Smart Hybrid IoT Home Security System Using ESP32 and AWS

Professor - Dr. Guy Tel-Zur

Ori Dagan (207698788)
Department of Electrical and Computer Engineering
Ben Gurion University of the Negev
Beer Sheva, Israel
oridaga@post.bgu.ac.il

Elad Elmakias (208195370)
Department of Electrical and Computer Engineering
Ben Gurion University of the Negev
Beer Sheva, Israel
eladelma@post.bgu.ac.il

***Abstract*—The rapid expansion of the Internet of Things (IoT) has facilitated the development of sophisticated residential security systems; however, traditional Cloud only streaming solutions often face significant challenges regarding "bandwidth bottlenecks" and high operational costs. This project proposes a cost effective, intelligent home security system utilizing a hybrid Edge-Cloud architecture to optimize data transmission and analytical accuracy. The system deploys an ESP32-CAM as an edge node to handle localized motion detection and image capture, subsequently offloading computationally intensive facial recognition tasks to Amazon Rekognition through a serverless AWS pipeline. By implementing an event-driven "Snap and Upload" methodology triggered by Passive Infrared (PIR) sensors, the architecture effectively "filters" the data stream at the edge, reducing cloud-bound transmission compared to continuous 24/7 surveillance models.**

Keywords—IoT, Edge Computing, Cloud Intelligence, Amazon Rekognition, Hybrid Architecture, Face Recognition, ESP32-CAM.

I. Introduction

Traditional home security primarily relies on mechanical key-lock systems, which are increasingly vulnerable to duplication and physical bypass in the modern digital landscape. While the Internet of Things has introduced digital surveillance as a solution, the current preference for centralized Cloud processing creates significant operational challenges. The reliance on "100% Cloud" solutions often fails to meet the low latency and high bandwidth demands of real-time security applications.

Contemporary surveillance infrastructures are increasingly burdened by a "data deluge" where continuous video streaming results in severe bandwidth bottlenecks. Transmitting high-resolution video frames (typically 25KB each) can saturate networks, causing the processing time to fall behind what is needed for real-time events. Furthermore, the high transmission fees and computational resource costs associated with cloud providers can make continuous monitoring business solutions impractical.

To overcome these limitations, this project proposes an intelligent "Snap and Upload" security framework based on a hybrid Edge-Cloud architecture. By utilizing an ESP32-CAM to perform localized "stateless" filtering, the system only transmits data during specific motion events, which reduces cloud-bound data transmission by up to 98% compared to continuous streaming and effectively mitigates network congestion [2]. While existing local-only systems utilize traditional algorithms like Local Binary Patterns (LBP) which offer approximately 80% accuracy [1], our architecture offloads complex analytical tasks to Amazon Rekognition. This leverages cloud-scale deep learning to achieve superior identification precision that resource constrained Edge devices cannot maintain locally.

Additionally, the system employs a PIR sensor to wake the hardware from a low-power "hibernation" state only when motion is detected, ensuring minimal energy consumption and addressing the limited battery life typical of remote Edge machines. By integrating localized Edge-triggering with Cloud-native intelligence, the proposed system provides a scalable, low-cost solution that bypasses the latency and throughput issues inherent in traditional 24/7 surveillance architectures.

We employed Amazon SNS to facilitate user communication. This service enables the system to dispatch an email notification immediately upon detecting an intruder.

II. Background and Related Work

The evolution of home security has transitioned from traditional mechanical locking mechanisms to increasingly sophisticated electronic and digital solutions. Historically, security was reliant on physical keys, which presented significant risks regarding unauthorized duplication and the potential for loss. The introduction of Radio Frequency Identification (RFID), biometric scanners, and password based keypads addressed some of these vulnerabilities by providing unique digital signatures for access control. However, these early digital systems often operated as isolated units lacking remote accessibility and real-time monitoring capabilities. The rise of IoT has redefined this landscape by enabling the integration of various sensors and high-performance microcontrollers with web and mobile applications, allowing homeowners to monitor and manage their security from any location

In the domain of IoT enabled security, Pawar et al. [1] proposed a system utilizing a Raspberry Pi 3 Model B+ as a central processing unit, integrated with PIR and ultrasonic sensors for motion detection and distance measurement. Their methodology employs the Local Binary Pattern (LBP) algorithm for local face recognition to automate door locking and unlocking. The system includes a notification framework that sends SMS and email alerts containing intruder images to the homeowner. Experimental results from the study demonstrated a facial recognition accuracy of approximately 80% under real-time testing conditions.

A primary advantage of the approach by Pawar et al. is its energy efficiency, achieved by using PIR sensors to maintain the system in a low power state until motion is detected. However, a significant disadvantage is the 80% accuracy limitation of the LBP algorithm, which is susceptible to errors in varied lighting or if facial features are partially obstructed. Furthermore, performing all

imaging locally on a Raspberry Pi is more computationally intensive and requires significantly more power compared to smaller microcontrollers. This limits the system's intelligence to the device's local hardware. Our proposed solution addresses these trade-offs by utilizing the *ESP32-CAM*, a more power efficient node, and offloading the task to *Amazon Rekognition*. This transition from local LBP to cloud-native deep learning increases identification accuracy to near perfect rates while significantly reducing the local computational load.

The technical challenges of such hybrid architectures are explored in depth by Silva et al. [2], who investigated the performance trade-offs between Cloud only and hybrid Edge-Cloud processing for data streams. Using real life datasets from CCTV footage and taxi logs, they measured how network bandwidth and data volume impact latency and throughput. Their research identified that "bandwidth bottlenecks" are a primary failure point for Cloud only systems, particularly when streaming high resolution video data which can cause processing to fall behind real-time.

The core benefit of the findings by Silva et al. is the demonstration that localized pre-processing or "filtering" at the edge can reduce the volume of data transmitted to the cloud by up to 98%. This hybrid model ensures stable latency even under fluctuating network connections. Nevertheless, they acknowledge the Edge devices are limited by restricted resources and battery life, making operations such as high level behavioral analysis unsuitable for Edge only deployment. Our project effectively bridges this gap by adopting the exact hybrid model recommended in their study. We utilize the ESP32-CAM for simple, stateless motion detection at the Edge to eliminate the bandwidth jam, but leverage the power of AWS for the "complex analytics" of facial recognition. This ensures our system maintains the low data transmission benefits found in Silva's research while providing the high-level intelligence that Pawar's local system lacked.

III. System Architecture

This section details the high-level design and structural components of the IoT Security System. The architecture is divided into three distinct layers: the Physical Perception Layer (Sensors & Peripherals), the Edge Processing Layer (Dual-Core Microcontrollers), and the Cloud Decision Layer (Serverless Backend).

The system utilizes a split-architecture approach, decoupling low-power sensor management from high-bandwidth image processing. This design choice was necessitated by the hardware constraints of the ESP32-CAM module, which utilizes nearly all available GPIO pins for the camera interface, leaving insufficient I/O for external security peripherals.

To resolve this, the system employs two specialized microcontroller units communicating via a dedicated UART serial bridge:

1. The Peripheral Unit (ESP32 DevKit V1): Acts as the central controller, managing state, user input, and environmental sensing.
2. The Vision Unit (ESP32-S3): Functions as a dedicated co-processor for image capture and encrypted Wi-Fi transmission.

The Peripheral Unit interfaces with the physical world through a specific set of modules designed to provide both security triggers and user feedback:

- Input/Output Interface: A 4x3 Matrix Keypad provides a traditional fallback authentication method (PIN code). Visual system status (Armed, Entry Delay, Alarm) is communicated via a 16x2 LCD Display driven by the I2C protocol to minimize pin usage.
- Sensor Grid: The system monitors two primary trigger zones:
    - Perimeter Access**:** A magnetic reed switch (simulated via button on GPIO 4) detects the physical state of the door.
    - Volumetric Presence: A PIR motion sensor (simulated on GPIO 2) provides instant threat detection within the secured area.
- Active Deterrence: A siren (driven by GPIO 23) serves as the immediate physical response mechanism to unauthorized access.

The backend architecture is built upon a Serverless Computing Model using Amazon Web Services (AWS). This architecture eliminates the need for always-on servers, reducing cost and maintenance while allowing the system to scale automatically from zero to thousands of requests.

The cloud architecture consists of four microservices:

- Compute (AWS Lambda): The core logic engine. It acts as an event-driven function that executes only when an image is received. This function orchestrates the flow between storage, analysis, and notification services.
- Storage (Amazon S3): Used as a dual-purpose repository. One bucket (incoming-captures) acts as a temporary buffer for evidence storage, while a second secure bucket (authorized-faces) serves as the immutable reference database for known biometric identities.
- Cognitive Analysis (Amazon Rekognition): An AI-as-a-Service (AIaaS) component that provides the biometric verification engine. It abstracts complex deep learning models, allowing the system to perform facial comparison and liveness detection without requiring local GPU hardware.
- Messaging (Amazon SNS): A publish/subscribe service that decouples the detection logic from the alerting mechanism. It handles the reliable delivery of critical security alerts (SMS/Email) to the end-user, independent of the camera's connectivity status.

This hybrid Edge-Cloud architecture ensures that critical sensing happens locally with zero latency, while computationally intensive biometric verification is offloaded to scalable cloud infrastructure.

IV. Software Implementations

The software architecture of the system is designed to prioritize reliability, responsiveness, and energy efficiency. It is implemented across three distinct execution environments: the real-time firmware on the Peripheral Unit, the high-throughput firmware on the Vision Unit, and the serverless backend logic on AWS Lambda.

To ensure deterministic behavior and prevent invalid system states, the Peripheral Unit operates on a Finite State Machine (FSM) model. This approach replaces simple procedural loops with a formal state transition diagram, guaranteeing that the system reacts predictably to sensor inputs based on its current context.

The system initializes in an IDLE state, where it awaits user input via the keypad or enters a low-power mode. Upon entering a valid master password, the system transitions to an EXIT_DELAY state, providing the user a ten-second window to vacate the premises before the sensors become active. Once this timer expires, the system enters the ARMED state.

In the ARMED state, sensor inputs trigger divergent behaviors. A trigger from the motion sensor (simulated via GPIO 2) is treated as an immediate high-priority threat, causing an instantaneous transition to the ALARM state. Conversely, a trigger from the perimeter door sensor (GPIO 4) initiates an ENTRY_DELAY state, providing a fifteen-second grace period for authorized users to disarm the system. If the user fails to provide a valid password or biometric verification within this window, the system escalates to the ALARM state, activating the siren and notifying the cloud backend.

Due to the decoupled hardware architecture, synchronized operation between the Peripheral Unit and the Vision Unit is achieved through a custom asynchronous serial protocol. The units communicate via a UART bridge operating at 9600 baud, utilizing a lightweight byte-oriented command set to minimize latency.

When a sensor trigger occurs, the Peripheral Unit transmits a command character **'C'** (Check), instructing the Vision Unit to wake from standby and capture an image. The Peripheral Unit then continues its FSM execution without blocking, maintaining responsiveness to keypad inputs while awaiting a reply.

The Vision Unit processes the request and returns a single status byte: 'K' (Known) if the cloud analysis confirms an authorized identity, or 'A' (Alert) if an intruder is detected. This asynchronous "fire-and-forget" mechanism ensures that high-latency network operations do not impede the critical real-time performance of the alarm system.

The serverless backend logic is encapsulated within a Python-based AWS Lambda function. This function serves as the central orchestration engine, executing a linear pipeline for every incoming request.

Upon receiving an HTTP POST request, the function first performs a security validation by verifying a custom *x-auth-token* header. This prevents unauthorized entities from submitting malicious data to the system. Once authenticated, the function decodes the Base64-encoded image payload and persistently stores the raw capture in an Amazon S3 bucket for forensic evidence.

The core biometric analysis is performed by invoking the Amazon Rekognition API. The incoming image is compared against a pre-indexed collection of authorized faces stored in a secure S3 bucket. The API returns a similarity confidence score; if this score exceeds the defined threshold of 85%, the function returns a success status to the hardware unit. If the face is unrecognized, the function triggers Amazon SNS to dispatch an immediate email alert to the system owner and returns a forbidden status, signaling the hardware to sound the alarm.

To extend operational longevity in battery-backed scenarios, the system implements a power management strategy centered on the ESP32's Light Sleep mode.

The Peripheral Unit monitors a system inactivity timer. If the system remains in the IDLE or ARMED state with no sensor activity for twenty seconds, the processor enters a low-power suspension state, disabling the CPU clock and non-essential peripherals while maintaining RAM retention.

Crucially, the system configures the Real-Time Clock (RTC) controller to monitor specific GPIO pins associated with the keypad, door sensor, and motion sensor. A state change on any of these lines generates a hardware interrupt, instantly waking the processor. This "sleep-until-interrupt" architecture allows the system to reduce power consumption significantly without compromising security responsiveness, as the wake-up latency is negligible.

V. Results and Discussion

The implementation of the Biometric IoT Security System successfully demonstrated the feasibility of decoupling sensor management from high-bandwidth biometric processing. The dual-microcontroller architecture proved robust in handling concurrent tasks, maintaining responsive user interactions on the Peripheral Unit while the Vision Unit executed network-intensive operations.

A critical performance metric for any real-time security system is the end-to-end latency—defined as the time elapsed between a sensor trigger and the receipt of an alert by the end-user. During testing, the system consistently performed within acceptable parameters for a home automation proof-of-concept, though network conditions significantly influenced total response time.
As seen in the following table.

The latency analysis indicates that the primary bottleneck is not the local processing on the ESP32 hardware, but rather the establishment of the secure HTTPS connection and the subsequent cloud processing. The Cold Start time of the AWS Lambda function also contributed

variably to the total delay. However, for a battery-operated video verification system, a delay of approximately five seconds is comparable to many commercial smart doorbell products, confirming the viability of the architecture.

TABLE I. LATENCY ANALYSIS

Operation Phase	**Average Latency (ms)**	**Notes / Bottleneck**
Sensor Trigger to Wake-Up	<1	UART Serial Overhead
Image Capture & Encoding	200	ESP32-S3 Hardware Limitations
WiFi Connection and SSL	1500	Network Signal Strength
Cloud Upload (HTTPS POST)	800	Payload Size (~30kB)
AWS Lambda Cold Start	600	Python Runtime Initialization
Rekognition Analysis	500	API Processing Time
SNS Alert Delivery	1500	Carrier Network Latency
Total System Latency	~5000	End-To-End Response Time

The facial recognition component, powered by Amazon Rekognition, demonstrated high accuracy when operating with a confidence threshold of 85%. This threshold effectively balanced the risk of False Acceptance (allowing an intruder) against False Rejection (locking out an owner), provided that the subject faced the camera directly under adequate lighting.

From an economic perspective, the serverless architecture proved highly cost-effective. Unlike traditional security monitoring services that require monthly subscription fees, this system incurs costs only during active events. The pricing model for AWS Rekognition and Lambda is volume-based, meaning a typical home usage pattern resulting in a few dozen triggers per month would result in operational costs measured in mere cents. This makes the solution scalable and accessible for low-cost residential deployment.

As a proof of concept, the current system identifies several avenues for optimization to transition from a prototype to a commercial-grade product.

Hardware Scaling: While the ESP32-S3 is capable, replacing the Vision Unit with a Single Board Computer (SBC) such as a Raspberry Pi would significantly enhance optical capabilities. An SBC would allow for the integration of higher-fidelity camera modules and local image pre-processing. Specifically, an algorithm could be implemented to capture a burst of images and locally select the frame with the highest clarity and best lighting before transmitting it to the cloud, thereby reducing false negatives due to motion blur.

Network Resilience: The current reliance on local Wi-Fi introduces a single point of failure; if the local network is severed, the system loses its intelligence. Future iterations would benefit from a dual-uplink strategy, integrating a cellular modem (LTE-M or NB-IoT) to provide a fallback communication channel independent of the home internet connection.

Security and Provisioning: Currently, network credentials are hardcoded into the firmware, which presents a security risk and limits deployment flexibility. A production version would implement a SoftAP provisioning mode, allowing the user to securely configure Wi-Fi credentials via a smartphone app during initial setup. Furthermore, physical hardening is required; the sensitive electronics must be housed in a tamper-resistant enclosure to protect against physical disablement.

Power Redundancy: To ensure continuous operation during power outages, a dedicated battery backup circuit with seamless failover capability should be integrated. Coupled with the existing deep-sleep software logic, this would ensure the system remains vigilant even in the absence of mains power.

References

[1] S. Pawar, V. Kithani, S. Ahuja, and S. Sahu, “Smart Home Security Using IoT and Face Recognition,” presented at the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), 2018, pp. 1–6. doi: 10.1109/ICCUBEA.2018.8697695.

[2] P. Silva, A. Costan, and G. Antoniu, “Investigating Edge vs. Cloud Computing Trade-offs for Stream Processing,” presented at the 2019 IEEE International Conference on Big Data (Big Data), 2019, pp. 469–474. doi: 10.1109/BigData47090.2019.9006139.

[3] https://github.com/Shmerlard/iot_project