Threat Modeling: A Developer's Approach

Dinesh Kumar Mohanty, Dragan Peraković
Research Scholar, Full-time professor,
Swiss School of Business Management,
kiitkp03@gmail.com, dragan.perakovic@fpz.unizg.hr

***Abstract*— Threat modeling is a process of identifying vulnerabilities during the design phase, where security engineers identify these vulnerabilities by analyzing data flow diagrams (DFD) and/or sequence flow diagrams (SFD). In this research paper we will be talking about real-time test cases and possible methodologies of identifying vulnerabilities with an example, which will help developers to identify vulnerabilities when they design the product flow. We will discuss possible remediation techniques that can be adopted by the developers when developing the product.**

***Index Terms*— Threat Modeling, Confidentiality, Integrity, Availability, Authentication, Authorization, Accountability**

I. INTRODUCTION

Threat modeling is a hypothetical approach to system design that aligns with the secure development life cycle (SDLC) using which developers developed a secure application; it is sometimes called the proactive approach of securing an application from different types of vulnerabilities, which can be adopted at different stages, like during the design phase, incremental developments, or even after development is completed [1].

The aim of the threat modeling process is to get a clear picture of various assets of the organization, the possible threats to these assets, and how and when these threats can be mitigated [2]. The end product of threat modeling is a robust security system [2].

CIA-AAA – Core concept for Developers

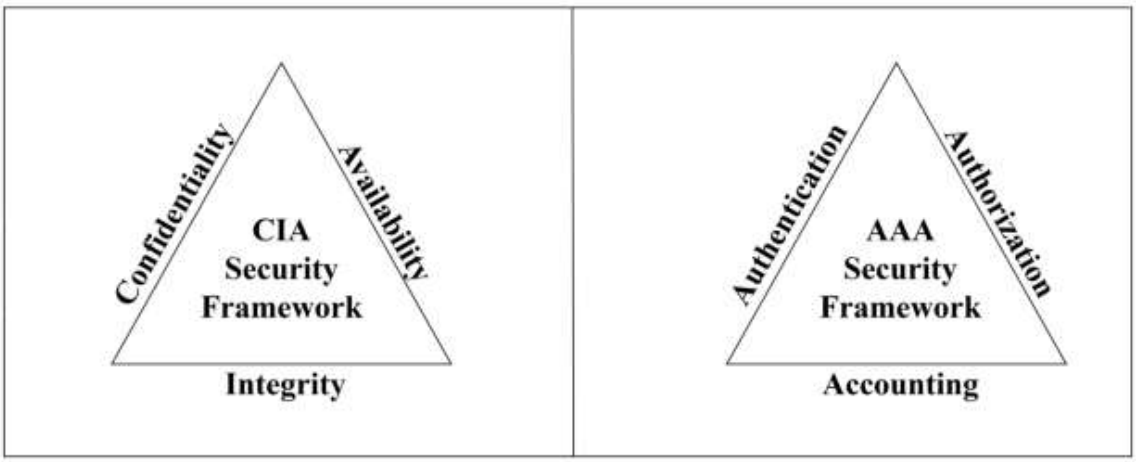

Figure 1: CIA-AAA

1. Confidentiality: In confidentiality we make sure that the data being transmitted between source and destination is secured by maintaining its secrecy. To achieve this, data gets encrypted so that if a malicious user is able to get the data by any means, like a man-in-the-middle attack, they cannot decrypt it, as the key that is used here (public and private key) is not with the attacker and is only accessible to the intended user and secure from attackers [3].

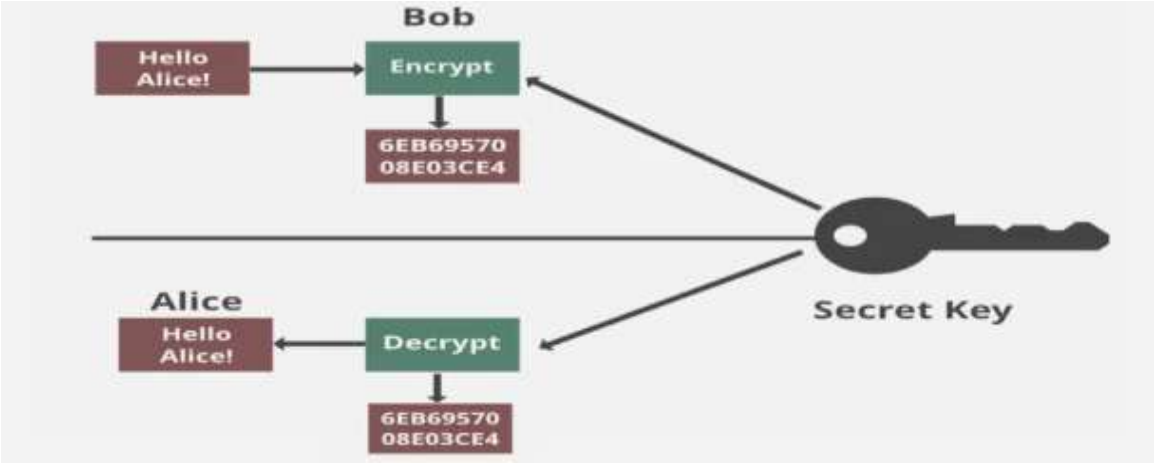

Figure 2: Confidentiality

2. Integrity: It makes sure that when received at the receiver's end, the data is not tampered with or altered by an attacker during transit or at rest [3].

3. Availability: It makes sure that when data is delivered at the receiver's end, the receiver is able to access it whenever required, which is mainly used against threats/attacks like DDOS and ransomware. One example would be when we release a product publicly, we release it with a hash of the file that makes sure that when downloaded by the users, it is not tampered with.

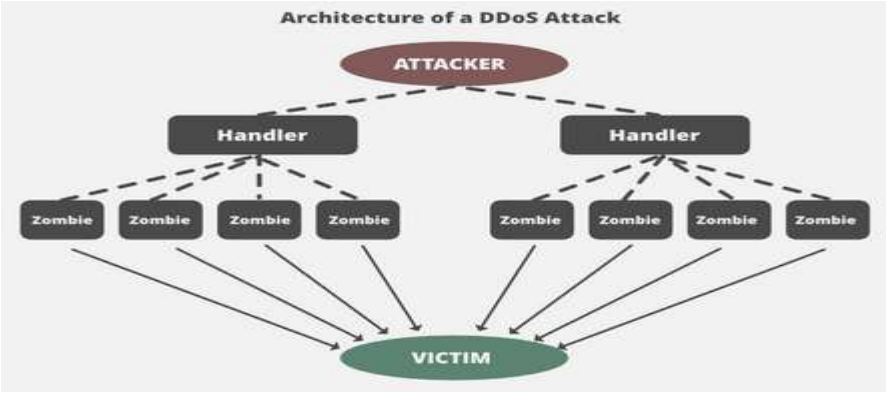

Figure 3: Availability

1. Authentication (Access Control): Verification and identification of the user or server attempting to gain access to resources. It's like showing your ID to get into a club. Example: Passwords, Certificates, Biometrics [5].
2. Authorization**:** According to [5]. Granting permissions or access rights to authenticated users based on policies. It's like getting a special pass to access certain areas.
3. Role-Based Access Control (RBAC): This provides a user access based on mapped roles, for example, like student and teacher access on a learning portal.
4. Attribute-Based Access Control (ABAC): This decides what actions you can perform when authenticated to a system or application. For example, teachers can set questions, and students can answer based on the question type.
5. Rule-Based Access Control: Here, rules set by administrators decide what you can access, usually based on conditions they have set. For example, an HR management portal where aspirants apply for the job but they cannot post any new jobs like HR can do.

6. Least Privilege Principle: This means only giving you access to what you need to do your job, with no extra privileges.
7. Accountability/Auditing**:** Tracking and recording user activities for monitoring, compliance, and forensic purposes. It's like a security camera watching who goes in and out. Example: Logging, Auditing, Reporting, and Compliance. [5].

CIA is all about protecting the most important parts of information during in-transit or at-rest, whereas AAA is all about making sure that proper authentication, access control (authorization), and accountability (auditing) are implemented within a system or asset. Companies use both models to validate that security systems are well capable of protecting data and to make sure that systems are used responsibly.

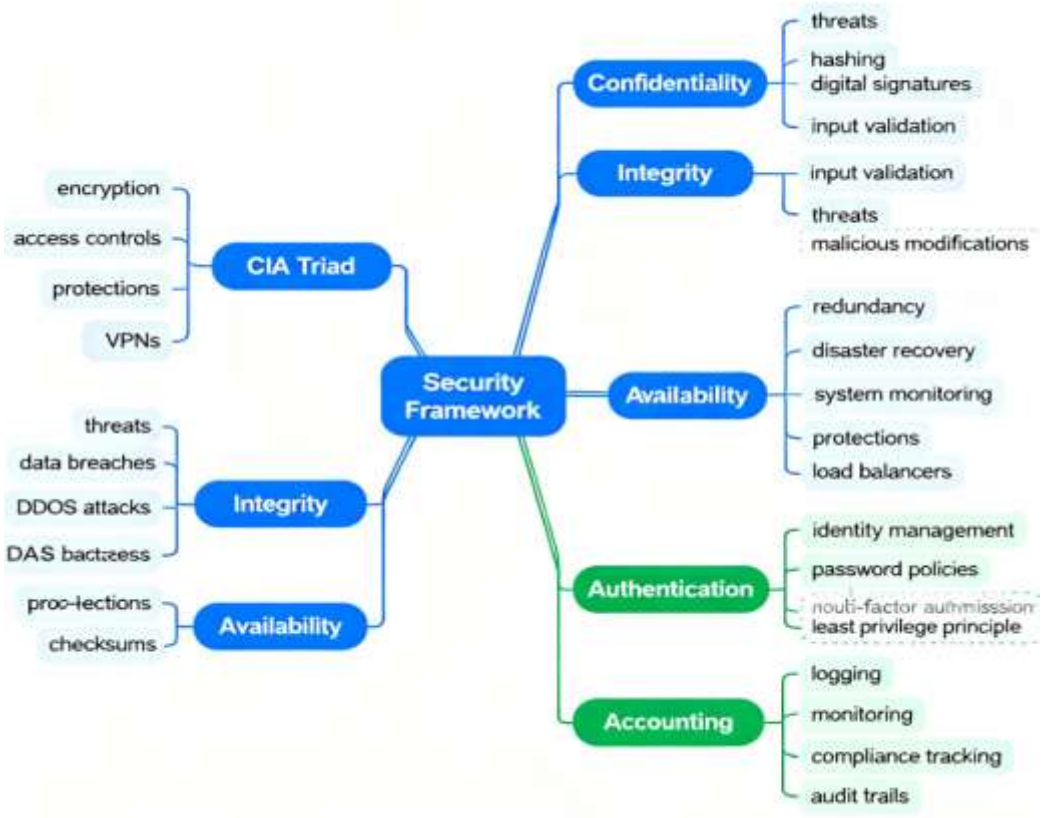

Figure 4: CIA-AAA Security Framework Examples

II. METHODOLOGY

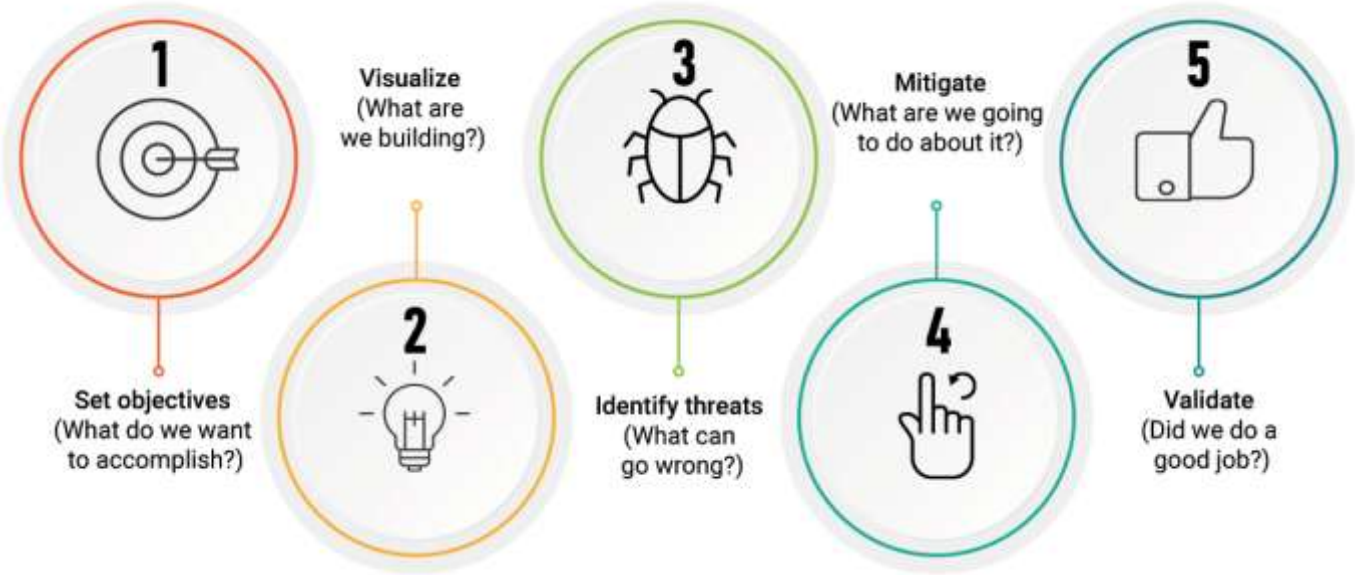

Figure 5: Methodology

Security engineers proactively work with the development team to identify threats and vulnerabilities in an application, system, and networks during the architecture design phase, and it can be accomplished at various stages, like the initial phase, during feature releases, or incremental stages, etc. During threat modeling the question usually asked is:

- What are we building? (Understanding product architecture, its components, and data flow between components.)
- What can go wrong? (This is the phase where we Identify threats and vulnerabilities based on the design.)
- What are we doing to protect it? (Define various security controls that can prevent attacks.)
- Did we do a good job? (One done validate mitigations via the penetration testing process and reassessing as and when needed.)

III. RESULT DISCUSSION

Threat modeling helps prioritize risks based on their likelihood and impact, reduces costly rework by addressing vulnerabilities at an early stage in development, ensures compliance with regulations (e.g., GDPR, PCI-DSS, HIPPA etc.,) and enhances team awareness of security best practices [6]. There are 13 main threat modeling methods present for identifying vulnerabilities, and they are:

1. PASTA, which stands for Process for Attack Simulation and Threat Analysis, is a risk analysis-based approach to identifying vulnerabilities that involves 7 stages (objectives of the assessment, scope of the assessment, application decomposition, threat identification, vulnerability analysis, attack simulation, and risk analysis) [7].
2. DREAD, whose full form is Damage Potential, Reproducibility, Exploitability, Affected Users, and Discoverability, is a framework that focuses on threat prioritization on these 5 criteria [8].
3. TRIKE (It is a risk-based approach that mainly focuses on business, operational, and technical aspects and the data flow of assets).
4. VAST (Visual, Agile, and Simple Threat) used techniques for identification and assessment of threats.
5. LINDDUN—It is a privacy-focused threat modeling framework that stands for Linkability, Identification, Non-repudiation, Detection, Disclosure of Information, Unawareness, and Non-compliance [9].
6. CVSS (Common Vulnerability Scoring System) is used to identify vulnerability with attributes like attack vector, complexity, privileges required, user interaction, scope, confidentiality, integrity, and availability; by selecting the values for these attributes, a score is generated based on which vulnerability's severity is defined [9].
7. OCTAVE stands for Operationally for Critical Threats, Assets, and Vulnerability Evaluation, which is a risk management framework that is mostly aligned with security and organizational goals [10].

8. Attack Trees—A formal, methodological way of describing the security of a system, based on various attacks. It is a representation of attacks against a system in a tree structure.
9. Security cards—It uses a deck of 42 cards to facilitate threat discovery activities: Human Impact (9 cards), Adversary's Motivations (13 cards), Adversary Resources (11 cards), and Adversary's Methods (9 cards) [11].
10. hTMM—The Hybrid Threat Modeling Method (hTMM), developed by the Software Engineering Institute in 2018, consists of a combination of SQUARE (Security Quality Requirements Engineering Method), Security Cards, and PnG activities [11]. The targeted characteristics of the method include no false positives, no overlooked threats, a consistent result regardless of who is doing the threat modeling, and cost-effectiveness [11].
11. Quantitative Threat Modeling Method—This method uses quantitative data to identify potential security threats, which involves gathering data on the assets, risks, threats, and vulnerabilities associated with a system or application [12]. This information is then analyzed, and a quantitative risk score is assigned to each potential threat that helps to prioritize potential threats based on their risk level and allocate resources accordingly [12].
12. Persona Non Grata (PnG)—PnGs represent archetypal users who behave in unwanted, possibly nefarious ways [13]. However, like ordinary personas, PnGs have specific goals that they wish to achieve and specific actions that they may take to achieve their goals. Modeling PnGs can therefore help us to think about the ways in which a system might be vulnerable to abuse and use this information to specify appropriate mitigating requirements [14]. The PnG approach makes threat modeling more tractable by asking users to focus on attackers, their motivations, and abilities. Once this step is completed, users are asked to brainstorm ideas about targets and likely attack mechanisms that the attackers would deploy [14]. The theory behind this approach is that if engineers can understand what capabilities an attacker may have and what types of mechanisms they may use to compromise a system, the engineers will gain a better understanding of targets or weaknesses within their own systems and the degree to which they can be compromised [14].
13. STRIDE is a threat modeling framework developed by Microsoft, which is our main focus in this paper, and sometimes security engineers call it the developer-centric approach of CIA-AAA for threat modeling.

Table—Stride Framework

Type of Threat	CIA-AAA Mapping	Explanation	Real-World Example
Spoofing	Authentication	Impersonating to be another user to gain system access	A fake login page of an original page to capture user credentials.
Tampering	Integrity	Un-AuthZ alteration of data or a system behavior.	Altering a database entry to change prices of a product in an e-commerce application.
Repudiation	Non-Repudiation	Denying any responsibility of an action due to lack of accountability	A user deleted a critical file and, when asked, denies that he performed the action, because audit logs are missing.
Information Disclosure	Confidentiality	Exposing of sensitive information (data) to unauthorized parties (say guest user)	An unsecured API endpoint exposing sensitive information without authorization.
Deniel of Service	Availability	Overloading a system to make it unavailable when required	Botnets attack a website with N number of requests, making it inaccessible to legitimate users (down).
Elevation of Privilege	Authorization	Gaining unauthorized high-level user access by exploiting a new or existing vulnerability	Exploiting a security misconfiguration vulnerability within an application to access admin or any higher user functionalities

Threat Modeling with an Example:

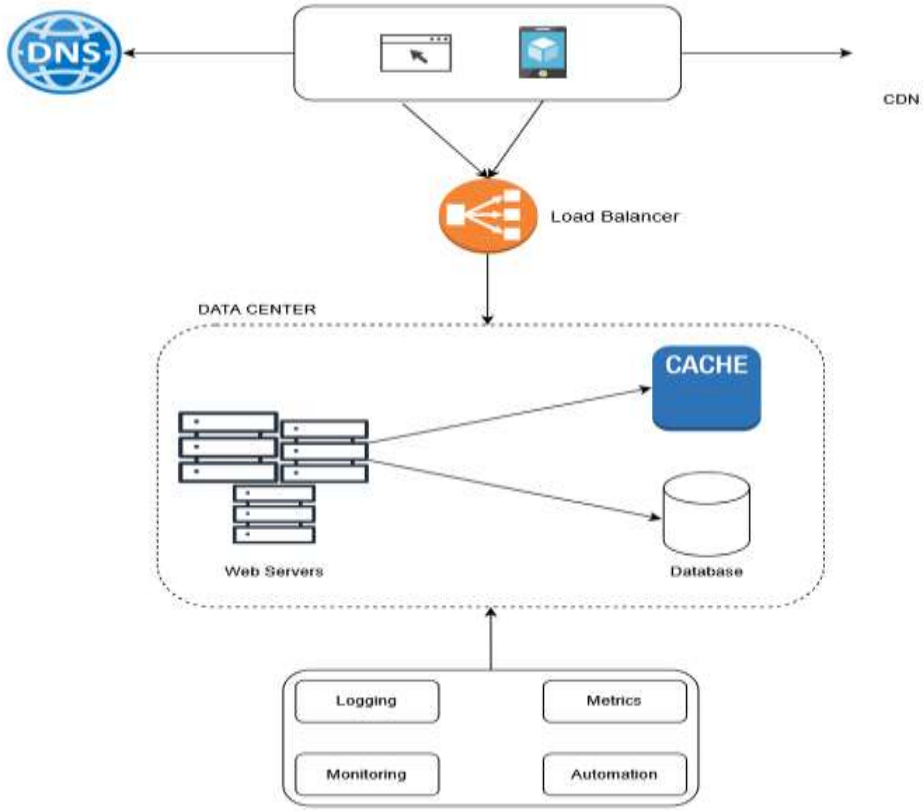

Figure 6: Application Architecture Diagram

This is the basic architecture diagram that explains how users connect to an application from a web or mobile application and the backend assets being used to set it up. In this flow diagram we need to identify assets, controls, and threats, or threat mapping.

1. Assets—These are the core components that an organization strives to protect. Components are both physical and digital assets. The more critical the asset is, the higher the probability of protecting it against potential threats.
2. Controls—Security mechanisms and policies implemented to protect the assets from various possible threats. These controls can be categorized into three primary types:
3. Administrative controls: These include policies, procedures, and governance frameworks, e.g., access management policies, security training, and risk assessments [15].
4. Technical controls: Used to protect digital assets such as firewalls, encryption, intrusion detection systems, and multi-factor authentication [15].
5. Physical controls: Used to protect physical assets and facilities like locks, surveillance systems, security personnel, and perimeter defenses, fences [15].
6. Threat mapping—It is the process of finding potential threats/vulnerabilities in the assets and controls in place; it also helps in assessing the likelihood of each threat occurrence and the potential impact it would have on the organization's operations. By mapping threats to assets, it helps security teams to prioritize the fixing of a vulnerability [15].
7. Trust Boundary—It is a logical perimeter within which components of a system are present that either store data or execute the process across the boundaries of a given network, system, or application (and services) [15].
8. Mapping of Assets: As per their names, the mapping happens, meaning Assets are mapped as A1, A2, …A(N), controls are mapped like C1, C2…C(N); and similarly, threats are mapped as T1, T2…T(N).

Table *5.2.2*

Asset Identification from Architecture Diagram

Asset ID	Asset	*Description*
A1	DNS Server	Resolves domain names to IP address
A2	Load Balancer	Distributes incoming traffics across multiple servers
A3	Web Application	Provides applications' core functionalities over Web
A4	Mobile Application	Allows users to access applications via mobile
A5	CDN (Content Delivery Network)	Caches static content such as JS and CSS codes close to users' area.
A6	Web Servers	Hosts web application, API services and processes user requests
A7	Cache Server	Speed up responses
A8	Database	Used to store user data such as login credentials, user PII information, etc.
A9	Datacenter	Physical infrastructure providers for housing servers, cache, CDNS
A10	Logging and Monitoring Tools	Uses to track metrics, storing user logs.

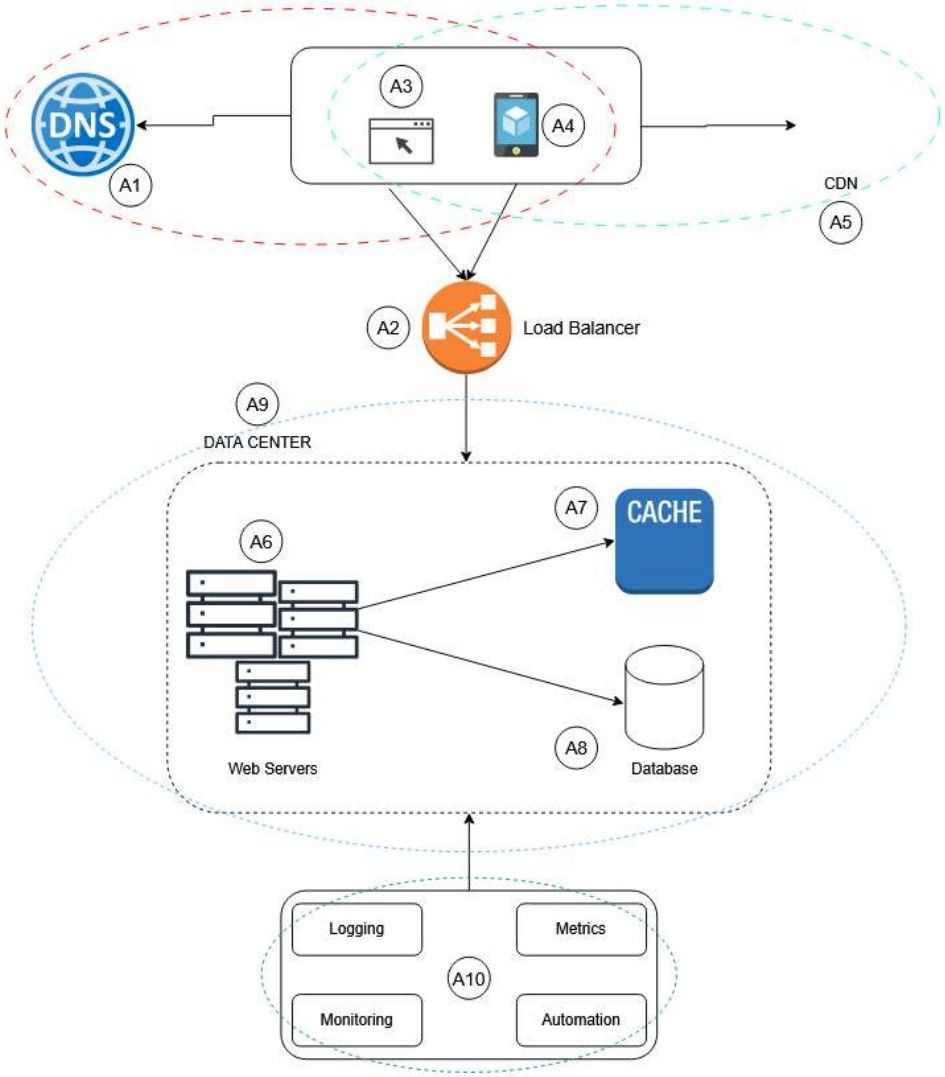

Figure 7: Application Architecture Diagram with Assets and trust Boundaries

TABLE 5: CONTROLS IDENTIFICATION FROM ARCHITECTURE DIAGRAM

Asset ID	Asset	*Description*
C1	DNSSEC	Cryptographic authentication to DNS responses, which ensures integrity and authenticity of DNS data to prevent spoofing and MITM attacks.
C2	DNS Filtering	Blocks access to malicious or unauthorized domains
C3	DNS Rate limiting	Limits the number of requests made to DNS servers by an IP address, which further prevents DDOS attacks
C4	SSL/TLS Encryption	This ensures that sensitive information that is transmitted is secure (encrypted during transit), preventing eavesdropping, data tampering, and MITM attacks.
C5	Authentication	Validating the authenticity of the servers interacting with CDN, which prevents DNS hijacking and spoofing.
C6	Access control policies	Defines and enforces access control policies for who can access cached content on edge servers. Policies can be user roles, country-based access, etc.
C7	Content Security Policies	Implementing CSP headers can mitigate web application vulnerabilities like XSS, data injection attacks, etc.
C8	Access Control	This ensures secure access to critical components like web servers, cache, database, etc.
C9	Network Segmentations	Segmentation of network into smaller, isolated environment preventing lateral movements
C10	Encryption	Encrypting sensitive data during in transit and at rest. Example AES encryption.
C11	Patching and Updates	Ensures that security vulnerabilities were patched on regular interval both on 3rd party libraries and on Operating systems
C12	Data backup and recovery	Backup ensure if any disaster happens, then we can store data from a backup server for availability.
C13	Monitoring and Logging	Ensures real-time detection and anomalies through log analysis and monitoring, which improves security incidents and incident response time

Asset	Spoofing (T1)	Tampering (T2)	Repudiation (T3)	Information Disclosure (T4)	Denial of Service (T5)	Elevation of Privilege (T6)
DNS (A1)	DNS cache poisoning (T1) - **C1, C2**	Malicious redirection (T2) - **C1, C2**	Lack of DNS logging (T3) - **C3**	Exposure of DNS queries (T4) - **C3**	DNS amplification attacks (T5) - **C3**	Not applicable
Load Balancer (A2)	IP spoofing (T1) - **C4**	Forged traffic routing (T2) - **C4**	Insufficient logging (T3) - **C11**	Intercepting traffic (T4) - **C4, C5**	Flooding attack (T5) - **C4**	Misconfigured access control (T6) - **C8**
Web App (A3)	Session hijacking (T1) - **C6**	Parameter tampering (T2) - **C6, C7**	Poor audit trails (T3) - **C11**	XSS, CSRF attacks (T4) - **C7**	Application layer DDoS (T5) - **C11**	Exploiting insecure APIs (T6) - **C8**
Mobile App (A4)	API key theft (T1) - **C6**	Altering mobile traffic (T2) - **C6**	Denied app usage logging (T3) - **C11**	Leaking app data (T4) - **C6**	Overloading requests (T5) - **C4, C5**	Exploiting outdated app versions (T6) - **C11**
CDN (A5)	Fake CDN nodes (T1) - **C5**	Modified cached content (T2) - **C5**	Insufficient logging (T3) - **C7**	Exposure of cached data (T4) - **C5**	CDN exhaustion attack (T5) - **C5**	Not applicable
Web Servers (A6)	Credential theft (T1) - **C8**	File upload tampering (T2) - **C8**	Weak access logging (T3) - **C11**	Server misconfigurations (T4) - **C9**	Volumetric DDoS (T5) - **C4**	Privilege escalation via vulnerabilities (T6) - **C8**
Cache (A7)	Fake cache poisoning (T1) - **C10**	Cache invalidation (T2) - **C10**	Cache operations logging (T3) - **C11**	Sensitive data leakage (T4) - **C10**	Cache overflow attacks (T5) - **C10**	Misconfigured permissions (T6) - **C8**
Database (A8)	Unauthorized connections (T1) - **C8**	Data injection (T2) - **C9**	Poor access tracking (T3) - **C11**	SQL injection attacks (T4) - **C9**	Overloading connections (T5) - **C4**	Exploiting weak database roles (T6) - **C8**
Data Center (A9)	Rogue devices (T1) - **C8**	Physical tampering (T2) - **C8**	Lack of physical audit logs (T3) - **C11**	Data exposure via backups (T4) - **C10**	Physical sabotage (T5) - **C8**	Unauthorized admin access (T6) - **C8**
Monitoring (A10)	Fake logs (T1) - **C11**	Log tampering (T2) - **C11**	Disabling audit mechanisms (T3) - **C11**	Sensitive log leaks (T4) - **C11**	Overloading telemetry systems (T5) - **C11**	Exploiting monitoring tool vulnerabilities (T6) - **C11**

Figure 8: Threat Mapping for the Diagram with Controls

Explanation of Controls:

1. C1, C2: DNSSEC, DNS Filtering & Blacklisting—Protects DNS traffic by stopping DNS spoofing, cache poisoning attacks, etc.
2. C3: DNS Rate Limiting—Helps protect against DDoS attacks targeting DNS services running in the servers.
3. C4: SSL/TLS Encryption – Data integrity of sensitive data.
4. C5: Origin Authentication—Checks the source of the traffic reaching the CDN or web application by verifying its source of origin.
5. C6: Access Control Policies – Limits and enforces role-based access to resources from unauthorized access.
6. C7: Content Security Policies (CSP)—Protections against content injection, such as XSS and CSRF, and data injections (images, CSS, JavaScript).
7. C8: Access Control (Physical & Logical) – This limit who can access data centers, servers, and databases and enforces the principle of least privilege access to sensitive assets.
8. C9: Network Segmentation—Stop lateral movement by isolating network components and reducing the attack surface.
9. C10: Data Encryption—Make sure data is encrypted during transit and at rest to prevent unauthorized access or leaks [16].
10. C11: Regular Patching, Monitoring, & Logging—Make sure systems are up to date with security patches, and activities are logged and monitored for signs of abnormal behavior [17].
11. C12: Data Backup & Recovery—Daily, weekly, and monthly backups of critical data to an offshore location ensure its availability and integrity in case of data loss or system failure during natural disasters [18].

IV. CONCLUSION

These traditional ways of threat modeling help organizations find vulnerabilities and also improve their threat intelligence systems. This puts organizations in a better position to stay ahead of potential attackers. This reduces the need for time-consuming manual analysis and supports ongoing security monitoring. Instead of thinking about threat modeling only in the context of the new Zero Trust architecture, which does not assume trust by default and requires strict access control, constant authentication, and micro-segmentation, it becomes a key part of how security is designed and managed.

In short, threat modeling has become an important part of security that is now required in today's world full of cyber dangers. Any company that starts thinking about security right from the beginning of building software can find possible threats ahead of time and stop them before they turn into actual attacks. A team of people, like security experts, developers, and business leaders, work together to create detailed threat models.

The proposed model (CIA-AAA) helps find risks and come up with good ways to deal with them. Threat models need to be checked and improved often to stay up-to-date with new threats. To handle cyber risks well, companies need to take action before problems happen, not just after. Using automation and making sure everything follows rules helps improve security. This approach makes organizations stronger, reduces the chances of attacks, and helps protect their digital stuff from growing threats. Investing in threat modeling now helps prevent future breaches and builds a strong security culture that keeps the business and its users safe for a long time.

REFERENCES

[1] W. Xiong and R. Lagerström, “Threat modeling–A systematic literature review,” *Comput. Secur.*, vol. 84, pp. 53–69, 2019.
[2] “What Is Threat Modeling? Definition, Process, Examples, and Best Practices - Spiceworks,” Spiceworks Inc. Accessed: Nov. 30, 2025. [Online]. Available: https://www.spiceworks.com/it-security/network-security/articles/what-is-threat-modeling-definition-process-examples-and-best-practices/
[3] B. Lundgren and N. Möller, “Defining Information Security,” *Sci. Eng. Ethics*, vol. 25, no. 2, pp. 419–441, Apr. 2019, doi: 10.1007/s11948-017-9992-1.
[4] “What is CIA Triad?,” GeeksforGeeks. Accessed: Oct. 16, 2025. [Online]. Available: https://www.geeksforgeeks.org/computer-networks/the-cia-triad-in-cryptography/
[5] “Understanding AAA Frameworks in Cybersecurity: Authentication, Authorization, and Accounting.” Accessed: Oct. 16, 2025. [Online]. Available: https://www.linkedin.com/pulse/understanding-aaa-frameworks-cybersecurity-accounting-sohail-%D1%85%D0%B0%D0%BA%D0%B5%D1%80--llxic
[6] R. Nagori, “Threat Modeling.” Accessed: Oct. 24, 2025. [Online]. Available: https://interview.rajanagori.in/threatmodel/
[7] Imperva, “What is Threat Modeling | Guide to Security Risk Management | Imperva,” Learning Center. Accessed: Nov. 19, 2025. [Online]. Available: https://www.imperva.com/learn/application-security/threat-modeling/
[8] S. Hussain, A. Kamal, S. Ahmad, G. Rasool, and S. Iqbal, “Threat modelling methodologies: a survey,” *Sci IntLahore*, vol. 26, no. 4, pp. 1607–1609, 2014.
[9] N. Naila, “Threat modelling technique for GDPR compliance based on logical reasoning,” 2024.
[10] J. Bridges, “IT Risk Management Process, Frameworks & Templates,” ProjectManager. Accessed: Nov. 19, 2025. [Online]. Available: https://www.projectmanager.com/training/it-risk-management-strategies
[11] N. Shevchenko, T. A. Chick, P. O’riordan, T. P. Scanlon, and C. Woody, “Threat modeling: a summary of available methods,” 2018.
[12] M. Thevarmannil, “10 Types of Threat Modeling Methodology To Use in 2025,” Practical DevSecOps. Accessed: Oct. 24, 2025. [Online]. Available: https://www.practical-devsecops.com/types-of-threat-modeling-methodology/
[13] N. R. Mead, F. Shull, K. Vemuru, and O. Villadsen, “A hybrid threat modeling method,” *Carnegie MellonUniversity-Softw. Eng. Inst.-Tech. Rep.-CMUSEI-2018-TN-002*, 2018.
[14] N. R. Mead, F. Shull, K. Vemuru, and O. Villadsen, “A hybrid threat modeling method,” *Carnegie MellonUniversity-Softw. Eng. Inst.-Tech. Rep.-CMUSEI-2018-TN-002*, 2018.

[15]K. A. Cochran and K. Reis, "Core Concepts in Cybersecurity," in *CompTIA Security+ (SY0-701) Certification Companion: Hands-on Preparation and Practice Guide*, K. A. Cochran and K. Reis, Eds., Berkeley, CA: Apress, 2025, pp. 13–72. doi: 10.1007/979-8-8688-1498-3_2.
[16]G. Andersen, "The Role of System Security Engineering in Advancing Sports Technology." Accessed: Nov. 19, 2025. [Online]. Available: https://moldstud.com/articles/p-the-role-of-system-security-engineering-in-sports-technology
[17]J. Mehta, "Top Software Vulnerabilities in 2024 - How to Identify and Prevent the?," SignMyCode - Blog. Accessed: Nov. 19, 2025. [Online]. Available: https://signmycode.com/blog/how-to-identify-and-prevent-the-top-software-vulnerabilities-in-2023
[18]DSCI, "India Cyber Threat Report." DSCI, Nov. 19, 2025. Accessed: Nov. 19, 2025. [Online]. Available: https://www.dsci.in/files/content/knowledge-centre/2024/India-Cyber-Threat-Report-2025.pdf