Good Practices for protecting Unmanned Aircraft Systems (UAS) against Cyber Security Threats

---

In collaboration with

Globals ITES Private Limited

Date: 18 April 2025

Table of Contents

This whitepaper titled 'Good Practices for protecting Unmanned Aircraft Systems (UAS) against Cyber Security Threats' has been written by The Indian Computer Emergency Response Team (CERT-In) in collaboration with Globals ITES Private Limited.

Definitions

Drone: A drone is a flying robot that can be remotely controlled or fly autonomously using software-controlled flight plans in its embedded systems, which work in conjunction with on-board sensors and a Global Positioning System (GPS).

Unmanned Aircraft Systems: Unmanned Aircraft Systems (UAS) is the totality of everything that makes a drone work including its GPS module, ground control module, transmission systems, camera, all the software, and the person on the ground controlling the drone.

Introduction

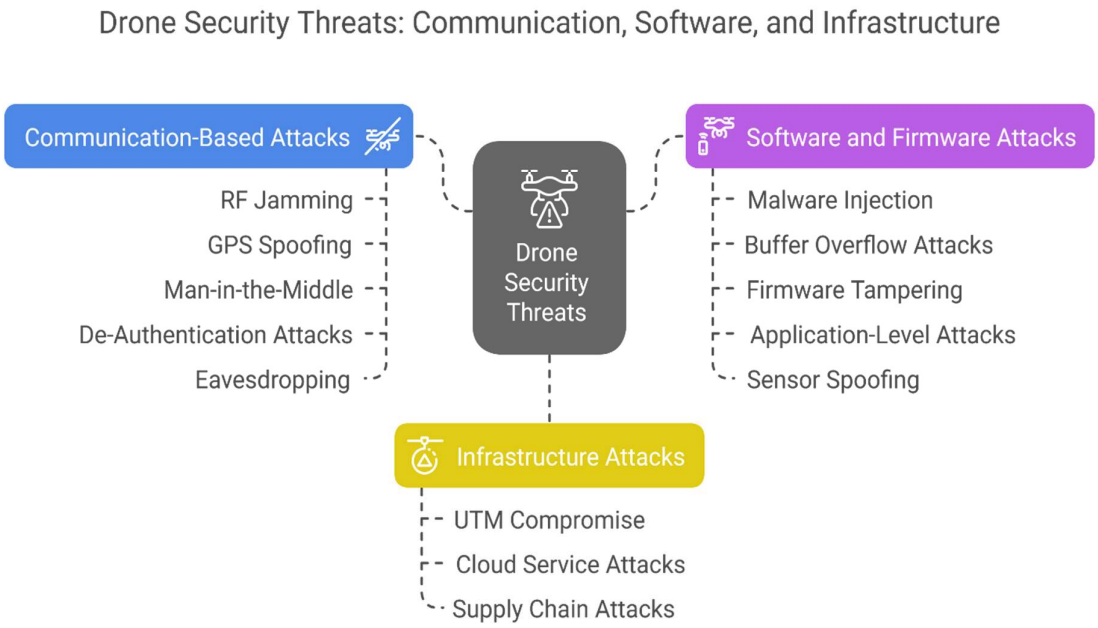
Unmanned Aircraft Systems (UAS) have experienced exponential growth in both commercial and recreational applications over the past decade. While initially limited to military applications, drones have now permeated numerous sectors including agriculture, infrastructure inspection, delivery services, and entertainment. This proliferation has been accompanied by increasing concerns regarding the cybersecurity posture of these systems and their vulnerability to attacks. As drones become more integrated into everyday operations, ensuring their cyber resilience has become paramount.

The cybersecurity landscape for drones is particularly complex because these systems represent a convergence of multiple technologies: aeronautics, wireless communications, sensing capabilities, and increasingly sophisticated software. This complexity creates numerous attack surfaces that malicious actors can exploit. Furthermore, as drones transition from standalone systems to connected nodes in larger networks-often referred to as the "Internet of Drones" (loD) the need for robust security measures becomes even more critical.

The rapid evolution of Unmanned Aircraft Systems (UAS) demand comprehensive cyber security posture to be adopted for safe and secure operations. There is a need for a multi-layered approach for UAS cyber resilience, encompassing secure-by-design principles, robust authentication mechanisms and a comprehensive incident response plan.

1. Threat Landscape for Unmanned Aircraft Systems (UAS)

The threat landscape for Unmanned Aircraft Systems (UAS), is evolving rapidly as their usage becomes more widespread. Some key aspects of the current threat landscape are:



1.1 Communication-Based Attacks

RF (Radio Frequency) Jamming: Deliberately broadcasting interference on the frequencies used by the drone's communication systems, preventing legitimate commands from reaching the aircraft or blocking telemetry data. Affected frequencies typically include bands commonly used for drone control and video transmission.

GPS (Global Positioning System) Spoofing: Transmitting false GPS signals to deceive the drone's navigation system, potentially causing it to fly to unintended locations or land in areas controlled by the attacker. This attack could exploit the unencrypted nature of civilian GPS signals.

Man-in-the-Middle (MitM): Intercepting and potentially altering communications between the drone and its controller, where unencrypted Wi-Fi communication is used.

De-Authentication Attacks: Forcing the disconnection of the drone from its legitimate controller by sending De-Authentication packets, particularly against Wi-Fi-connected drones.

Eavesdropping: Passively monitoring the data transmitted between the drone and GCS (Ground Control Station), potentially capturing sensitive video feeds, telemetry data, or command sequences.

1.2 Software and Firmware Attacks

Malware Injection: Installing malicious code on the drone's software/firmware systems leading to persistent malware injection, unauthorized access, data theft, or disruption of operations. Malware injections can be done through compromised updates or through gaining unauthorized physical access.

Exploitation of vulnerabilities: Exploiting vulnerabilities in the software such as improper input handling, memory corruptions, buffer overflows etc. potentially leading to remote code execution or denial of service.

Firmware Tampering: Modifying the drone's firmware to alter functionality, bypass safety controls or introduce vulnerabilities/backdoors.

Application-Level Attacks: Exploiting vulnerabilities in the companion applications used to control drones, which often run on smartphones or tablets.

Sensor Spoofing: Manipulating the inputs to the drone's sensors (beyond just GPS), such as using directed light to confuse optical sensors generating false readings for other navigational instruments.

1.3 Infrastructure Attacks

UTM (UAS Traffic Management) Compromise: Attacking the UTM systems to disrupt air traffic coordination or inject false information.

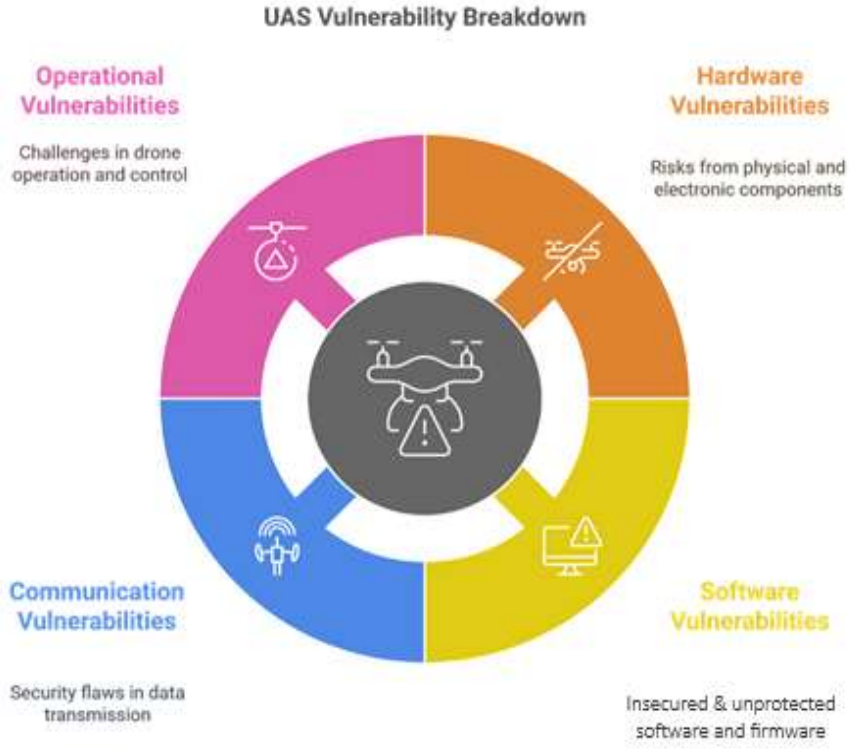
Cloud Service Attacks: Targeting the cloud-based services used by drone's platforms for services such as data storage, or firmware updates.

Supply Chain Attacks: Threat actors may infiltrate the manufacturing or distribution process, embedding malicious firmware or hardware components. This can lead to unauthorized access, remote exploits or data leaks potentially compromising the integrity and security of drone operations.

As drone technology rapidly develops, its potential applications are being reimagined, including its use in cyberattacks. Unauthorized capture of drones by malicious actors could result in their misuse to endanger people, hinder operations, or inflict damage on infrastructure.

2. Vulnerabilities in Current UAS Systems

There are some specific weaknesses inherent in the current UAS implementations across hardware, software, and communication systems.



2.1 Hardware Vulnerabilities

Electronic Speed Controller (ESC) Vulnerabilities: Some ESCs store firmware in volatile memory that requires uploading upon power-up, creating an opportunity for malicious firmware insertion.

Sensor Reliability Issues: Many commercial drones use consumer-grade sensors that lack hardware-level security protections and may be susceptible to manipulation or spoofing.

Physical Access Risks: Limited physical security protections on many commercial drones could allow attackers with physical access to extract memory contents or install hardware implants.

Inadequate Secure Boot Mechanisms: Lack of secure boot capabilities to verify the integrity of firmware before execution.

2.2 Software and Firmware Vulnerabilities

Unencrypted Storage: Critical configuration files and flight logs are often stored without encryption, allowing unauthorized access to sensitive data.

Insecure Update Mechanisms: Many UAS lack cryptographic verification of firmware updates, allowing potential installation of compromised firmware.

Default Credentials: UAS components usually ship with default passwords that users neglect to change, providing easy access for attackers.

Insufficient Input Validation: Many drone control applications lack proper validation of user and network inputs, creating opportunities for injection attacks.

Debugger and Development Backdoors: Development interfaces if left enabled in production firmware, could provide unauthorized access to internal systems.

2.3 Communication Vulnerabilities

Unencrypted Transmissions: Many consumer and some commercial UAS transmit control commands and telemetry data without encryption.

Weak Authentication: Inadequate or weak authentication mechanisms (such as vulnerable WPA2 (Wi-Fi Protected Access 2)) between drone and GCS (Ground Control Station) allow unauthorized access to the devices.

ADS-B (Automatic Dependent Surveillance-Broadcast) Insecurities: The ADS-B protocol used by some UAS for position reporting lacks adequate authentication, allowing spoofing of position data.

Cellular Network Vulnerabilities: UAS using cellular networks may be susceptible to IMSI (International Mobile Subscriber Identity) catchers and other known cellular network attacks.

2.4 Operational Vulnerabilities

Limited Resilience to Signal Loss: Many systems fail in an unsafe manner when communications are disrupted, rather than gracefully degrading their functionality.

Inadequate Geofencing Implementation: Geofencing restrictions could be bypassed through simple software modifications or GPS spoofing.

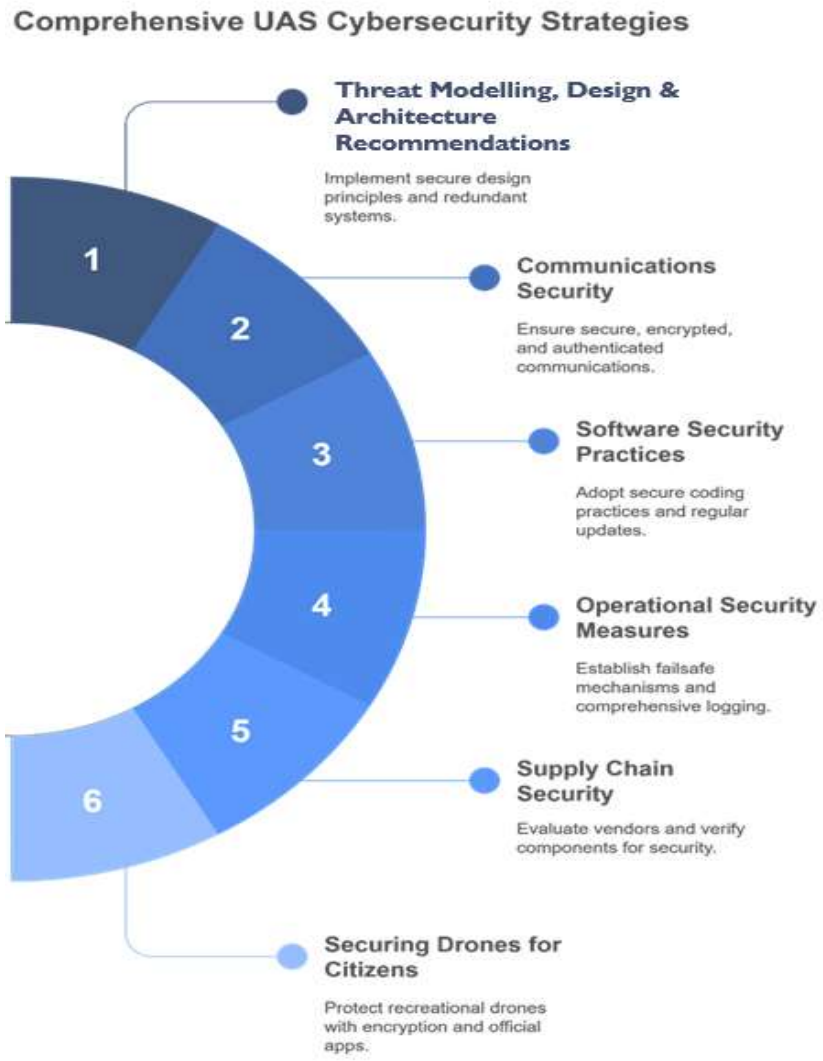
Remote ID Vulnerabilities: Emerging remote identification systems may expose drone operators to privacy risks or location tracking.

Insufficient logging: Systems lacking comprehensive logging capabilities may hinder forensic investigation.

These vulnerabilities highlight the need for improved security practices across the UAS ecosystem, from design and manufacturing to operational considerations.

3. Security Recommendations and Mitigation Strategies

Good practices recommended for enhancing UAS cybersecurity across various aspects of the system lifecycle are listed as follows:



3.1 Threat Modelling, Design and Architecture Recommendations

Threat modelling is the process of evaluating assets to identify potential vulnerabilities using theoretical scenarios, practical security techniques, system diagrams, testing methods, and tools. It also includes recommending corrective actions and policies to mitigate risks.

3.1.1 Threat Modelling Approaches

Multiple frameworks exist for modelling threats which could be adopted to UAS, each offering different perspectives on the security landscape.

3.1.1.1 PASTA Threat Modelling Framework

At the strategic level, to mitigate risks as a business problem consider PASTA (Process for Attack Simulation and Threat Analysis) threat modelling or similar methods. PASTA is a risk-centric threat modelling method, where the focus is on the highest and most relevant risks that can affect the business.

PASTA has seven distinct stages. Each stage feeds information into the next stage. Each stage adds to the information known about the object in scope, its business/technical environment, potential threats involved, and its risks.

The stages of PASTA threat modelling:

1. Identify assets and define the application's architecture.
2. Define the application's threat environment.
3. Decompose the application functionally and detail how attackers might exploit weaknesses.
4. Identify important attack scenarios.
5. Conduct a structured analysis of the identified attack scenarios
6. Identify possible threat agents.
7. Prioritize and mitigate the identified threats.

3.1.1.2 NIST Risk Assessment Framework

The NIST (National Institute of Standards and Technology) risk assessment framework helps determining likelihood of threat exploiting a vulnerability & assessing potential impact.

According to NIST 800-30, the basic steps for conducting a risk assessment are:

1. Identify Threat Sources and Events
2. Identify Vulnerabilities and Predisposing Conditions
3. Determine the Likelihood of Occurrence
4. Determine the Magnitude of Impact
5. Determine Risk

3.1.1.3 STRIDE Framework

The STRIDE framework provides a structured approach to identifying and mitigating specific categories of security threats:

a) Spoofing: Impersonating another user or system.
b) Tampering: Modifying data or code.

c) Repudiation: Denying having performed an action.
d) Information disclosure: Exposing information to unauthorized parties.
e) Denial of service: Making a system or resource unavailable.
f) Elevation of privilege: Gaining capabilities without proper authorization.

By combining multiple approaches and methodologies, organizations can build a robust threat model to pinpoint and mitigate potential vulnerabilities before they are exploited.

Secure-by-Design Principles: Implement security considerations from the earliest stages of drone development rather than as afterthoughts.

Defence-in-Depth Strategy: Deploy multiple layers of security controls to protect against failures of any single security mechanism.

Cyber Range: Implement UAS cyber ranges to conduct red-teaming exercises to simulate cyber-attacks and test for cyber resilience.

Isolation and Segmentation: Separate critical systems from non-critical ones, using hardware and software boundaries to limit the impact of compromises.

Minimized Attack Surface: Reduce unnecessary services, remove debugging interfaces, and disable unused ports in production systems.

Secure Boot Implementation: Ensure that only verified firmware can be executed through cryptographic validation during the boot process.

Hardware Security Modules: Integrate dedicated security chips for storing cryptographic keys and performing sensitive operations.

Redundant Systems: Implement redundancy for critical functions to maintain essential operations even if primary systems are compromised. Enable redundant navigation systems, such as visual and inertial navigation, to reduce reliance on GPS.

3.2 Communications Security

End-to-End Encryption: Implement strong encryption for all command and control communications as well as data transmissions. Encryption ensures that even if communication is intercepted, the data remains unreadable to unauthorized parties.

Strong Authentication: Use mutual authentication between all components of the UAS ecosystem, including between drones and controllers, drones and UTM systems, and within drone swarms.

Frequency Hopping: Employ spread spectrum techniques to enhance resistance to jamming and interception.

Signal Integrity Monitoring: Continuously monitor signal quality and patterns to detect jamming or spoofing attempts.

Secure Key Management: Implement robust processes for key generation distribution, storage, and rotation.

Bandwidth Reservation: When using cellular networks, utilize quality of service mechanisms to ensure UAS communications receive appropriate priority.

3.3 Software Security Practices

Organizations may refer 'Guidelines for Secure Application Design, Development, and Implementation & Operations'[1] and 'Guidelines on Information Security Practices for Government Entities'[2] which help them to establish a firm and robust application security baseline in application development lifecycle as well as procedures to protect their cyber infrastructure from prominent threats.

Secure Development Lifecycle: Follow established secure coding practices and conduct regular security reviews throughout development.

Regular Security Updates: Maintain a robust update mechanism for addressing vulnerabilities promptly, with cryptographic verification of updates.

Input Validation: Implement thorough validation of all inputs, especially those received over network interfaces.

Static and Dynamic Analysis: Conduct both static code analysis and dynamic testing to identify security flaws before deployment.

Vulnerability Management: Establish processes for tracking, prioritizing and addressing discovered vulnerabilities.

Secure APIs: Ensure that all application programming interfaces implement proper authentication, authorization, and input validation. Refer Best Practices for API security provided in Annexure.

3.4 Operational Security Measures

Failsafe Mechanisms: Implement appropriate responses to detected attacks, such as return-to-home functions when communication integrity is compromised.

Comprehensive Logging: Maintain detailed logs of all significant system events, protected from tampering, for security monitoring and forensic purposes.

Intrusion Detection Systems: Deploy monitoring systems capable of identifying abnormal behaviour patterns that may indicate attacks.

Restrict Unsafe Flying: Conduct vulnerability and risk assessment of the operating area and avoid flying drones in areas with known cybersecurity threats or high-risk environments. Conducting a security assessment of the operating area helps identify potential vulnerabilities.

Use secure communication for remote drone operation: Use secure communication channels to protect against interception. Avoid using open or unsecured Wi-Fi networks for drone operation.

Geofencing Enforcement: Implement cryptographically secured geofencing that cannot be easily bypassed.

Training and Awareness: Train personnel on cybersecurity awareness and threat mitigation techniques to reduce human error risks. Ensure that operators understand security risks and best practices for secure operation.

Regular Security Assessments: Conduct periodic penetration testing and security reviews to identify new vulnerabilities.

3.5 Supply Chain Security

Vendor Assessment: Evaluate the security practices of component suppliers and software providers. Ensuring that vendors follow strict cybersecurity standards reduces the risk of compromised equipment.

Component Verification: Implement processes to verify the authenticity of components before integration.

SBOM (Software Bill of Materials): Maintain comprehensive inventories of all software components, including third-party libraries. Conduct Firmware Reverse Engineering to assess firmware integrity and backdoors. Refer CERT-In guidelines on SBOM[3] released in October 2024.

HBOM (Hardware Bill of Materials): Document all hardware components to facilitate risk assessment and vulnerability management.

Secure Manufacturing Processes: Ensure that manufacturing facilities maintain appropriate security controls to prevent tampering.

By implementing these recommendations in a manner appropriate to specific use cases and risk profiles, UAS operators and manufacturers can significantly improve the security posture of their systems against the evolving threat landscape.

3.6 Securing Drones for Citizens (Recreational & Hobby Use) and Farmers

Social media vloggers/hobbyist drone users face risks such as hacking, privacy concerns, theft, and regulatory fines. While their security needs are less complex than industries, they should still follow essential best practices.

Enable Encryption: Many consumer UAS support video and telemetry encryption. Users should enable this feature to prevent unauthorized interception. Encrypting stored data ensures personal footage is safe even if the drone is stolen/lost.

Use Official Apps Only: Users should download drone control apps only from official sources (Google Play, Apple Store) to avoid malware-infected clones. Malicious apps can steal user credentials, hijack controls, or install spyware.

Use strong password: Use strong, unique passwords for drone controllers and applications to prevent unauthorized access. Also, consider using multi-factor authentication (MFA) where possible to add an additional layer of security.

Training and Awareness: Provide awareness and training to farmers at district levels on the use of drones and its cyber security threats.

4. Conclusion

The rapid advancement and widespread adoption of Unmanned Aircraft Systems, necessitates thorough and multi-layered mitigation strategies. The UAS ecosystem encompasses complex interactions between hardware, software, communications systems, and operational procedures, each of which may present exploitable vulnerabilities to malicious actors.

The threat landscape highlights that UAS face a broad spectrum of attack vectors, including disruption of communication systems, exploitation of software vulnerabilities and physical tampering. The motivations behind these attacks are equally diverse, ranging from criminal intent to APT (Advanced Persistent Threat) activities, underscoring the need for security approaches tailored to specific threat models and risk assessments.

As drones become more autonomous and interconnected, the security stakes continue to rise, particularly when these systems interact with public information infrastructure or operate in densely populated areas.

Key recommendations for enhancing UAS cybersecurity include:

- ✓ Conducting a risk based threat modelling
- ✓ Adopting secure-by-design principles from the earliest stages of development
- ✓ Implementing strong authentication and encryption for all communications
- ✓ Ensuring robust software security practices throughout the development lifecycle
- ✓ Deploying comprehensive monitoring and logging capabilities
- ✓ Establishing resilient operational procedures to respond to security incidents
- ✓ Maintaining rigorous supply chain security measures
- ✓ Staying informed about evolving threats and mitigation strategies

By taking a proactive, comprehensive approach to UAS cybersecurity that addresses the full spectrum of threats and vulnerabilities across the entire system lifecycle, stakeholders can help ensure that the remarkable potential of UAS technology can be realized without unacceptable security risks.

5. References

1. Guidelines for Secure Application Design, Development, Implementation & Operations
https://www.cert-in.org.in/PDF/Application_Security_Guidelines.pdf

2. Information Security Best Practices for Government entities
https://www.cert-in.org.in/PDF/guidelinesgovtentities.pdf

3. Technical Guidelines on SBOM
https://www.cert-in.org.in/PDF/SBOM_Guidelines.pdf

4. API Security: Threats, Best Practices, Challenges, and Way forward using AI, page 7-8.
https://www.cert-in.org.in/PDF/CIWP-2023-0001.pdf

Best Practices for API security

An API (Application Programming Interface) is a data connection allowing data to be shared with other applications. Some of the best practices to keep in mind when designing and creating APIs to protect against API attacks are:

1. Authentication and Authorization

- Use strong authentication mechanisms such as API keys, OAuth, or JWT (JSON Web Tokens).
- If using tokens like JWT, set appropriate expiration times and implement secure token management practices to prevent token misuse or replay attacks.
- Implement granular access control to limit API access based on user roles and permissions.
- Always validate user credentials and tokens before granting access to sensitive data.

2. API Gateway and Firewall

- Employ an API gateway for centralized security enforcement, monitoring, and management.
- Implement web application firewall (WAF) to protect against common web threats.

3. Data Protection and Secure Communication

- Encrypt sensitive data using appropriate encryption algorithms and key management.
- Apply data masking techniques to hide sensitive information in logs and responses.
- Use secure communication protocols to prevent eavesdropping and man -in-the-middle attacks.
- Employ secure headers and practices to prevent information leakage.

4. Input Validation and Sanitization

- All user inputs should be validated and sanitized to prevent injection attacks (e.g., SQL injection, XSS) and parameter manipulation.

5. Output Encoding

- Encode output to protect against HTML/JavaScript injection (XSS) and other data manipulation attacks.

6. Rate Limiting and Throttling

- Implement rate limiting and throttling mechanisms to prevent abuse of the API and DDoS attacks by limiting the number of requests from a single client within a specific time frame.

7. Error Handling and Logging

- Ensure proper error handling to avoid exposing sensitive information.
- Implement comprehensive logging for monitoring and auditing purposes.

8. Cross-Origin Resource Sharing (CORS)

- Configure CORS properly to restrict which domains can access the API from the client-side, thereby preventing unauthorized cross-origin requests.

9. Secure Storage of Secrets

- Store API keys, credentials, and sensitive data securely using encryption and access controls.

10. Regular Security Assessments

- Conduct regular security assessments of APIs such as penetration testing, security audits and code reviews to identify potential vulnerabilities and security flaws.

11. Education and Documentation

- Clear documentation should be provided containing steps to use the API securely, including examples of proper authentication and authorization methods.

12. Privacy Protection

- Minimize data collection and storage to only what is necessary.
- Comply with relevant privacy regulations and obtain user consent for data processing.
- Integrate privacy considerations from the initial stages of API development. Perform a Privacy Impact Assessment (PIA) to identify and mitigate potential privacy risks.

13. Secure Development Lifecycle (SDLC)

- Integrate security considerations into the entire API development process.
- Conduct security training for developers to raise awareness of secure coding practices.

Source: API Security: Threats, Best Practices, Challenges, and Way forward using AI[4].