Vulnerable App Threat Model

Owner: Richard Ndung'u
Reviewer:
Contributors:
Date Generated: Tue Sep 30 2025

Executive Summary

High level system description

This is a vulnerable web app threat model. It outlines the web app communication and threats prone to

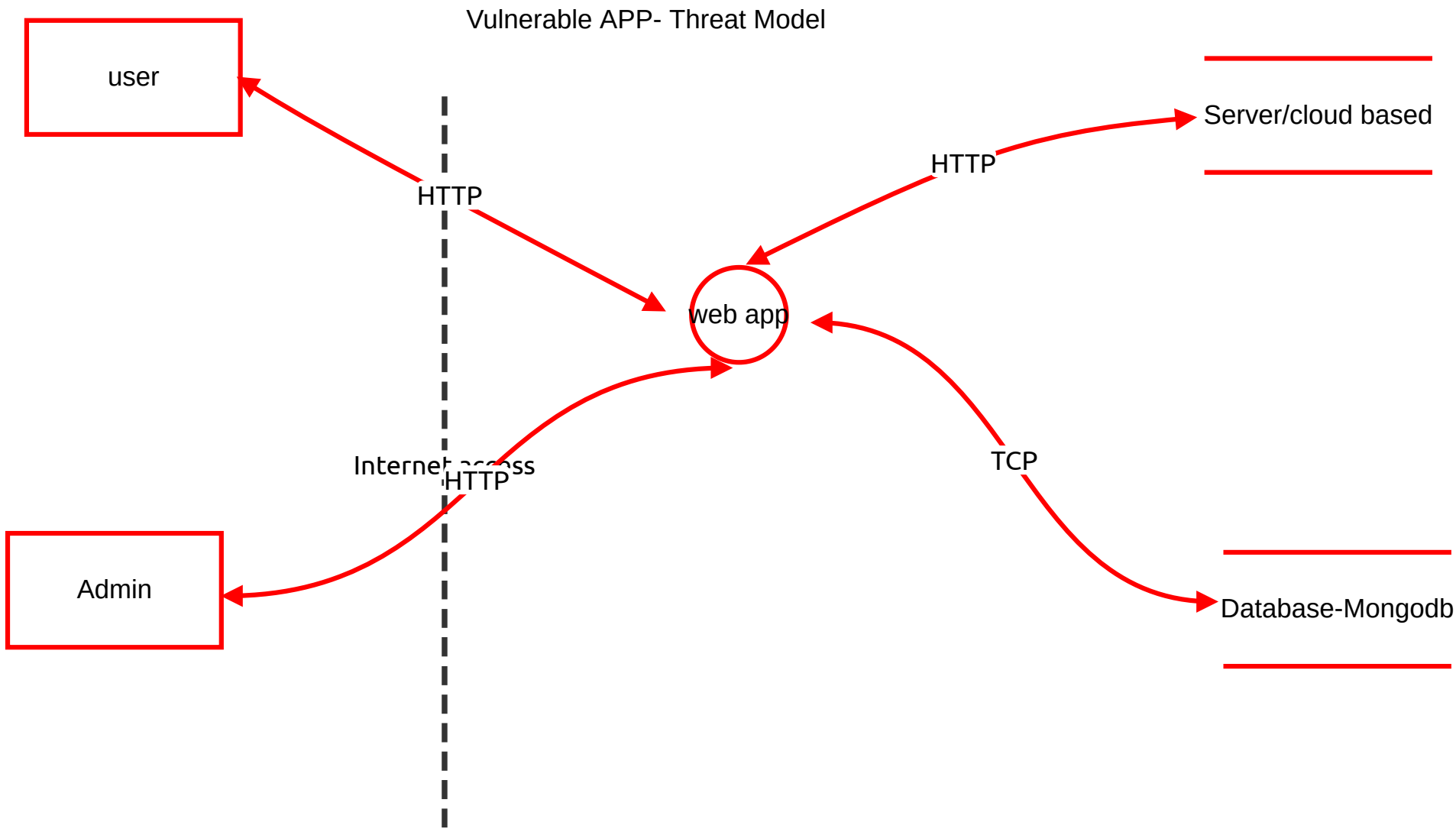
The system allows communication between:
Normal users (clients) who interact with the API through a browser.
High-level users (administrators) who access privileged endpoints such as user deletion.
The backend API which processes authentication, authorization.
The database (MongoDB) which stores sensitive assets such as user credentials, tokens, and profile information.
Supporting components like application logs and backups that may also hold sensitive data.

The model applies the STRIDE methodology (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privilege) to identify security threats across communication paths and trust boundaries. Each threat is mapped to its root cause and mitigation, showing where the application is prone to

Summary

Total Threats	24
Total Mitigated	0
Total Open	24
Open / Critical Severity	15
Open / High Severity	6
Open / Medium Severity	0
Open / Low Severity	0
Open / TBD Severity	3

New STRIDE diagram



New STRIDE diagram

web app (Process)

Description: The core service that handles authentication, user sessions, and business logic.

Issues JWT tokens, validates requests, and interacts with the database

Number	Title	Type	Severity	Status	Score	Description	Mitigations
13	Spoofing	Spoofing	Critical	Open		Spoofing: Accepting weak JWTs or tokens signed with leaked secrets.	Implement strong secrete policy and Implement RBAC Implement Rate limiting on log ins Implement input validation
14	Tampering	Tampering	Critical	Open		Tampering: Attackers could modify requests (e.g., role field) to bypass controls	Ensure strict HTTPS implementation
15	Repudiation	Repudiation	Critical	Open		Web users may deny involvement if logs are not implemented	Implement safe logs storage for audits
16	Information Disclosure	Information disclosure	Critical	Open		Information Disclosure: Poor error handling could expose stack traces or DB schema.	Ensure Error handing does not reveal backend schema
17	Denial of Service	Denial of service	High	Open		Denial of Service: No rate-limiting on login leads to brute-force and resource exhaustion.	Implement rate-limiting to ensure bots and brute force is not possible

user (Actor)

Description: Interacts with the web app through a browser or API

Registers, logs in, and accesses personal data via the API

Number	Title	Type	Severity	Status	Score	Description	Mitigations
1	The Application is Vulnerable to Spoofing Attack	Spoofing	Critical	Open		Threat Actors can gain access of other users information since the application is vulnerable to IDOR	Implement (Role based Access Control) RBAC. By this user A can not get User B information
10	Repudiation: User may deny malicious actions	Repudiation	High	Open		Repudiation: Normal users may deny malicious actions if proper audit logging is missing. It will not be possible to trace the users without proper audit logs	System log audit is recommended , app logs to be stored safely in case an audit is required.

Admin (Actor)

Description: High level privileges,can view users and delete them

Number	Title	Type	Severity	Status	Score	Description	Mitigations
5	Admin Spoofing	Spoofing	Critical	Open		Threat actor can gain admin privileges by modifying JWT token which has a weak secrete key and has been hard-coded. Threat actors can get access and delete users from the data base	Implement strict JWT secrete management policy Implement strict RBAC
12	Repudiation	Repudiation	High	Open		Admins may deny their action the web app	Enable safe logs storage

Server/cloud based (Store)

Description: Provides a platform for back-end & front-end to run

Number	Title	Type	Severity	Status	Score	Description	Mitigations
6	Server tampering	Tampering	TBD	Open		Threat actors can tamper with server running the application; Misconfiguration or exposed ports allow remote modification of services.	Ensure the servers ports are not open an configurations if on cloud are done properly
18	Information Disclosure	Information disclosure	High	Open		Information disclosure: Poorly configured servers may leak users information	Ensure server configurations are secure,only open necessary ports

Database-Mongodb (Store)

Description: Stores users data like names ,email ,passwords

Number	Title	Type	Severity	Status	Score	Description	Mitigations
3	Information Disclosure	Information disclosure	High	Open		The data base leaks users information, user email, passwords, user id	Provide remediation for this threat or a reason if status is N/A
4	DOS	Denial of service	High	Open		Threat actors can prevent users from accessing their information from the data base	Implement RBAC
7	Tampering	Tampering	Critical	Open		Tampering: Exposed or unsecured DB allows attackers to alter or delete data.	Implement RBAC

HTTP (Data Flow)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
27	Tampering	Tampering	Critical	Open		Tampering: Misconfigured firewall allows malicious inbound traffic.	Use Firewalls both in cloud and local host
28	DOS	Denial of service	Critical	Open		Denial of Service: Open ports on local host,database may experience Denial of service	Ensure localhost or cloud are configured not to expose open ports or sensitive routes Monitor traffic for anomalies and block malicious IPs. Place server behind a reverse proxy / WAF (e.g., Nginx, Cloudflare).

TCP (Data Flow)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
24	Tampering	Tampering	TBD	Open		Tampering: If DB is exposed to the internet, attackers can alter records directly.	Restrict DB access to localhost or private subnet only. Enable authentication on MongoDB with strong credentials.
25	Information disclosure	Information disclosure	TBD	Open		Information Disclosure: Weak password storage (plaintext or MD5) leaks sensitive data.	Store passwords using bcrypt or Argon2 with salt. Implement input validation to prevent NoSQL injection
26	DOS	Denial of service	Critical	Open		Denial of Service: Un optimized queries or injection attacks could lock DB resources.	Implement Rate limiting to access databases

HTTP (Data Flow)

Description: User communication with the API

Number	Title	Type	Severity	Status	Score	Description	Mitigations
19	Tampering	Tampering	Critical	Open		Tampering: Requests (like role=admin) can be modified in transit without TLS.	use HTTPS to implement TLS that encrypts communication over the internet.
20	Information disclosure	Information disclosure	Critical	Open		Information Disclosure: Credentials (email, password) leak if sent over HTTP.	Use HTTPs Ensure back end does not return users information
21	DOS	Denial of service	Critical	Open		Denial of Service: Repeated brute-force login attempts overwhelm the login endpoint.	Implement rate limiting

HTTP (Data Flow)

Description: Admin access to users information and endpoints

Number	Title	Type	Severity	Status	Score	Description	Mitigations
22	Tampering	Tampering	Critical	Open		Tampering: Without proper RBAC checks, attackers can manipulate admin requests.	Implement strict RBAC Use least privilege: separate admin and normal user accounts.
23	Privilege Escalation	Information disclosure	Critical	Open		Elevation of Privilege: If a normal user tampers with a JWT, they could call admin endpoint	RBAC implementation