Risk-Centric Threat Analysis of an MQTT-Based Disaster Relief System Using PASTA.

Hussain, Ahlam, Hussam, Rowida

***Abstract* - A disaster relief communication system is analyzed using the Process for Attack Simulation and Threat Analysis (PASTA) framework to identify and mitigate security threats. The system provides a communication system in post-disaster scenarios with no stable pre-existing telecommunication infrastructures. The system leverages MQTT messaging through an Eclipse Mosquitto broker and volunteer-operated drones. We model the system's threats across PASTA's seven stages, aligning business objectives with technical scope, decomposing the application architecture, and enumerating threat scenarios, including network eavesdropping, message spoofing, denial-of-service attacks, and device compromise. Using standardized libraries (CAPEC attack patterns, CWE weaknesses, CVE vulnerabilities, and MITRE ATT&CK tactics), we assess vulnerabilities in each component and simulate potential attack paths. A risk and impact analysis uses qualitative and quantitative measures to prioritize risks. The results underscore the importance of proactive threat modeling in ensuring the reliability of disaster response systems even under adversarial conditions.**

***Index Terms* - Message Queuing Telemetry Transport (MQTT), Process for Attack Simulation and Threat Analysis (PASTA), Disaster Relief, MITRE**

I. Introduction

The growing number and severity of natural disasters expose major weaknesses in traditional emergency communication systems. Events like earthquakes, floods, and hurricanes can knock out cell towers, internet infrastructure, and power grids, resulting in survivors being cut off from help and making it difficult for emergency teams to coordinate. In these high-pressure situations, having a communication system that's quick to deploy, flexible, and doesn't rely on existing infrastructure can be a lifesaver. This report explores a disaster response system that addresses this issue using MQTT (Message Queuing Telemetry Transport), a lightweight protocol that works well even with limited bandwidth and unstable connections. The system uses drones operated by volunteers and cloud-connected components to create a decentralized communication network.

These drones act as mobile communication hubs, using MQTT and an Eclipse Mosquitto broker to connect users and rescue units in real time. This setup is fast to deploy and scalable, ideal for areas where traditional infrastructure is down. However, it comes with significant cybersecurity risks, mainly because MQTT doesn't have built-in security features. A strong threat modeling approach is needed to keep the system secure and functional, even under hostile conditions.

A. MQTT Security Challenge.

When disasters strike, they often wipe out telecom infrastructure, cutting off victims and rescue teams. This communication blackout can delay aid, block vital information, and increase the risk of casualties. Protocols like MQTT are used. Its publish-subscribe model makes it perfect for low-bandwidth, unstable networks commonly found in disaster zones.

While MQTT is efficient, it wasn't designed with security in mind. It lacks basic protections like encryption, authentication, or access control, making it vulnerable to attack, tampering, unauthorized access, denial-of-service, and spying. These weaknesses can have devastating consequences in emergencies, possibly causing mission-critical failures and the loss of lives.

B. Technical background.

The system relies on the Eclipse Mosquitto MQTT [1] broker to handle message traffic between drones, command software, and cloud services. This setup allows devices to communicate asynchronously—messages are sent to specific "topics," and only devices subscribed to those topics receive them. This makes the system scalable and efficient, especially in rough conditions with weak connectivity.

The architecture includes several components: drones with MQTT that send mission data from victims and to rescue teams, a command center app for monitoring, and cloud infrastructure for data storage and coordination. While this design adds reliability and fault tolerance, it also increases the system's vulnerability, particularly at the broker level and over unprotected wireless links. Each

part of the system could become an entry point for hackers trying to disrupt communications or steal information.

C. PASTA Framework for Threat Modelling.

The report uses the PASTA [2] (Process for Attack Simulation and Threat Analysis) framework to meet the demanding security needs of this disaster communication system. PASTA is a risk-focused threat modeling method that aligns cybersecurity analysis with business and technical goals. Unlike traditional methods like STRIDE, PASTA simulates real-world attacker behavior through seven structured steps, making it especially useful for systems in unstable or high-risk environments.

These seven stages start by defining objectives and end by simulating attacks and evaluating risks. Each step builds on the last, helping teams trace how vulnerabilities can lead to real-world damage. PASTA's structured and adaptable process helps prioritize the most critical threats and tailor protections to the system's design, making it ideal for communication systems in dangerous or unpredictable settings.

D. Objectives.

This report evaluates the cybersecurity strength of a drone-based MQTT communication system by applying the PASTA framework. Because this system supports life-saving efforts during emergencies, it focuses on ensuring it stays online, keeps data accurate, and protects sensitive information. Using the PASTA framework, the report will guide the threat analysis from defining goals to mapping out attacks and recommending solutions.

By grounding this analysis in the real needs of disaster response, the goal is to help build secure, reliable communication tools that improve coordination and situational awareness when it matters most.

II. System Architecture & Overview

System overview

The disaster relief system is prepared and created to restore communication in urgent situations when the usual infrastructure, such as cell towers and internet service, is unavailable. It adds edge devices like smartphones, drones, local servers running Mosquitto (an MQTT broker), and optional cloud components to assemble a strong, safe, and flexible communication network. This part outlines every section of the architecture and how they perform together to ensure life-saving information runs where it must be, even under challenging conditions.

A. Purpose and Rationale

During Hurricane Katrina, one of the most critical failures occurred within the communication infrastructure. Cell towers and public radio systems failed due to the loss of power and the physical damage caused to telecommunication assets. Emergency responders could not communicate with rescue teams such as Firefighters, EMS, or police. All remaining backup systems were overwhelmed, leading to areas not having contact for days[3].

Similarly, in the aftermath of the 2010 Haiti earthquake, the destruction of infrastructure led to the collapse of many GM towers that were situated on rooftops and crumbled during the earthquake. This and the damage to state-owned telecom facilities severely disrupted telecommunications [4].

The MQTT-based disaster relief system is ideal for acting as an alternative communication network due to its lightweight messaging protocol, which is perfect for low bandwidth. MQTT allows users to transmit relatively small messages quickly and has fallback mechanisms in case of routing issues.

Its main aim is to provide credible communication when usual options are unviable. The system permits victims and responders to communicate over a local wireless network that may work separately from the internet infrastructure. As an alternative, it utilizes a lightweight messaging protocol called MQTT, which is ideal for unstable and low-bandwidth domains.

B. Key components and architecture

This system covers many central and vital parts that work together in a chain to deliver assistance requests from victims to urgent personnel. At the edge, victims utilize smartphones to transmit distress messages through an app. These messages are transmitted wirelessly to the volunteer drones flying near the victim. The drones connect to a local MQTT broker that forwards the messages to a leading center, where the staff responsible for responding can view them.

The general data flow will look like this:

1. The phone of the victim → MQTT message → Drone relay
2. Drone → Local broker
3. Broker → Lead center application
4. If available: Local broker → Cloud server for support and backup

Communication Workflow Example

- Not a\Active Workflow:
  - Victim transmits MQTT message → Drone pass on to the broker.
  - Broker path to field command application.
  - Dashboard always updates with the position, place and urgency.
- Active Workflow (after it gets connected):
  - Information send to cloud broker
  - Encrypted and preserved in cloud SQL
  - MQTT broker bridges to cloud
  - National groups observe dashboards remotely

Following these steps will permit the system to work with or without the Internet and scale quickly when new parts are added.

The system has a lot of significant parts running together:

- Smartphones with a particular app to transmit distress messages.
- Drones flying on top of the area gather and transmit these messages.
- A leading center in the cloud to process, preserve, and display the data.
- A dashboard for rescue teams to look at the messages and make decisions.

The system's architecture has the following main components:

1. Victim Device (User App)

Victims utilizes a smartphone app to issue aid messages (subject: aid/request) through Wi-Fi. Every single message has a metadata like timestamp, GPS coordinates, and severity.

2. Volunteer Drones

Drones fly above the disaster zone, taking actions as mobile Wi-Fi hotspots and MQTT clients. They depend on messages that are from victims to the MQTT broker and publish situational data (images, coordinates) to subjects such as drone/data.

3. MQTT Broker (Mosquitto)

The MQTT broker gets all of the messages. It executes subject-based message routing, applying TLS encryption, and utilizes Access Control Lists (ACLs) to stop unofficial publishing/subscribing.

Subjects include:

- Aid/request
- rescue/instructions
- drone/data
- system/logs

4. Field Command Center Application

The main control console subscribes to all of the MQTT subjects and envisions them on a dashboard. It encrypts data prior to local storage and helps bidirectional communication with drones or users.

5. Cloud Broker and Database

Once the internet is accessible, the MQTT broker crosses to a cloud broker (on AWS). This sends picked subjects for focused access. The cloud side covers:

- SQL database for constant storage
- Network segmentation for the cloud protection
- Certificate-based authentication for the trusted connections

C. MQTT Broker

The MQTT broker is placed in the heart of the system and works on a local device such as a rough laptop, Raspberry Pi, or even a pfSense firewall. This broker utilizes Mosquitto, a known open-source tool that performs similarly to a main post office: all messages pass through it, and it ensures everyone gets to the correct area or spot.

The broker takes charge of subjects such as:

- Assist/request messages from victims
- Rescue/status updates from the staff who reply

While Mosquitto is lightweight and flexible, its default configurations are inherently insecure. It fails to verify connection attempts, which is a substantial threat in a real-world deployment. Additionally, it doesn't utilize encryption (TLS) [1] unless specifically configured. If the communication channels are left open, attackers might eavesdrop, tamper with authentic transmitting MQTT messages, or even spoof illegitimate traffic. This system structure looks into these risks and introduces security layers that the report will discuss in upcoming parts.

The system is divided into zones with strict security controls:

- Zone 1: Local Wi-Fi Network (unlock)

Victim devices utilize unlocked Wi-Fi to get to connect to the drone. TLS is enforced between the victim application and MQTT broker.

- Zone 2: Drone-to-Broker Link

Drones get connected through utilizing MQTT on top of TLS and validate with device certificates to stop and block impersonation.

- Zone 3: Broker-to-Cloud Bridge

MQTT messages are sent just on top of a secure TLS channel. The cloud broker confirms all connections through utilizing X.509 certificates.

- Zone 4: SQL Data Stores

The local and the cloud databases encrypt important data and enforce role-based entry.

D. Volunteer Drones

Volunteer drones play a critical role in the disaster relief architecture by acting as airborne intermediary components for message transmission. The volunteer drones fly on top of the affected places to perform tasks such as message relaying. These drones gather distress signals transmitted from smartphones on the ground. Their mobility allows them to reach areas inaccessible by ground teams or where standard communication infrastructure is unreachable.

The drones in this system process two tasks:

1. Message Relay: This involves capturing distress signals from the victims' disaster app and then transmitting them wirelessly to an MQTT broker.
2. Telemetry Collection: Gather situational data such as GPS coordinates or other readings that might assist responders.

Drones are mobile Wi-Fi hotspots that can dynamically extend the system's coverage area. Their

ability to reposition themselves based on real-time mission updates grants the system flexibility that traditional static telecommunication protocols cannot provide.

Drones connect to the MQTT broker wirelessly and transmit data they collect, such as GPS coordinates or severity levels.. All data received can be stored in the event of connectivity being lost. Once a stable connection is reestablished, the drones forward the collected information, ensuring that critical data is not lost.

However, these drones introduce physical and cybersecurity risks. Due to their low operational altitude, they are susceptible to tampering and unauthorized access by attackers. Drone-to-broke communications must be secured using lightweight encryption protocols to counteract those specific attack attempts. Security monitoring solutions such as Wazuh [5] can be deployed to monitor logs and detect anomalous activity.

Since these drones operate beyond the field personnel, they are susceptible to physical capture (From power depletion, crashes, or intentional recovery efforts). If adversaries gain access to the drone hardware, it may result in severe consequences. The impacts of physical drone retrieval include the ability to extract information about the network architecture, credentials, and locally stored data.

Adding to this, to block the drones (particularly the untrusted or spoofed drones) from flooding the system with unreal or too much messages, we apply:

- Rate Limiting:

Every drone has a limited number of messages it could issue each time window. This blocks flooding even if the drone comes to term.

- Behavioral Filtering:

The broker observes message frequency and drops traffic that override configured thresholds.

- Authentication Filters:

Just the devices with the right certificates may be issued to MQTT. Any kind of drone that does not have a recognized cert is rejected directly.

- Victim Location Validation:

The field application cross-references GPS locations with the drone telemetry to find unsure or not possible locations (wrong emergency signals).

These mitigations make sure continuity of service, correct and right victim tracking, and defense against message spoofing.

E. Field command center application

The field command center application serves as the intelligence hub of the disaster relief system. Operating on hardware by first responders, the application receives all distress signals relayed by the drones via the MQTT broker. It processes incoming data and displays the information on the dashboard. This dashboard provides an overview of the affected area, showing locations of victims and prioritizing requests based on the severity level that was coupled with them.

This dashboard includes:

- A real-time map presents assistance requests, drone positions, and places
- A dashboard for filtering via urgency
- Tools for transmitting instructions back to drones or victims

This dashboard provides responders with full visibility of the situation. Since each message comes via the broker, this app becomes the central control point for making decisions. Quick response is important during emergencies, and this app aids the responders in taking action quickly and correctly based on recent data.

F. Cloud Integration & Database

The disaster relief system is designed to function in disconnected environments but also includes cloud integration for extended coordination. This cloud component becomes active when internet connectivity is established via appropriate telecommunication.

1. Purpose of Cloud integration

Cloud integration supports the following goals:

- Centralized database
- Long-term storage for post-incident analysis and auditing
- Redundancy and fault tolerance, ensuring data is not lost in the case of hardware failure.

2. Broker Bridging Mechanism

When internet access is restored, the local MQTT broker establishes a bridge connection with the cloud, which allows MQTT topics to be forwarded. The cloud could be hosted on secure infrastructures such as AWD, Azure, or private servers, which serve as a central node that supports analytics and coordination.

3. Database Design

Two layers of databases are employed in the system:

Local database (Field database)

- Lightweight SQL.
- Stores distress messages and severity.
- Operates entirely offline for faster operations for first responders.

Cloud database:

- Persistent storage.
- Receives synchronized data from the broker.
- Supports historical data analytics.

The Local Database:

- Lightweight SQL

  Preserves:

  - Device_id
  - Timestamp location
  - Message_id
  - Urgency
- Runs even if not available, enables the local search and dashboard filters

Cloud SQL Database:

- Hosted on the AWS RDS or Azure MySQL
- Preserves and keeps synchronized MQTT subjects from the field. It supports:
  - Disaster recovery
  - Historical data analytics
  - Multi-location coordination

Backup and Redundancy

- Local information is hide until a cloud sync happens
- The backups are encrypted and in a new version
- The sync intervals are depend on the connectivity state

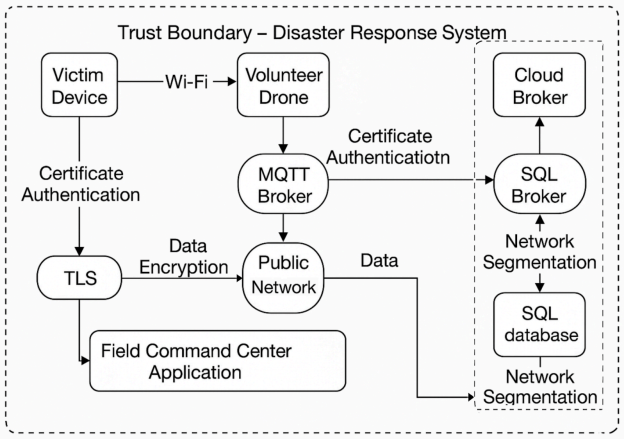

Figure 1. UML disaster response system

III. Threat Modelling Overview

PASTA (Process for Attack Simulation and Threat Analysis) is a risk-focused threat modeling approach that connects organizational goals with security needs through methodical evaluation and simulated attacks [3]. It features a seven-stage sequence that steers analysts from defining the system's intent to identifying threats and evaluating potential risks. Unlike check-style methods such as Stride, PASTA focuses on the attack flow that attackers might take to exploit the system. Such an approach allows organizations to align their defenses with operational priorities. Its adaptable nature makes it well-suited for evolving systems across the different phases of development and post-deployment.

A. Stage 1: Define Objectives

Stage one of PASTA lays the foundation by clarifying business and security goals. Business objectives clarify the intended outcome of the completed system, while security goals ensure that business objectives are protected through security principles such as confidentiality, integrity, and availability. This stage intertwines security into the overall development of a system rather than treating security as an isolated requirement.

B. Stage 2: Define Technical Scope

In the second stage, the technical scope of the system is outlined by identifying all essential components, technologies, and any third-party services involved. This includes databases, servers, APIs (Application Programming Interfaces), networks, and software applications. Defining what falls within or outside this scope helps ensure thorough analysis and reduces the risk of missing key system elements.

A vast scope can weaken the system's overall security. A larger attack surface increases vulnerability, and security may become less effective as mitigation resources are spread too thin across non-critical areas. The technical scope should be as focused as possible to manage risks efficiently while meeting the system's core business requirements.

C. Stage 3: Application Decomposition

The application decomposition stage involves breaking down the system into functional elements, such as user roles, workflows, dataflows, trust boundaries, and external actors. This step helps clarify the system's operation by showcasing how data moves across components and where trust assumptions are made. Decomposition highlights the identification of potential weaknesses or misuse scenarios.

Table 1: Model components & their definitions

Model Components	**Definition**
Application Components	Software elements within a system that process, transmit, or manage data. E.g., APIs, Services, modules. These are the building blocks that provide function to an application
External Actors	Entities outside the system boundary that interact with the system. E.g., users, third-party systems, or external services. They can initiate or respond to system processes
Assets	System components that

	need protection from attacks. E.g., User data and credentials.
Stores	Data repositories store information, such as databases, File systems, and cloud storage. Stores can hold sensitive or critical data.
Trust Boundaries	Boundaries where trust or security control changes, such as between a user device and a server. Crossing these trust boundaries often requires some type of authentication.

Tools like Data Flow Diagrams (DFDs) are standard in this stage because they visually represent how information traverses the system. These diagrams show external actors, processes, data stores, trust boundaries, and data flows. The visual representation of these components enables analysts to map use cases to specific components and assess which areas of the system are exposed to risk. Clear operational models of the system prepare the groundwork for identifying threats and designing attack flow models to be used in later stages.

D. Stage 4: Threat Analysis

In stage four, the focus is on identifying and categorizing potential threats based on the analysis done in previous stages. With a comprehensive picture of system components, use cases, and data flows, analysts can evaluate how an attacker might exploit the system. This stage involves creating an attack scenario landscape—a representation of threat actors, techniques, and attack vectors relevant to the system under analysis.

Threats are classified using frameworks such as STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege), which assist analysts in categorizing the nature and intent of different attacks. Resources such as MITRE ATT&CK (Adversarial Tactics, Techniques, & Common Knowledge), CAPEC (Common Attack Pattern Enumeration & Classification), and CWE (Common Weakness Enumeration) provide insight into adversarial behaviours, tactics, and known vulnerabilities. These well-documented references shed light on potential attack patterns, which can provide analysts further insight into mitigating them.

E. Stage 5: Vulnerability & Weakness Analysis

Stage five focuses on identifying vulnerabilities within the system that could be exploited by threats outlined in the fourth stage (Threat Analysis). These vulnerabilities exist in the system's design, implementation, configuration, or deployment. This phase correlates known weaknesses to the components identified in the third stage (Technical Scope), creating a clearer picture of risk exposure.

Resources such as CWE and CVE map vulnerabilities to specific components or system behaviours. Analysts consider other configuration flaws, insecure default settings, and insufficient authentication controls. In addition to known issues, misuse cases are also evaluated because they can reveal overlooked logic flaws or privilege escalation paths.

This stage focuses on understanding where the system is weak and how these weaknesses align with threat behaviour. It is used as the foundation for the attack modelling phase.

F. Stage 6: Attack modelling

Stage six utilizes the insights gathered from stage five (Vulnerability & Weakness analysis) to simulate how a realistic attack could unfold within the system. This phase involves creating attack scenarios that demonstrate the steps an attacker might take to exploit system vulnerabilities. The goal is to visualize an attack's progression, identify potential entry points, and evaluate how the system would respond to each stage of an attack.

Attack flow diagrams and attack trees illustrate the relationships between system weaknesses, threat vectors, and attacker goals. Each branch or node in an attack tree represents a method or decision point an attacker could use. Attack flows provide actions based on real tactics, techniques, and procedures (TTPs).

Resources such as MITRE ATT&CK, CAPEC, CWE, and CVE support the modelling process by linking previously identified system weaknesses to known exploitation patterns and their corresponding defense strategies in the case of MITRE D3FEND. This structured visualization highlights where existing controls can be bypassed. By simulating attacks using these tools, organizations can assess their systems' resilience and prioritize certain defenses.

Table 2: Databases used in threat modelling

Name	**Description**	**Use Case in Threat Modelling**
CVE	**Common Vulnerabilities and Exposures** is a public list of known cybersecurity vulnerabilities. [6]	Identifies specific vulnerabilities in a system component to assess the risk.
CVSS	**Common Vulnerability Scoring System** is a standard method for rating the severity of vulnerabilities. [7]	Helps prioritize threats based on their respective impact and exploitability.
CWE	**Common Weakness Enumeratio**n is a list of standard software and	Guides the identification of architectural / code-level flaws.

	hardware weakness types. [8]	
CWSS	**Common Weakness Scoring System** is a framework for scoring software weaknesses based on risk. [9]	Helps evaluate the severity of weaknesses identified through the use of CWE
CAPEC	**Common Attack Pattern Enumeration and Classification** is a catalog of known attack patterns used by adversaries.[10]	Assists in modeling potential attack vectors and adversary behaviour.
CCE	**Common Configuration Enumeration** is a system used to identify system configuration issues. [11]	Helps assess and manage insecure system configs during design and setup.
MITRE ATT&CK	**MITRE ATT&CK (Adversarial Tactics, Techniques and Common Knowledge)** is an accessible knowledge base of adversary tactics and techniques based on real-life examples. [12]	Helps simulate realistic attack paths by mapping known adversary behaviours to system components and processes. Assists in validating attack trees during the attack modelling phase.
MITRE D3FEND	**MITRE D3FEND (Detection, Denial, and Disruption Framework Empowering Network Defense)** is a knowledge graph of cybersecurity countermeasures that maps to ATT&CK techniques and mitigates adversary behaviours [13]	Assists in selecting technical mitigations for identified threats mapped to specific ATT&CK tactics. Useful in post-mitigation design.

G. Risk and Impact

The final PASTA stage involves evaluating the likelihood and impact of the attack scenarios developed in stage 6. The objective is to assess each threat's potential consequences regarding how it affects business and security objectives and prioritize risks accordingly. Multiple methods are used to evaluate this.

I.Risk matrix

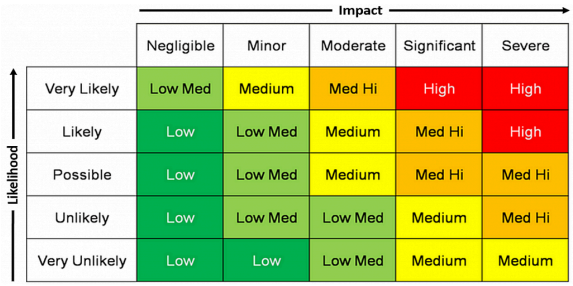

Figure 2. A risk matrix with different impact and likelihood categories [13]

A risk Matrix is a visual tool for determining a risk's priority by plotting the likelihood of a threat occurring against the impact it could have. By categorizing threats into separate levels, the matrix supports more informed planning and resource allocations.

II.CVSS & CWSS

CVSS (Common Vulnerability Scoring System) and CWSS (Common Weakness Scoring System) are more data-driven approaches. They employ a scoring system to measure the system's threat exposure.

The CVSS framework provides a numerical score between 0 and 10, directly reflecting how severe a vulnerability is. It considers both the exploitability and impact of an exploit while considering its:

- Attack Vector.
- Attack Complexity.
- Privileges Required & User Interaction.
- Impacts on Confidentiality, Integrity, Availability.

The base score provides a view of severity and helps system engineers identify which vulnerabilities pose the most significant risks and should be prioritized.

The CWSS framework provides a way to rank software weaknesses similar to those found in the CWE catalog. By assessing the likelihood and context in which a weakness could be exploited, a numerical score between 0 and 100 is output. The CWSS score is calculated based on:

- Base Finding Metrics (Severity of the weakness)
- Attack Surface Metrics
- Environmental Metrics (How relevant is the weakness to the system's mission?

In the context of PASTA, CWSS helps assess threats linked to insecure architecture, weak input validation, or poor access controls, which could all be exploitable despite not being active vulnerabilities.

III. Residual Risk Analysis

In threat modelling, eliminating all the risks within a system is often impossible, so a system engineer focuses on reducing the risk to an acceptable level. This process is known as “Residual Risk Analysis.”

In the context of PASTA, residual risk is assessed after simulating attacks, mapping threats to vulnerabilities, and applying appropriate mitigation tactics. Original threat scenarios are reviewed, and their probability of success is estimated. Then, they are compared to a review of the same threat scenario, but after controls are implemented,

Residual risk analysis is the final validation step in the PASTA framework. It confirms whether the secure architecture is resilient enough under realistic threat conditions.

$$RR = (Vuln \times Threat \times Impact) \div Mitgations$$

This stage correlates risks with the system's defined business and security objectives, ensuring that defensive resources are allocated efficiently by first targeting high-impact, high-likelihood risks. This phase results in mitigation strategies and supports strategic decision-making by understanding where the system is most vulnerable and what consequences may arise from those vulnerabilities.

IV. Threat modelling applied

A. Stage 1: Objectives

a. Business Objectives

Table 3 outlines the key requirements and objectives for ensuring the safe, secure, and efficient operation of the MQTT-based disaster Relief System.

Table 3: Model components & their definitions

Req ID	Business Security Objectives	Objct. Ref	System & Compliance Requirements
MQTT -1	Keep the communication during disasters (High Availability)	1.1, 1.2	1. Configure the MQTT broker with TLS and authentication to prevent and block hijacked connections. 2. Ensure the broker is hosted on resilient, low-power devices with battery backup. 3. Utilize redundant brokers to hold the failover and traffic balancing.
MQTT -2	Safeguard information and data in transit between the victims, drones, and the command center	2.1, 2.2	1. Apply TLS on MQTT messages. 2. Encrypted channels are required between the drone and broker and the broker and cloud. 3. Use firewall rules to prevent and stop unapproved ports and protocols.
MQTT -3	Find and Stop Denial-of-Service Attacks	3.1, 3.2	1. Put in the rate-limiting and connection quotas on MQTT brokers. 2. Observe and watch the traffic utilizing Wazuh or Suricata for anomalies. 3. Harden MQTT configs via limiting the max-inflight messages and enabling DoS alerts.
MQTT -4	Keeps the message's integrity and trustworthiness	4.1, 4.2	1. Digitally sign MQTT messages (if practical). 2. Support topic structure and apply input sanitization. 3. Block and stop message replay by implementing timestamp validation.
MQTT -5	Block and stop unauthorized entry to the MQTT broker and drone clients	5.1, 5.2	1. Enable the username/password authentication on all MQTT endpoints. 2. Give a special client ID with ACLs, the Access Control Lists. 3. Periodically rotate the broker credentials or the tokens.
MQTT -6	Minimize operational disruption because of the device tampering	6.1, 6.2	1. Put an anti-tamper detection on drones and the field hardware. 2. Disable the auto-connect for the unauthenticated customers. 3. Perform drone log backup and alert when the physical has been accessed.

The primary mission is to save lives and accelerate disaster recovery by providing an emergency communication service when conventional channels fail. This includes enabling stranded civilians to call for help, coordinating rescue team efforts, and improving situational awareness through drone surveillance data. The system must be highly available and reliable under disaster conditions; downtime or failure could result in unmet rescue needs.

This system plays a mission-critical role in bridging communication gaps in crisis zones. For instance, a collapsed bridge or flooded road reported through the system helps rescue teams reroute operations and avoid danger zones. If any information is delayed or lost, it can cost lives or delay any medical aid.

Although speed and uptime are the primary business goals, there’s also a need to uphold ethical responsibility. Respecting the privacy of disaster victims, such as shielding their identities and locations, helps preserve trust in the response effort and prevents secondary harm such as exploitation or misinformation.

In addition, business objectives include interoperability and scalability. The system should support future expansion and function across various geographical or regulatory domains, requiring a flexible yet secure infrastructure.

b. Security Objectives (CIA Triad)

To meet the system's business goals, the system must uphold core cybersecurity principles: confidentiality, integrity, and availability. It should particularly focus on maintaining service uptime and the accuracy of critical information.

High availability is essential during crises. The system must withstand unexpected surges in traffic and intentional denial-of-service attempts. Backup power options and redundant broker setups can help maintain communication, even when one component is

compromised. Without this level of resilience, distress calls may go unanswered.

Integrity is equally essential. Unauthorized message alterations, such as faking an SOS call or forging drone sensor data, could mislead rescue efforts. Verifying the authenticity of messages, whether through digital signatures or secure app-level authentication, helps ensure that responders are acting on real, untampered information.

Confidentiality remains essential, but it may be less prioritized during emergencies. The system should still prevent casual interception of sensitive data by applying encryption for wireless communications and limiting access to stored records. While usability is critical in the field, lightweight security options like pre-shared keys or simplified credential systems can still offer a reasonable level of protection without burdening the user.

Finally, any implemented security measures must support, not hinder, response operations. Controls should be as invisible as possible to end users. This system should also meet any cybersecurity standards that local or international disaster response agencies set.

c. Mapping Objectives to Threat Impact

Clearly defined business and security objectives are a foundation for identifying and evaluating the system's vulnerabilities. Any threat that compromises availability or integrity should be treated as a high-priority risk, particularly in a disaster response system where interruptions may have life-threatening consequences.

For example, system downtime during an emergency can prevent distress messages from reaching rescue personnel, leading to delays in assistance or even loss of life. Likewise, if a message is tampered with or spoofed, responders may act on false information, potentially misdirecting critical resources.

By mapping objectives to potential threat impacts, it becomes possible to assess how each security breach aligns with or opposes the system's mission. A denial-of-service attack that disables the MQTT broker contradicts the objective of continuous communication. Similarly, injecting false data into drone telemetry violates the requirement for accurate situational awareness.

A brief business impact assessment supports this alignment by estimating the consequences of specific attacks. These include operational delays, privacy violations, loss of public trust, and safety risks. This context is essential for later stages of threat modeling, where threat severity must be prioritized and mitigation strategies evaluated for their effectiveness at preserving mission-critical functionality.

B. Stage 2: Define Technical Scope

a. In-scope components

The technical scope of the disaster relief system includes all core components for enabling secure communication during emergencies. These components form the basis of the threat model and represent the assets most likely to be targeted by adversaries. One key element is the client device, typically a smartphone or IoT-enabled unit running the emergency application. These devices serve as endpoints for sending or receiving MQTT messages. Their local storage, app configuration, GPS input, and wireless interfaces are all within scope, given the risk of malware, unauthorized access, or spoofing attempts by adversaries.

Another critical component is the MQTT broker, implemented via the Mosquitto platform and typically deployed on a local Raspberry Pi or laptop. This broker facilitates message routing and may use persistence to store session data. The broker's software configuration, authentication settings, data directories, network interfaces, and host operating system are in scope. Any tools or services used for administration, such as the password file or ACL configurations, also represent areas where attackers could attempt privilege escalation or manipulation.

Ad-hoc network infrastructure, specifically the wireless environment created by responders or drones, is also included in the scope. This environment contains SSID security, WPA2 encryption, mesh networking, and any packet forwarding or relay services drones use. Given the openness of wireless communication, threats like packet injection, rogue access point creation, or jamming must be considered. If not adequately protected, communication between drones, clients, and the broker is vulnerable to eavesdropping and tampering.

Also, drones themselves are considered part of the technical scope. These mobile nodes typically run MQTT clients and possess various sensors and communication hardware. Attackers could exploit vulnerabilities in drone firmware, wireless interfaces, or MQTT configurations to hijack devices or access sensitive data. Because drones may store credentials or cache messages, the loss or compromise of a single drone could expose broader system assets. Physical security concerns are also relevant concerning cyber impact, such as the theft of a drone that stores broker credentials.

The field command system, including laptops or local servers that process and visualize data, is a high-value component within the scope. This system may include the dashboard application, local databases, data visualization software, and operating system services. Threats may emerge through physical access, open ports, or network exposure, particularly if adversaries gain proximity to the disaster zone.

Lastly, cloud infrastructure used for backup and remote access is in scope. This includes a cloud MQTT broker and an associated database that may store message logs or drone telemetry. The bridge connection between local and cloud brokers introduces additional risk, especially if unsecured credentials or misconfigured TLS channels are exploited. Cloud configurations, API access

tokens, and environment variables that store authentication data are also relevant in threat modeling.

Table 4: Define Technical Scope

ID	Component Name	Business Goals ID	Justification
1	Client Device (Victim/Responder)	MQTT-1, MQTT-2, MQTT-5	Take action as the endpoint for emergency communication. They are vulnerable to malware, information sniffing, and spoofing with no secure configuration or encryption.
2	MQTT Broker (Mosquitto)	MQTT-1, MQTT-3, MQTT-4, MQTT-5	Central routing hub for all of the MQTT messages. Broker misconfigurations or weak credentials could permit privilege escalation, DoS, or data leakage.
3	Wireless Network Infrastructure	MQTT-1, MQTT-3	It incorporates the drone Wi-Fi mesh and field-deployed SSIDs. Susceptible to rogue APs, jamming, and packet injection if not properly secured
4	Drone Device (MQTT Client)	MQTT-1, MQTT-4, MQTT-5	The mobile node is handing over data. It could be physically compromised or exploited via the firmware, credentials, or cached messages.
5	Field Command System	MQTT-1, MQTT-3, MQTT-6	The system processes and visualizes arriving messages. It is at risk of attack through network display, unencrypted services, or even local access.
6	Cloud Infrastructure	MQTT-1, MQTT-2, MQTT-4,	It stores the backup and helps the broker bridge. Cloud misconfigurations and exposed keys generate attack opportunities
		MQTT-6	over public networks.

b. Trust boundaries

The architecture includes multiple trust boundaries that separate secure zones from potentially hostile or less-controlled environments. One major trust boundary exists at the local wireless network where smartphones, drones, and the MQTT broker communicate. Devices connected to this network are generally trusted once authenticated, but entry to the network depends on Wi-Fi security settings. An external attacker can easily breach the trust boundary within range if the network uses a weak password or lacks encryption.

Another trust boundary exists between the local broker and the cloud infrastructure. This connection traverses the public internet or satellite links and must be treated as untrusted. Any data crossing this boundary requires proper authentication, encrypted channels, and verification mechanisms to prevent man-in-the-middle attacks or data injection. The broker must validate the source and structure of any message from the cloud to avoid backdoor compromise.

Physical boundaries also form a key part of the threat landscape. Attackers must generally be near the disaster area to join the wireless network. This limits access but does not eliminate it, especially if the password is shared widely among volunteers. Devices such as drones and the broker host must be protected physically, as compromising them can allow bypass of application-layer defenses. In addition, logical trust boundaries exist at software process levels. For example, the MQTT broker process is trusted but must handle all incoming client messages as untrusted input.

c. Out of scope considerations

Certain elements are deliberately excluded from the technical scope of this threat model to maintain focus and manage complexity. Operational aspects of disaster response, like how rescue teams conduct field operations or navigate the environment, are not considered unless they intersect directly with the system's digital components. While these operations rely on communication infrastructure, the threat model does not address physical logistics or coordination strategies outside the network.

General physical sabotage, like physically destroying a device or cutting power to the server, is considered out of scope unless it results in a cyber-related impact. For example, the theft of a victim’s phone is not analyzed unless the attacker uses it to access MQTT services. Likewise, long-term data governance, retention policies, or regional regulatory compliance are not explored in detail, though privacy best practices are acknowledged in the objectives.

External systems that do not integrate with the MQTT-based communication framework are also

excluded. These include unrelated public internet services or national emergency systems that do not interface with the broker or cloud database. Hardware tampering beyond drones, such as dismantling laptops or routers manually, is treated as a physical threat outside the scope of this cyber-focused analysis.

By clearly defining what is and is not within scope, the threat modeling process is kept focused on critical assets and communications pathways. This precision ensures that the system's attack surface is neither underestimated nor diluted with unrelated concerns. The next section builds on this scoped architecture to analyze how data flows, where weaknesses may exist, and what specific attack scenarios must be considered.

C. Stage 3: Application Decomposition

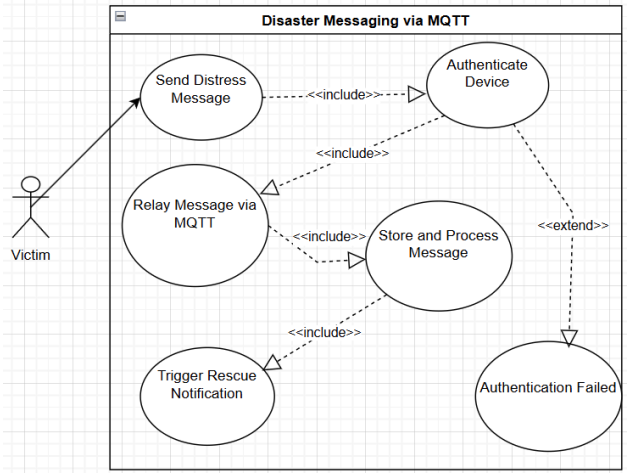

Figure 3. Use Case 1 - Submit Distress Message and Enable Rescue Coordination In

In Use Case 1, Figure 2, the mobile application's victim interface allows the user to log into the system and send an SMS distress message. Starting with the Stream Capture Phase, the user types a message that has a basic structure including GPS coordinates, level of urgency, and other optional free text phrases. The message is sent using a local wireless network (WiFi) to a drone, which acts as a mobile MQTT client. Ensure that the Authenticate Device use case gets executed preemptively before the message is accepted. This prevents unauthorized devices from interacting with the communication network and guarantees that only authenticated devices can access the system. Otherwise, the system moves to the Authentication Failed extension for further actions, captures the attempt for auditing, and thus terminates any potential exploitable situations that may arise from these attacks.

After completing the authentication process, the drone sends the message in the Relay Message via the MQTT step. In this phase, information is given to the local MQTT broker, which holds and directs the message to the central system. Later, the central broker proceeds to execute the Store and Process Message step, in which the message is stored and set up for analysis. During this period, the command center application analyzes the message and gains structured parts like location, time, and threat level. Subsequently, the system invokes the Trigger Rescue Notification use case, which modifies the operational dashboard and notifies first responders. This procedure ensures that messages in need of attention undergo verification, are securely sent, and their contents are efficiently understood, enabling fast and well-coordinated actions in case of a disaster.

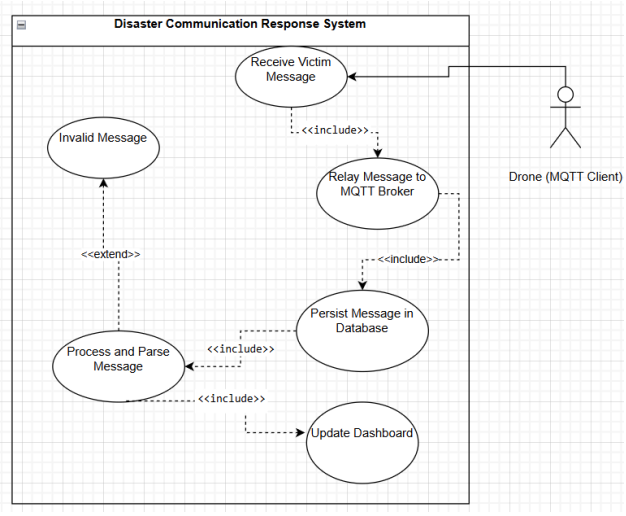

Figure 4. Use Case 2 – Drone Message Relay and Field Command Response

In Use Case 2, Figure 4, the drone operates as a mobile MQTT client, relaying distress messages from victims to the command infrastructure. The process begins with the Receive Victim Message step, where the drone captures a message sent over a local wireless network by a user device. This message typically includes emergency content such as GPS location and urgency level. Immediately after reception, the drone executes the Relay Message to MQTT Broker step. This ensures that the message is handed off to the local MQTT broker, which is configured to receive, queue, and manage MQTT topics relevant to the disaster response workflow.

Once the broker receives the message, it triggers the Persist Message in Database use case, storing the information in a cloud or locally hosted database, depending on connectivity. This guarantees message retention and prepares the data for further processing. Following storage, the Process and Parse Message step is invoked by the command center application. During this phase, the message is analyzed to extract structured data, such as the sender's location, the message type, and severity indicators. The application then updates the system's operational dashboard through the Update Dashboard use case, providing first responders and analysts with real-time situational awareness. If the message fails schema validation or contains corrupted data, the system triggers the Invalid Message extension to log and isolate the entry without affecting live operations. This flow ensures data integrity and operational visibility, allowing field responders to take swift, informed action

based on trusted input relayed by drones.

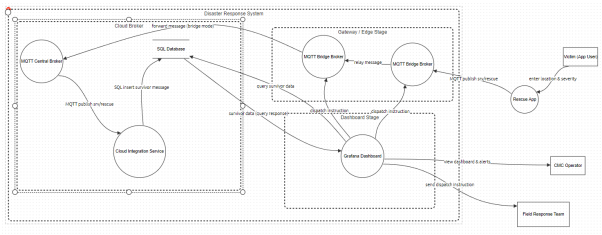

Figure 5. Disaster Response System Data Flow Diagram (DFD) for Victim Communication and Rescue Coordination

The DFD version of the diagram focuses on the subsystems of the disaster response systems like Client Stage, Gateway/Edge Stage, Cloud Broker Stage, and Dashboard Stage relative to information pertaining to reliability and timeliness in capturing messages for communication. With the Client Stage as the starting point, the victim's activity includes utilizing the Rescue App which permits transmission of structured messages containing GPS tags, free text fields, and levels of severity in MQTT protocol format. These messages are sent over local Wi-Fi and ad-hoc networks to their subnet drones.

Once the message is published, the drone that captures the message acts as an MQTT Bridge Broker in Gateway/Edge Stage. In appliances without direct links to the Central Broker, the drone applies an interdrone bridge to forward the message to another drone. Whichever the broker's visibility at one given time, the message waits defers to the cloud. Through the bridge mode of MQTT, the drone passes the message to the Central Broker which is through internet connection. During the passage, the message is encapsulated in a TLS tunnel that guarantees security, and the drone ensures integrity and confidentiality of the message.

In the Cloud Broker Phase, the internal message reconsolidation takes place at the Cloud Integration service interface with the MQTT Central Broker Cloud Integration Service. This service then exfiltrates critical message payload fields along with other relevant information and places it into a SQL Database for storage in order to facilitate advanced future analysis and visualization.

From the SQL Database, the Grafana Dashboard retrieves the pertinent data during the Dashboard Phase. The Dashboard displays incident data in real-time like receiving signals from distress GPS beacons and their intensity categorization. This is crucial for the CMC Operator in order to grasp the situation and adjust his actions accordingly. Simultaneously, dashboard activity and user interactions are stored in the Log Database and Audit Trail database for incident traceability and auditing purposes.

In this flow, the CMC Operator interacts with the Field Response Team for the coordination of rescue actions on the system's risk classification and data-packed map. The DFD shows the data flow and functional dependencies through multiple security layers and how victim data is captured and securely archived, visualized, interacted with, and managed during actual disasters.

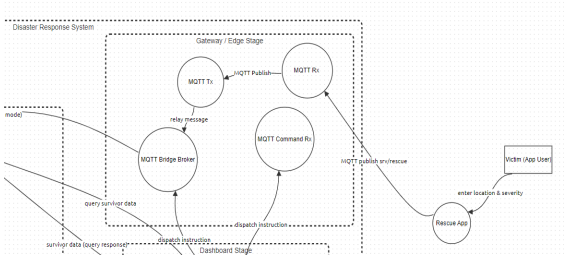

Figure 6. Decomposed Drone Data Flow – Gateway/Edge Stage

The composed perspective of the Gateway/Edge Stage highlights the architecture's operational subsystems within the volunteer drone system. After the Rescue App sends a distress call, it is first managed by the MQTT Rx unit, which receives incoming publish requests on subscribed topics. This unit guarantees reception of data sent via Wi-Fi or mesh network payloads.

Next, the incoming Command Rx listens for Command Center issued dispatch commands, control messages, or route updates. After validation along with message filtering, victim information alongside command replies are relayed to the MQTT Bridge Broker. The Bridge Broker is responsible for sending secured information to an adjacent broker or the cloud, depending on current connectivity.

The outgoing messages will pass through the various subscribed services: other drones on location or cloud-hosted brokers. The data is finally supplied to the MQTT Tx unit, which handles outgoing MQTT messages. This twiddle exchange method promotes file transfer availability through unmanned operation in the absence of direct access, all while ensuring secure directional information flow through interlocks.

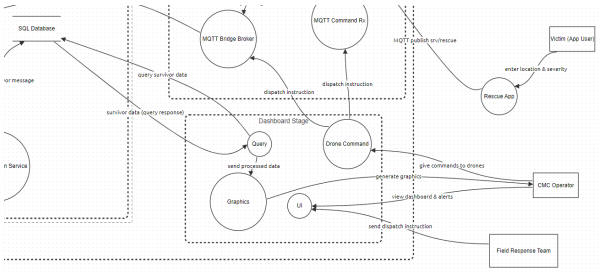

Figure 7. Decomposed Dashboard Stage – Grafana Architecture

The decomposed view of the Dashboard Stage focuses on execution order for visualization-related data retrieval and query functions. Capture of survivor messages in the SQL Database invokes the Query module to fetch and extract filtered entries that are relevant within the scope of a specific time window, a certain location, or the urgency of the incident.

The dataset Graphic Engine receives includes drone telemetry and SOS clustering points with risk silhouetted overlays that mark stricken areas. The CMC Operator can observe the lower subordinate level graphics provided by the graphics engine in the form of user interface of the system.

The responding/cache drone is executed through the MQTT Command format issued by the operator. The instruction set is again sent through MQT, Bridge Broker, and directed to the drones or responders pertinent to the classification.

Examination of the decomposition allows visibly refreshing recalibrating TransDragon with realign Grafana used for data streams whereby it collates visualization offering gauges regard readiness level opt engagement functions serving enable responsive measures alert triggering struck situation stratified intervention graph

D. *Stage 4: Threat analysis*

a. Threat actor profiles

The disaster relief system faces a variety of potential adversaries, each with distinct goals and capabilities. One group of concern includes curious bystanders who are individuals with moderate technical skills who are physically present in the disaster area. These actors might attempt to intercept unencrypted communications using tools like Wi-Fi sniffers, hoping to uncover sensitive details like victim locations or identify unattended zones for exploitation.

Another threat is the malicious insider. These are volunteers or responders who already have authorized access to the system and either intentionally misuse their privileges or unintentionally serve as vectors for attacks if their devices become compromised. Insider threats are especially dangerous because they can bypass some authentication layers and access critical functions.

Remote external attackers represent a third profile. These cybercriminals or state-sponsored entities can exploit system weaknesses from afar, targeting cloud interfaces, exposed APIs, or vulnerable devices used by volunteers. Their attacks are often sophisticated and may aim to compromise system integrity or disrupt services.

Hacktivists and terrorist groups also present a risk. These adversaries may not be interested in gaining access to sensitive data, but rather in causing disruption. For example, they might spread misinformation during a crisis or launch denial-of-service attacks to cripple the system's communication flow.

While not malicious, accidental threats like human error or system misconfiguration can have consequences similar to intentional attacks. These incidents can lead to data corruption or exposure of sensitive information.

b. Stride Categories

Applying the STRIDE framework to the system reveals multiple categories of threats. Spoofing is a serious concern, as attackers might impersonate legitimate devices or users to inject false data or gain unauthorized access. For example, a rogue device might present itself as a trusted drone or broker to send misleading information.

Tampering is another high-risk category. Adversaries may alter messages during transmission or manipulate stored data, such as changing a help request's coordinates or modifying drone telemetry. Such actions could mislead responders or delay assistance to victims.

Repudiation involves the denial of actions or events. Without GOOD authentication and logging mechanisms, it becomes difficult to trace malicious activity back to a source. A compromised user might send harmful commands and later deny involvement, undermining system accountability.

Information disclosure risks arise when attackers gain unauthorized access to sensitive data. This can occur through unencrypted wireless transmissions, unsecured cloud storage, or improperly configured broker permissions. Such breaches can expose personal details of victims or operational plans.

Denial of service (DoS) attacks target system availability. Attackers might flood the MQTT broker with bogus requests, jam wireless signals, or exploit vulnerabilities to crash services. These actions can paralyze communication during emergencies.

Elevation of privilege is a concern. So, if attackers can gain higher access than intended by stealing credentials or exploiting software flaws, they could control core system functions, manipulate data, or lock out legitimate users.

c. Mapp to CAPEC, CWE, MITRE Att&ck

To provide structure and validation, threats are mapped to known attack patterns and tactics using standard cybersecurity libraries. CAPEC-151. For example, addresses identity spoofing, which applies to scenarios where an attacker impersonates a drone or MQTT client. CAPEC-94 and CAPEC-158 cover message manipulation and network sniffing, reflecting how an attacker might tamper with messages or gather sensitive data from unencrypted communications.

CAPEC-125 highlights flooding attacks aimed at overwhelming the broker, while CAPEC-148 addresses content spoofing, such as injecting false emergency messages that appear credible. These mappings show how known techniques can be applied to this specific system context. The CWE catalog identifies common software and configuration weaknesses. CWE-311 relates to the lack of encryption for sensitive data, which is a concern if TLS is not enforced on MQTT connections. CWE-798 addresses the risk of hard-coded credentials, which could allow attackers easy entry if discovered. CWE-120 refers to unsafe memory handling, like buffer overflows, which are especially relevant given vulnerabilities historically associated with Mosquitto.

The MITRE ATT&CK framework also provides relevant classifications. T1040 reflects network sniffing

behaviors, T1498 represents denial-of-service techniques like flooding and jamming, and T1078 deals with misuse of valid credentials. T1059 addresses how attackers may use command interpreters if they gain access to brokers or cloud servers. By linking threats in the system to these established models, the analysis remains grounded in real-world attacker behaviors and known exploitation patterns. This alignment supports future stages of the PASTA model, including vulnerability assessment and the design of appropriate countermeasures.

Table 5: Threat Analysis

Category	Description
Threat Actor	1. Curious bystanders: They are individuals with moderate technical knowledge physically present in disaster zones who use tools such as Wi-Fi sniffers to intercept unencrypted MQTT messages for curiosity. 2. Malicious insiders: authorized volunteers or responders who intentionally misuse their access or pose a threat if their devices are compromised. These actors can bypass authentication and directly affect system operations. 3. Remote cybercriminal: exposed to cloud services or unsecured broker ports, who disrupt services and steal any sensitive data. 4. hacktivists or terrorist groups: they are motivated by ideology, these groups may launch misinformation campaigns or DoS attacks to cripple communication. 5. Unintentional actors: Human error, such as misconfiguration or device misuse, can lead to exposure without malicious intent
Attack Vectors	1. Wi-Fi sniffing and packet capture: Unencrypted MQTT traffic can be captured and then analyzed to discover any sensitive credentials. 2. API abuse: Here, the attackers send unauthorized requests to exposed MQTT topics or REST interfaces. 3. denial of service: the flooding MQTT broker or drones with excessive messages causes service crashes or, in extreme cases may delays. 4. Malware on field devices: In this case, a compromised drone or laptop could inject malicious data.
System Vulnerabilities	1. Lack of authentication on the MQTT broker: Accepting stranger clients allows attackers to subscribe freely. 2. Unencrypted MQTT traffic: In this case, without TLS, any data in transit can be intercepted or altered. 3. Weak logging and audit trails: the inability to trace the source of unauthorized actions leads to repudiation threats. 4. No rate limiting: In this case, the broker overload can be easily achieved through flooding attacks because of the lack of control on message frequency or sessions.
Potential Impact	1. Disrupted Emergency Communication: The victim's messages could not reach the dashboard because of the drone or broker failure. 2. Exposure of the Victim Information: Names, locations, and SOS messages could be leaked to the attacker. 3. Responder Misdirection: Tampered or spoofed messages might distract rescue teams from the actual signals.
	4. Public Trust Damage: System downtime or misinformation while the disaster is happening could cause people to lose faith in digital coordination tools.

E. *Stage 5: Vulnerability Analysis*

Table 6: Disaster Response System Asset Vulnerability Assessment

System Component	Vulnerability / Weakness	Threat Mapping	Severity	CWE/CVE
MQTT Broker (Mosquitto)	The default configuration permits unauthenticated client connections.	Spoofing,	High	CWE-306, CVE-2021-34431
MQTT Broker (Mosquitto)	Vulnerability allows denial of service (DoS) attacks via malformed MQTT packet crashes.	Denial of Service	High	CVE-2021-34431, CVE-2021-41039
MQTT Broker (Mosquitto)	TLS encryption will not be performed without a specific manual configuration.	Information Disclosure	High	CWE-319
Drone MQTT Clients	Absence of rate-controlling mechanisms permits overwhelming through continuous publish/subscribe requests.	Denial of Service	High	CWE-400
Drone Message Routing	Security breaches, such as encryption and authentication, are not applied in drone-to-drone MQTT message forwarding.	Tampering, Spoofing	Medium	CWE-319, CWE-306
Dashboard (Grafana)	Absence of multi-factor authentication leads to looser protection enforcement.	Elevation of Privilege	Medium	CWE-284, CVE-2022-21703
SQL Database	Accepting user-supplied messages without validation or cleansing prior to saving.	Tampering	High	CWE-89
Log Database	Guarding against unwanted altered entries is not present; alongside, the	Repudiation	Medium	CWE-778

	ability to track changes of those entries that have not been safeguarded is absent.			

The table above outlines primary weaknesses captured for the MQTT-based disaster relief system. Every weakness has an associated STRIDE threat type as well as an appropriate CWE or CVE mapping. The severity column captures the impact of each flaw on the system. These vulnerabilities will inform the attack modeling activities in the next phase.

F. Stage 6: Attack Modelling

Figure 8. The attack flow of the MQTT-based disaster relief system [14]

The MQTT-based disaster relief system exhibits a wide attack surface showcased by a mulitutde of initial access vectors. Even with certain mitigation strategies in place, several of these vectors remain partially viable. Given the range of the threat landscape, this report will focus on certain impactful attack flows from the larger model. Each of these flows demonstrates how an adversary can pivot from initial access to impactful system compromises. This attack flow showcases the need for multiple mitigation strategies to be in place concurrently to protect the integrity of the system.

a. Attack Scenario 1: DoS

Attack flow: Unauthorized MQTT Client Access → Broker Resource Exhaustion → Message Routing Failure → System-wide Communication Breakdown

This attack flow demonstrates how an adversary can take advantage of weakly authenticated access to the MQTT broker and exploit protocol limitations to flood message queues, resulting in communication failure across the disaster relief system.

Initial access:

The attack begins with the adversary establishing access to the system from one of the following techniques:

- A rogue client joining an open / poorly secured MQTT topic
- A compromised drone or field device with hardcoded credentials
- Exploitation of open ports due to misconfiguration

Broker Resource Exhaustion

Using the authenticated or impersonated connection, the attacker sends a flood of published messages with high loads to widely subscribed topics.. The broker is designed for lightweight delivery, and the increased amount of messages could lead to:

- Memory bloat from message buffering
- Resource exhaustion from holding messages

Communication breakdown

As internal buffers fill, the broker experiences message loss, subscription timeout, or even denies connection requests. Dropped legitimate communications could be:

- Victim SOS messages
- Drone operation commands
- Relay traffic.

Impact

- Loss of availability of the system during operational periods
- Paused operations due to deadlocks in communication
- Risk to victim's life, as field teams fail to receive and act on information.
- Drones may crash or lead to physical security risks if attackers extract stored information

Classifications

- CWE-400: Uncontrolled Resource Consumption
- CWE-693: Protection Mechanism Failure (due to lack of rate limits)
- CAPEC-130: Excessive Allocation
- MITRE ATT&CK Tactic: Impact (T1499 - Endpoint DoS)

b. Attack Scenario 2: Message Tampering via Man-in-the-Middle

Attack flow: Unencrypted MQTT Traffic → Rogue Access point/Wifi Sniffing → Message Interception → Message spoofing

This attack scenario targets the absence of encryption and integrity within the communication system. In the case of a disaster, the increased traffic activity makes it

easier for messages to be tampered with and go unnoticed. These environments make it ideal for adversaries to position themselves as either passive listeners or active intermediaries.

Initial access:

- Close proximity to a drone.
- Rogue drone within a system.
- Lack of message signing.

Message Interception & Tampering:

Without TLS encryption, MQTT messages, which are typically human-readable, are exposed to eavesdropping, allowing attackers to inspect the system and tamper with messages mid transit.

- Memory bloat from message buffering
- Resource exhaustion from holding messages

Impact

- Rescue teams may be sent to false coordinates, which might result in legitimate victims being ignored due to overwritten/dropped messages.
- Field operators waste time and resources verifying false information
- Drones may operate based on false commands.
- Repeated misinformation will result in a loss of trust in the system.
- SQL injection attacks on the system

Classifications

- CWE-319: Clear Text Transmission of Sensitive Information
- CWE-345: Insufficient Verification of Data Authenticity
- CAPEC-94: Man-in-the-Middle Attack
- MITRE ATT&CK Tactic: Credential Access (T1557.002 - Adversary-in-the-Middle: Wireless)
- CWE-89: SQL Injection

c. Attack Scenario 3: Credential Exposure & Lateral Movement

Attack flow: Unencrypted MQTT Traffic → Rogue Access point/Wifi Sniffing → Message Interception → Credential Extraction → Lateral Movement

This attack flow explores how the absence of transport-layer encryption in the communication infrastructure enables adversaries to obtain sensitive credentials & tokens in transit for the goal of escalating their access across the system.

Initial access:

The adversaries passively sniff MQTT messages being transmitted with tools such as Wireshark. With attackers being able to understand the messages due to a lack of encryption, they are able to obtain:

- Credentials.
- Client IDs, Topic names, and session parameters.
- Session tokens or cookies..

Credential Harvesting & Reuse.

With valid credentials and topic structures. Attackers:

- Reconnects with the MQTT broker using the stolen identity
- Gains access to topics that the compromised client had authentication for.
- Subscribes to sensitive feeds.

Lateral movement & Persistent Access:

With valid credentials and topic structures. Attackers:

- Are able to move laterally by identifying other devices on the network using scanning tools.
- May gain access to other services if the credentials are reused.
- Maintain persistence on the broker or implementing rogue clients using valid credentials.

Impact

- The attack surface is expanded.
- Adversaries gain long-term access to the internal system without raising flags on the backend.
- Ability to bridge the devices to a backend system
- Integrity is undermined.

Classifications

- CWE-522: Insufficiently Protected Credentials
- CWE-319: Clear Text Transmission of Sensitive Information
- CAPEC-137: Parameter Injection
- MITRE ATT&CK Tactic: Credential Access (T1557.002 - Adversary-in-the-Middle: Wireless)
- MITRE ATT&CK Tactic: Lateral Movement (T1550.002 - Use Alternate Authentication Material)

G. *Stage 7: Risk & Impact Analysis*

Stage 7 of the PASTA Framework focuses on assessing the overall risk and impact of the system by enumerating the cybersecurity threats associated with the MQTT Disaster relief system. This phase prioritizes threats based on their likelihood and impact to disrupt mission-critical objectives such as communication, data integrity, and system resilience by building upon the threat and vulnerability data gathered in earlier stages. This structured analysis starts the development of mitigation strategies, which are detailed in the later subsections. Residual risk is also considered, offering a view of the systems' exposure even after implementing controls.

I. MQTT Broker Threat Enumeration

Table 7: MQTT Broker Threat Enumeration

Title / Type	Description	CWE / CAPEC
Location	[A passive attacker] [on the	CWE-319

Tracking via Cleartext [Information Disclosure]	same network] can [eavesdrop on unencrypted MQTT Messages, which leads to [tracking disaster victims], negatively impacting [victim safety and privacy.	
Message Tampering (Broker -> Drone) [Tampering]	A [man-in-the-middle attacker] on [MQTT] Channel can [Modify broker messages], which leads to drone [misbehaviour], negatively impacting [data and control integrity]	CWE-345
Broker Subscriber List Exposure [Information Disclosure]	[An attacker with access to/logs] can [extract subscriber list], which leads to [system topology inference], negatively impacting [system confidentiality].	CWE-200

The MQTT broker is the central component of the MQTT disaster relief communication, and attacks performed on it could lead to systematic failures. One of the most critical threats identified is how MQTT is not set up with encryption in its default settings [1], which can lead to unencrypted messages being transmitted. In the current design, traffic between clients, drones, and brokers is prone to passive and active interception. A passive attacker can pick up critical private information, such as the disaster victims' GPS coordinates and the scenario's operational status. Without encryption (e.g, TLS), this vulnerability (CWE-319) leaves the system open to all types of information transmitted to be exposed on all communication channels.

Another notable threat involves message tampering, where a man-in-the-middle attacker could intercept and modify MQTT messages between the broker and drones. This threat (CWE-345) may result in incorrect instructions, loss of data instructions, or even compromised data, which might lead to more significant issues in the system. The ability for attackers to tamper with communication within the system can jeopardize the safety of both the victims and the rescue teams.

II. Drone Client Threat Enumeration

Table 8: Drone Client Threat Enumeration

Title / Type	**Description**	**CWE / CAPEC**
Drone Message Flooding [Denial of Service]	An attacker floods the MQTT channel with excessive or malformed messages, exhausting drone resources like battery and processing power. This disrupts the ability of drones to relay	CWE-400 CAPEC-125
	genuine distress calls.	
Expedition List Tampering [Tampering]	[An attacker] [with access to the drone system] can [modify the local subscription list], which leads to [messages being misrouted or intercepted], negatively impacting [message routing integrity].	CWE-494
Rescue Team Impersonation [Spoofing]	[An attacker] [with MQTT publishing access and topic knowledge] can [pose as a rescue team, which leads to [misrouted missions or data leakage], negatively impacting [trust and data integrity].	CAPEC-62
Spoofed Distressed Origin [Spoofing]	[An attacker impersonates a legitimate disaster victim] and [publishes a spoofed distress messages using MQTT tools, due to the lack of enforced strong client authentication.] This can lead to the system unnecessarily deploying rescue resources to fictitious locations]	CWE-290

The drone-client subsection of the system serves as both a collector of MQTT messages and a relay system, which is the link between victims, responders, and brokers. Drones are set up in an uncontrolled environment to automatically relay data to the correct endpoints, which leads to a primary concern: message flooding. If an attacker transmitted excessive MQTT messages, the communication system would be overloaded with having to figure out the proper routes for these messages to reach the appropriate endpoints, resulting in dropped messages, slow response time, and even performance issues, which could render drones unusable and prone to physical interference. Preventing this threat (CWE-400, CAPEC-125) will require traffic validations and rate-limiting mechanisms.

Beyond Denial of Service risks, the drone system is also vulnerable to tampering with local configuration files. There are two methods of achieving this goal: physical access, which could severely increase the likelihood of CWE-400 / CAPEC-125 being performed successfully, with physical hardware retrieval in mind. The other method is through malware or the attacker's familiarity with the system's default config settings (CWE-494). This threat could also affect message routing integrity, emphasizing the need for read-only file

permissions and secure booting techniques to be implemented.

Without proper authentication or topic access control, impersonation (CAPEC-62) could lead to false alerts, misrouted missions, and data leakage, negatively impacting the system's trust. Mitigations in this area should focus on certification validations.

III. Dashboard & Database threat enumeration

Table 9: Model components & their definitions

Title / Type	Description	CWE / CAPEC
Dashboard Misuse [Repudiation]	[An unauthorized user with dashboard access] can [view sensitive operational data], which leads to [mission strategy compromise], negatively impacting [responder and victim safety].	CWE-552
Dashboard Data Leakage [Information Disclosure]	[A malicious user with dashboard access] can [operate without logging], which leads to [unaccounted data changes], negatively impacting [accountability].	CWE-778
SQL Injection in message [Tampering]	[An attacker] [with access to message publishing] can [send malicious SQL commands via MQTT fields], which leads to [manipulation or deletion of mission data], negatively impacting [system reliability and trustworthiness].	CWE-89
SQL Database Disclosure [Information Disclosure]	[An attacker] [with unauthorized cloud access or misconfiguration exploitation can extract mission information from the SQL database, which leads to [privacy violations or preemptive attacks], negatively impacting the [confidentiality of mission operations].	CWE-200
Compromised Data Processing [Tampering]	[An attacker] [with backend or insider access] can [manipulate raw data During processing, which leads to incorrect analysis and resource misallocation] negatively impacting [rescue mission effectiveness].	CWE-74

The dashboard and database layer of the system acts as the interface for mission management and coordination with the rescue team. This component processes incoming MQTT messages, stores incident records, and displays information to the operators. Due to its central role in the MQTT disaster relief system, it is a high-value target for attackers. One notable threat is SQL injection, where an attacker embeds incoming MQTT messages with harmful SQL instructions. These inputs (CWE-89) can manipulate or delete mission-critical data, disrupt coordination, or erase distress messages if not properly sanitized. Since MQTT fields can include free-form inputs, this threat is high on the priority list.

Another serious concern is database exposure, where attackers with unauthorized access to the database or the ability to exploit misconfigured environments can extract sensitive data on the mission and the victims. This maps to CWE-200 (Exposure of sensitive information to an Unauthorized actor).

Qualitative risk Matrix (Impact x likelihood)

Table 10: Risk Matrix

Impact	Risk Matrix Score	Threat Treatment
Disrupted Emergency Communication (Drone Flooding) MQTT flooding overloads drones, causing message loss, delays, or power failures	4B	**Mitigate** • Rate limiting • Max in-flight caps • Anomaly detection (Wazuh)
Victim Location Exposure (Cleartext MQTT) Unencrypted traffic can reveal sensitive information such as GPS locations and distress messages.	5B	**Mitigate** • TLS Encryption • Enforced certificate validation • Encrypted channel enforcement across all endpoints
False Messaging (Broker -> Drone Tampering) Adversaries may modify MQTT messages mid-transit.	4B	**Mitigate** • Input validation • Message signing • Client authentication
Backend Data Corruption (SQL Injection) Unencrypted MQTT fields allow malicious SQL execution.	3B	**Mitigate** • Parameterized queries • Input sanitization • Signed message enforcement • Authentication for DB access • Isolated DB layer
Dashboard Misuse Unauthorized users access or modify dashboard content untracked.	3B	**Mitigate** • Audit logging • Role-based access control (RBAC) • Log forwarding
Rescue Team	5D	**Mitigate**

Impersonation An attacker with access to mission data can physically approach victims or field responders, posing as legitimate rescue teams and potentially cause abductions or physical harm.		• Restrict mission data access • Endpoint hardening • Access audit • Mission data minimization on devices
Expedition List tampering (Local Config modification) The attacker modifies subscription lists, impacting routing integrity.	3C	**Mitigate** • Read-only permissions • Secure boot configuration • Config hashing • Access control to authorized users

Table 10 outlines the prioritized risks facing the MQTT disaster relief systems by evaluating the threats using impact severity and likelihood. Each threat scenario reflects a real-world vulnerability present within the system and its components. High-risk impacts like Victim Location Exposure (5A) can leak GPS coordinates due to unencrypted MQTT traffic. This disclosure threatens victims' privacy and can enable more complex and impactful threats, such as Rescue Team Impersonation (5D), and let attackers infer about the system using MQTT metadata. In the scenario “Rescue team Impersonation”, an attacker could physically approach a victim under the guise of an authorized responder, potentially endangering those in need of assistance. While this attack is considered less likely due to the operational effort required, the impact is critical, justifying its high-priority classification.

Other threats, such as Drone Flooding (4B) and False Messaging via Broker Tampering (4B), pose risks to system availability, rescue team resources, and message integrity. Threats such as Dashboard Misuse (3B) and SQL injection (3B) highlight administrative and backend vulnerabilities. Each danger is matched with a suggested mitigation strategy. The Impact x Likelihood risk matrix ensures that high-impact risks are addressed and enables cybersecurity engineers to prioritize the threats accordingly.

H. *Post Mitigation*

Residual risk

Residual risk analysis quantifies the updated risk after mitigations have been applied. In our analysis, we adapt the traditional residual risk formula to more accurately reflect how more mitigations result in a lower residual risk.

The standard approach multiplies the effectiveness of the mitigations together, and the numerical representation of these mitigations are percentages, and when multiplied, leads to a lower number. Thus, leading the amount of mitigations to be negatively proportional to the residual risk, which should not apply. To address this, we calculate the effectiveness with the following formula:

$$EMF=1-i=1\prod n(1-ei)$$

Where ei is the effectiveness of a mitigation strategy, this model preserves the realism of diminishing returns. This formula only works when the presented mitigation strategies are not redundant.

1) Disrupted Emergency Communication (Drone Flooding)
- Risk Matrix Score: 4B
- Vulnerability (p1): 0.7
- Threat Likelihood (p2): 0.7
- Impact: 4
- Mitigations:
  - Rate limiting (0.8)
  - Inflight caps (adjusted for partial redundancy) (0.6)
  - Anomaly detection (Wazuh) (0.75)
- EMF = 0.98
- Residual Risk = **1.98**

2) Victim Location Exposure (Cleartext MQTT)
- Risk Matrix Score: 5B
- Vulnerability (p1): 0.8
- Threat Likelihood (p2): 0.8
- Impact: 5
- Mitigations:
  - TLS Encryption (0.85)
  - Certificate validation (0.8)
  - Encrypted channel enforcement (0.75)
- EMF = 0.9925
- Residual Risk = **3.22**

3) False Messaging (Broker -> Drone Tampering)
- Risk Matrix Score: 4B
- Vulnerability (p1): 0.7
- Threat Likelihood (p2): 0.6
- Impact: 4
- Mitigations:
  - Input validation (0.7)
  - Message signing (0.8)
  - Client authentication (0.75)
- EMF = 0.985
- Residual Risk = **1.71**

4) Backend Data Corruption (SQL Injection)
- Risk Matrix Score: 3B
- Vulnerability (p1): 0.65
- Threat Likelihood (p2): 0.5
- Impact: 3
- Mitigations:
  - Parameterized queries (0.85)
  - Input sanitization (0.8)
  - Message signing (0.75)
- Authentication for DB access (0.8)
- Isolated DB layer (0.7)
- EMF = 0.99955
- Residual Risk = **0.975**

5) Dashboard Misuse
- Risk Matrix Score: 3B
- Vulnerability (p1): 0.6

- Threat Likelihood (p2): 0.6
- Impact: 3
- Mitigations:
  - Audit logging (0.7)
  - Role-Based Access Control (0.75)
  - Log forwarding (0.7)
- EMF = 0.9775
- Residual Risk = **1.11**

6) Rescue Team Impersonation
- Risk Matrix Score: 5D
- Vulnerability (p1): 0.6
- Threat Likelihood (p2): 0.3
- Impact: 5
- Mitigations:
  - Restrict mission data access (0.75)
  - Endpoint hardening (0.7)
  - Access audit (0.7)
  - Mission data minimization (0.8)
- EMF = 0.9955
- Residual Risk = **0.9**

7) Expedition List Tampering (Local Config Modification)
- Risk Matrix Score: 3C
- Vulnerability (p1): 0.6
- Threat Likelihood (p2): 0.3
- Impact: 3
- Mitigations:
  - Read-only permissions (0.7)
  - Secure boot configuration (0.75)
  - Config hashing (0.7)
  - Access control (0.75)
- EMF = 0.994375
- Residual Risk = **0.54**

Table 11: Updated Risk Matrix

Impact	**Residual risk score**	**Residual Risk Level**	**Residual Risk Matrix Score**
Disrupted Emergency Comms	1.98	Medium	4B
Victim Location Exposure	3.22	High	5B
False Messaging	1.71	Medium	4B
Backend Data Corruption	0.975	Low	3B
Dashboard misuse	1.11	Medium	3B
Rescue Team Impersonation	0.9	Low	5D
Expedition List Tampering	0.54	Low	3C

From the analysis, Victim Location Exposure remains the most critical residual risk with a high score of 3.22, primarily due to its significant impact on confidentiality and how it directly relates to real-world consequences and other impacts. Conversely, Backend Data Corruption, Rescue Team Impersonation, and Expedition List tampering are rated lower due to the applied mitigation strategies such as endpoint hardening and secure configurations. Overall, the table highlights the success of layered defenses reducing most critical threats to acceptable levels, and also highlights impacts that would need continued monitoring and vigilance.

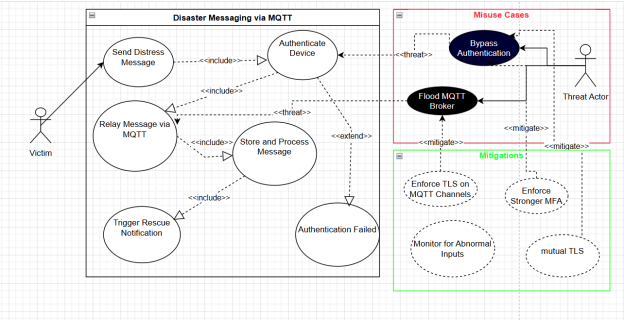

Figure 9. Mitigated Use Case 1 [14]

The Updated Use Case 1 in figure 9 depicts secure communications by highlighting notable vulnerabilities that can be leveraged by malicious actors for submitting distress messages over MQTT during disaster responses. The flow starts from a victim's device authenticating to the system and sending a structured distress message containing the GPS coordinates and urgency levels. The victim's message is later relayed by a drone through MQTT, where it is processed by the broker and forwarded to the command center for rescue coordination.

It is, however, possible for attackers to circumvent authentication frameworks and claim to be victims or inject phony messages. Moreover, they can try to misuse an MQTT broker by sending an overwhelming amount of information, thereby causing a denial-of-service (DoS) attack and interrupting crucial communication. These threats endanger message reliability, enable spoofing, and overall, compromise the system's availability and trustworthiness of rescue operations.

To nullify such threats, the system employs several security controls. Firstly, mutual TLS authentication ensures client and broker validation; secondly, stronger Multi-Factor Authentication (MFA) reduces the risk of unauthorized access, and real-time detection of anomalies through Wazuh/Suricata marks questionable actions. Within the defenses are encrypted telemetry and secured communication via MQTT, which protects the entire traffic. Furthermore, rate limiting coupled with DoS attack mitigation and strict spoofing validates attacks while increasing system strength in adversary conditions.

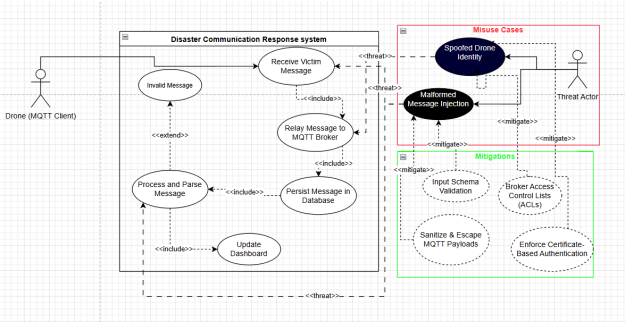

Figure 10. Mitigated Use Case 2 [14]

The updated use in figure 10 cases offer a detailed examination of the most critical threat scenarios within the disaster response communication system, focusing on weaknesses at the Victim Message Submission (Use Case 1) and Drone Message Relay with Field Command Coordination (Use Case 2). The new diagrams contain specific misuse cases designed to show how attackers can exploit system weaknesses, such as bypassing device authentication, injecting malformed MQTT payloads, spoofing drones, or overwhelming the broker to obstruct communications.

Every abuse case is defined with applicable mitigations aimed at increasing system security. These include mutual TLS authentication, ACLs, payload sanitization, schema validation, and Wazuh or Suricata anomaly detection. By addressing these particular concerns and incorporating appropriate layered defenses, the use cases improve the reliability and accessibility of the disaster communication framework and strengthen trust in system performance. Coordination and information flows performed during emergency operations ensure reliability.

I. *Mitigation Strategies*

Based on the threat enumerations and risk analysis conducted in stage 7, we apply mitigation strategies to reduce the systems' exposure to:

- Spoofing
- Tampering
- Denial of Service
- Information Disclosure

These mitigations are specifically designed to fit the system components and align with the confidentiality, integrity, and availability goals identified in Stage 1. Each mitigation strategy was drafted with its technical feasibility in mind when used in a disaster-response setting.

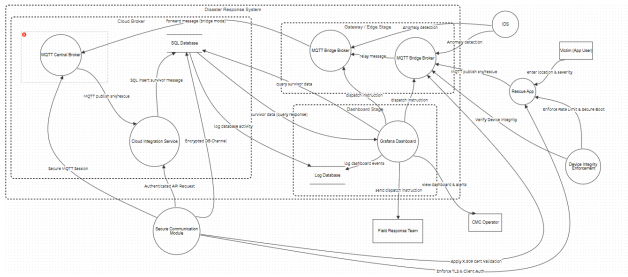

Figure 11. Mitigated DFD [14]

Figure 11 showcases the mitigations that were specifically designed to align with the confidentiality, integrity, and availability goals identified in stage . Each mitigation strategy was drafted with its technical feasibility in mind when used in a disaster-response setting. Figure 11 has the following mitigation strategies applied:

i . Communication Security

The Secure Communication Module undertakes all necessary measures to prevent eavesdropping and tampering by enforcing TLS on all MQTT communications, including to and from communications with the User App, Bridge Broker, Central Broker, and Cloud. The module also makes sure the broker is set up with client certificates where traffic is strictly over port 8883 TLS, not plaintext.

This solves CWE-319 (Missing encryption) and T1040 (Network Sniffing) by ensuring that the sensitive data is encrypted while in transit.

Also, aligns with MITRE D3FEND Techniques:

- D3-ENC (Encrypted Channel Establishment)
- D3-TFC (Traffic Flow Confidentiality)

The same protections are applied to the Cloud and dashboard subsystems allowing for total end to end security.

ii. Authentication and Authorization

The MQTT broker will implement client authentication and topic level authorization. Drone, command app, and user app as captured devices authenticate alongside presenting their credentials and inputs with corresponding hashes. Mosquitto can be configured to accept connections only upon providing a username and password, as well as an ACL mapping that dictates access on a per topic publish/subscribe basis.

This is relevant to CWE-306 (Missing Authentication) and CWE-862 (Missing Authorization) while also in some way mitigating control T1078 (Valid Accounts Misuse) by adding those controls centered around accounts.

Supports MITRE D3FEND techniques:

- D3-AUAA (User Authentication and Authorization relayed from D3 text ver)
- D3-MA (Message Authentication)
- D3-ACD (Access Control Design).

Secondary Countermeasures target users who do not possess permission to connect, spoofers who will not pass an authentication check. If somehow, a user with stolen credentials gets access, they are locked out of changing the topics they do not control. For example, an attacker without a required certificate cannot readily masquerade as the broker or some client. For agility or emergency purposes, we may temporarily adopt a simpler method such as having a single key for all devices that share the same classification.

iii. Network Security Enhancements

For the wireless communication layers, the Secure Communication Module will implement WPA2-Enterprise for device-specific authentication, where possible. In cases where such enforcement is impractical, the system may revert to using WPA2-PSK with a rigorously generated pre-shared key. This key will be changed as required by authorized staff to preserve the integrity of access control.

At the broker layer, Device Integrity Enforcement applies DoS mitigation through:

- Rate limiting on MQTT broker (e.g. maxconnections, maxinflight_messages)
- Post isolation through firewalls
- Minimizing broker privileges
- Frequent patching

These mitigations address CWE-400 (Uncontrolled Resource Consumption) and T1498 (Network Denial of Service) of the MITRE ATT&CK framework.

- D3-DA005 (Network DOS Protection)
- D3-DA0018 (Rate limit)
- D3-DA001 (Patch Validation)
- D3-SI (Service Isolation)

IV. Backend Data Protections

The backend sanitizes all incoming MQTT data prior to its entry into the SQL database. This preemptive action harnesses control over vulnerability exploitation of CWE-89. This control reduces injection attack attempts and corresponds to T1190 (Exploit Public-Facing Application). It also composes with D3-DATA (Data Validation) and ensures all data is checked prior to any further action being taken on it.

The Secure Communication Module enforces that highly privileged authentication credentials and encryption are employed for any database transactions, and protects from unauthorized access. In conjunction, these correspond to:

- D3-AUAA (User Authentication & Authorization)
- D3-ENC (Encrypted Channel Establishment)

In addition, the IDS supports complete backend logging by means of capturing system events and logging them into the remote Log Database. This advanced mechanism ensures security by countering logging insufficiency or CWE-778 and ensures accountability, traceability, and detection of potential anomalous behavioral activities.

Aligned MITRE D3FEND technique:

- D3-LMA (Log Monitoring and Analysis)

V. Final Remarks

A. Key findings

Applying the PASTA framework to the MQTT-based disaster relief system revealed a range of threat scenarios that target traditional IT components and context-specific elements. Threats such as drone flooding, abusing the MQTT protocol, and SQL injection emerged as significant threats due to their ability to compromise the system's availability and data integrity. Particularly notable threats revolved around information disclosure, such as unencrypted MQTT messages and dashboard misuse. These information disclosure threats could physically impersonate rescue team members and system topology inference. These findings confirm that while MQTT is an easily implementable, lightweight communication protocol that is advantageous in disaster environments, additional security layers must be implemented to ensure mission-critical reliability and assist in saving lives.

The analysis also highlighted how various attack surfaces intersect across different components of the MQTT system. For example, MQTT topic spoofing was tied to authorization misconfigurations. Another example is how drone relay functions introduced choke points that could lead to denial-of-service attacks. These disruptions, in turn, could lead to physical access to drone configuration files due to drone failures caused by denial-of-service attacks. This cross-component threat interaction reinforces the necessity of end-to-end threat modelling that covers not only individual components but the system as a whole.

B. Summary of Mitigation Strategy

The mitigation strategies developed were carefully aligned with the threats identified while keeping the architecture's operational constraints in mind. TLS encryption was applied to all MQTT communication components, ensuring encrypted channels between users, drones, brokers, and the cloud. Mutual certificate validation was also included in the mitigation strategies to prevent unauthorized devices from communicating with the system and to mitigate man-in-the-middle attacks. Broker-level access control lists were enforced to ensure no unauthorized users could publish or subscribe to sensitive topics, significantly reducing the risk of spoofed or malicious messages disrupting coordination.

To protect the system's integrity, message authentication and certification make it harder to interfere with or tamper with incoming messages and spoof new MQTT traffic. Parameterized SQL queries and input sanitization procedures were implemented on the backend to protect the system from injection threats. Role-based

access control was applied to the dashboard interface, ensuring only authorized users could perform their respective actions. Audit logging and remote log forwarding were included to support incident response capabilities and system monitoring. Each mitigation was selected for its positive impact on the system's security and how it fits its business objectives.

C. Challenges, limitations

One of the challenges encountered was trying to achieve the mission's cybersecurity objectives and reach the mission's business objectives. For example, while mutual authentication and certificate systems are considered strong security practices, implementing them in an environment where the components are not suited for tasks beyond message relaying could harm the mission overall. The balance in achieving the business and security goals required thoughtful considerations.

Another challenge encountered stems from the assumed trust in physical assets. With drones being recognized as “safe” devices from physical reach, the assumption that they are physically secure could lead to potential issues, especially in a disaster environment where the likelihood of physical access to drones is increased. Physical access to these devices can expose the hardware to theft, tampering, or damage. While measures like secure boot and read-only configurations were proposed, they cannot eliminate physical risks and do not address other, more specific threats possible with physical access to the drone.

Due to the lightweight nature of MQTT and the drone-based communication system, several mitigation strategies were ultimately excluded. Some of these approaches might have been viable had the team had access to the physical system, allowing for a clearer understanding of the hardware limitations being worked with. One such strategy was the implementation of full-disk encryption, which could have significantly strengthened protections against information disclosure in the event of a physical device capture. This option was dismissed for two reasons: first, its power demands were deemed too intensive for a device known not to have a large energy capacity; secondly, the likelihood of an adversary physically retrieving a field device remained uncertain. Given these factors, the mitigation techniques mentioned in the report were considered a more balanced approach between feasibility and effective risk reduction.

Another strategy explored but not adopted was periodic cryptographic key rotations within the encrypted communication channels. While this would have further reduced the likelihood of a successful exploitation of accessing sensitive data, it was ultimately excluded due to concerns over synchronization between the drones and the slight increase in communication traffic. In a resource-constrained and closely connected environment, minimizing system load was ultimately prioritized over the implementation of this mechanism.

Lastly, while the PASTA framework provided a structured and risk-centric approach to evaluating threats, its staged methodology may not fully reflect the dynamic conditions of a real disaster environment. In different scenarios where threats evolve rapidly and system components are deployed ad hoc, the linear progression of PASTA stages can limit real-time adaptability. Also, the framework’s reliance on detailed architectural decomposition assumes a level of system maturity that may not exist in early deployments. These limitations suggest that in future iterations, PASTA could be supplemented with continuous monitoring frameworks to support evolving risks in crisis response systems.

D. Recommendations for future work

Future efforts should revisit discredited mitigation strategies that were set aside due to assumptions about certain system component constraints. Because this analysis was conducted without a functional prototype or hands-on testing, the team had to make conservative estimates regarding hardware capabilities and bandwidth limitations. It might be possible for the actual system to prove capable of supporting mitigations such as full-disk encryption, cryptographic key rotation, or more frequent and detailed logging. As such, future security engineers working on this project are encouraged to reevaluate dismissed mitigations. Doing so may reveal that some trade-offs were unnecessary and that the evolving industry of embedded systems might improve the limitations of such devices.

A key recommendation for future work is validating the threat model through simulations replicating realistic attacks. Such testing scenarios include packet replay vulnerabilities, authentication bypasses, and topic spoofing. This testing should also explore edge cases that would not be possible without a hands-on prototype. Another cause of concern is onboarding new drones and ensuring they are updated to meet the configuration of current drones that were patched to take into account previously unconsidered cyber threats.

Additionally, improvements to situational awareness, real-time threat intelligence feeds, and monitoring could be implemented into the dashboard. The system's logging infrastructure could be improved to power a dynamic anomaly detection layer by statistically analyzing logs from previous attack attempts and coupling them with machine learning. Over time, this anomaly detection algorithm will improve to provide more insight into the system's security downfalls, and additional improvements can be made.

E. Conclusion

In conclusion, this report demonstrated a comprehensive risk-centric analysis of a disaster relief system built on MQTT-based drone communications using the PASTA threat modelling framework. The analysis identified critical vulnerabilities that balanced security, practicality, and performance by aligning business and security objectives. This layered defensive approach addresses typical IT threats and accounts for real-world operational constraints, such as limited power. While certain advanced protections were excluded due to

hardware limitations, the final architecture has implemented mitigation strategies ideal for threats in a volatile environment. This work lays a foundation for future improvements and encourages further development and exploration of the lightweight yet effective security controls for this MQTT-based system.

V. References

[1] Eclipse Foundation, "MQTT Version 3.1.1," *OASIS Standard*, 2014. [Online]. Available: https://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html . [Accessed: 15-Apr-2025].

[2] VerSprite, "PASTA Threat Modeling Framework," *ThreatModeling.com*. [Online]. Available: https://threat-modeling.com/pasta-threat-modeling/. [Accessed: 15-Apr-2025].

[3] U.S. House of Representatives, *A Failure of Initiative: Final Report of the Select Bipartisan Committee to Investigate the Preparation for and Response to Hurricane Katrina*, 109th Congress, 2nd Session, House Report 109-377, Feb. 15, 2006. [Online]. Available: https://www.govinfo.gov/content/pkg/CRPT-109hrpt377/pdf/CRPT-109hrpt377.pdf

[4] L. Wentz, "Haiti Information and Communications Observations," ICT4Peace Foundation, Sep. 2010. [Online]. Available: https://ict4peace.org/wp-content/uploads/2010/09/Haiti-Summary-Observations-V9_Larry-Wentz_PDF.pdf

[5] Wazuh, "The Open Source Security Platform," *Wazuh Documentation*, 2024. [Online]. Available: https://documentation.wazuh.com/current/index.html. [Accessed: 16-Apr-2025].

[6] MITRE Corporation, "Common Vulnerabilities and Exposures (CVE)," *CVE - MITRE*. [Online]. Available: https://cve.mitre.org/. [Accessed: 15-Apr-2025].

[7] FIRST.org, "Common Vulnerability Scoring System v3.1: Specification Document," *Forum of Incident Response and Security Teams*, 2019. [Online]. Available: https://www.first.org/cvss/specification-document .[Accessed: 15-Apr-2025].

[8] MITRE Corporation, "Common Weakness Enumeration (CWE)," *CWE - MITRE*. [Online]. Available: https://cwe.mitre.org/. [Accessed: 15-Apr-2025].

[9] MITRE Corporation, "Common Weakness Scoring System (CWSS)," *CWSS - MITRE*. [Online]. Available: https://cwe.mitre.org/cwss/ . [Accessed: 15-Apr-2025].

[10] MITRE Corporation, "Common Attack Pattern Enumeration and Classification (CAPEC)," *CAPEC - MITRE*. [Online]. Available: https://capec.mitre.org/. [Accessed: 15-Apr-2025].

[11] MITRE Corporation, "Common Configuration Enumeration (CCE)," *CCE - MITRE*. [Online]. Available: https://cce.mitre.org/ . [Accessed: 15-Apr-2025].

[12] MITRE Corporation, "MITRE ATT&CK Framework," *MITRE ATT&CK®*. [Online]. Available: https://attack.mitre.org/. [Accessed: 15-Apr-2025].

[13] MITRE Corporation, "MITRE D3FEND Framework," *MITRE D3FEND™*. [Online]. Available: https://d3fend.mitre.org .. [Accessed: 15-Apr-2025].

[14] Hussain Zainal, *CYSE587 Security Threat Modeling Projects*, GitHub repository, 2025. [Online]. Available: https://github.com/MarioDS15/CYSE587