



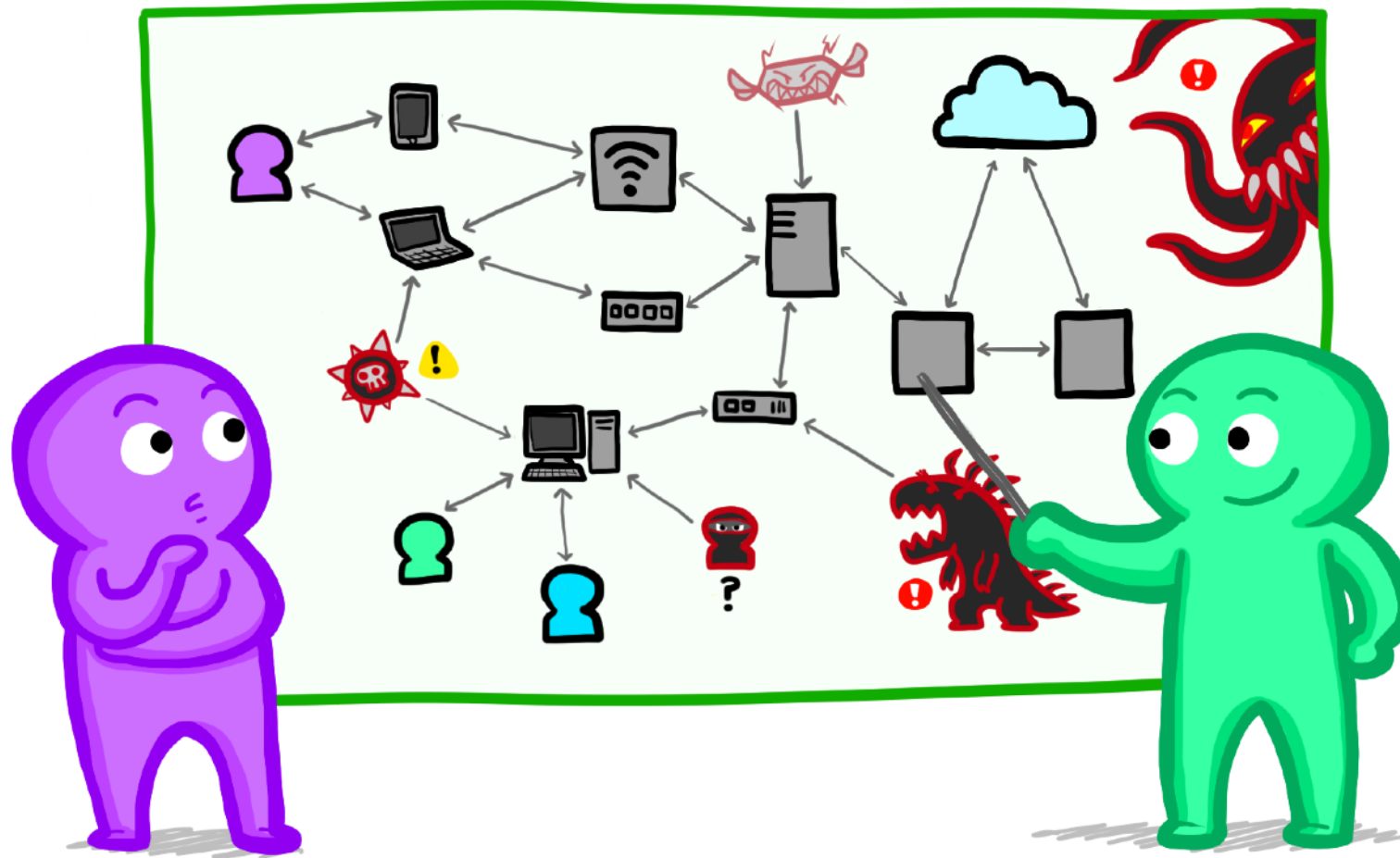
**Sanjeev Jaiswal**

Security Architect, Flipkart

# Threat Modeling Primer

The earlier, the better

# Mindset plays an important role here!



# Who am I

- 15+ years of Experience (5 in Dev, 10+ in Security)
- Security Architect, Flipkart
  - Security Head in Lifesight for 2.5 years
  - India Lead, AppSec Team in Epam
- Application Security
- Cloud Security (AWS, GCP)
- Programming: Perl, Python
- **Areas of Interest:**
  - Security Architecture
  - GenAI Security

# Agenda

- Setting up the context
- What and Why Is Threat Modeling (TM)
- When We Should Use Threat Model
- How to implement the Threat Model
- STRIDE Fundamentals
- What's Next

.

# Setting up the context

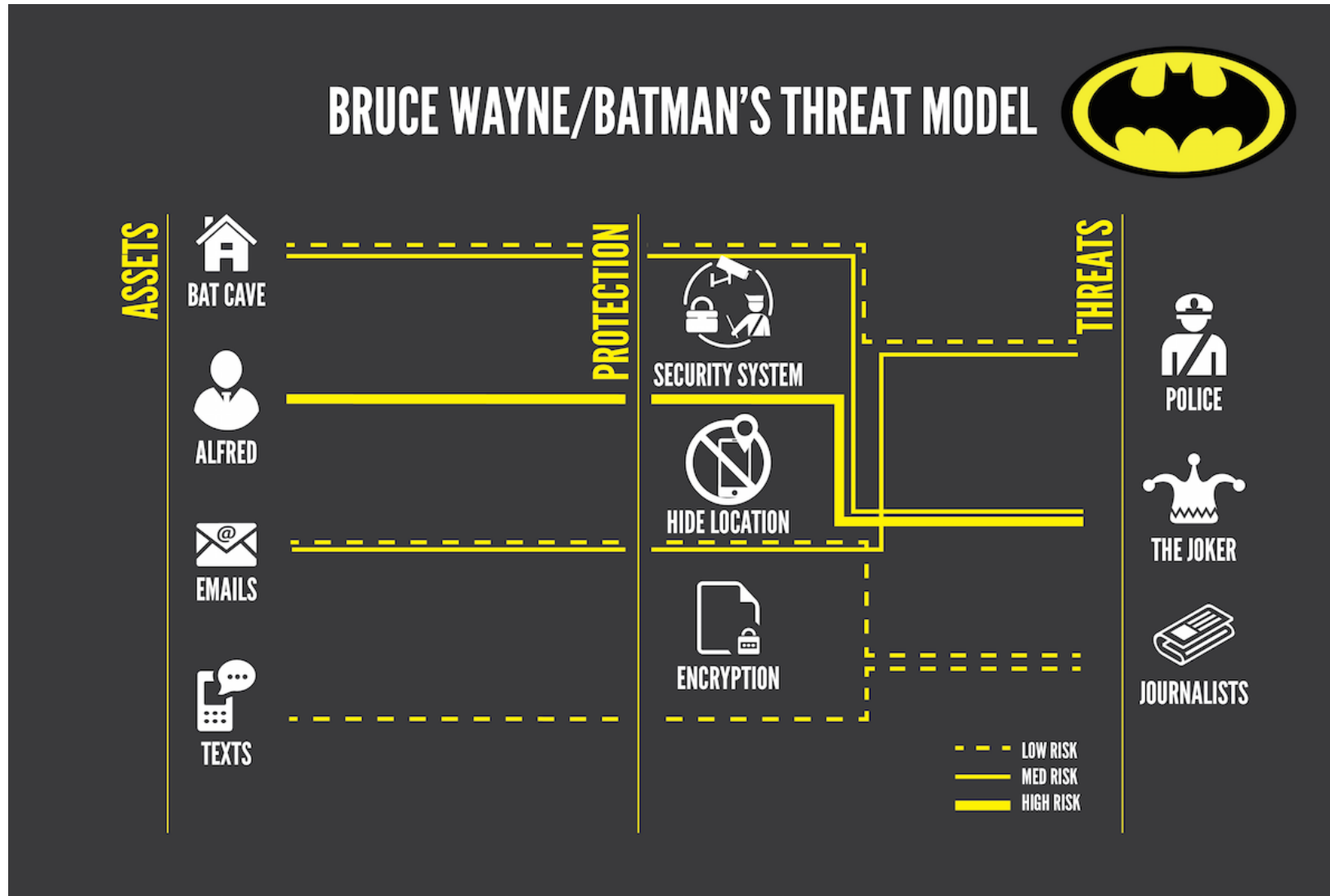
- You understand Data Flow Diagram
- It is foundational session
- Won't cover detailed scenario based TM
- Won't cover Rapid TM, Agile TM, Automated TM, TM using LLM
- This session will be the base for above all

# Curious case of helmet

Does having helmet is enough?

- Types of helmets: which one?
- Having a helmet is enough?
- Is a low-quality helmet okay?
- Wearing a helmet just to avoid a fine?
- What about the helmet expiry date?
- Do we need a helmet upgrade?

# Batman needs Threat Modeling too!



# What is Threat Modeling

- Design/Model of a system/application from a security point of view
- A list of potential threats
- A list of actions to mitigate each threat
- Validating the threats and verifications of action taken.



# Threat Modeling is not

- a Risk management tool
- a Penetration Testing activity
- only for a new system
- only for security teams
- about security checklists
- a replacement for secure coding

# Why Threat Modeling

- To build a secure system/application
- Define and build required controls
- Identify threats early and evaluate their risk
- Document threats, controls, risks & Mitigations
- Security test cases to be performed by pentesters

# When to use Threat Modeling

- The sooner, the better
- Ideally, at the design phase
- Whenever system changes
- After an incident - lessons learned
- Possibly at CI/CD?

# Threat Modeling Myths

- Thinking like an attacker while threat modeling
- This process is only for experts or for Architects
- Only inflow, no outflow and reverse as well
- Thinking one size fits all
- Neglecting business impact
- Focusing on vulnerabilities not the threats.

# Before we start Threat Modeling

## Start with these 4 Questions

1. What are we building?
2. What can go wrong?
3. What are we going to do about it?
4. Did We do a good enough job?

# Threat Modeling Types

**Basically 3 types:**

1. Attacker Centric
2. Application Centric
3. Asset Centric

# Different TM frameworks



- **STRIDE** (most common one)
- PASTA
- OCTAVE (GRC focused)
- LINDUNN (Privacy focused)
- Trike (Risk based)
- VAST
- ATT&CK (Attack Tress - Pentest Focused)
- hTMM (Best of STRIDE + ATT&CK)

## **ELEVATION OF PRIVILEGE**

underprivileged user gains privileged access

**E**

## **SPOOFING**

Impersonating another user  
to gain unauthorised access

**S**

## **DOS**

Service/resource is unavailable for use

**D**

# **STRIDE**

Developer  
focused

**T**

## **TAMPERING**

Unauthorised alteration of data

## **INFORMATION DISCLOSURE**

Unauthorised access to confidential  
information

**I**

## **REPUDIATION**

I didn't do it. Do you have proof?

**R**



# STRIDE vs CIAAN

	Threat	Property Violated	Threat Definition
<b>S</b>	Spoofing	Authentication	Pretending to be something or someone, that's not you
<b>T</b>	Tampering	Integrity	Altering something on the network, storage, memory etc.
<b>R</b>	Repudiation	Non-Repudiation	Claiming that you didn't do when asked for proof.
<b>I</b>	Information Disclosure	Confidentiality	Spilling the beans, error messages, error logs, response headers, robots.txt, etc.
<b>D</b>	Denial of Service	Available	If I can't use it, I won't let others also use it. Service interrupted/unavailable
<b>E</b>	Elevation of Privilege	Authorization	It seems exciting to get VIP treatment!

# OWASP Top 10 (2021) vs STRIDE

	OWASP Top 10	STRIDE
<b>A1</b>	Broken Access Control	Tampering (T), Elevation of Privilege (E)
<b>A2</b>	Cryptographic Failures	Information Disclosure(I), Spoofing(S)
<b>A3</b>	Injectons	Tampering(T), Elevation of Privilege(E)
<b>A4</b>	Insecure Design	Tampering(T), Elevation of Privilege(E), Information Disclosure(I)
<b>A5</b>	Security Misconfigurations	Information Disclosure(I), Tampering(T), Elevation of Privilege(E)
<b>A6</b>	Vulnerable and Outdated Components	Tampering(T), Denial of Service(D)
<b>A7</b>	Identification and Authentication Failures	Spoofing(S), Elevation of Privilege(E)
<b>A8</b>	Software and Data Integrity Failures	Repudiation(R), Tampering(T), Elevation of Privilege (E)
<b>A9</b>	Security Logging and Monitoring Failures	Repudiation(R), Information Disclosure(I)
<b>A10</b>	Server Side Request Forgery (SSRF)	Spoofing(S), Information Disclosure(I), Tampering (T)

# OWASP Top 10 for LLMs vs STRIDE

	OWASP Top 10 for LLMs	STRIDE Category	Example
LLM01	Prompt Injection	Tampering (T), Elevation of Privilege (E)	Jailbreaking an LLM to bypass ethical filters.
LLM02	Insecure Output Handling	Information Disclosure(I), Tampering (T)	LLM output reveals confidential API keys or credentials.
LLM03	Training Data Poisoning	Tampering(T), Elevation of Privilege(E)	Poisoning a dataset to bias AI-generated decisions.
LLM04	Model Denial of Service (DoS)	Denial of Service(D)	Sending recursive prompts that cause infinite loops.
LLM05	Supply Chain Vulnerabilities	Information Disclosure(I), Tampering(T), Spoofing(S)	Hosting a fake model with a backdoor.
LLM06	Sensitive Information Disclosure	Information Disclosure(I)	An AI assistant leaks a user's financial data.
LLM07	Insecure Plugin Design	Spoofing(S), Elevation of Privilege(E), Tampering(T)	A malicious plugin extracts sensitive data from an LLM.
LLM08	Excessive Agency	Elevation of Privilege (E)	LLM executes unintended code that modifies system files.
LLM09	Overreliance on LLM Output	Spoofing(S), Information Disclosure(I)	Using AI-generated code with hardcoded credentials.
LLM10	Model Theft	Spoofing(S), Information Disclosure(I), Tampering (T)	Reverse-engineering an LLM's weights to extract IP.

# Terms used in Threat Model

## Terms that you will use

- **Asset:** What do you want to protect?
- **Threat:** What's a potential negative impact or outcome?
- **Vulnerability:** Spotted Weakness? Threat can be sensed?
- **Attack:** How to take advantage of the Vulnerability?
- **Mitigation:** How can we reduce the damage?
- **Security Control:** protection at places?

# Threat vs Vulnerability vs Risk

Threat → A burglar who wants to break into your house.

Vulnerability → An unlocked front door.

Risk → The chance that the burglar enters your house through the unlocked door and steals your valuables

A burglar (**threat**) can break into your house if you leave the door unlocked (**vulnerability**), increasing the chance of your valuables being stolen (**risk**).

# DFD for Threat Model

## Data Flow Diagram Symbols

### Elements

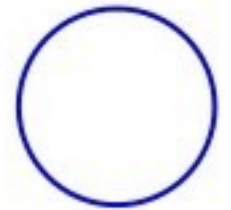
- Process
- Multi-Process
- Data Flow
- Trust Boundary
- Data Store
- External Entity



External  
Entity



Complex-Process



Process



Data Store


















Dataflow



Privilege  
Boundary

# STRIDE per element

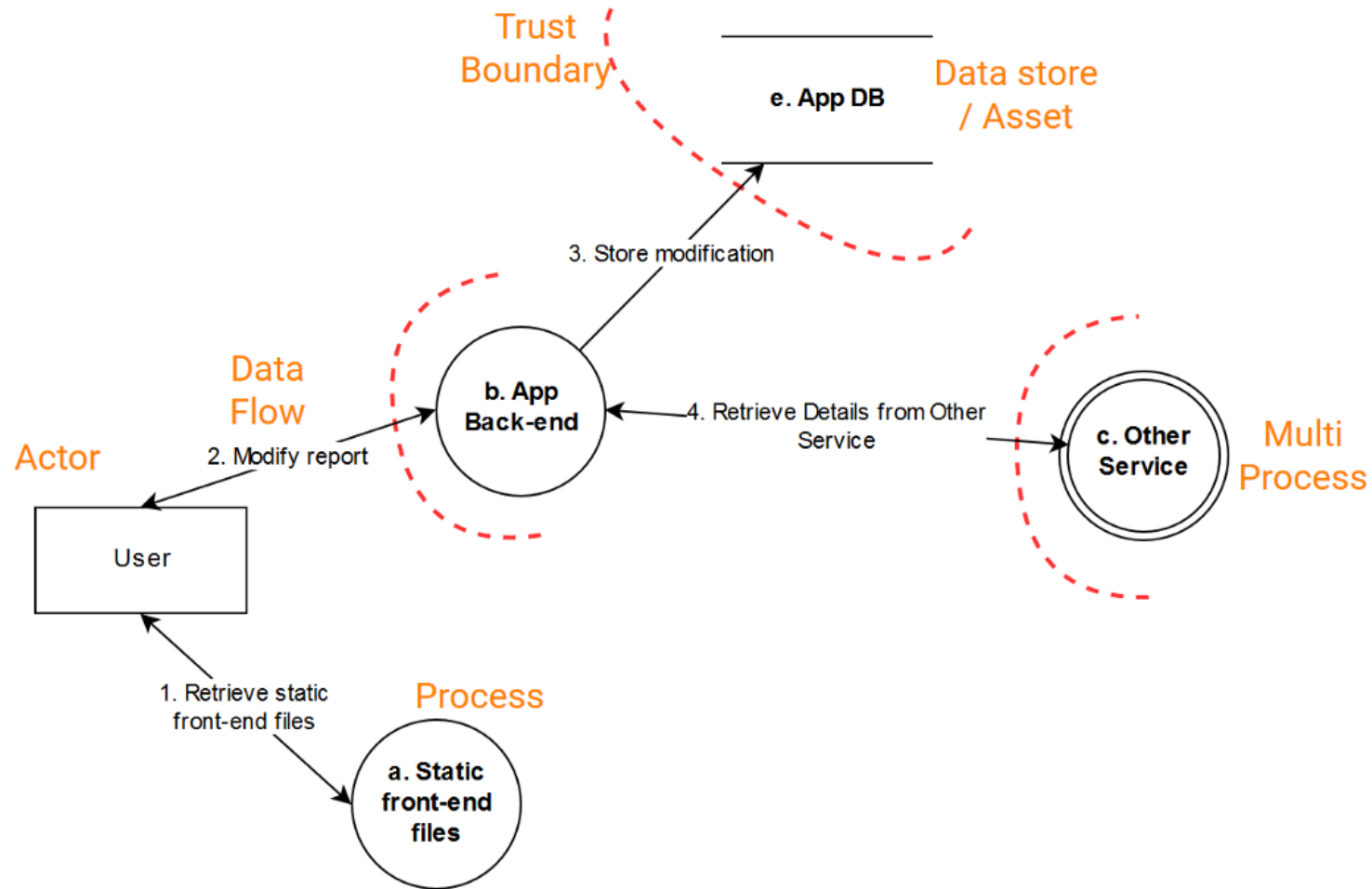
	Spoofing	Tampering	Repudiation	Information Disclosure	Denial of Service	Elevation of Privilege
External Entity						
Process						
Data Store						
Data Flow						

# Steps to perform Threat Model

- Identify Entry/Exit Points
- Decompose the Application as deep as you can
- Identify the assets (crown jewels)
- Identify the trust levels
- Map the threats associated with assets
- Map the security controls against each assets
- Iterate till all assets are covered
- Prepare a threat report based on TM performed



# Sample Threat Model



# Threat Modeling Tools

- **White boarding**
- MS Threat Modeling tool
- [OWASP Threat Dragon Project](#)
- **Draw.io**
- IriusRisk
- SecuriCAD by foreSeeti
- SD Elements
- Threagile
- PyTM
- Cairis

# Demo



# 1. Web Login

**Scenario:** A user logs in to a web application using a **username & password**.

**Objective:** Identify security threats and mitigations for authentication.

**Steps:**

- Define the System & Assets (Dissect the system)
- **Identify Threats** (Using STRIDE Model)
- Mitigations

# STRIDE and Mitigations

1. **Spoofing:** Attacker tries credential stuffing or brute force attacks.
  2. **Tampering:** Intercepting login requests (MITM attack).
  3. **Repudiation:** No logging of failed login attempts.
  4. **Information Disclosure:** Application leaks error messages (e.g., *"Invalid username"* vs. *"Invalid password"*)
  5. **Denial of Service (DoS):** Multiple login attempts slow down the server.
  6. **Elevation of Privilege:** Broken authentication allows privilege escalation.
1. Enforce strong password policies
  2. Implement MFA
  3. Use rate limiting and lockout mechanisms
  4. Secure login data with TLS encryption
  5. Use OAuth/OpenID Connect instead of passwords

## 2. Contact Form on a Website

**Scenario:** A public **contact form** allows users to submit messages via a web form

**Objective:** Find common web security risks

**Steps:**

- Define the System & Assets (Dissect the system)
- **Identify Threats** (Using STRIDE Model)
- Mitigations

# STRIDE and Mitigations

	Threat	Example	Mitigation
<b>S</b>	Fake or automated bot submissions	An attacker submits fake messages pretending to be another user	<ul style="list-style-type: none"> <li>Implement CAPTCHA</li> <li>Enforce authentication for sensitive forms</li> </ul>
<b>T</b>	Altering form fields in requests	Attacker modifies hidden fields (e.g., changing an email field to redirect messages elsewhere)	<ul style="list-style-type: none"> <li>Validate form input on the server</li> <li>Use HTTPS to protect in-transit data</li> </ul>
<b>R</b>	No tracking of form submissions	A user submits an abusive message and denies responsibility	<ul style="list-style-type: none"> <li>Log IP addresses and timestamps</li> <li>Enable audit logging</li> </ul>
<b>I</b>	Email addresses or messages leaked	Form displays entered email IDs in a confirmation message visible to others	<ul style="list-style-type: none"> <li>Do not echo sensitive input in responses</li> <li>Implement proper access controls</li> </ul>
<b>D</b>	Flooding the form with automated spam	Bot submits thousands of requests, overwhelming the backend	<ul style="list-style-type: none"> <li>Implement rate limiting</li> <li>Block repeated submissions from the same IP</li> </ul>
<b>E</b>	Form submission leads to unintended admin actions	Attacker injects admin-related parameters (e.g., role=admin) in the request	<ul style="list-style-type: none"> <li>Perform strict input validation</li> <li>Implement least privilege access control</li> </ul>

# 3. LLM-based systems: Threat Modeling (Exercise)

**Scenario:** An LLM powered chatbot allows you to ask cybersecurity questions

**Objective:** Find common OWASP Top 10 LLM issues and map with STRIDE

## Steps:

- Define the System & Assets (Dissect the system)
- **Identify Threats** (Using STRIDE Model)
- Mitigations

## LLM System Components

- **User:** Sends queries to the LLM.
- **LLM Model API:** Processes the request and generates a response.
- **Training Data Source:** Provides data to train or fine-tune the model.
- **Plugin/Third-party Integrations:** Extends functionality (e.g., retrieval-augmented generation, external APIs).
- **Storage & Logs:** Stores interactions, logs, or training data.



# STRIDE and Mitigations

	Threat	OWASP LLM Risks	Mitigation
<b>S</b>	Fake model API responses, malicious external plugins	LLM05: Supply Chain, LLM09: Overreliance	Validate plugins and AI-generated content
<b>T</b>	Malicious input manipulates LLM responses, poisoned datasets introduce bias	LLM01: Prompt Injection, LLM03: Training Data Poisoning	<ul style="list-style-type: none"> <li>• Input sanitization,</li> <li>• Secure dataset controls</li> </ul>
<b>R</b>	No logging of external plugin actions	LLM07: Insecure Plugin Design	Secure logging and API request monitoring
<b>I</b>	LLM leaks confidential user data, API keys in responses	LLM02: Insecure Output Handling, LLM06: Sensitive Info Disclosure	<ul style="list-style-type: none"> <li>• PII masking,</li> <li>• Access control for sensitive queries</li> </ul>
<b>D</b>	Recursive queries overload model inference	LLM04: Model DoS	<ul style="list-style-type: none"> <li>• Rate limiting,</li> <li>• Request validation</li> </ul>
<b>E</b>	LLM executes unauthorized actions, attacker extracts model weights	LLM08: Excessive Agency, LLM10: Model Theft	<ul style="list-style-type: none"> <li>• Permission controls,</li> <li>• encryption of model weights</li> </ul>

# Look around you for Threat Modeling

1. While crossing the road
2. Inspecting car before leaving for a trip
3. Flying private plane from India to UK
4. Wankhede stadium for final match b/w India vs Pakistan
5. Going to hill stations/pilgrimage in peak time
6. Buying or building a new property
7. Buying a used vehicle

# Threat Modeling (TM) FAQs



1. What are the advantages of TM in an organisation?
2. How can we prevent threats even before deployment?
3. Who should be responsible for TM
4. Who should be involved in TM sessions?
5. How can I rate something which does not exist
6. What frameworks or methodologies can we use for TM
7. How do we measure the success of a TM program?
8. How can we balance TM without slowing down development?
9. How do we ensure Threat Modeling is a continuous process and not a one-time exercise?
10. What are the biggest challenges in adopting Threat Modeling in an organization?
11. Can AI/automation tools replace manual TM?

The background is a solid green color with various faint, light-green geometric patterns. These include a grid of squares in the top-left and bottom-left corners, a vertical column of plus signs on the left, a vertical column of circles on the right, and several horizontal and vertical lines scattered throughout.

# What's Next

# Explore these

- DevSecOps Threat Model
- Infra Threat Model
- Cloud Threat Model
- AI Threat Model
- [PASTA](#) (Attacker Focused)
- OCTAVE (Practice Focused)
- VAST (Enterprise Focused)

# Learning Resources

- Threat Modeling Book by Adam Shostack
- Learn Threat Modeling for Security Professionals
- OWASP Application Threat Modeling
- Threat Modeling CheatSheet
- Threat Playbook by we45 (Interesting One)
- Docker Container Security and STRIDE

# Few more learning resources

- [Microsoft Secure-SDL: Threat Modeling](#)
- [OWASP Application Threat Modeling](#)
- [Threat Modeling why how when \(Nice Article\)](#)
- [Kubernetes Threat Model \(pdf\)](#)
- [Docker Security: Threat Modeling](#)
- [Container as a Service Threat Analysis \(pdf\)](#)

# My Social Channels



[cybercloud.guru](https://cybercloud.guru)



[github.com/jassics](https://github.com/jassics)



[twitter.com/jassics](https://twitter.com/jassics)



[linkedin.com/in/jassics](https://linkedin.com/in/jassics)



