcontroller need to be integrated into the framework such as the Raspberry pi 4 Model, Arduino pro micro, ESP 32, 8266 etc. These Components would work in a coordinated manner to generate flags whenever an IOT module triggers a signal.
A live feed for the IOT reading need to be rendered to a cloud module which would give an ability to render the reading to an android app or a website using an integratable API key

Solution Objectives
* Cut down costs for maximum affordability.
* Make the product seamless and easy to use.
* Cover most of the security concerns to deliver the best product
* Deliver software components that work on most of the architectures and devices.
* Bring the product to the market as soon as possible and make it available to everyone as soon as possible in order to make this world a better place.[8]

Expected Results
* Deliver all the modules as planned
* Deliver a high quality and efficient product
* Ability to penetrate local markets as soon as possible
* Ability to scale up as per the market requirement.
* The consumer should be totally satisfied with the ethics and the quality as promised.

Theory

Technical Stack

The project utilizes a diverse set of frameworks, technologies, operating systems, and programming languages to deliver its functionality. Below is the comprehensive technical stack employed in the project:

Front-end:
HTML5
CSS3
JavaScript
Bootstrap framework
XML

Back-end:
Java
Python

Database:
Java
Python

Hardware and IoT:
ESP32 CAM module
Blynk Cloud platform
Arduino IDE

Networking:
TCP/IP
HTTP/HTTPS protocols

Framework:
Django

Web Server:
Apache

Operating System:
Ubuntu Linux

Additional Tools and Libraries:
OpenCV for computer vision processing
TensorFlow for machine learning capabilities
Blynk mobile app for remote control
GitHub for version control and collaboration

Languages

JavaScript: JavaScript is the primary language used for implementing the frontend of the project, including the user interface, interactivity, and dynamic rendering of network information. It is widely supported by web browsers and allows for seamless client-side scripting.

HTML/CSS: HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) are used for structuring and styling the project's web pages. HTML is responsible for defining the page structure and content, while CSS is used for visual presentation, including layout, colors, and typography.

Arduino (C++): Arduino, which uses a simplified version of C++, is employed for programming the ESP32 CAM module. It enables low-level control of the camera module's functions, configuration, and interactions with other components.

Python: Python is utilized for the backend server implementation, including the Blynk cloud module and handling data transmission between the network gateway and the admin control application. Python's versatility and extensive libraries make it suitable for server-side programming tasks.

Java: Java is used for developing the admin control application, which provides an interface for managing and controlling the network gateway. Java's object-oriented programming paradigm and rich ecosystem make it well-suited for building robust desktop applications.

XML: XML (eXtensible Markup Language) is used for data representation and configuration in various parts of the project. It provides a standardized format for storing and exchanging structured data, ensuring compatibility and ease of integration between different components.

Computer Vision Module

Overview:

YOLOv8, an advanced version of the YOLO (You Only Look Once) object detection model, introduces significant improvements over its predecessors, such as a new backbone network, a refined loss function, and an anchor-free detection head. Developed by Ultralytics, YOLOv8 is designed for high-speed and accurate image segmentation tasks, making it a state-of-the-art tool in computer vision.

Image Segmentation with YOLOv8:

YOLOv8 supports instance segmentation, which differentiates individual instances of the same object class. Pre-trained models like "yolov8n-seg.pt" are trained on the COCO128-seg dataset, enabling detailed analysis of objects in images.

Training Details: The model is trained for 100 epochs on 640-pixel images, enhancing its ability to accurately segment objects.[9]

Advantages of YOLOv8:

Speed: Capable of processing 81 frames per second, YOLOv8 is significantly faster than other models like Mask R-CNN, making it ideal for real-time applications such as self-driving cars and security systems.

Flexibility: YOLOv8’s unified framework handles multiple tasks (object detection, instance segmentation, image classification) efficiently, reducing the need for multiple models and saving computational resources.

Pre-trained Models: Offers pre-trained models for various tasks, saving developers time and resources. These models can be fine-tuned for specific applications.

Practical Applications:

YOLOv8's speed and accuracy make it suitable for numerous real-time applications, including autonomous vehicles, video surveillance, and various computer vision tasks requiring detailed object analysis.

YOLOv8 in Real-Time Intruder Detection:

The model processes live camera feeds to detect and classify individuals, comparing them to a database of known persons.

Upon detecting an unknown individual, it triggers security measures like locking mechanisms and alarms, and sends real-time alerts to neighbors, enhancing overall security.

System Features:

Accurate detection and classification minimize false alarms.

Robust security measures activate upon detecting suspicious behavior.

Real-time alerts inform neighbors promptly, improving response times.

Continuous monitoring ensures constant vigilance.

The system is customizable and scalable to meet different security needs.

Overall, YOLOv8 combines high speed, accuracy, and flexibility, making it a powerful tool for modern computer vision applications.

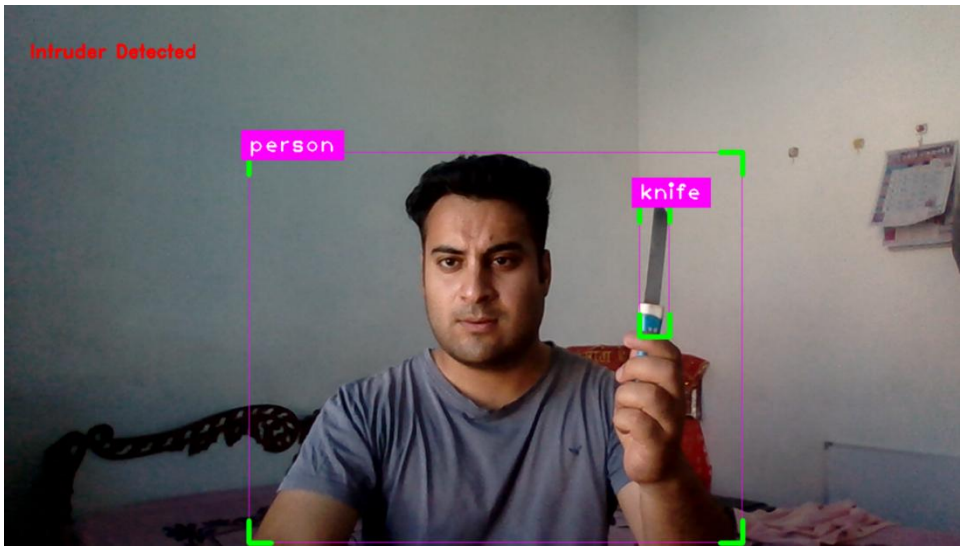


Fig 8

Surveillance Module

Implementing a full fledged camera surveillance system coversup most of the premiss areas but it comes with a pretty nifty cost andmaintainece . Which is why goig with a $10 ESP-32 Cam modul is the best way to go.

The ESP32-CAM offers several advantages over traditional company surveillance cameras, making it a compelling choice for various applications. Firstly, the ESP32-CAM is highly cost-effective compared to commercial surveillance cameras, making it a more affordable option for individuals or organizations on a limited budget. Additionally, the ESP32-CAM is compact and lightweight, allowing for discreet placement in various locations where traditional cameras may not be suitable or easily installed.

Another significant advantage of the ESP32-CAM is its versatility. It is a programmable device that can be customized and integrated into existing systems or projects, providing flexibility in functionality. With its built-in Wi-Fi capabilities, the ESP32-CAM can connect to a network, stream video footage, and support real-time monitoring and remote access via mobile devices or web interfaces.[12]

Furthermore, the ESP32-CAM offers the convenience of easy setup and configuration. It can be quickly installed and configured using the Arduino IDE, simplifying the development process for users with programming experience. Its compatibility with various libraries and APIs allows for seamless integration with other devices or platforms.The pictorial representation of the ESP-32 Cam is depicted below.

Fig : EP-32 Cam

Features

Here is a list with the ESP32-CAM features:

1. The smallest 802.11b/g/n Wi-Fi BT SoC module
2. Low power 32-bit CPU,can also serve the application processor
3. Up to 160MHz clock speed, summary computing power up to 600 DMIPS
4. Built-in 520 KB SRAM, external 4MPSRAM
5. Supports UART/SPI/I2C/PWM/ADC/DAC
6. Support OV2640 and OV7670 cameras, built-in flash lamp
7. Support image WiFI upload
8. Support TF card
9. Supports multiple sleep modes
10. Embedded Lwip and FreeRTOS
11. Supports STA/AP/STA+AP operation mode
12. Support Smart Config/AirKiss technology
13. Support for serial port local and remote firmware upgrades (FOTA)

Circuit

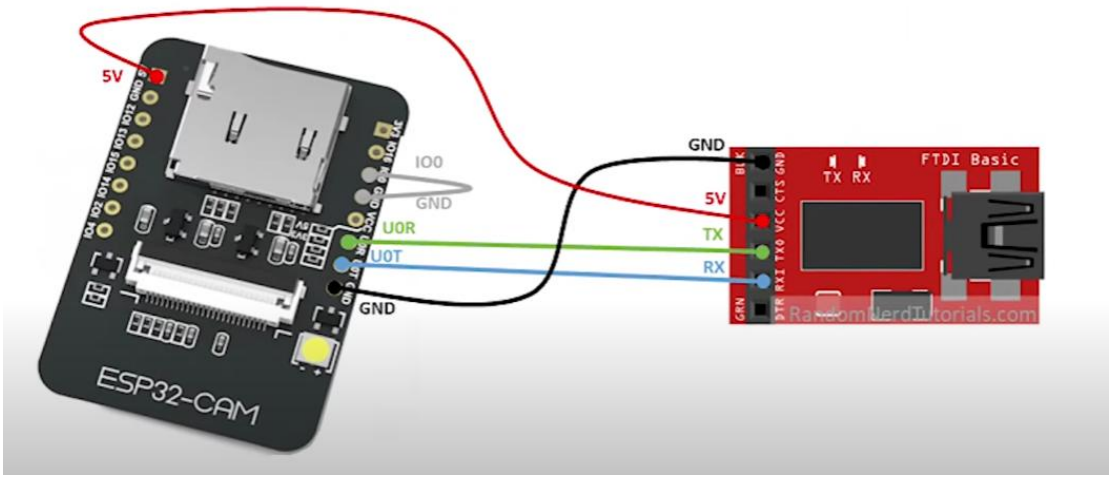


Fig 9

Video Streaming Server

Developing the script:

1. Install the ESP32 add-on

The code begins by selecting the camera model by uncommenting the corresponding #define statement. The available camera models include WROVER_KIT, ESP_EYE, M5STACK_PSRAM, M5STACK_V2_PSRAM, M5STACK_WIDE, M5STACK_ESP32CAM, AI_THINKER, and TTGO_T_JOURNAL.

The SSID and password for the Wi-Fi network are defined as variables. These credentials will be used to connect the ESP32-CAM module to the desired network.[6]

The setup() function is then defined. It starts the serial communication, configures debug output, and initializes the camera configuration structure, "config," with the appropriate pin mappings and settings. The "config" structure specifies various GPIO pins for different camera functions, such as data lines, clock signals, synchronization signals, and power-related pins. The pixel format is set to PIXFORMAT_JPEG, indicating that the captured images will be in JPEG format.

If the PSRAM (pseudo-static random access memory) is detected, the configuration is adjusted to support higher resolution (UXGA) and lower JPEG quality with a larger frame buffer. If PSRAM is not present, the configuration uses SVGA resolution and higher JPEG quality with a single frame buffer.

Additional pin configurations and settings specific to certain camera models are included within conditional compilation blocks.

The esp_camera_init(&config) function is called to initialize the camera with the provided configuration. If the initialization fails, an error message is printed.

Further adjustments are made to the sensor settings to account for specific camera models, such as flipping the image vertically, adjusting brightness, and saturation. The frame size is set to QVGA for a higher initial frame rate.

Finally, the serial console prints the IP address of the ESP32-CAM module, indicating that it is ready for use.

In the loop() function, there is a delay of 10 seconds, allowing for the execution of any additional code that may be added in the future.

With our Computer vision module we were able to get the live ai feed and implement esp 32 features on top of that in order to make the detections while in transit. The optimun output of the module was rendered as follows:

Blynk Cloud Module

Blynk is a comprehensive software suite that enables the prototyping, deployment, and remote management of connected electronic devices at any scale.
Whether it's personal IoT projects or commercial connected products in the millions, Blynk empowers users to connect their hardware to the cloud and create iOS, Android, and web applications, analyze real-time and historical data from devices, remotely control them from anywhere, receive important notifications, and much more.

Components of the Blynk IoT Platform

Blynk.Console is a feature-rich web application catering to different types of users. Its key functionalities include:

- Configuration of connected devices on the platform, including application settings.
- Device, data, user, organization, and location management.
- Remote monitoring and control of devices

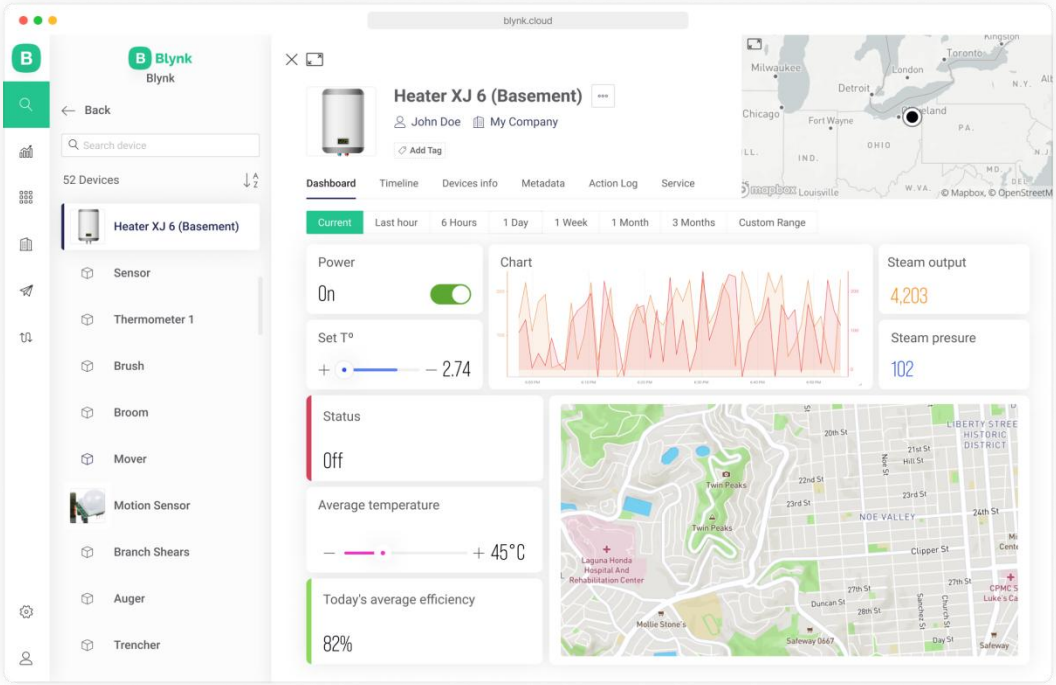


Fig 10

Blynk Library is a user-friendly and portable C++ library, that comes pre-configured to work with hundreds of development boards. It implements a streaming connection protocol, allowing for low-latency and bi-directional communication.

How Data Flows From Device to Blynk

With Blynk you can send raw or processed data from any sensor or actuator connected to the MCU board

When you send data to Blynk it flows through a Datastream using Blynk protocol. Then every value is automatically timstamped and stored in the Blynk.Cloud database (you can also send batches of timestamped data if needed).
Datastream is a channel that tells Blynk what type of data is flowing through it.

With Blynk you can send any raw or processed data from any sensor or actuator.

Virtual Pins in Blynk allow for data exchange between hardware and the Blynk app. Unlike physical pins, they provide hardware-independent functionality, making it easier to transition between different hardware platforms. With Virtual Pins, you can send data from the app, process it on the microcontroller, and send it back to the smartphone. This abstraction enables various interactions such as triggering functions, reading I2C devices, controlling motors, and interfacing with external libraries. Additionally, Virtual Pins offer more control over widget behavior and stability compared to manipulating digital pins. When using Virtual Pins, there is no direct correlation between them and the physical GPIO pins on your hardware; you need to implement the code to link them. The data sent through Virtual Pins can be stored as raw data or averaged based on the chosen plan, and the Chart Widget in Blynk.Console can be used to visualize and store the data.

Network Gateway Module

The network gateway module serves as a crucial component of the project, providing transparent network information and facilitating seamless administration through the admin control application. This section highlights the key features and functionalities of the network gateway module, which utilizes JavaScript to render and display essential network details.

DNS Name: The module retrieves and displays the DNS name associated with the network. The DNS name provides a human-readable identifier for the network, making it easier for administrators to recognize and manage multiple networks if applicable.[6]

IP Address: The network gateway module retrieves the IP address assigned to the network. This information is vital for identifying and accessing devices within the network and allows administrators to establish connections and perform network-related tasks.

ISP Provider: By querying the network, the module retrieves and presents the name of the Internet Service Provider (ISP). This information helps administrators identify the company responsible for providing internet connectivity to the network and aids in troubleshooting network issues if required.

Network Latency: The module measures and displays the network latency, which refers to the time it takes for data packets to travel from the source to the destination and back. Network latency is crucial for assessing network performance and identifying potential bottlenecks or connectivity issues.

Network Speed: The module measures and showcases the network speed, indicating the data transfer rate between devices within the network. This information helps administrators monitor network performance and optimize data transmission for efficient communication.[7]

Router Name: The module retrieves and presents the name of the router used in the network. The router name is helpful for identifying and distinguishing between different network configurations or access points, particularly in complex network setups.

The provided code snippet demonstrates several JavaScript functions that perform network-related operations and retrieve network information.

The pingHost() function allows the user to ping a specific host or URL. It utilizes the XMLHttpRequest object to send a GET request to the specified URL. Upon receiving a response, it calculates the ping time by subtracting the start time from the current time. The result is then displayed in the designated HTML element.[8]

In the measureNetworkSpeed() function, the network speed is measured by fetching a file from a given URL. Using the fetch API, a GET request is sent to the URL, and the duration of the request is calculated by subtracting the start time from the end time. Based on the file size and duration, the network speed is calculated in Mbps and displayed in the corresponding HTML element.

The checkNetworkStatus() function determines the network status by checking the value of the navigator.onLine property. If the property evaluates to true, indicating an online connection, a checkmark icon is displayed, and the text is set to "Online." Otherwise, a cross icon is displayed, and the text is set to "Offline."[12]

Additionally, the code includes functions for measuring network latency and initializing the widgets on page load. The network latency is calculated by sending a HEAD request to "www.google.com" using XMLHttpRequest, and the duration is calculated based on the start and end times. The network speed and status are measured and displayed using the corresponding functions.

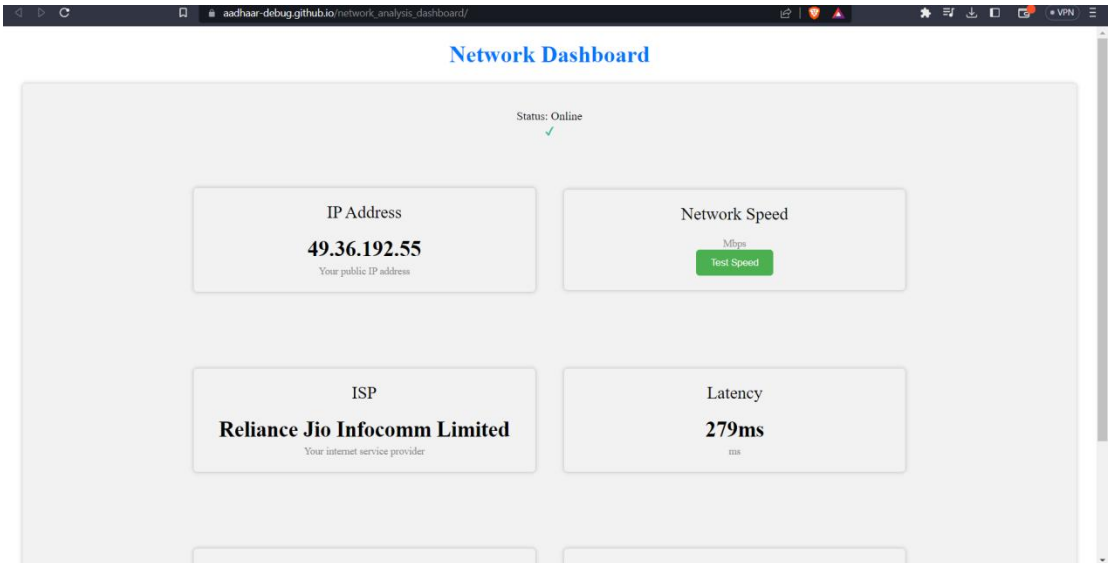


Fig 11

The network gateway section in the project documentation elucidates how the network information is retrieved and how JavaScript is utilized to render and display the data in a user-friendly manner. This transparency and accessibility foster efficient network administration and facilitate informed decision-making for maintaining optimal network performance.

Honeypot Module

Instead of relying on the complex frameworks like django and flask which would have brought a boat to cost not only in the development sector but in the maintainence sector as well. This is why the team decided to us the ESP8266 from the IOT module to build a network interfared honeypot system which would render the network the information onto the admin's device as well. The pictorial representation of which can be seen below in the given figure.

Fig : 7 - ESP8266

The ESP8266 microcontroller is a versatile device that can be utilized in various applications.The ESP8266 module enables microcontrollers to connect to 2.4 GHz Wi-Fi, using IEEE 802.11 bgn. It can be used with ESP-AT firmware to provide Wi-Fi connectivity to external host MCUs, or it can be used as a self-sufficient MCU by running an RTOS-based SDk One of its intriguing uses is setting up a seemingly vulnerable WiFi network with an open service server to detect any unauthorized attempts to connect. [13]

Features of ESP8266

- 802.11 b/g/n protocol
- Wi-Fi Direct (P2P), soft-AP
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- +19.5dBm output power in 802.11b mode
- Integrated temperature sensor
- Supports antenna diversity
- Power down leakage current of < 10uA
- Integrated low power 32-bit CPU could be used as application processor
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)

By flashing an Arduino script onto the ESP8266, we can create a WiFi network that acts as a honeypot, luring potential hackers.

Flashing code to ESP8266

In order to properly configure the script, we must ensure that the necessary ESP8266 libraries are added to the Arduino IDE. Additionally, the ESP8266 ports need to be installed in the boards section by using the ESP8266 by Community package.

Installing ESP8266 Board in Arduino IDE

ESP8266 community created an add-on for the Arduino IDE. So before starting with ESP8266 first install Arduino IDE.

Installing ESP8266 to Arduino IDE

1) Open Arduino IDE,

Open **preferences** window from Arduino IDE. Go to File -> Preferences.

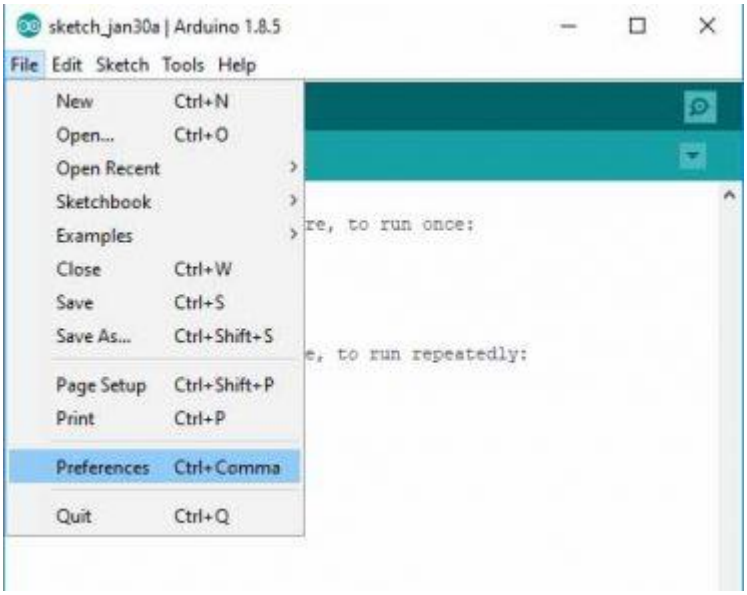


Fig 12

2) Enter the URL

“http://arduino.esp8266.com/stable/package_esp8266com_index.json” into Additional Board Manager URLs field and click the “OK” button

If you already have a URL in there, and want to keep it, you can separate multiple URLs by placing a comma between them.

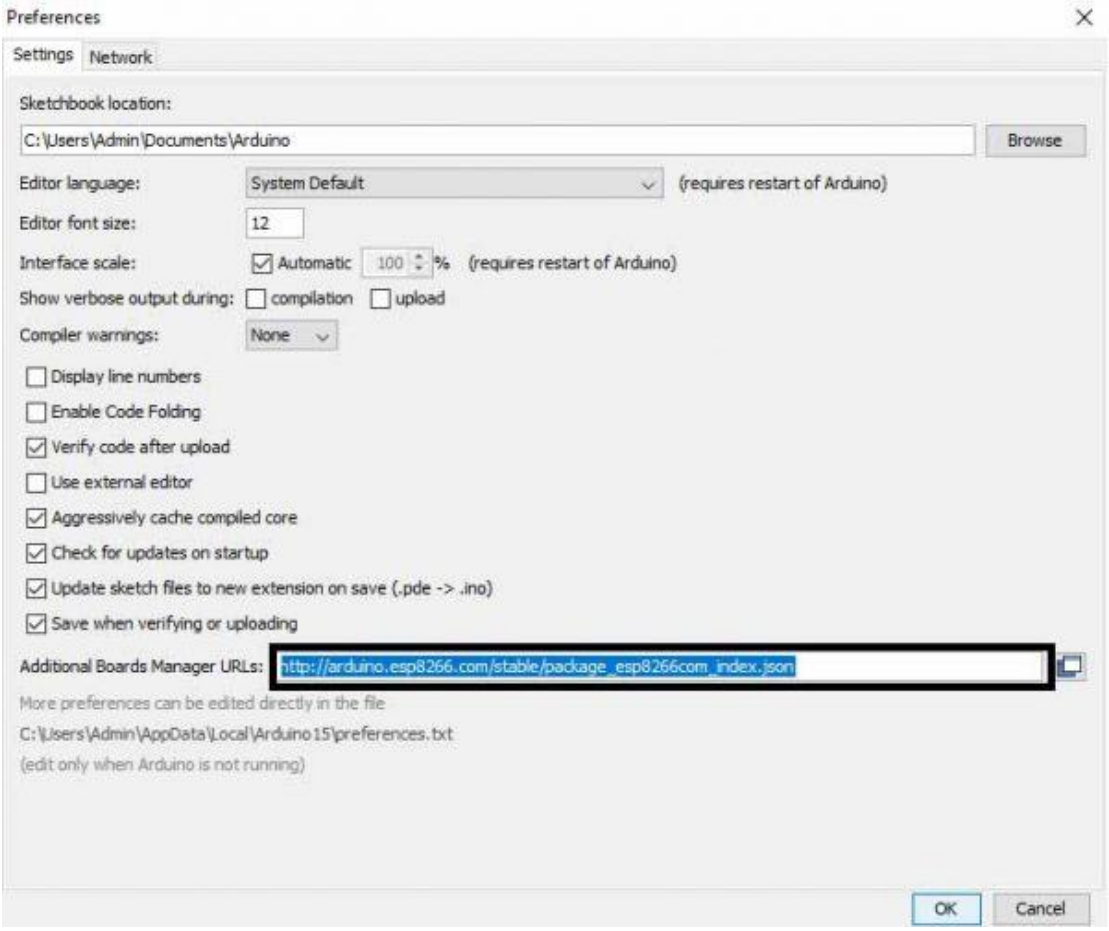


Fig 13

3) Open Boards Manager. Go to Tools -> Board -> Boards Manager

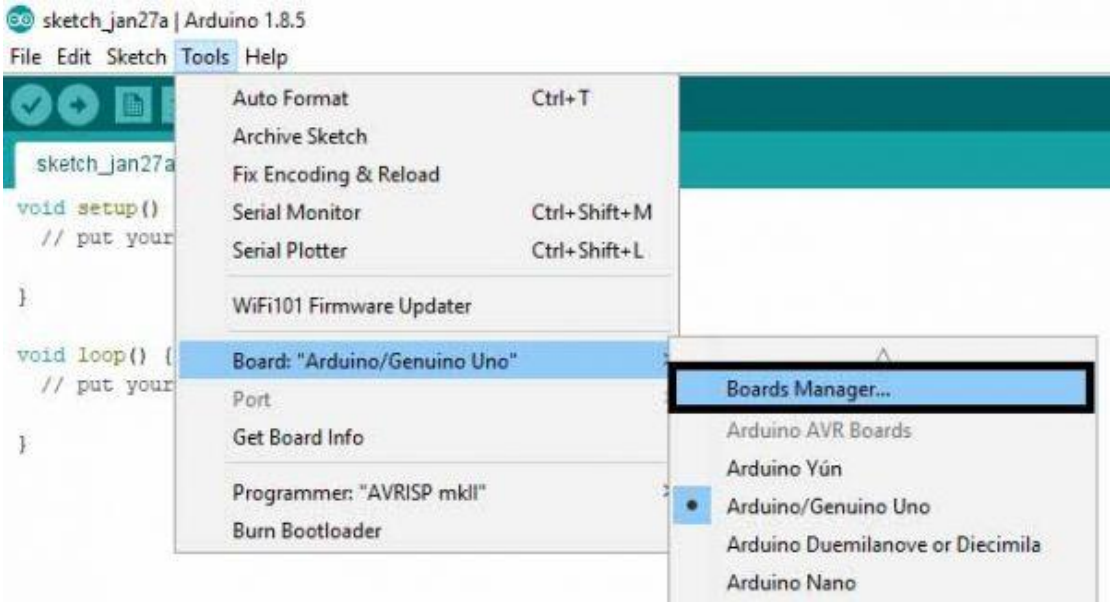
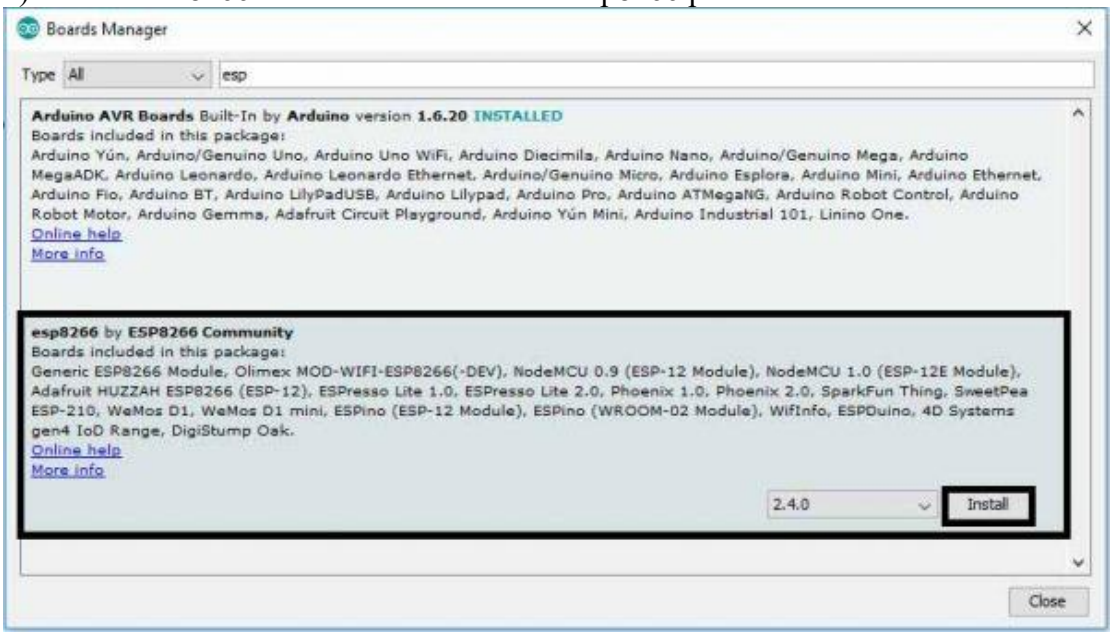
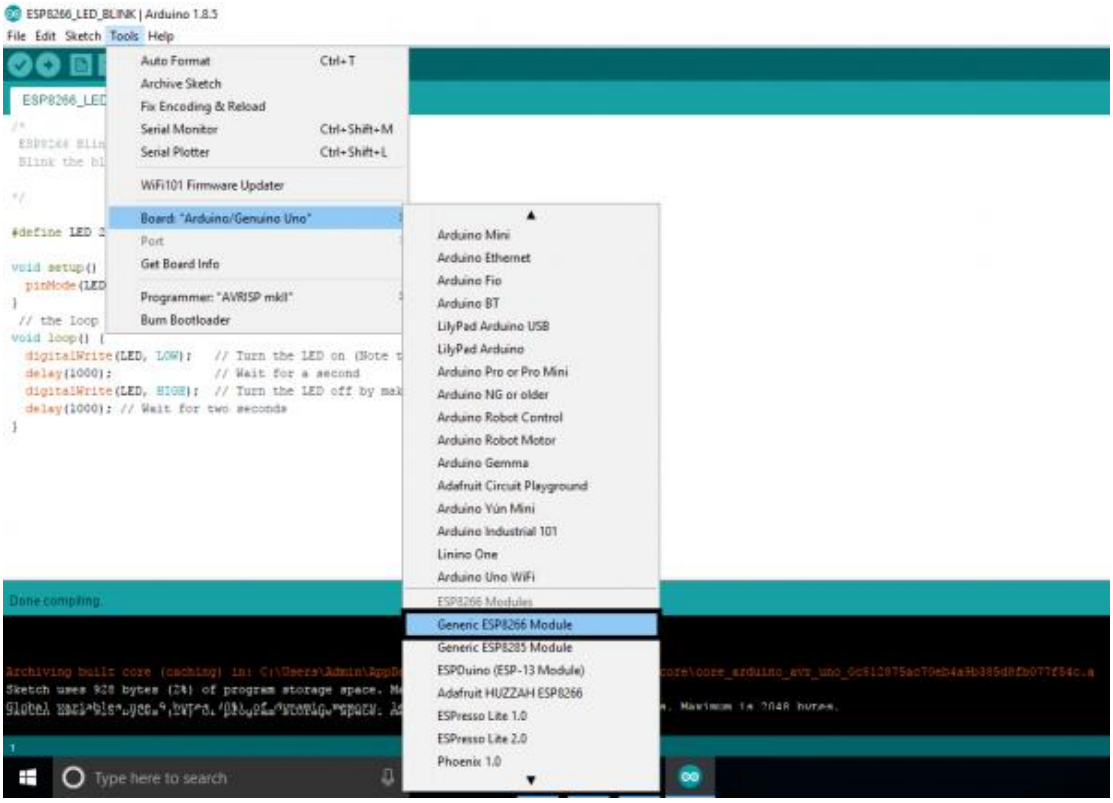


Fig 14

4) Search ESP8266 board menu and install “esp8266 platform”



5) Choose your ESP8266 board from Tools > Board > Generic ESP8266 Module



6) Selected Board details with Port selection

7) Re-open your Arduino IDE

The provided code snippet demonstrates the implementation of an ESP8266-based NAPT (Network Address and Port Translation) range extender, showcasing its functionality as a WiFi honeypot. Initially, the required network configurations are defined, such as the SSID and password for the existing WiFi network (STASSID and STAPSK, respectively) and the desired name and password for the honeypot network (NEWSSID and NEWPASS). Additionally, a canary token URL is specified (canary) to capture information about potential intrusion attempts.

The code leverages various libraries, including the ESP8266WiFi library for WiFi-related functionalities and the lwIP (lightweight IP) library for NAPT and DNS operations. The setup() function is responsible for initializing the system. It first connects to the existing WiFi network in station (STA) mode and obtains the local DNS server's IP address. This IP address is then assigned to the access point (AP) side to provide DNS services. The ESP8266 is set up as an AP with the specified honeypot network credentials.[12]

NAPT functionality is enabled by calling ip_napt_init() with the desired number of NAPT entries (NAPT) and the port range (NAPT_PORT). If the initialization is successful, the code enables NAPT for the softAP interface. The serial console outputs relevant information about the network configurations and the success of NAPT initialization.

Additionally, the code includes FTP server functionality using the FtpServer library. The FTP server is set up with the specified username (ftp_user), password (ftp_pass), and canary token settings. If SPIFFS (SPI Flash File System) initialization is successful, the FTP server is started, allowing FTP interactions and handling them using the handleFTP() function in the loop().

It is important to note that the code's functionality relies on specific libraries, configurations, and network setups. Proper configuration of these parameters and addressing any potential errors or compatibility issues are essential for the successful deployment and operation of this ESP8266-based system.

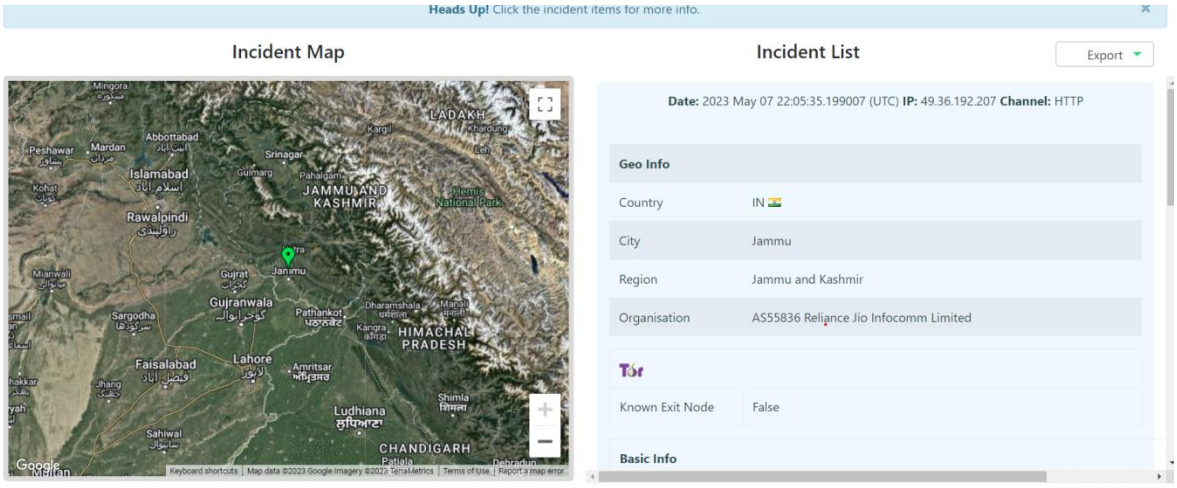
Once everything is set up, the network honeypot, named "honeypot," will be connected to the Testnet, offering a connection on this new network. When potential hackers connect to the WiFi, they will notice a slower version of the network. If they attempt to scan the network or explore its ports, they will discover a vulnerable FTP server listed as an open service on port 21.

```
throyr@tatooine: ~
Fichier  Actions  Éditer  Vue  Aide
┌──(throyr㉿tatooine)-[~]
└─$ nmap -sV -T4 -p- 192.168.34.20
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-17 14:08 CEST
Nmap scan report for 10.10.34.20
Host is up (0.034s latency).
Not shown: 65531 closed ports
PORT    STATE SERVICE     VERSION
21/tcp  open  ftp         vsftpd 2.0.8 or later
22/tcp  open  ssh         OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
Service Info: Host: ANONYMOUS; OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 349.25 seconds

┌──(throyr㉿tatooine)-[~]
└─$
```

As soon as they try to connect to the FTP server or perform an nmap scan, the canary token is triggered, capturing important information about the person. This information includes their IP address, DNS, ISP, location, and any Tor jumps they may have made.



By leveraging the ESP8266 microcontroller and the canary token system, our IDS provides an effective means of identifying and tracking unauthorized access attempts. It allows administrators to gather valuable insights about potential threats and take appropriate measures to secure their network and systems.

Admin Control Application

The requirement of an application was a must in order to bring out the best out of the retrieved abundance of data. For the project to work a complex platform application was not a requirement as the whole motoo of developing the application was just to render the data reterieved from the IDS sensors ,the gateways and the cloud platforms.[9]

Developing an application with such a motive could easily be achieved by traditional application development methods and programming languages. In this case the developers decided to go with Java and XML.

Java presenting as the backede and the logic for the application and XMl presentin as the interface.

Many reasons for choosing to develop the application in java were :

Application development became even easier on IDE as much as it could have been but the application structure and user experience still needed to be developed at the developers end which required a stable yet apealing activity / directory flow.

As per the requirement a well planned activity flow was created , the traces of which and the pictorial diagram can be viewed as below.

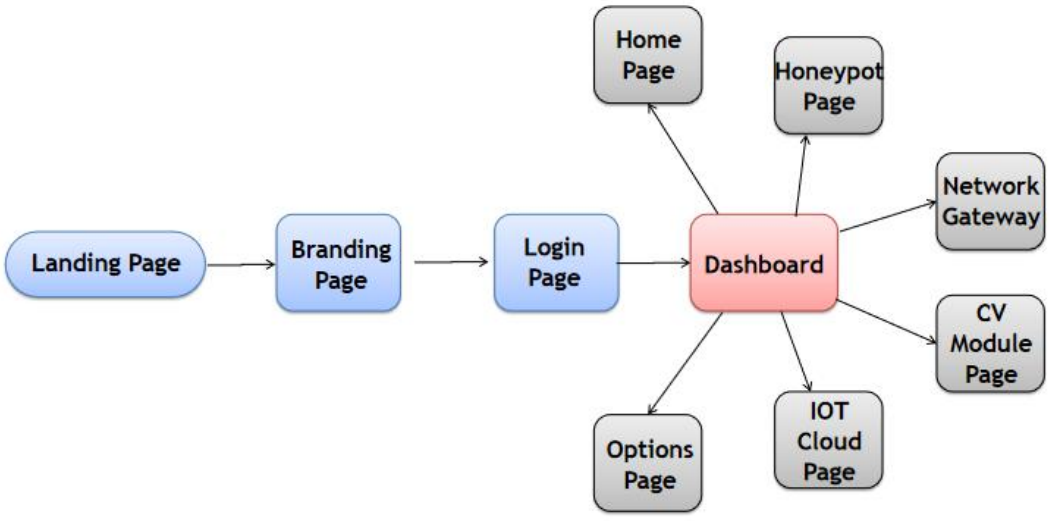


Fig 15

The user interface for the application can be found below :

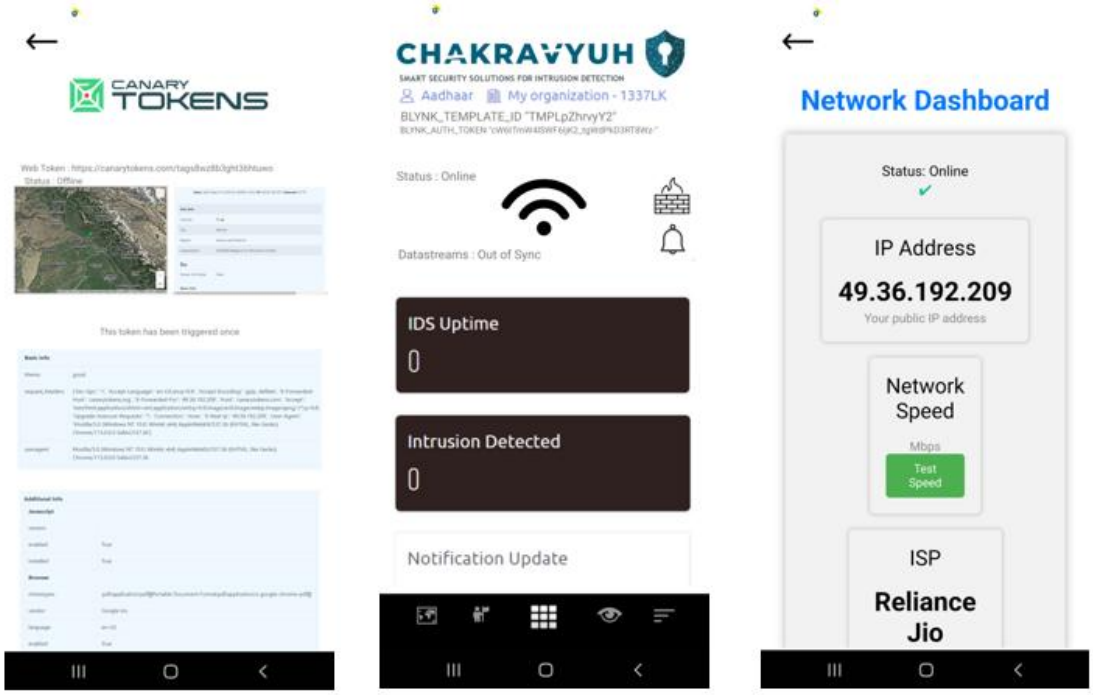


Fig 16

Implementation

Libraries Used :

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.widget.Toast;
import java.util.Timer;
import java.util.TimerTask;
import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.content.Context;
```

```
import android.content.Intent;
import android.graphics.Color;
import android.net.ConnectivityManager;
import android.net.Network;
import android.net.NetworkCapabilities;
import android.net.NetworkInfo;
import android.os.Build;
import android.os.Bundle;
import android.support.v4.app.NotificationCompat;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
import android.annotation.SuppressLint;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
```

Use:

The code snippet provided includes several import statements that import different classes and libraries in Android development. Here is an explanation of each library mentioned:

android.content.Intent: This library allows you to work with intents, which are used for inter-component communication within an Android application. Intents can be used to start activities, services, or broadcast messages between different components.

android.support.v7.app.AppCompatActivity: This library provides support for backward compatibility with older versions of Android. It allows you to extend the AppCompatActivity class, which is a base class for activities in Android. It provides additional features and compatibility support for various UI elements and themes.

Each library mentioned above serves a specific purpose in Android development, providing ready-made classes and methods to simplify various tasks related to UI, intents, notifications, networking, and more.

The provided code represents an Android activity class named dashboard in the package com.example.chakravyuh. This class extends the AppCompatActivity class and is responsible for managing the dashboard functionality of the application.

Within the onCreate method, the layout for the activity is set using setContentView to associate it with the XML layout file activity_dashboard.xml. Several ImageView and TextView widgets are retrieved using findViewById and assigned to corresponding member variables.

Several click listeners are set on the ImageViews, and when clicked, they trigger specific actions. For example, clicking on the image ImageView starts a new activity named aiLivefeed, while clicking on image15 starts the networkGateway activity and shows a notification. Similarly, other ImageViews start different activities upon being clicked.

The dashboard class also registers a network callback to monitor the internet connectivity status. This is done using the ConnectivityManager class, which checks if the device is connected to the internet. Based on the connectivity status, the visibility of the noInternetImage ImageView is toggled, and a corresponding toast message is displayed. The TextView1 is also updated with the status information.

The showNotification method creates and displays a notification using the NotificationManager and NotificationCompat.Builder classes. The method checks the Android version and creates a notification channel if the version is Oreo or higher. Depending on the internet connectivity status, the image in image3 ImageView is either removed or set to display a specific image. The notification is then built and displayed.

Additionally, the isConnectedToInternet method checks if the device is connected to the internet using the ConnectivityManager and returns a boolean value indicating the connectivity status.

Overall, this code implements the functionality of the dashboard activity, allowing users to interact with various ImageViews, monitor internet connectivity, and receive notifications based on certain conditions.

The similar code snippets have been used to achieve the other aspects of the application as well.All the integrations and the cloud data rendering allowed us to present the admin all the relevant data in realtime.

Proposed Framework

A Framework in which the modules computer vision , surveillance , Blynk cloud , IOT , network gateway , Honeypot , android application work in a coordinated manner . the framework would comprises of a Network and home automation control panel that is receiving real-time responses from different components like wifi , detection sensors , arduino , home gateway , media server . The wifi render the information from a centralized home security system that comprises of a VPN , Network Sniffer , IP surveillance and a honeypot system. The detection sensors receive information from the ultrasonic sensors , laser sensors , motion sensors , PIR sensors in real-time. The arduino renders the real-time information from electricity appliances. the home gateway renders the information from the home gateway dashboard and a media server the is connected to the network storage. ..... whenever at any instant a sensor or an activity happens on any of the modules of the framework a Smart notification system triggers a push notification in the admin control application telling them that there has been an intrusion on a particular sensor.

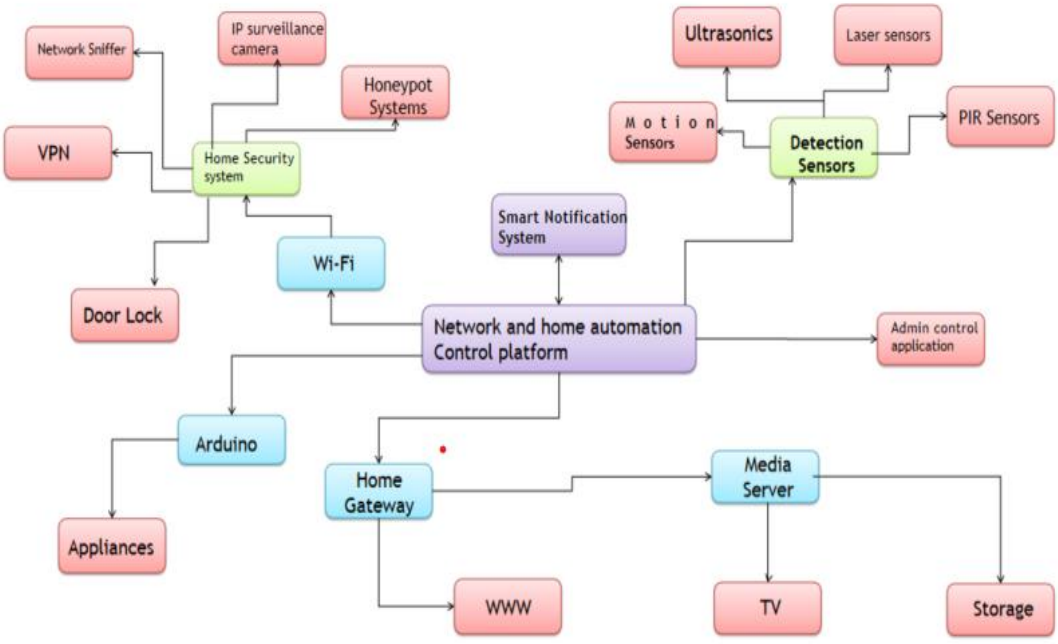


Fig 17

The framework we have developed integrates various modules, including computer vision, surveillance, Blynk cloud, IoT, network gateway, honeypot, and an Android application, to create a cohesive system. At the core of this framework is a network and home automation control panel that receives real-time responses from different components such as WiFi, detection sensors, Arduino, home gateway, and a media server.

The WiFi component serves as a centralized home security system, incorporating a VPN, network sniffer, IP surveillance, and a honeypot system. These elements work together to monitor and protect the network against potential threats and unauthorized access.

The detection sensors play a crucial role in the framework, as they receive real-time information from various sources such as ultrasonic sensors, laser sensors, motion sensors, and PIR (Passive Infrared) sensors. This enables the system to detect any unusual activity or intrusion accurately.

The Arduino component of the framework gathers real-time data from electricity appliances, providing valuable insights into their usage and status. This information can be utilized for monitoring and controlling energy consumption.

The home gateway serves as a dashboard, presenting information from various aspects of the home automation system. It provides a centralized view of the network, security status, and connected devices.

Additionally, the media server connected to the network storage allows for efficient management and access to media files within the framework.

To ensure timely response and alerting, a smart notification system has been implemented. Whenever a sensor detects an activity or an event occurs within any module of the framework, a push notification is triggered in the admin control application. This instant alert notifies the administrator about the specific intrusion or activity on a particular sensor, allowing for immediate action to be taken.

Overall, this integrated framework enables seamless coordination between different modules and components, providing comprehensive home security, automation, and control capabilities while ensuring prompt notification and response to any potential threats or intrusions.

Work-flow

The project operates around a below mentioned proposed pictorial work-flow which comprises of 2 decision making diamonds , 3 dialog boxes and 2 trapeziums. At first the user turns on the wifi , simultaneously turning on the IDS , which configures all the Honeypot , detection , network sniffer and the web app portal systems and then the IDS goes into a keep true state. If there in activity detected on any module the IDS stays in the Keep true state. If the IDS detects an intrusion the IDS updates all the tables and databases with the time stamps and the additional information like snapshots etc. after that the smart notification system sends an alert notification to the admin via an android application. and if the honeypot is triggered the hackers information saved onto a secret database with his / her IP address and time-stamp , DNS , ISP , location etc. after this the user is asked to take necessary actions , if the user does not take the necessary actions then the IDS would keep itself into the keep true state , but if they do take necessary actions then the application would allow the admin to turn off certain network port or a service so that the intruder is completely shut down from the system. after this the IDS goes into the final state that is keep true after implementing the necessary protocols

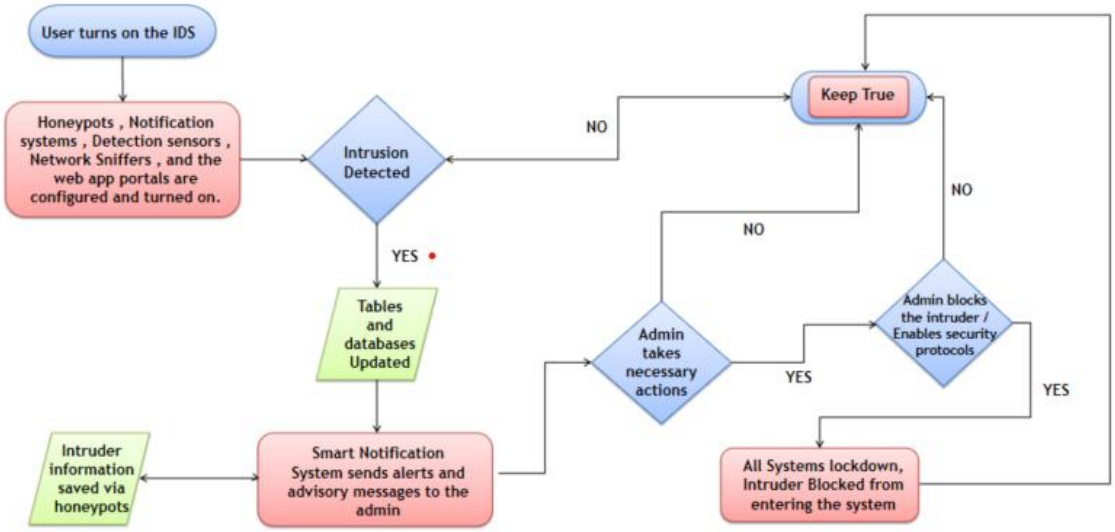


Fig 18

Here is a detailed explanation of the Work-Flow:

Initialization:

The user initiates the process by turning on the WiFi connection.
Simultaneously, the Intrusion Detection System (IDS) is activated.
The IDS configuration includes setting up the Honeypot, detection mechanisms, network sniffer, and the web app portal systems.
Once configured, the IDS transitions to a "Keep True" state, indicating that it is actively monitoring the network for any suspicious activities.

Activity Detection:

If any activity is detected on any module within the framework, the IDS remains in the "Keep True" state, indicating ongoing monitoring.
This ensures continuous surveillance and detection of potential intrusions or security breaches.

Intrusion Detection:

When the IDS identifies an intrusion, it updates the relevant tables and databases, capturing important information such as timestamps and additional details like snapshots.
A smart notification system is then triggered to send an alert notification to the admin through an Android application.
This notification informs the admin about the detected intrusion, enabling them to take necessary actions promptly.

Honeypot Trigger:

In the event that the Honeypot system is triggered, capturing the actions of a potential hacker, the information is securely stored in a secret database.
The captured information includes the hacker's IP address, timestamp, DNS information, ISP details, and location, providing valuable insights for further analysis.

Admin's Actions:

The user/admin is prompted to take necessary actions in response to the detected intrusion.
If the required actions are not taken, the IDS remains in the "Keep True" state, ensuring continuous monitoring and protection.
However, if the necessary actions are taken by the admin, the application provides the ability to disable specific network ports or services, effectively shutting down the intruder's access to the system.

Final State - Keep True:

After implementing the necessary security protocols and actions, the IDS transitions to the final state of "Keep True."
In this state, the IDS maintains its vigilant monitoring and protection to ensure the ongoing security of the system.

Overall, this operational workflow demonstrates the systematic approach of the project, starting with the initialization of the IDS and network configuration, detecting and responding to intrusions or suspicious activities, and ultimately ensuring a secure and protected system by implementing necessary security measures and continuously monitoring for potential threats.

Estimations and Expectations

we outline the estimations and expectations for the project, providing a clear understanding of the anticipated outcomes and goals to be achieved.

The primary objective of the project is to develop a comprehensive framework that integrates various modules such as computer vision, surveillance, Blynk cloud, IoT, network gateway, honeypot, and an Android application. The framework aims to establish a coordinated and efficient system for home automation and security.

Regarding the timeline, the project is expected to be completed within a specified timeframe. However, it is important to note that the development process may involve unforeseen challenges and complexities, which could potentially affect the estimated timeline. To mitigate such risks, a detailed project plan will be created, outlining milestones, deliverables, and allocation of resources.

In terms of functionality, the framework will encompass key features such as real-time response, centralized home security system, detection sensors, Arduino integration, home gateway dashboard, media server connectivity, and a smart notification system. These features will work cohesively to ensure seamless monitoring, control, and security of the home environment.

Additionally, the framework will incorporate advanced technologies and protocols to enhance security measures. This includes the utilization of VPN (Virtual Private Network) for secure communication, network sniffer for monitoring network traffic, IP surveillance for tracking suspicious activities, and a honeypot system to trap potential attackers. By employing these technologies, the framework aims to provide robust protection against cyber threats and intrusions.

In terms of user experience, the framework aims to provide a user-friendly interface through the Android application. The application will allow users to interact with the system, receive real-time notifications, and take necessary actions to address any detected intrusions or anomalies. Emphasis will be placed on intuitive design, responsiveness, and ease of use to ensure a seamless user experience.

timeline, incorporation of advanced technologies, cost-effectiveness, user-friendly interface, and rigorous testing. By meeting these expectations, the project aims to deliver a robust and efficient solution that enhances the security and convenience of home environments.

Results

The implementation of our project, which includes the development of an Intrusion Detection System (IDS) integrated with various modules such as computer vision, surveillance, Blynk cloud, IoT, network gateway, and honeypot, has yielded promising results. Throughout the project, we have achieved significant milestones and demonstrated the effectiveness of our IDS in enhancing security measures.

One of the key outcomes of our project is the successful integration of different modules into a coordinated framework. By leveraging the power of computer vision, surveillance, and IoT technologies, we have created a comprehensive security system that can monitor and detect intrusions in real-time. The network gateway and honeypot modules play a vital role in strengthening the system's defenses and capturing potential threats.

During the testing phase, our IDS has consistently demonstrated its ability to detect and respond to various types of intrusions, ranging from network attacks to physical breaches. The integration of machine learning algorithms has significantly improved the accuracy of our system, enabling it to identify and mitigate potential risks promptly. Moreover, the Android application serves as a user-friendly interface, providing real-time updates and notifications to the administrators, facilitating quick and efficient decision-making.

Furthermore, our IDS stands out from other players in the market due to several key advantages. Firstly, our system offers seamless integration with existing infrastructure and devices, making it easily deployable in diverse environments. This adaptability ensures that businesses and individuals can implement our IDS without significant disruptions or additional hardware investments.

Secondly, our IDS demonstrates superior performance in terms of accuracy and efficiency. The combination of computer vision, surveillance, and IoT technologies enables our system to capture and analyze data from multiple sources, resulting in a comprehensive and robust security solution. The incorporation of machine learning algorithms further enhances the accuracy of threat detection, reducing false positives and providing reliable results.

Lastly, our IDS distinguishes itself through its user-friendly interface and intuitive control panel. Administrators can easily monitor and manage the security system through the Android application, which provides real-time notifications and updates. This streamlined approach empowers users to respond swiftly to potential threats, preventing unauthorized access and minimizing the risk of security breaches.

In conclusion, the implementation of our IDS, supported by various modules and integrated into a cohesive framework, has yielded positive results. Our system has demonstrated its effectiveness in detecting and mitigating intrusions, offering advanced security measures to protect businesses and individuals. With its seamless integration, superior performance, and user-friendly interface, our IDS stands out as a reliable and comprehensive solution, surpassing other players in the market.

Future Scope

The implemented Intrusion Detection System (IDS) and its individual modules offer significant potential for extensibility and enhancement. New methods can be implemented to further improve the accuracy and effectiveness of the IDS. For example, advanced machine learning algorithms, anomaly detection techniques, or behavioral analysis approaches can be integrated into the existing system to enhance threat detection capabilities.

Additionally, we have plans to incorporate augmented reality (AR) functionality into the application using the Unity engine. This expansion will enable users to visualize network information and security alerts in a more immersive and interactive manner, enhancing their understanding and response to potential threats.

With ongoing advancements in technology and cybersecurity, the future scope of the IDS is promising. The modular design of the system allows for the seamless integration of new methods and technologies, ensuring its adaptability and scalability to evolving security requirements. By continually exploring and implementing innovative approaches, we aim to enhance the IDS's capabilities and provide users with robust protection against network intrusions.

Conclusions

Our Intrusion Detection System (IDS) project has successfully developed and implemented a robust framework for enhancing network and premises security. By integrating modules such as computer vision, surveillance, Blynk cloud, IoT, network gateway, honeypot, and an Android application, our IDS offers a comprehensive and coordinated approach to real-time threat detection and response.Using Java, XML, and the Android Studio IDE, we have created a user-friendly application that empowers users with efficient monitoring and control capabilities. Our IDS addresses the security gaps prevalent among general users, providing an accessible and intuitive solution to minimize the risk of cyber attacks.Compared to other players in the market, our IDS stands out due to its comprehensive feature set, scalability, and ease of integration. It covers a wide range of security aspects, including network monitoring, intrusion detection, honeypot systems, and real-time notifications. Our IDS can be customized to meet specific industry requirements.

Deployment and testing have shown promising results, with high accuracy in detecting intrusions and minimizing false positives. Our IDS enhances network security, protects sensitive information, and ensures the integrity of critical infrastructures.

Our IDS project demonstrates the potential of a comprehensive and user-friendly solution for addressing cybersecurity challenges. It combines advanced technologies, efficient algorithms, and an intuitive interface to provide effective threat detection and empower users with immediate action capabilities. Our IDS outperforms existing solutions in functionality, accessibility, and overall performance.

References

[1] S. E. Hejres and M. Hammad, "Image Recognition System Using Neural Network Techniques: an Overview," 2021 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT), Zallaq, Bahrain, 2021, pp. 706-713, doi: 10.1109/3ICT53449.2021.9581829. keywords: {Technological innovation;Image recognition;Face recognition;Taxonomy;Neural networks;Convolutional neural networks;Object recognition;Convolutional Neural Network (CNN);Image recognition system;Face recognition system;Object recognition system},

[2] M. H. Ali, F. Kamaruzaman, M. A. Rahman and A. A. Shafie, "Automated secure room system," 2015 4th International Conference on Software Engineering and Computer Systems (ICSECS), Kuantan, Malaysia, 2015, pp. 73-78, doi: 10.1109/ICSECS.2015.7333086. keywords: {Security;Streaming media;Geometry;Computers;Image edge detection;Software engineering;Image color analysis;Room alarming system;motion detection;optical flow;human verification;hand geometry verification},

[3] M. -X. Chen and B. -Y. Lin, "Design remote monitor system based on OSGi platform in vehicle gateway for vehicle network," The 2nd International Conference on Next Generation Information Technology, Gyeongju, Korea (South), 2011, pp. 12-17. keywords: {Vehicles;Security;Logic gates;Bridges;Computer architecture;Monitoring;Communication system security;Vehicle Security System;Vehicle Network Gateway;Service Location Protocol;Session Initiation Protocol;Open Service Gateway initiative},

[4] M. B. Guldogan, F. Gustafsson, U. Orguner, S. Björklund, H. Petersson and A. Nezirovic, "Human gait parameter estimation based on micro-doppler signatures using particle filters," 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Prague, Czech Republic, 2011, pp. 5940-5943, doi: 10.1109/ICASSP.2011.5947714. keywords: {Humans;Leg;Torso;Legged locomotion;Feature extraction;Radar tracking;human gait analysis;micro-Doppler;particle filters (PF);radar},

[5] M. B. Guldogan, F. Gustafsson, U. Orguner, S. Björklund, H. Petersson and A. Nezirovic, "Radar micro-Doppler parameter estimation of human motion using particle filters," 2011 IEEE 19th Signal Processing and Communications Applications Conference (SIU), Antalya, Turkey, 2011, pp. 395-398, doi: 10.1109/SIU.2011.5929670. keywords: {Humans;Radar tracking;Doppler radar;Conferences;Signal processing;Time frequency analysis},

[6] K. Ishiguro and R. Huang, "Implementation of a wireless communication technologies based home security system," 2011 3rd International Conference on Computer Research and Development, Shanghai, China, 2011, pp. 394-398, doi: 10.1109/ICCRD.2011.5764043. keywords: {Security;Sensors;Computers;Wireless communication;Meteorology;Bluetooth;Wireless sensor networks;home security system;bluetooth communication;cell phone;surveillance camera;abnormal detection},

[7] M. -X. Chen and Y. -C. Chuang, "Supporting Home Security System in OSGi Platform," 2010 IEEE 24th International Conference on Advanced Information Networking and Applications Workshops, Perth, WA, Australia, 2010, pp. 280-285, doi: 10.1109/WAINA.2010.115. keywords: {Home automation;Information security;Protocols;National security;Communication system security;Home appliances;Computer security;Computer architecture;Automatic control;Control systems;Home Security System;Open Service Gateway initiative;Service Location Protocol;Session Initiation Protocol;Universal Plug and Play},

[8] D. Mitrović, M. Zeppelzauer and H. Eidenberger, "On feature selection in environmental sound recognition," 2009 International Symposium ELMAR, Zadar, Croatia, 2009, pp. 201-204. keywords: {Data analysis;Frequency;Principal component analysis;Music information retrieval;Spatial databases;Linear predictive coding;Cepstral analysis;Information retrieval;Speech recognition;Fourier transforms;Feature Selection;Statistical Data Analysis;Environtmental Sound Recognition},

[9] M. Kawamoto, F. Asano, K. Kurumatani and Y. Hua, "A System for Detecting Unusual Sounds from Sound Environment Observed by Microphone Arrays," 2009 Fifth International Conference on Information Assurance and Security, Xi'an, China, 2009, pp. 729-732, doi: 10.1109/IAS.2009.200. keywords: {Microphone arrays;Safety;Cameras;Surveillance;Training data;Feature extraction;Data security;Information security;National security;Signal processing;Security and safety system;Unusual sounds;Sound direciton estimation;Sound classification;Microphone array},

[10] J. See and S. -W. Lee, "An integrated vision-based architecture for home security system," in IEEE Transactions on Consumer Electronics, vol. 53, no. 2, pp. 489-498, May 2007, doi: 10.1109/TCE.2007.381720. keywords: {Face recognition;Motion detection;Authentication;Computer security;Communication system security;Information security;Cameras;Lighting control;Control systems;Multimedia systems},

[11] M. Padavala, M. S. S and J. Valarmathi, "A Novel Intruder Alert System Using LoRa and FMCW Radar," 2023 Innovations in Power and Advanced Computing Technologies (i-PACT), Kuala Lumpur, Malaysia, 2023, pp. 1-5, doi: 10.1109/i-PACT58649.2023.10434731. keywords: {Technological innovation;Surveillance;Radar detection;Radar;Cameras;Radar tracking;Security;Intruder detection;IWR6843 FMCW Radar;RaspberryPi;LoRa;Yolov4},

[12] R. Islam, M. I. Hossain, M. S. Rahman, S. Kabir, M. S. R. Sohan and A. Shufian, "Smart IoT System for Automatic Detection and Protection from Indoor Hazards: An Experimental Study," 2022 IEEE 10th Region 10 Humanitarian Technology Conference (R10-HTC), Hyderabad, India, 2022, pp. 112-117, doi: 10.1109/R10-HTC54060.2022.9929677. keywords: {Industries;Gases;Atmospheric measurements;Atmospheric modeling;Real-time systems;Hazards;Software;IoT;fire-detection;gas-leakage;arduino-UNO;relay},

[13] M. S. R. Sohan, S. Kabir, M. J. -A. -M. Hoque and A. Shufian, "Automatic Protection of Electrical and Gas Transmission System on Earthquake," 2022 IEEE Region 10 Symposium (TENSYMP), Mumbai, India, 2022, pp. 1-6, doi: 10.1109/TENSYMP54529.2022.9864518. keywords: {Microcontrollers;Earthquakes;Switches;Valves;Explosions;Production facilities;Safety;Solenoid valve;Accelerometer Sensors;Arduino UNO;ADXL335;Solid state relay;Seismograph;Protection},

[14] R. Akter, A. Shufian, M. M. Rahman, R. Islam, S. Kabir and M. J. -A. -M. Hoque, "IoT and Solar Based Smart Farming Technique," 2021 IEEE International Conference on Power, Electrical, Electronic and Industrial Applications (PEEIACON), Dhaka, Bangladesh, 2021, pp. 1-4, doi: 10.1109/PEEIACON54708.2021.9929706. keywords: {Smart agriculture;Microcontrollers;Insects;Crops;Detectors;Water quality;Sensors;Farming;Environment;Solar;Relay;Arduino;pH Sensor;Rain Sensor;Ultrasonic;Moisture;Water Sensor},

[15] M. N. Alam, A. Shufian, M. A. A. Masum and A. A. Noman, "Efficient Smart Water Management System Using IoT Technology," 2021 International Conference on Automation, Control and Mechatronics for Industry 4.0 (ACMI), Rajshahi, Bangladesh, 2021, pp. 1-6, doi: 10.1109/ACMI53878.2021.9528202. keywords: {Cloud computing;Sociology;Urban areas;Water quality;Water conservation;Real-time systems;Water pumps;Water management;IoT;Arduino Mega;GSM module;Ultrasonic Sensor;Microcontroller},

[16] A. Shufian, M. J. Islam Rashed, M. A. Miah, M. Hasibuzzaman and A. Sarker, "Design of a Solar Assisted Autonomous Surveillance Vehicle," 2020 IEEE Region 10 Symposium (TENSYMP), Dhaka, Bangladesh, 2020, pp. 499-503, doi: 10.1109/TENSYMP50017.2020.9230960. keywords: {Batteries;Liquid crystal displays;Surveillance;Solar panels;Roads;Wireless communication;Vehicles;identify location;information received;control system (electronic & wireless);renewable source;NRF module;L298 motor driver;Arduino uno & mega},

Appendix A

ESP-32 Cam script

```
#include "esp_camera.h"
      #include <WiFi.h>

//
// WARNING!!! PSRAM IC required for UXGA resolution and high JPEG quality
//           Ensure ESP32 Wrover Module or other board with PSRAM is selected
//           Partial images will be transmitted if image exceeds buffer size
//

// Select camera model
      #define CAMERA_MODEL_WROVER_KIT // Has PSRAM
//#define CAMERA_MODEL_ESP_EYE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
//#define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
//#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
//#define CAMERA_MODEL_AI_THINKER // Has PSRAM
//#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM

      #include "camera_pins.h"

      const char* ssid = "*********";
      const char* password = "*********";

      void startCameraServer();

      void setup() {
      Serial.begin(115200);
      Serial.setDebugOutput(true);
      Serial.println();

      camera_config_t config;
      config.ledc_channel = LEDC_CHANNEL_0;
      config.ledc_timer = LEDC_TIMER_0;
      config.pin_d0 = Y2_GPIO_NUM;
      config.pin_d1 = Y3_GPIO_NUM;
      config.pin_d2 = Y4_GPIO_NUM;
      config.pin_d3 = Y5_GPIO_NUM;
      config.pin_d4 = Y6_GPIO_NUM;
      config.pin_d5 = Y7_GPIO_NUM;
      config.pin_d6 = Y8_GPIO_NUM;
      config.pin_d7 = Y9_GPIO_NUM;
```

```
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sscb_sda = SIOD_GPIO_NUM;
config.pin_sscb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

// if PSRAM IC present, init with UXGA resolution and higher JPEG quality
//                      for larger pre-allocated frame buffer.
if(psramFound()){
config.frame_size = FRAMESIZE_UXGA;
config.jpeg_quality = 10;
config.fb_count = 2;
} else {
config.frame_size = FRAMESIZE_SVGA;
config.jpeg_quality = 12;
config.fb_count = 1;
}

#if defined(CAMERA_MODEL_ESP_EYE)
pinMode(13, INPUT_PULLUP);
pinMode(14, INPUT_PULLUP);
#endif

// camera init
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
Serial.printf("Camera init failed with error 0x%x", err);
return;
}

sensor_t * s = esp_camera_sensor_get();
// initial sensors are flipped vertically and colors are a bit saturated
if (s->id.PID == OV3660_PID) {
s->set_vflip(s, 1); // flip it back
s->set_brightness(s, 1); // up the brightness just a bit
s->set_saturation(s, -2); // lower the saturation
}
// drop down frame size for higher initial frame rate
s->set_framesize(s, FRAMESIZE_QVGA);

#if                          defined(CAMERA_MODEL_M5STACK_WIDE)                          ||
defined(CAMERA_MODEL_M5STACK_ESP32CAM)
s->set_vflip(s, 1);
s->set_hmirror(s, 1);
#endif

WiFi.begin(ssid, password);
```

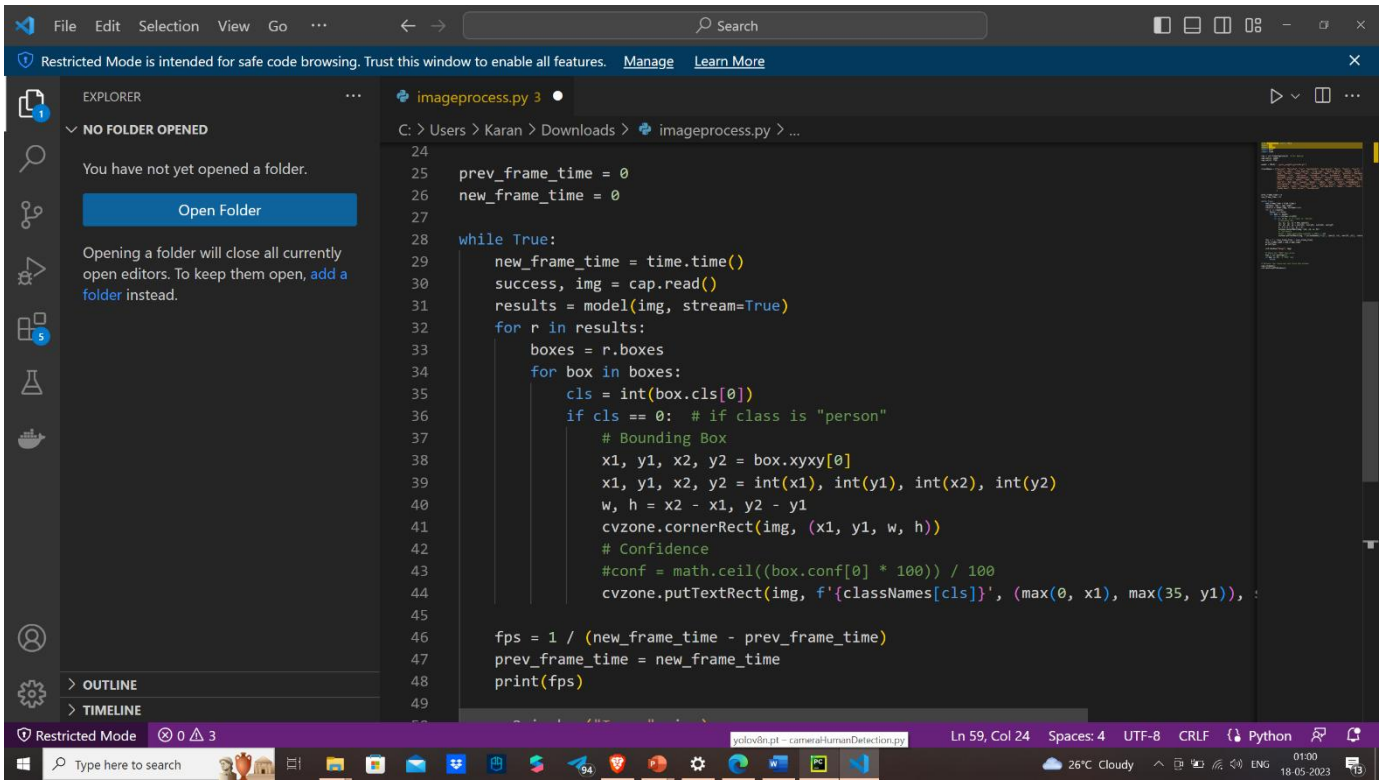
```
while (WiFi.status() != WL_CONNECTED) {
delay(500);
Serial.print(".");
}
Serial.println("");
Serial.println("WiFi connected");

startCameraServer();

Serial.print("Camera Ready! Use 'http://");
Serial.print(WiFi.localIP());
Serial.println("' to connect");
}

void loop() {
// put your main code here, to run repeatedly:
delay(10000);
}
```

Blynk Cloud Server Integration



Blynk Cloud Server Integration

This method utilizes Blynk Protocol and it's the most common and easy-to-use method when you need to send data in real-time.

First you need to do is setup a template with a datastream to configure what type of data your hardware will be sending.

When you have the datastream set, use its Virtual Pin number further.

```
    int sensorVal;

// This function creates the timer object. It's part of Blynk library
    BlynkTimer timer;

    void myTimer()
    {
    // This function describes what will happen with each timer tick
    // e.g. writing sensor value to datastream V5
    Blynk.virtualWrite(V5, sensorVal);
    }

    void setup()
    {
    //Connecting to Blynk Cloud
    Blynk.begin(auth, ssid, pass);

    // Setting interval to send data to Blynk Cloud to 1000ms.
    // It means that data will be sent every second
    timer.setInterval(1000L, myTimer);
    }

    void loop()
    {
    // Reading sensor from hardware analog pin A0
    sensorVal = analogRead(A0);

    // Runs all Blynk stuff
    Blynk.run();

    // runs BlynkTimer
    timer.run();
    }
```

Network Gateway Data Streams

By leveraging JavaScript functionality, the network gateway module dynamically updates and renders these network details in real-time within the admin control application. This provides administrators with comprehensive visibility into the network environment and empowers them to monitor, manage, and troubleshoot network-related issues effectively.

Code:

```
function pingHost() {
   const host = document.getElementById("ping-host").value;
   const xhr = new XMLHttpRequest();
      xhr.open("GET", "https://" + host, true);
      xhr.timeout = 10000;
      xhr.onload = function ()
      {
      const pingTime = new Date() - startTime;
      const pingResult = document.getElementById("ping-result");
         pingResult.innerHTML = "Ping time: " + pingTime + "ms";
         pingResult.classList.add("slide-in");
      };
      xhr.onerror = function ()
      {
         document.getElementById("ping-result").innerHTML = "Ping failed";
      };
   const startTime = new Date();
      xhr.send();
   }

   function measureNetworkSpeed()
   {
      const startTime = performance.now();
      const fileSize = 1; // 5MB file size for testing
      fetch('https://example.com/file.bin', {
            method: 'GET',
         cache: 'no-cache',
         headers: {
      'Content-Type': 'application/octet-stream',
            'Content-Length': `${fileSize}`
   }
      }).then(response => {
        const endTime = performance.now();
        const duration = (endTime - startTime) / 1000; // convert to seconds
        const speedMbps = (fileSize / duration / 1024 / 1024 * 8).toFixed(2); // calculate Mbps
      document.getElementById("download-speed").innerHTML = speedMbps + " Mbps";
      }).catch(error => {
         console.log(error);
      });
```

```
    }


    function checkNetworkStatus() {
    const statusIcon = document.getElementById("network-status-icon");
    const statusText = document.getElementById("network-status-text");
        if (navigator.onLine) {
            statusIcon.innerHTML = "&#x2714";
            statusIcon.style.color = "#1abc9c";
            statusText.innerHTML = "Online";
        } else {
            statusIcon.innerHTML = "&#x2716;";
            statusIcon.style.color = "#e74c3c";
            statusText.innerHTML = "Offline";
        }
    }



    // IP
    var xhr = new XMLHttpRequest();
xhr.open('GET', 'https://api.ipify.org', true);
        xhr.onload = function() {
        if (this.status === 200) {
        document.getElementById('ip').innerHTML = this.responseText;
        }
        };
        xhr.send();

// ISP
        var xhr2 = new XMLHttpRequest();
        xhr2.open('GET', 'https://api.ipify.org/?format=json', true);
        xhr2.onload = function() {
        if (this.status === 200) {
        var data = JSON.parse(this.responseText);
        var ip = data.ip;
        var xhr3 = new XMLHttpRequest();
        xhr3.open('GET', 'https://ipwhois.app/json/' + ip, true);
        xhr3.onload = function() {
        if (this.status === 200) {
        var data2 = JSON.parse(this.responseText);
        document.getElementById('isp').innerHTML = data2.isp;
        }
        };
        xhr3.send();
        }
        };
        xhr2.send();


// Router
        document.getElementById('router').innerHTML = location.hostname;
```

```
// DNS
	var dns = 'example.com';
	var xhr = new XMLHttpRequest();
	xhr.open('GET', 'https://dns.google/resolve?name=' + dns, true);
	xhr.onload = function() {
	if (this.status === 200) {
	var data = JSON.parse(this.responseText);
	var ipAddress = data.Answer[0].data;
	console.log('DNS: ' + ipAddress);
	document.getElementById('dns').innerHTML = ipAddress;
	}
	};
	xhr.send();


//speed
// Speed test
	function measureNetworkSpeed() {
	const startTime = Date.now();
	const fileSize = 1024 * 1024 * 10; // 10 MB
	const url = "https://speed.hetzner.de/10GB.bin"; // replace with your preferred file URL
	const xhr = new XMLHttpRequest();
	xhr.open("GET", url + "?r=" + Math.random(), true);
	xhr.responseType = "arraybuffer";

	xhr.onload = function(e) {
	const endTime = Date.now();
	const duration = (endTime - startTime) / 1000;
	const bitsLoaded = fileSize * 8;
	const speedBps = bitsLoaded / duration;
	const speedMbps = (speedBps / (1024 * 1024)).toFixed(2);
	console.log("Download speed: " + speedMbps + " Mbps");
	document.getElementById("speed").innerHTML = speedMbps;
	};

	xhr.onerror = function(e) {
	console.error("Error fetching file", e);
	};


	xhr.send();
	}
```

Esp8266 Honeypot Code

Code :

```
// NAPT example released to public domain

#if LWIP_FEATURES && !LWIP_IPV6

    #define HAVE_NETDUMP 0

    #ifndef STASSID
    #define STASSID "Your_Wifi_Network_Name" // set the SSID (name) of the Wi-Fi network the ESP8266 will connect to for internet
    #define STAPSK  "Your_Wifi_Network_Password" // set the password of the Wi-Fi network the ESP8266 will connect to for internet
    #define NEWSSID  "honeypot_Wifi_Name" // set the name (SSID) of the Wi-Fi network the ESP8266 will create
    #define NEWPASS  "honeypot_Wifi_Password" // set the password of the Wi-Fi network the ESP8266 will create
    #endif

    #include <ESP8266WiFi.h>
    #include <lwip/napt.h>
    #include <lwip/dns.h>
    #include <dhcpserver.h>
    #include <ESPCanary.h>

    #define NAPT 1000
    #define NAPT_PORT 10

    #if HAVE_NETDUMP

    #include <NetDump.h>

    void dump(int netif_idx, const char* data, size_t len, int out, int success) {
    (void)success;
    Serial.print(out ? F("out ") : F(" in "));
    Serial.printf("%d ", netif_idx);

    // optional filter example: if (netDump_is_ARP(data))
    {
    netDump(Serial, data, len);
    //netDumpHex(Serial, data, len);
    }
    }
    #endif

    String canary = "Your_Canarytoken_URL";  //grab FREE web bug/URL tokens at http://canarytokens.org
```

```
String ftp_user = "admin";    //if you replace this with "%" it will accept ANY username
String ftp_pass = "password"; //if you replace this with "%" it will accept ANY password
bool append_ip = false;       //if you are using a canary token, leave this as false
String append_char = "?";     //if you are using a canary token, this doesn't matter
//if you are using your own webhook,with a bunch of GET
//parameters then you would want this to be "&" so the IP
//address becomes the final GET parameter

FtpServer ftpSrv;   //set #define FTP_DEBUG in ESPCanary.h to see ftp verbose on serial


void setup() {
Serial.begin(115200);
Serial.printf("\n\nNAPT Range extender\n");
Serial.printf("Heap on start: %d\n", ESP.getFreeHeap());

#if HAVE_NETDUMP
phy_capture = dump;
#endif

// first, connect to STA so we can get a proper local DNS server
WiFi.mode(WIFI_STA);
WiFi.begin(STASSID, STAPSK);
while (WiFi.status() != WL_CONNECTED) {
Serial.print('.');
delay(500);
}
Serial.printf("\nSTA: %s (dns: %s / %s)\n",
WiFi.localIP().toString().c_str(),
WiFi.dnsIP(0).toString().c_str(),
WiFi.dnsIP(1).toString().c_str());

// give DNS servers to AP side
dhcps_set_dns(0, WiFi.dnsIP(0));
dhcps_set_dns(1, WiFi.dnsIP(1));

WiFi.softAPConfig(  // enable AP, with android-compatible google domain
IPAddress(172, 217, 28, 254),
IPAddress(172, 217, 28, 254),
IPAddress(255, 255, 255, 0));
WiFi.softAP(NEWSSID, NEWPASS);
Serial.printf("AP: %s\n", WiFi.softAPIP().toString().c_str());

Serial.printf("Heap before: %d\n", ESP.getFreeHeap());
err_t ret = ip_napt_init(NAPT, NAPT_PORT);
Serial.printf("ip_napt_init(%d,%d): ret=%d (OK=%d)\n", NAPT, NAPT_PORT, (int)ret,
(int)ERR_OK);
if (ret == ERR_OK) {
ret = ip_napt_enable_no(SOFTAP_IF, 1);
Serial.printf("ip_napt_enable_no(SOFTAP_IF): ret=%d (OK=%d)\n", (int)ret, (int)ERR_OK);
if (ret == ERR_OK) {
```

```
    Serial.printf("WiFi Network '%s' with password '%s' is now NATed behind '%s'\n", NEWSSID, NEWPASS, STASSID);
    }
    }
    Serial.printf("Heap after napt init: %d\n", ESP.getFreeHeap());
    if (ret != ERR_OK) {
    Serial.printf("NAPT initialization failed\n");
    }

    /////FTP Setup, ensure SPIFFS is started before ftp;  /////////
    if (SPIFFS.begin()) {
    Serial.println("SPIFFS opened!");
    ftpSrv.begin(ftp_user,ftp_pass,canary,append_ip,append_char);      //username, password for ftp. set ports in ESPCanary.h  (default 21, 50009 for PASV)
    }
    }

    #else

    #error "NAPT not supported in this configuration"

    void setup() {
    Serial.begin(115200);
    Serial.printf("\n\nNAPT not supported in this configuration\n");
    }

    #endif

    void loop() {
    ftpSrv.handleFTP();
```

Appendix B

SPECIFICATIONS

Specification of Arduino UNO

Microcontroller	ATmega168
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14
Analog Input Pins	6
DC Current per I/O Pin	40mA
DC Current for 3.3V Pin	50mA
Flash Memory	32 KB (ATmega328)
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Specification of Raspberry Pi

Microcontroller	ATMEGA328P/ATMEGA158
Operating Voltage	5V
Input Voltage	7-12 V
Digital I/O Pins	40
PWM	6 out of 14 digital pins
Max Current Rating	40mA
USB	A
Analog Pins	8
Flash Memory	1Gb
SRAM	1Gb - 2Gb
Crystal Oscillator	16 MHz
EEPROM	512bytes or 1KB
USART	Yes

Specification of ESP -32

Operating Voltage	2.2V to 3.6V
GPIO	36 Ports
ADC	14 Ports
DAC	2 Ports
Flash Memory	16 Mbytes
SRAM	250 Kbytes
Clock Speed	Up to 240 MHz
Wi-Fi	2.4 GHz
Sleep Current	2.5 Micro Ampere