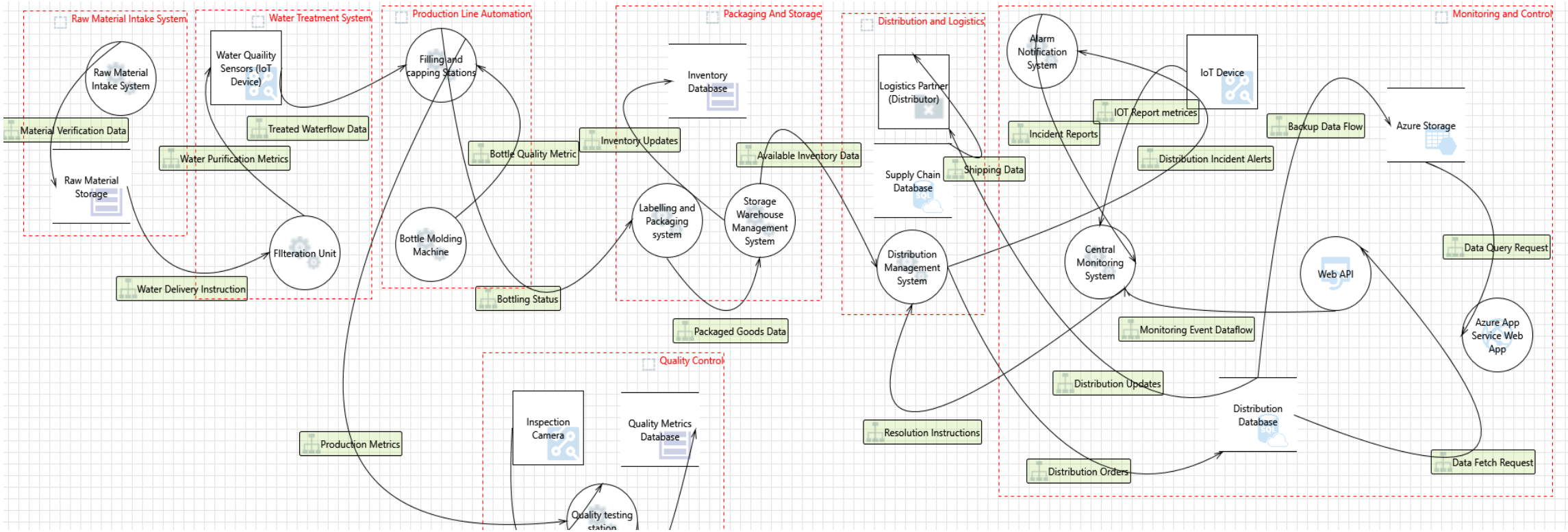
THREAT ANALYSIS ON SCM OF INDUSTRIAL MANUFACTURING(WATER BOTTLE PLANT) MINI-PROJECT 2024-25

Submitted to:

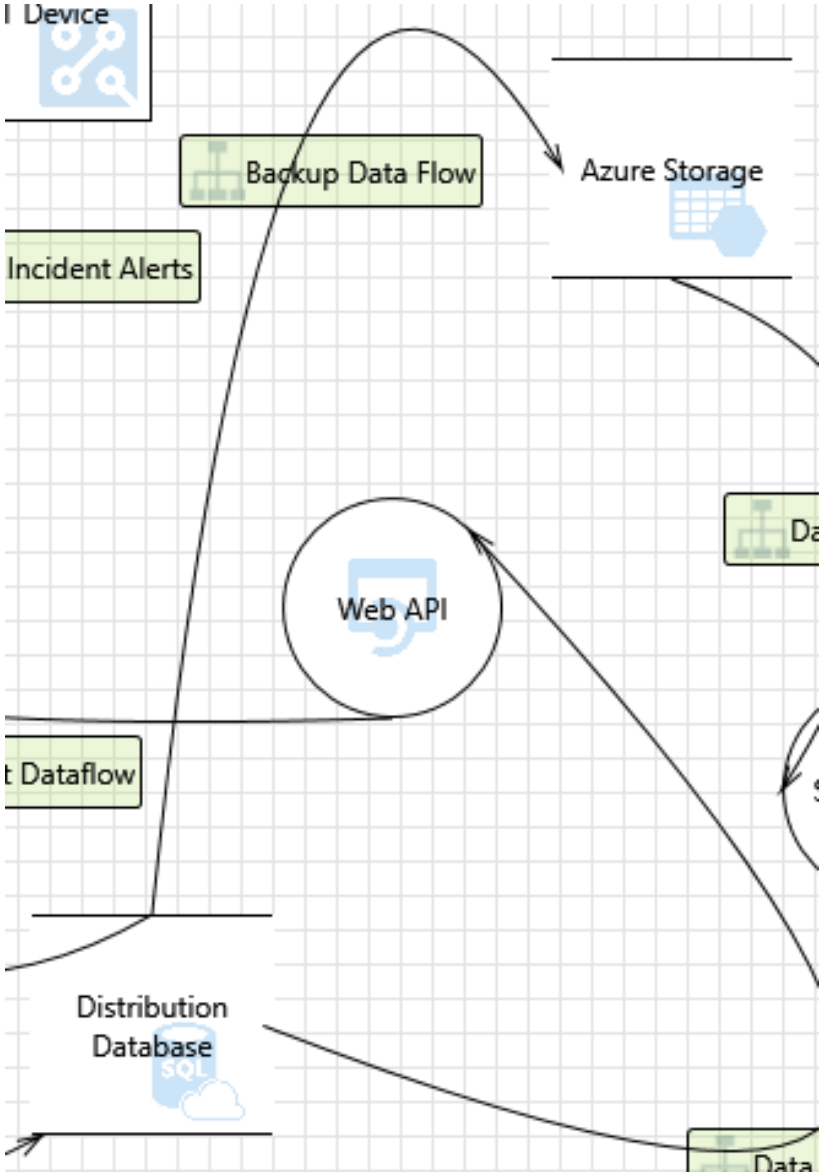
Dr. Ashish Kumar Dwivedi



THREAT ANALYSIS PLAN LAYOUT:

- **Mitigation Implemented:** 36
- **Not Started:** 7
- **Not Applicable:** 7
- **Completed:** 36
- **Total Threats Analyzed:** 43

Interaction: Backup Data Flow



1. An adversary can gain unauthorized access to Azure Storage due to weak access control restrictions [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary can gain unauthorized access to Azure Storage due to weak access control restrictions
Possible Mitigation(s):	• **SAS and SAP:** • Use **Shared Access Signatures (SAS)** and **Stored Access Policies (SAP)** for limited, time-bound access. • **References:** SAS Best Practices and SAP Guidelines. • **Role-Based Access Control (RBAC):** • Enforce **least privilege** roles and audit permissions regularly. • **Reference:** RBAC in Azure Storage. • **Network Security:** • Restrict access to trusted IPs and **virtual networks** via **Service Endpoints** or **Private Endpoints**. • **Reference:** Secure Networking.
SDL Phase:	Implementation

2. An adversary can gain unauthorized access to Azure Storage instances due to weak network configuration [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary can gain unauthorized access to Azure Storage instances due to weak network configuration

Possible Mitigation(s):	**Possible Mitigation(s):**
	• **Restrict Network Access:**
	  • Configure **Virtual Network (VNet) service endpoints** or **Private Endpoints** to limit access to trusted networks only.
	  • **Reference:** Network Security Best Practices.
	• **Firewall and IP Rules:**
	  • Implement **firewalls** to block public network access and restrict IP ranges.
	  • Allow only trusted IP addresses and subnets for communication.
	• **Secure Transport Layer:**
	  • Enforce **HTTPS-only** communication to protect data in transit.
	  • Disable legacy protocols such as SMBv1 and enable TLS 1.2 for encryption.

SDL Phase:	Implementation

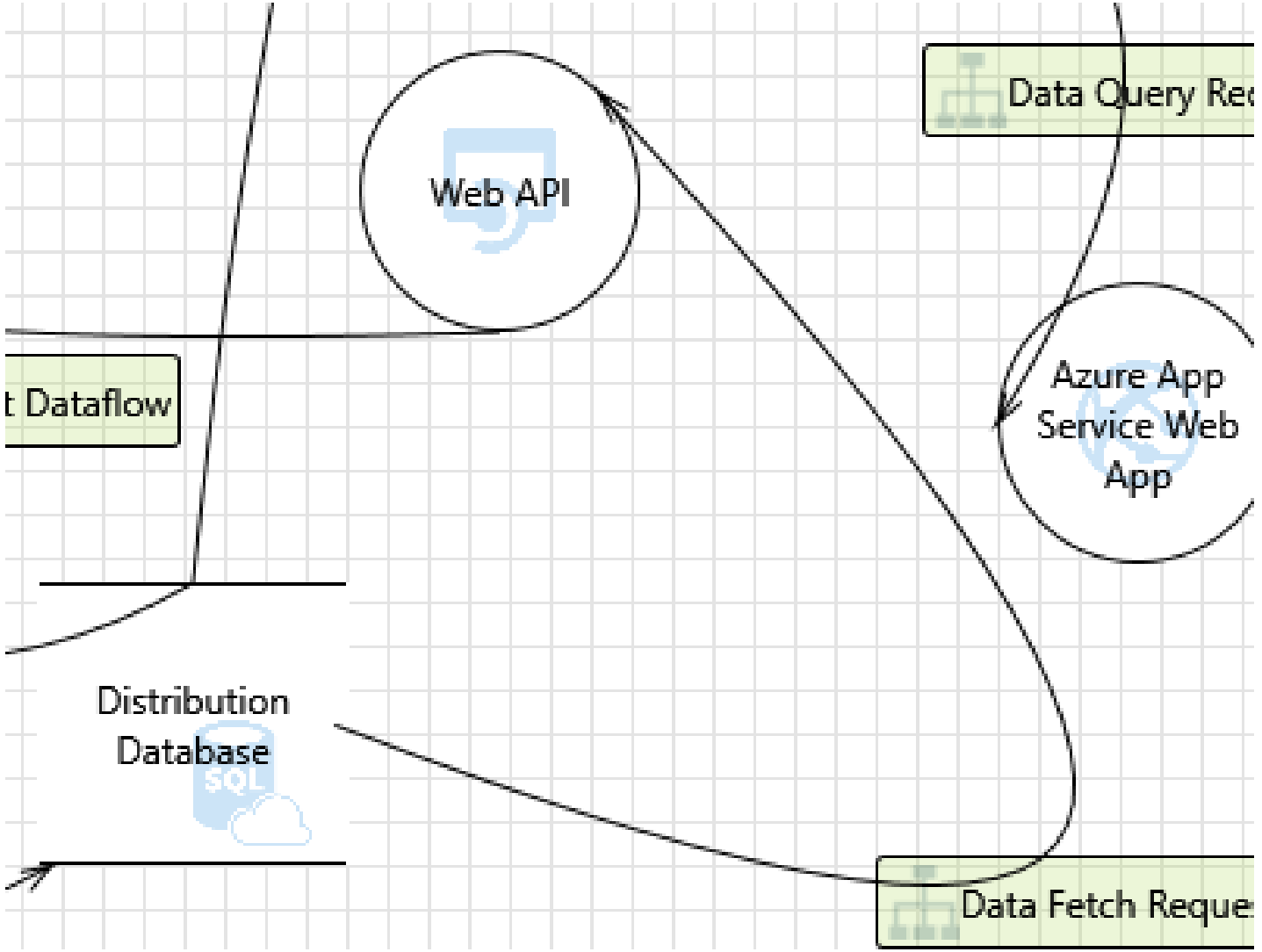
3.An adversary can abuse an insecure communication channel between a client and Azure Storage [Priority: Medium]

Category:	Information Disclosure
Description:	An adversary can abuse an insecure communication channel between a client and Azure Storage
Possible Mitigation(s):	• **Enforce HTTPS for Communication:** Enable the **Secure Transfer Required** option to force HTTPS for all communications with Azure Storage. Learn More • **Implement Client-Side Encryption:** Encrypt sensitive data before uploading it to Azure Storage to protect against unauthorized access. • **Validate Certificates:** Ensure SSL/TLS certificates are valid and trusted to prevent man-in-the-middle (MITM) attacks.
SDL Phase:	Implementation

4. An adversary can deny actions on Azure Storage due to lack of auditing. [Priority: Medium]

Category:	Repudiation
Description:	Proper logging of all security events and user actions builds traceability in a system and denies any possible repudiation issues. In the absence of proper auditing and logging controls, it would become impossible to implement any accountability in a system.
Possible Mitigation(s):	• **Enable Azure Storage Analytics**: Activate logging for Azure Storage to track access and actions, ensuring traceability. • **Implement Source-Level Auditing**: Audit calls to Azure Storage at the source to improve accountability. • **Enable Azure Monitor**: Use Azure Monitor to collect logs and set up alerts for suspicious activity.
SDL Phase:	Implementation

Interaction: Data Fetch Request



5. An adversary can gain access to sensitive data by performing SQL injection through Web API. [Priority: High]

Category:	Tampering
Description:	SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table or as metadata. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed.
Possible Mitigation(s):	• **Use Type-Safe Parameters**: Ensure that Web API data access utilizes type-safe parameters to avoid direct concatenation of user inputs into SQL queries, preventing SQL injection. • **Implement Input Validation**: Validate and sanitize all user inputs to ensure they conform to expected formats, rejecting any suspicious or unexpected inputs that may be used for SQL injection. • **Use Stored Procedures**: Instead of dynamic SQL queries, use stored procedures with parameterized queries to ensure safer handling of user inputs and avoid SQL injection vulnerabilities.
SDL Phase:	Implementation

6. An adversary may spoof Distribution Database and gain access to Web API . [Priority: High]

Category:	Spoofing
Description:	If proper authentication is not in place, an adversary can spoof a source process or external entity and gain unauthorized access to the Web Application
Possible Mitigation(s):	• **Strong Authentication** - Implement **OAuth 2.0** or **JWT tokens** with expiration and signature validation to authenticate requests securely. • **Mutual TLS (mTLS)** - Enforce **mutual TLS authentication** to validate both client and server identities, preventing unauthorized connections. • **API Gateway Security** - Use an **API Gateway** with built-in authentication, rate limiting, and anomaly detection to protect API endpoints.
SDL Phase:	Design

7. Attacker can deny a malicious act on an API leading to repudiation issues. [Priority: High]

Category:	Repudiation
Description:	Attacker can deny a malicious act on an API leading to repudiation issues
Possible Mitigation(s):	• **Auditing and Logging** - Enable **detailed logging** of all API requests, responses, and user actions with timestamps to create a tamper-proof audit trail. • **Digital Signatures** - Use **digital signatures** to verify the authenticity and integrity of transactions, ensuring non-repudiation. • **Log Integrity Protection** - Secure logs using **encryption** and **hashing** techniques to prevent tampering and unauthorized modifications.
SDL Phase:	Design

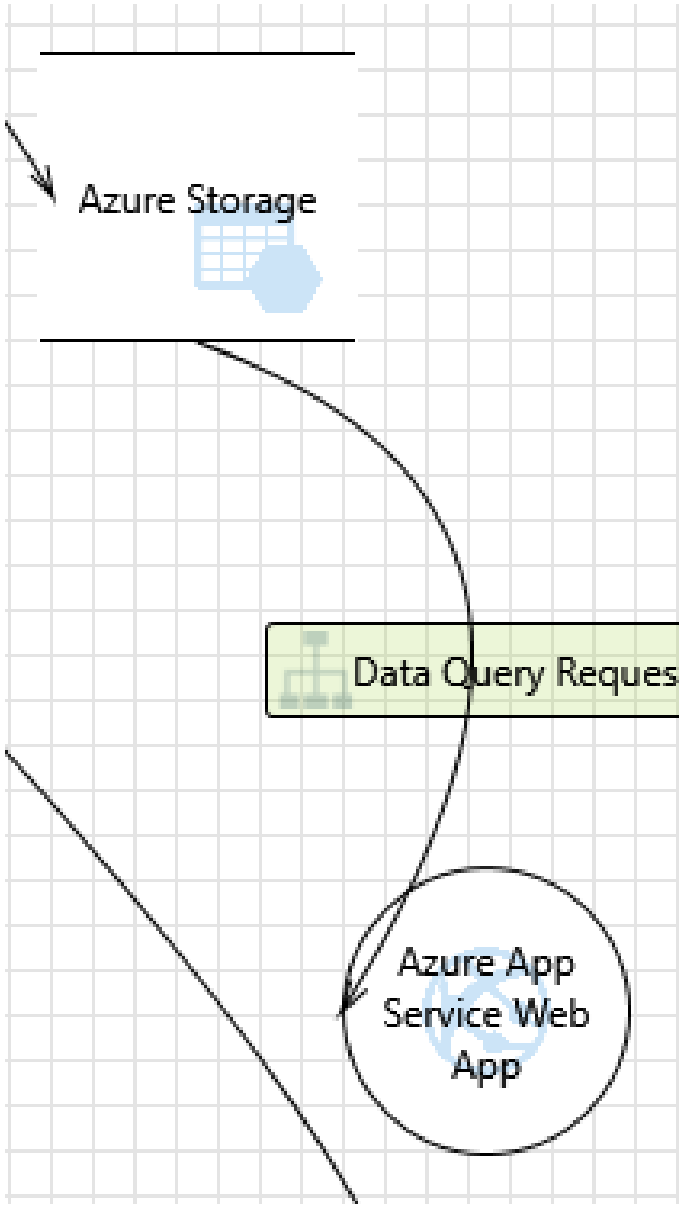
8. An adversary can gain access to sensitive data stored in Web API's config files. [Priority: Medium]

Category:	Information Disclosure
Description:	An adversary can gain access to the config files. and if sensitive data is stored in it, it would be compromised.
Possible Mitigation(s):	• **Encryption of Sensitive Data** - Encrypt sensitive sections of configuration files using strong encryption algorithms (e.g., AES-256). • **Environment Variables** - Store sensitive data, such as API keys and database credentials, in **environment variables** instead of configuration files. • **Access Control** - Apply **strict file permissions** to limit access to configuration files, ensuring only authorized processes or users can read them.
SDL Phase:	Implementation

9. An adversary may gain unauthorized access to Web API due to poor access control checks. [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary may gain unauthorized access to Web API due to poor access control checks
Possible Mitigation(s):	• **Role-Based and Attribute-Based Access Control (RBAC/ABAC)** • Enforce **RBAC** or **ABAC** to restrict access based on user roles or attributes, ensuring users only access authorized resources. • **Authorization Middleware** • Use **ASP.NET Core Authorization Middleware** to validate user permissions for each API endpoint before processing requests. • **Least Privilege Principle** • Apply the **least privilege principle** to ensure minimal access rights are granted by default, reducing exposure to privilege escalation attacks.
SDL Phase:	Implementation

Interaction: Data Query Request



10. An adversary may perform action(s) on behalf of another user due to lack of controls against cross domain requests. [Priority: High]

CATEGORY:	**ELEVATION OF PRIVILEGES**
Description:	An adversary may perform action(s) on behalf of another user due to lack of controls against cross domain requests
Possible Mitigation(s):	• **CORS Policy Configuration** • Allow requests **only from trusted origins** by configuring **CORS policies** explicitly in the Web API. Avoid using wildcard (*) origins. • **CSRF Protection** • Implement **anti-CSRF tokens** in API requests to prevent unauthorized actions from malicious domains. • **Authentication Headers** • Require **authentication tokens (e.g., JWT)** or session cookies with **SameSite=strict** attribute to validate legitimate requests.
SDL Phase:	Implementation

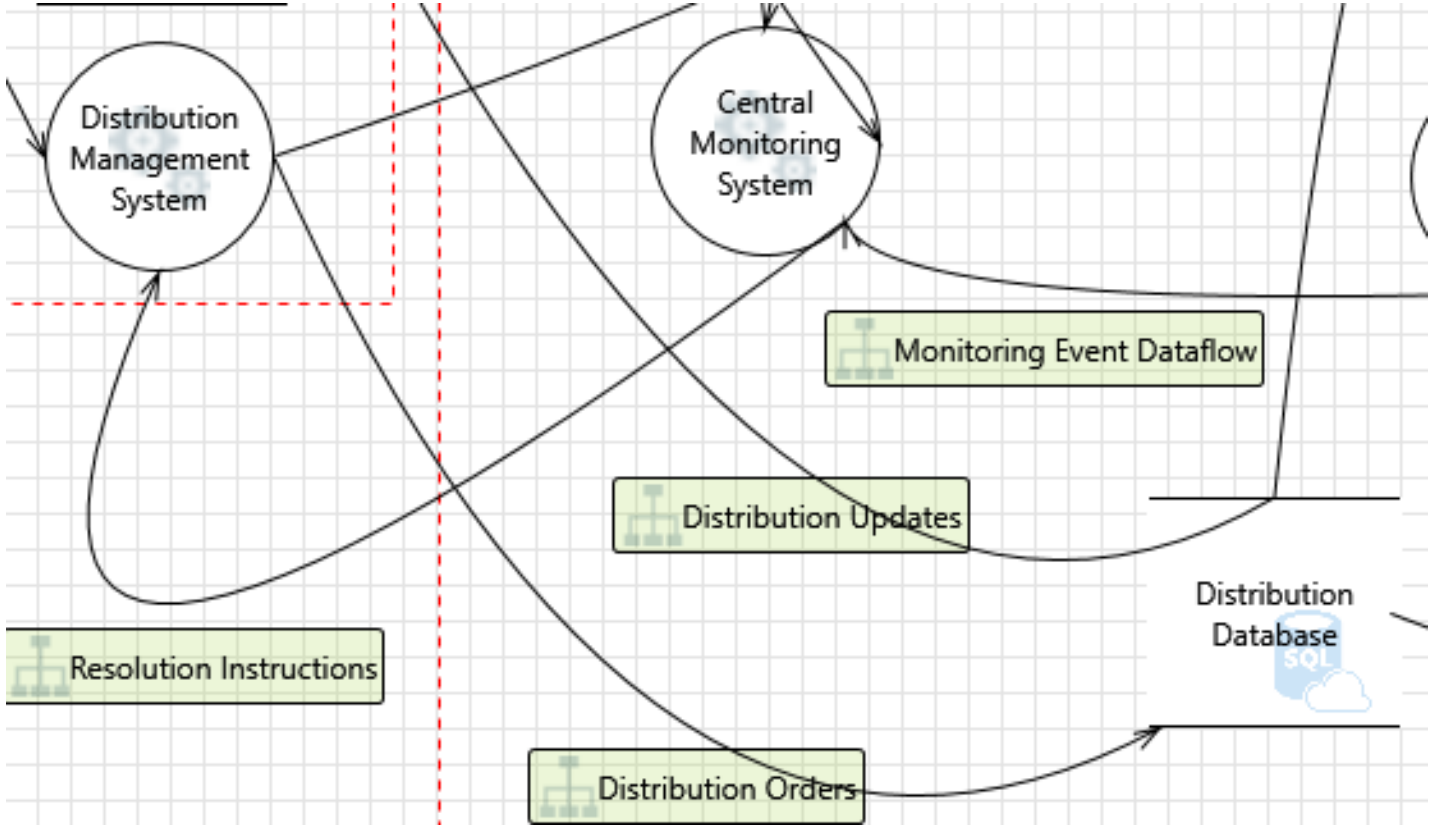
11. An adversary may gain unauthorized access to Azure App Service Web App due to weak network configuration. [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary may gain unauthorized access to Azure App Service Web App due to weak network configuration
Possible Mitigation(s):	• **IP Whitelisting** • Restrict access to the Web App by **allowing only trusted IP addresses** or ranges through Azure's **Access Restrictions** feature. • **Virtual Network (VNET) Integration** • Integrate the Azure App Service with a **VNET** to isolate traffic and allow access only from specific subnets. • **Private Endpoints** • Enable **Private Link/Private Endpoints** to ensure traffic flows securely within the Azure network, avoiding exposure to the public internet.
SDL Phase:	Implementation

12.) An adversary may block access to the application or API hosted on Azure App Service Web App through a denial of service attack. [Priority: High]

CATEGORY:	**DENIAL OF SERVICE**
Description:	An adversary may block access to the application or API hosted on Azure App Service Web App through a denial of service attack
Possible Mitigation(s):	•**Azure DDoS Protection** •Enable **Basic or Standard Azure DDoS Protection** to automatically mitigate network-level denial of service attacks. •**Application-Level Throttling** •Implement **throttling** mechanisms at the application level (e.g., per user, per session, per API) to limit request rates and maintain service availability. •**API Management** •Use **Azure API Management** to control and monitor API traffic, including rate limiting, IP filtering, and security policies, to protect against DoS attacks.
SDL Phase:	Implementation

Interaction: Distribution Orders



13. An adversary can gain unauthorized access to Azure SQL database due to weak account policy. [Priority: High]

Category:	Elevation of Privileges
Description:	Due to poorly configured account policies, adversary can launch brute force attacks on Distribution Database
Possible Mitigation(s):	• **Azure Active Directory (AAD) Authentication** • Use **Azure Active Directory** authentication for SQL Database connections to eliminate the need for SQL authentication and improve account management and security. • **Least Privilege Principle** • Ensure that **least-privileged accounts** are used for accessing the SQL Database, granting only the necessary permissions for each account. • **Brute Force Protection** • Enable **SQL Database firewall rules** and **multi-factor authentication (MFA)** for users to mitigate brute-force attacks. Additionally, use **account lockout policies** to prevent repeated failed login attempts.
SDL Phase:	Implementation

14. An adversary can read confidential data due to weak connection. [Priority: High]

Category:	Information Disclosure
Description:	An adversary can read confidential data due to weak connection string configuration.
Possible Mitigation(s):	• **Secure Connection String Configuration** • Ensure the connection string includes **encrypt=true** and **trustservercertificate=false** to enforce encryption and verify the server certificate, protecting against man-in-the-middle attacks. • **Environment Variables for Sensitive Data** • Store connection strings and other sensitive information in **environment variables** or **Azure Key Vault** to prevent hardcoding credentials in application code. • **Least Privilege Database Access** • Ensure that the connection string is used with a **least-privileged account** that only has the necessary permissions to perform its tasks, minimizing the impact of potential exposure.
SDL Phase:	Implementation

15. An adversary can gain unauthorized access to Azure SQL DB instances due to weak network security configuration. [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary can gain unauthorized access to Azure SQL DB instances due to weak network security configuration.
Possible Mitigation(s):	•**Network Security Group (NSG)**: Use NSGs to define rules for controlling incoming and outgoing traffic to the Azure SQL Database, restricting access to trusted networks only. •**Virtual Network (VNet) Integration**: Implement Virtual Network service endpoints or private endpoints to ensure that traffic between your application and Azure SQL Database flows securely within the Azure network.
SDL Phase:	Implementation

16.An adversary having access to the storage container (e.g. physical access to the storage media) may be able to read sensitive data.[Priority: High]

Category:	Information Disclosure
Description:	An adversary having access to the storage container (e.g. physical access to the storage media) may be able to read sensitive data.
Possible Mitigation(s):	•**Enable Transparent Data Encryption (TDE)**: TDE ensures that data is encrypted at rest, making it unreadable without proper decryption keys. This protects sensitive data from unauthorized access even if the storage media is physically compromised. •**Use Always Encrypted**: With Always Encrypted, sensitive data is encrypted on the client side before it is sent to the database. This ensures that sensitive data is encrypted both in transit and at rest, and only authorized applications can decrypt it. •**Data Masking**: Implement dynamic data masking to limit the exposure of sensitive information to users who do not have the appropriate permissions to view the actual data.
SDL Phase:	Implementation

17.A compromised identity may permit more privileges than intended to an adversary due to weak permission and role assignments.[Priority: High]

Category:	Information Disclosure
Description:	An adversary can read confidential data due to weak connection string configuration.
Possible Mitigation(s):	• **Secure Connection String Configuration** • Ensure the connection string includes **encrypt=true** and **trustservercertificate=false** to enforce encryption and verify the server certificate, protecting against man-in-the-middle attacks. • **Environment Variables for Sensitive Data** • Store connection strings and other sensitive information in **environment variables** or **Azure Key Vault** to prevent hardcoding credentials in application code. • **Least Privilege Database Access** • Ensure that the connection string is used with a **least-privileged account** that only has the necessary permissions to perform its tasks, minimizing the impact of potential exposure.
SDL Phase:	Implementation

18.An adversary can gain long term, persistent access to an Azure SQL DB instance through the compromise of local user account password(s). [Priority: High]

Category:	**Information Disclosure**
Description:	An adversary can read confidential data due to weak connection string configuration.
Possible Mitigation(s):	• **Secure Connection String Configuration** • Ensure the connection string includes **encrypt=true** and **trustservercertificate=false** to enforce encryption and verify the server certificate, protecting against man-in-the-middle attacks. • **Environment Variables for Sensitive Data** • Store connection strings and other sensitive information in **environment variables** or **Azure Key Vault** to prevent hardcoding credentials in application code. • **Least Privilege Database Access** • Ensure that the connection string is used with a **least-privileged account** that only has the necessary permissions to perform its tasks, minimizing the impact of potential exposure.
SDL Phase:	Implementation

19. An adversary can deny actions performed on Distribution Database due to a lack of auditing. [Priority: Medium]

Category: Repudiation

Description: An adversary can deny actions performed on Distribution Database due to a lack of auditing.

Possible Mitigation(s):

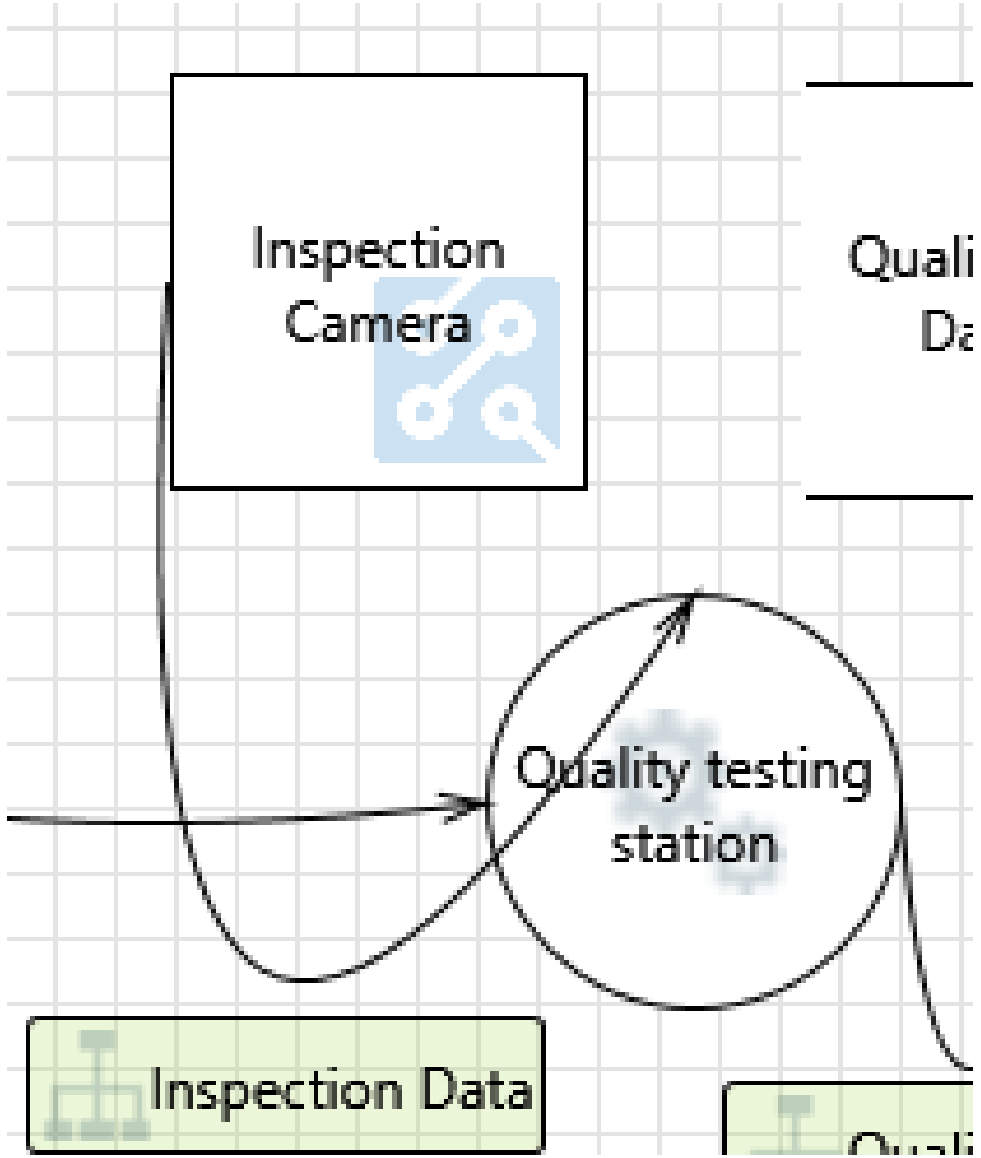
- **Enable Database Auditing**
- Enable **auditing** on Azure SQL Database to log all critical actions, providing a traceable history of events and reducing the risk of repudiation.
- **Threat Detection and Alerts**
- Configure **Azure SQL Database Threat Detection** to detect and alert on **anomalous activities**, ensuring that any suspicious behavior is flagged for immediate attention.
- **Log Monitoring and Review**
- Implement regular **log review** and monitoring processes to detect any unauthorized or malicious actions, ensuring actions on the database can be tracked and audited effectively.

SDL Phase: Design

20. An adversary may abuse weak Distribution Database configuration. [Priority: High]

- **Category:**Elevation of Privileges**Description:**An adversary may abuse weak Distribution Database configuration.**Possible Mitigation(s):Enable SQL Vulnerability Assessment**
- Enable **SQL Vulnerability Assessment** on Azure SQL Database to assess and identify security misconfigurations, vulnerabilities, and potential attack surfaces.
- **Apply Security Best Practices**
- Follow **Azure security best practices** to harden the database configuration, such as configuring proper firewalls, enforcing strong authentication, and minimizing permissions granted to users.
- **Regular Database Security Audits**
- Perform regular **database security audits** to ensure that any weaknesses or misconfigurations are identified and remediated in a timely manner.
- **SDL Phase:**Implementation

Interaction: Inspection Data



21.An adversary may tamper the OS of a device and launch offline attacks. [Priority: High]

Category:	Tampering
Description:	An adversary may launch offline attacks made by disabling or circumventing the installed operating system, or made by physically separating the storage media from the device in order to attack the data separately.
Possible Mitigation(s):	• **Full Disk Encryption** • Implement **Bitlocker** or similar encryption to encrypt the OS and any additional partitions of the IoT device, ensuring that data remains protected even if the device is tampered with or the storage media is separated. • **Secure Boot and Trusted Execution** • Enable **Secure Boot** to prevent unauthorized OS modifications by ensuring that only signed and trusted firmware is loaded during startup. • **Physical Security Measures** • Apply **physical tamper-evident mechanisms** to protect devices from unauthorized access, ensuring that any attempts to physically separate or tamper with the storage media are detectable.
SDL Phase:	Design

22.An adversary may tamper Inspection Camera and extract cryptographic key material from

it .[Priority:High]

Category:	Tampering
Description:	An adversary may partially or wholly replace the software running on Quality testing station, potentially allowing the replaced software to leverage the genuine identity of the device if the key material or the cryptographic facilities holding key materials were available to the illicit program. For example an attacker may leverage extracted key material to intercept and suppress data from the device on the communication path and replace it with false data that is authenticated with the stolen key material.
Possible Mitigation(s):	•**Hardware Security Module (HSM)**: Use HSMs to securely store cryptographic keys, ensuring they cannot be easily extracted or tampered with by unauthorized software. •**Encryption of Sensitive Data**: Ensure all sensitive data, including key material, is encrypted both at rest and during transmission to prevent interception and misuse.
SDL Phase:	Design

23. An adversary may exploit known vulnerabilities in unpatched devices. [Priority: High]

Category: Tampering

Description: An adversary may leverage known vulnerabilities and exploit a device if the firmware of the device is not updated

Possible Mitigation(s):

- **Automated Firmware Updates**
- Implement a robust process for the **automated updating of device firmware**, ensuring that all connected devices receive timely security patches and updates.
- **Cloud-Based Firmware Management**
- Leverage the **Cloud Gateway** to continuously monitor and manage the firmware versions of connected devices, ensuring all devices are compliant with the latest security updates.
- **Vulnerability Scanning**
- Regularly conduct **vulnerability scans** to identify any unpatched devices and promptly apply the necessary updates or patches to mitigate potential exploits.

SDL Phase: Design

24. An adversary may exploit unused services or features in Quality testing station. [Priority: High]

Category: Elevation of Privileges

Description: An adversary may use unused features or services on Quality testing station such as UI, USB port etc. Unused features increase the attack surface and serve as additional entry points for the adversary

Possible Mitigation(s):

- **Disable Unused Services and Features**
- Ensure that only essential services and features are enabled on the **Quality testing station**. Disable any unused services, ports, or interfaces (e.g., USB ports, UI features) to reduce the attack surface.
- **Least Privilege Principle**
- Apply the **least privilege principle** by limiting the access permissions for all services, ensuring that even if unused services are inadvertently enabled, they cannot be exploited.
- **Regular Security Audits**
- Perform **regular security audits** to identify and eliminate unused or unnecessary services or features, ensuring that the configuration is continuously secure.

SDL Phase: Implementation

25. An adversary may gain unauthorized access to privileged features on Inspection Camera. Priority: High]

Category: Elevation of Privileges

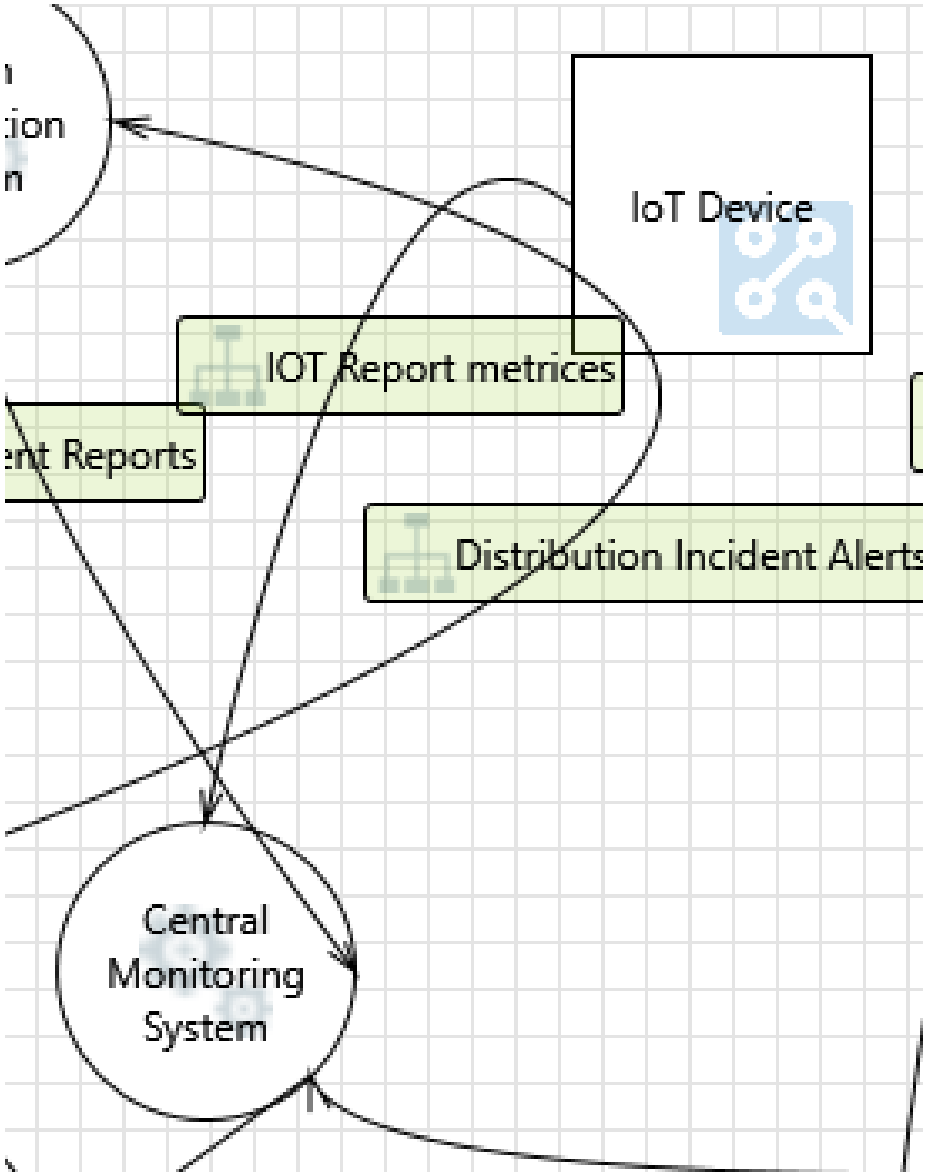
Description: An adversary may get access to admin interface or privileged services like WiFi, SSH, File shares, FTP etc., on a device

Possible Mitigation(s):

- **Secure Admin Interfaces with Strong Credentials**
- Ensure that all **admin interfaces** (e.g., Wi-Fi, SSH, file shares, FTP) are protected by **strong, complex passwords** and multi-factor authentication (MFA) wherever possible.
- **Limit Access to Privileged Features**
- Restrict access to **privileged services** by applying **role-based access control (RBAC)** and only allowing authorized users to perform administrative actions.
- **Regular Audits and Monitoring**
- Implement **auditing and monitoring** to track access to privileged features, and configure alerts for any suspicious activity related to admin interfaces or sensitive services.

SDL Phase: Implementation

Interaction: IOT Report metrices



26. An adversary may tamper IoT Device and extract cryptographic key material from it . [Priority: High]

Category:	Tampering
Description:	An adversary may partially or wholly replace the software running on Central Monitoring System, potentially allowing the replaced software to leverage the genuine identity of the device if the key material or the cryptographic facilities holding key materials were available to the illicit program. For example an attacker may leverage extracted key material to intercept and suppress data from the device on the communication path and replace it with false data that is authenticated with the stolen key material.
Possible Mitigation(s):	• **Key Storage Security**: Use **Hardware Security Modules (HSMs)** or **Trusted Platform Modules (TPMs)** to store cryptographic keys securely within the IoT device. • **Encrypted Storage**: Ensure that cryptographic keys are stored in **encrypted form** using strong algorithms and not in plaintext. • **Secure Boot**: Implement **secure boot** to ensure that only trusted, verified firmware can run on the device, preventing unauthorized software replacement.
SDL Phase:	Design

27. An adversary may exploit unused services or features in Central Monitoring System . [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary may use unused features or services on Central Monitoring System such as UI, USB port etc. Unused features increase the attack surface and serve as additional entry points for the adversary
Possible Mitigation(s):	• **Disable Unused Features**: Disable all unused services, features, and ports (such as UI, USB ports) to minimize the attack surface. Only enable the necessary components required for the system's functionality. • **Device Hardening**: Apply **device hardening** practices, such as disabling default accounts, removing unnecessary software, and setting up proper security configurations.
SDL Phase:	Implementation

28. An adversary may gain unauthorized access to privileged features on IoT Device . [Priority: High]

Category:	Elevation of Privileges
Description:	An adversary may get access to admin interface or privileged services like WiFi, SSH, File shares, FTP etc., on a device
Possible Mitigation(s):	• **Strong Authentication**: Secure all admin interfaces with **strong authentication mechanisms**, such as multi-factor authentication (MFA), to prevent unauthorized access to critical services like WiFi, SSH, File shares, FTP, etc. • **Access Control**: Implement **role-based access control (RBAC)** to limit access to admin features and sensitive services. Ensure that only authorized users or systems can access privileged interfaces. • **Encryption**: Use **encryption** (e.g., SSL/TLS for HTTP, SSH for terminal access) to secure communication between the device and external interfaces, protecting against man-in-the-middle attacks.
SDL Phase:	Implementation

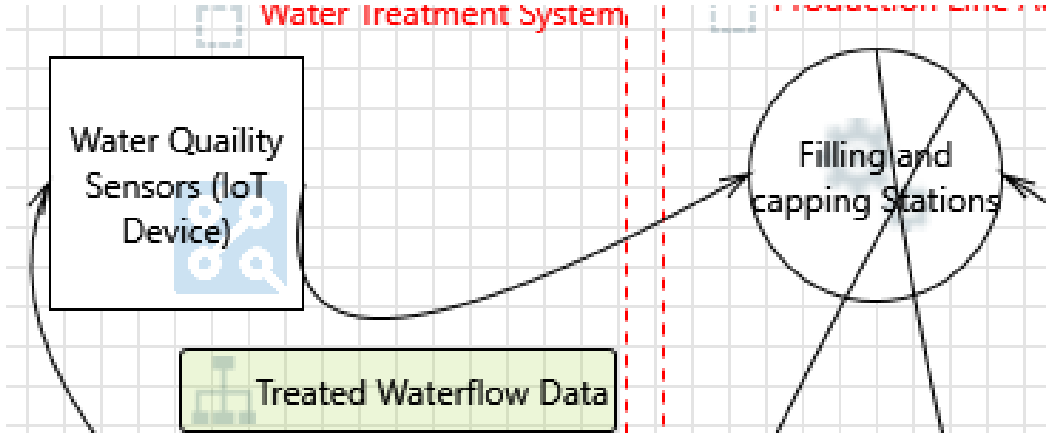
29.An adversary may tamper the OS of a device and launch offline attacks [State: Not Started] [Priority: High]

Category:	Tampering
Description:	An adversary may launch offline attacks made by disabling or circumventing the installed operating system, or made by physically separating the storage media from the device in order to attack the data separately.
Possible Mitigation(s):	•**Encrypt OS and Partitions**: Encrypt the IoT device's operating system and storage partitions using BitLocker or similar to protect data from unauthorized access if the device is tampered with or storage is separated. •**Secure Boot**: Implement secure boot to ensure the device only runs trusted software, preventing tampering with the OS or firmware. •**Tamper Detection**: Use tamper-evident hardware or sensors to detect physical tampering and set up alerts for unauthorized access.
SDL Phase:	Implementation

30.An adversary may exploit known vulnerabilities in unpatched devices [Priority: High]

Category:	Tampering
Description:	An adversary may leverage known vulnerabilities and exploit a device if the firmware of the device is not updated
Possible Mitigation(s):	•**Automated Firmware Updates**: Implement automated processes that ensure firmware on IoT devices is regularly updated to address security vulnerabilities and patch known issues. •**Vulnerability Scanning**: Use vulnerability scanning tools to identify outdated firmware versions and ensure compliance with the latest security patches. •**Cloud Gateway Integration**: Ensure the Cloud Gateway securely manages and pushes firmware updates to connected devices to keep them protected against emerging threats.
SDL Phase:	Implementation

Interaction: Treated Waterflow Data



31.An adversary may tamper Water Quaility Sensors (IoT Device) and extract cryptographic key material from it .[Priority: High]

Category:	Tampering
Description:	An adversary may partially or wholly replace the software running on Filling and capping Stations, potentially allowing the replaced software to leverage the genuine identity of the device if the key material or the cryptographic facilities holding key materials were available to the illicit program. For example an attacker may leverage extracted key material to intercept and suppress data from the device on the communication path and replace it with false data that is authenticated with the stolen key material.
Possible Mitigation(s):	• **Hardware Security**: Use **HSMs** or **TPMs** for secure key storage and **secure boot** for verified software loading. • **Key Management**: Implement **key rotation**, **expiration policies**, and **key wrapping** for encrypted storage. • **Firmware Security**: Apply **signed firmware updates** and ensure **encrypted update channels**.
SDL Phase:	Design

32.An adversary may exploit unused services or features in Filling and capping Stations.[Priority: High]

Category:	Elevation of Privileges
Description:	An adversary may use unused features or services on Filling and capping Stations such as UI, USB port etc. Unused features increase the attack surface and serve as additional entry points for the adversary
Possible Mitigation(s):	• **Disable Unused Services and Features Turn off unused services** (e.g., USB ports, UI features) to minimize the attack surface and prevent adversaries from exploiting them. • Use **configuration management tools** to regularly check and enforce the minimum set of active services. • **Apply Least Privilege Principle** • Restrict access to critical services, ensuring that only authorized personnel can interact with them. • Implement **role-based access control (RBAC)** to limit who can enable or interact with these services. • **Physical Security Controls** • Disable **USB ports** and other physical entry points through the device's firmware or hardware settings. • Secure access points to prevent unauthorized physical access to critical components.
SDL Phase:	Implementation

33. An adversary may gain unauthorized access to privileged features on Water Quaility Sensors (IoT Device).[Priority: High]

- **Category:**Elevation of Privileges**Description:**An adversary may get access to admin interface or privileged services like WiFi, SSH, File shares, FTP etc., on a device

Possible Mitigation(s):

Strong Authentication for Admin Interfaces

- Use **strong, unique credentials** for accessing admin interfaces and privileged services like WiFi, SSH, FTP, etc., to prevent unauthorized access.
- **Multi-Factor Authentication (MFA)**
- Enable **multi-factor authentication (MFA)** for all privileged interfaces to add an additional layer of security beyond just usernames and passwords.
- **Role-Based Access Control (RBAC)**
- Implement **Role-Based Access Control (RBAC)** to limit access to sensitive features only to authorized users, following the principle of least privilege.
- **SDL Phase:**Implementation

34. An adversary may tamper the OS of a device and launch offline attacks.[Priority: High]

Category: Tampering

Description: An adversary may launch offline attacks made by disabling or circumventing the installed operating system, or made by physically separating the storage media from the device in order to attack the data separately.

Possible Mitigation(s):

- **Full Disk Encryption**
- Implement **Bitlocker** or a similar full-disk encryption tool to encrypt the operating system and additional partitions of the IoT device to prevent unauthorized offline access to data.
- **Secure Boot**
- Enable **Secure Boot** to ensure that only authorized, signed firmware and OS are loaded, preventing tampering with the OS.
- **Physical Security**
- Use physical security measures such as **tamper-evident seals**, **secure enclosures**, or **locks** to prevent attackers from accessing the device storage directly.

SDL Phase: Design

35. An adversary may exploit known vulnerabilities in unpatched devices . [Priority: High]

Category: Tampering

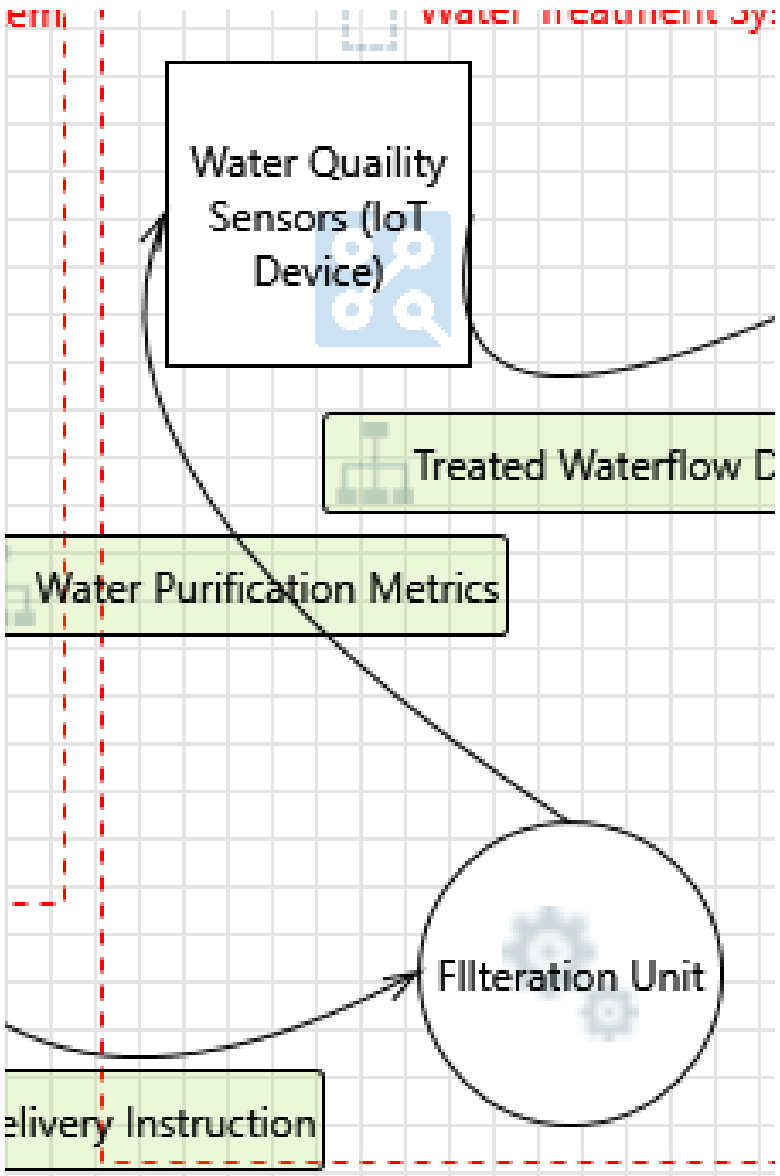
Description: An adversary may leverage known vulnerabilities and exploit a device if the firmware of the device is not updated

Possible Mitigation(s):

- **Automated Firmware Updates**
- Implement an **automated process** in the Cloud Gateway to ensure that connected devices are regularly checked for firmware updates and patched promptly.
- **Vulnerability Scanning**
- Perform **regular vulnerability scanning** on all devices to identify and address security flaws before they can be exploited.
- **Access Control and Device Authentication**
- Use strong **device authentication** and **access control** policies to limit unauthorized access to devices, especially in case of outdated firmware.

SDL Phase: Design

Interaction: Water Purification Metrics



36. An adversary may execute unknown code on Water Quality Sensors (IoT Device). [Priority: High]

Category: Tampering

Description: An adversary may launch malicious code into Water Quality Sensors (IoT Device) and execute it

Possible Mitigation(s):

- **Secure Boot and Firmware Validation**
  - Ensure the device only boots firmware that is cryptographically signed and validated to prevent unauthorized code execution.
- **Code Signing for Updates**
  - Enforce digital signatures for all software updates and patches to verify authenticity and integrity before installation.
- **Firmware Integrity Monitoring**
  - Implement runtime integrity checks using cryptographic hashes to detect and respond to unauthorized modifications.

SDL Phase: Design

THANK YOU