# Co-Modeler: AI-Driven Threat Modeling

PRESENTED BY

Shankar Chebrolu

President & Co-founder, CSA Triangle Chapter

Director of Security Architecture, Red Hat

Mar 20, 2026

# Agenda

- Threat Modeling Primer – 30 min
- Co-Modeler Tool & Demo – 15 min
- Q/A - discussion – 15 min

# Threat Modeling: Identifying Potential Threats

- A structured and repeatable process to identify threats and mitigate them against valuable assets in a system
- Secure systems cannot be built without understanding the potential threats
- Threat Modeling could  be used for:
  - Modeling a system
  - Identify Threats
  - Analyze Vulnerabilities
  - Design, Implement & Verify Mitigations

# Threat Modeling – alignment to NIST CSF

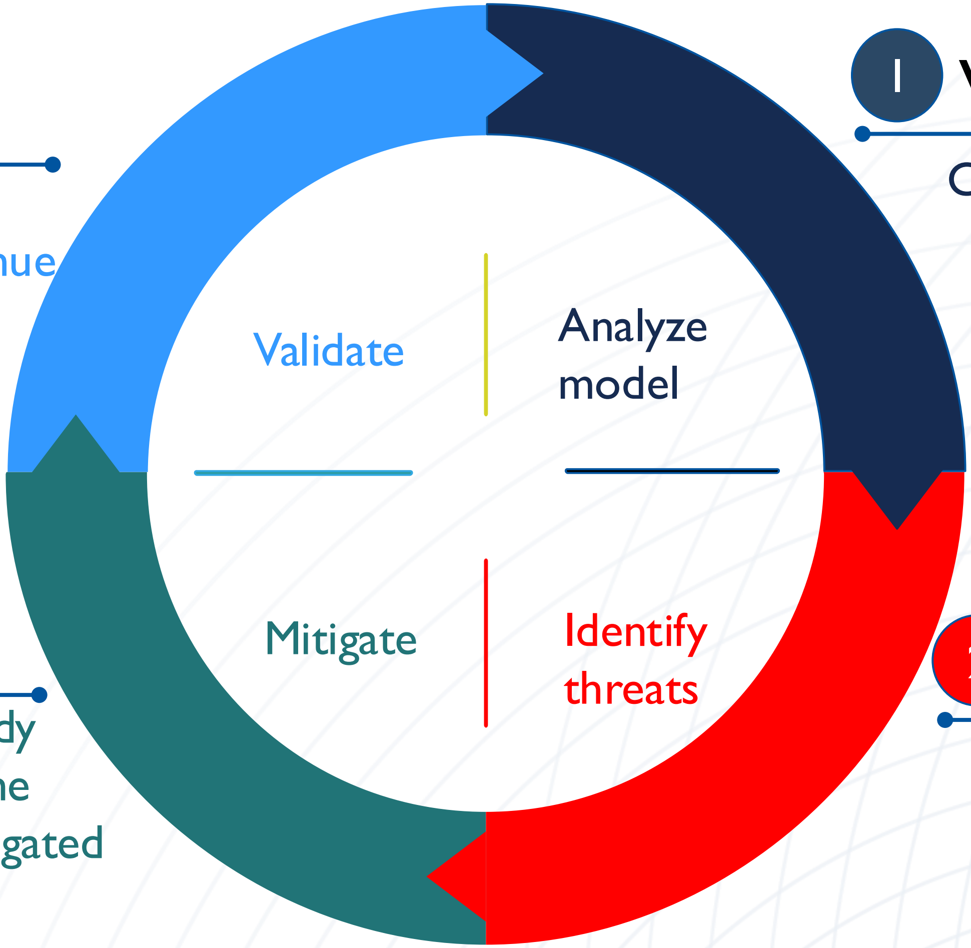
| Function | Category | Sub-category |
|---|---|---|
| **IDENTIFY (ID)** | **Risk Assessment (ID.RA):**<br>The organization understands the cybersecurity risk to organizational operations (including mission, functions, image, or reputation), organization assets, and individuals | **ID.RA-3: Threats,** both internal and external, are **identified** and **documented** |

# Threat Modeling Vs Threat Intelligence

| | Threat Modeling (TM) | Threat Intelligence (TI) |
|---|---|---|
| **Alignment** | Security architecture / design portion of secure development lifecycle (SDL) | Security operations |
| **Relevance** | Identifying threats in a particular system before it is deployed in production | Comprehensive list of threats to a whole organization w.r.t. Systems already in production/laptops/workstations etc. |
| **What's in Common** | In NIST-CSF, both TM and TI maps into Risk Assessment (ID.RA-3)<br>IDENTIFY (ID) → Risk Assessment (RA) → Threats are identified and documented (ID.RA-3) | |

# Threat Modeling Process

(The four-question framework by Adam Shostack)



1 **What are we working on?**

Create an architectural diagram

2 **What can go wrong?**

Analyze the model to identify potential threats

3 **What are we going to do about it?**

Indicate which threats are already mitigated and determine how the remaining threats would be mitigated

4 **Did we go a good job?**

Validate the system against recorded threat model. Continue to mitigate any open issues

# Threat Modeling Classification: STRIDE

| Classification | Definition | Sample Threats |
|---|---|---|
| **S**poofing | Impersonating users or services | ▪ Pretending to be valid user or stealing API keys<br>▪ Pretending to be valid LLM |
| **T**ampering | Modifying code or data | ▪ Modifying code (or library), data on a system<br>▪ Modifying a packet as it traverses the network<br>▪ Modifying training data, models, or inference pipelines |
| **R**epudiation | ▪ Claiming to have not performed an action<br>▪ Denying responsibility for AI system actions | ▪ Remove record of modification of a file or logs<br>▪ Remove record of deletion of a system resource |

# Threat Modeling Classification: STRIDE

| Classification | Definition | Sample Threats |
|---|---|---|
| **I**nformation disclosure | Exposing information to someone not authorized to access | ▪ Attackers use API queries to learn about model behavior or extract training data (membership inference)<br>▪ Model inversion, membership inference attacks.<br>▪ Sniffing network traffic to read sensitive data in transit<br>▪ Launching SQL injection attach to read sensitive data from DB table(s) |
| **D**enial of service (DoS/DDoS) | ▪ Overloading model APIs or corrupting inputs to degrade performance.<br>▪ Deny or degrade service to users | ▪ Flooding the API with spammy or malformed emails to exhaust system resources or degrade accuracy |
| **E**levation of privilege | Gain capabilities without proper authorization | ▪ Allowing a limited user to switch to an admin user without authorization or validation logic<br>▪ Regular users gain admin access to retrain or override model predictions |

# Threat Modeling Elements

- **Actor**: Users (typically humans)
- **Datastore:** Databases, Filesystems, LDAP, Cookies, Memory-Cache
- **Data Flow**: HTTPS, IPSEC, RPC
- **Process (runs code)**: Web application/service, LLM, OS process, any business logic running in a server (web server, app server, database)

# STRIDE applicability to Threat Modeling Elements

| | Spoofing | Tampering | Repudiation | Information disclosure | Denial of service | Elevation of privilege |
|---|---|---|---|---|---|---|
| **Actor** | ✓ | | ✓ | | | |
| **Process** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Datastore** | | ✓ | ✓ | ✓ | ✓ | |
| **Dataflow** | | ✓ | | ✓ | ✓ | |

# Threats and Risk Mitigations

| Threat(s) | Risk Mitigation | Control(s) |
|---|---|---|
| **S**poofing | Strong **Authentication** | Use 2FA / Biometric Authentication |
| **T**ampering | Protect Data **Integrity** | Use Strong cryptography for one-way hashing |
| **R**epudiation | **Non-Repudiation** | ▪ Use digital signatures<br>▪ Implement Log monitoring |
| **I**nformation disclosure | Protect Data **Confidentiality** | Use Strong cryptography for encryption of data-in-transit and data-at-rest |
| **D**enial of service (DoS/DDoS) | Ensure **Availability** | Enforce throttling to control resources |
| **E**levation of privilege | Enforce **Authorization** | Enforce principle of least privilege (RBAC / ABAC) |

# CROSS WALK BETWEEN STRIDE (threats) and NIST CSF

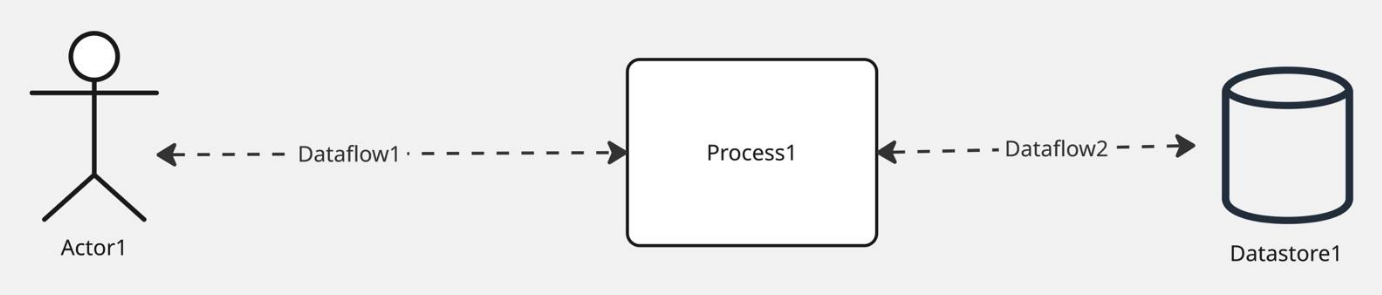
| Threat(s) | Risk Mitigation | NIST CSF 1.1 Control(s) |
|---|---|---|
| **S**poofing | Strong **Authentication** | PR.AC-7: Users, devices, and other assets are authenticated |
| **T**ampering | Protect Data **Integrity** | PR.DS-2: Data-in-transit is protected |
| **R**epudiation | **Non-Repudiation** | ▪ PR.DS-2: Data-in-transit is protected<br>▪ DE.AE-3: Event data are collected and correlated from multiple sources and sensors |
| **I**nformation disclosure | Protect Data **Confidentiality** | PR.DS-1: Data-at-rest is protected |
| **D**enial of service (DoS/DDoS) | Ensure **Availability** | ▪ PR.AC-5: Network integrity is protected<br>▪ PR.PT-4: Communications and control networks are protected<br>▪ DE.CM-1: The network is monitored to detect potential cybersecurity events |
| **E**levation of privilege | Enforce **Authorization** | PR.AC-4: Access permissions and authorizations are managed, incorporating the principles of least privilege and separation of duties |

# CROSS WALK BETWEEN STRIDE (threats) and CSA CCM / AICM

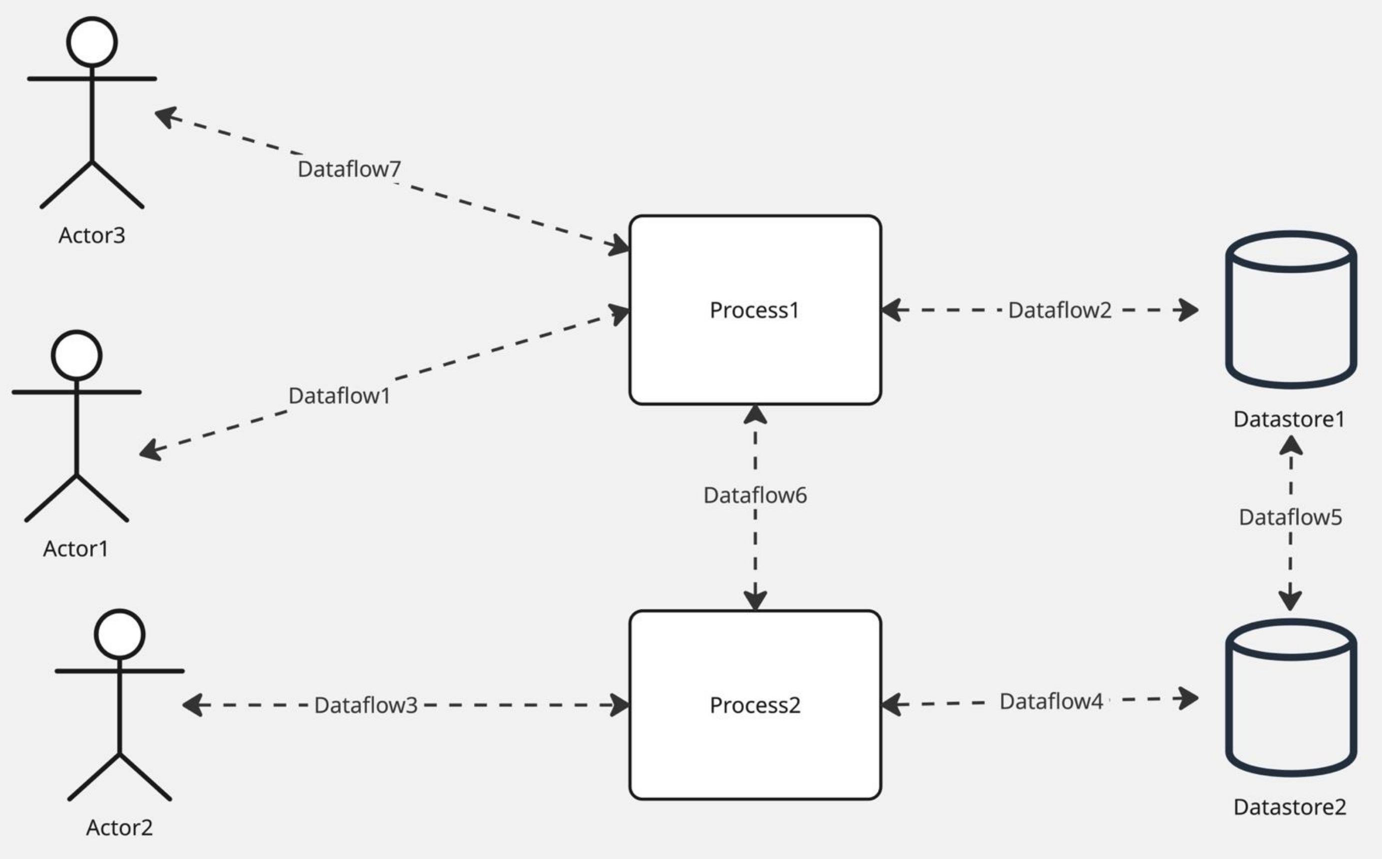
| Threat(s) | Risk Mitigation | CCM Control Domain | *"Sample"* Control(s) |
|---|---|---|---|
| **S**poofing | Strong **Authentication** | Identity & Access Mgmt (IAM) | **IAM-14:** Strong Authentication: MFA |
| **T**ampering | Protect Data **Integrity** | Cryptography, Encryption & Key Management (CEK) | **CEK-03:** Data Encryption<br>**CEK-08:** Key Management |
| **R**epudiation | **Non-Repudiation** | Logging & Monitoring | **LOG-03:** Log Monitoring & Alerting |
| **I**nformation disclosure | Protect Data **Confidentiality** | Cryptography, Encryption & Key Management (CEK) | **CEK-03:** Data Encryption<br>**CEK-08:** Key Management |
| **D**enial of service (DoS/DDoS) | Ensure **Availability** | Business Continuity Mgmt & Operational Resilience (BCR) | **BCR-06:** Service Continuity Exercises<br>**BCR-10:** Response Plan Exercise<br>**BCR-11:** Redundancy |
| **E**levation of privilege | Enforce **Authorization** | Identity & Access Mgmt (IAM) | **IAM-05:** Enforce the Principle of Least Privilege |

# Threat Modeling Exercise

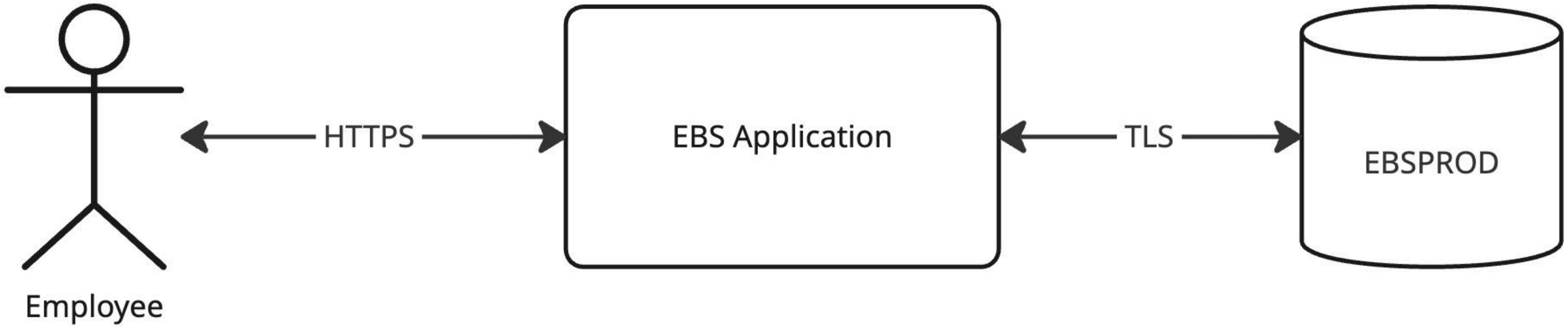
- Threat Modeling starts with an architecture diagram
- List down each architectural component
- Use the STRIDE applicability matrix to figure out potential threats for each element
- Figure out one or more controls to mitigate the risks due to those threats
- Refer:
  - A step-by-step guide to create your first threat model (template included)
  - Workshop video recording on youtube



Sample architecture diagrams

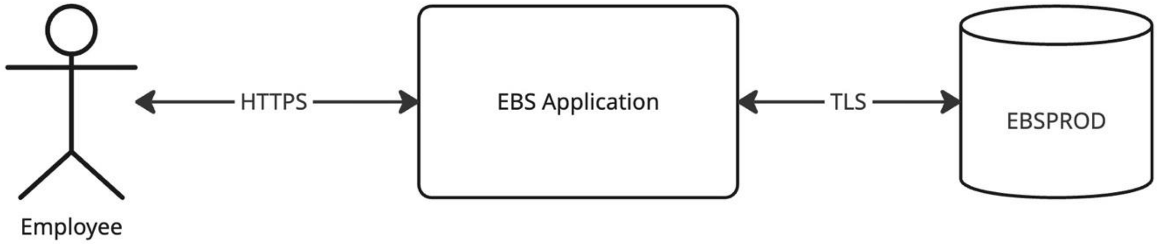


# Threat Modeling Exercise - Sample Architecture Diagram - 1



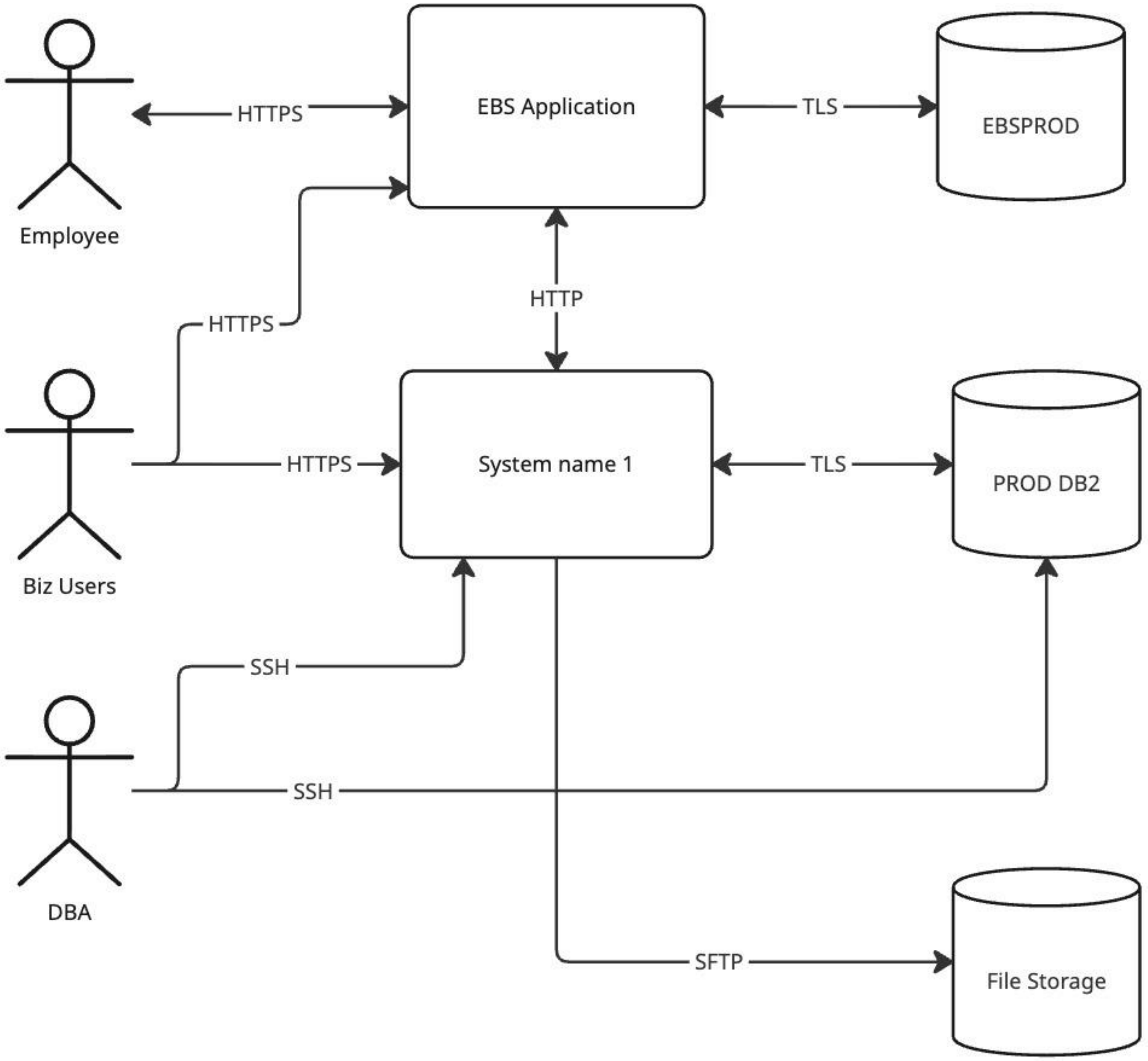
| Arch. Artifact Labels | TM Element<br>(actor, datastore, dataflow, process) | Applicable Threats | Risk Mitigation<br>(by implementing Security Controls) |
|---|---|---|---|
| Employee | | | |
| | | | |
| HTTPS | | | |
| | | | |
| | | | |
| EBS Application | | | |
| TLS | | | |
| EBSPROD | | | |

# Threat Modeling Exercise - Sample Architecture Diagram - I



| Arch. Artifact | TM Element | Applicable Threats | Risk Mitigation<br>(by implementing Security Controls) |
|---|---|---|---|
| Employee | Actor | Spoofing | Strong Authentication: 2FA / MFA |
| | | Repudiation | Logging & Monitoring |
| HTTPS | Dataflow | Tampering | Encryption of Data-in-transit |
| | | Information disclosure | Encryption of Data-in-transit |
| | | Denial of service | Secure Config of Network, Tiers/Zones |
| EBS Application | Process | Spoofing | PKI Certs / Secure Cert Mgmt |
| | | Tampering | Secure App Dev, SAST, Pen Testing |
| | | Repudiation | Logging & Monitoring |
| | | Information disclosure | Secure App Dev, SAST, Pen Testing |
| | | Denial of service | Secure Config of Web/Appservers |
| | | Elevation of privilege | Strong Authorization (RBAC/ABAC) |
| EBSPROD | Datastore | Tampering | Secure Config of DBs/servers |
| | | Repudiation | Logging & Monitoring |
| | | Information disclosure | Encryption of Data-at-rest |
| | | Denial of service | Secure Config of DBs/servers, Throttling |
| TLS | Dataflow | Tampering | Encryption of Data-in-transit |
| | | Information disclosure | Encryption of Data-in-transit |
| | | Denial of service | Secure Config of Network, Tiers/Zones |

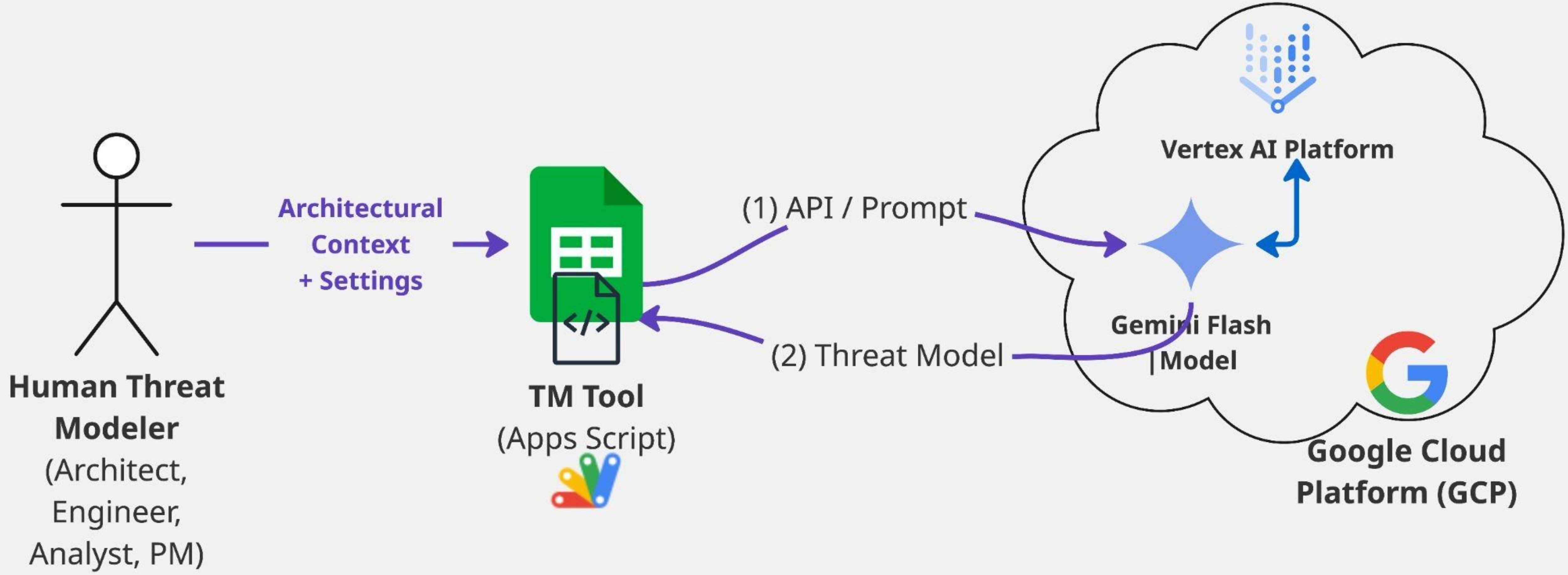
# Threat Modeling Exercise - Sample Architecture Diagram – 2



# How to Scale or Automate Threat Modeling?

- **The Manual Bottleneck:** Traditional threat modeling requires 100+ person-hours for moderately complex architectures.
- **Excessive Toil:** Architects spend too much time decomposing architecture diagrams, identifying components, manually cross-referencing them against applicable threats (eg. STRIDE) and risk mitigating security controls
- **Inconsistent Quality:** Manual mapping is highly susceptible to human error and fatigue.
- **The Opportunity:** Because threat modeling is a structured, repeatable process, it is a prime candidate for AI-driven automation.

# Co-Modeler: Tool Architecture



# Co-Modeler: Tool

- **Automates Component Extraction:** Uses multimodal LLM (Flash) to identify architectural artifacts / elements from an architecture diagram
- **Human-in-the-Loop Prompting:** "Architectural Context" input allowing human experts to guide the semantic reasoning of the systems/arch. diagrams before threat modeling begins.

# API call to Google Vertex AI

```
modelId = 'gemini-2.5-flash';
```

```
https://${LOCATION}-aiplatform.googleapis.com/v1/projects/
${PROJECT_ID}/locations/${LOCATION}/publishers/google/models/
${modelId}:generateContent;
```

# Prompt to Gemini 2.5 Flash

Act as a Senior Open Hybrid Cloud Security Architect. Analyze this architecture diagram.

Identify all Actors, Processes, Datastores, and Dataflows. Output strictly raw JSON (NO markdown/backticks) as an array of objects with these exact keys:

'displayName': (Actor, Process, Datastore, or Dataflow)

'label': Diagram text for the element

'description': Brief technical function

'threats': Array of STRIDE threats based STRICTLY on this matrix:

* Actor: Spoofing, Repudiation
* Process: All 6 STRIDE threats
* Datastore: Tampering, Repudiation, Information Disclosure, Denial of Service
* Dataflow: Tampering, Information Disclosure, Denial of Service

Inside the 'threats' array, each object must have:

'threatType': The STRIDE category

'threatDescription': How the threat applies to this component

'recommendedMitigation': Architectural security controls

# High Level Control Flow

```
[ 👤 User / Security Architect ]
        |
        | 1. Enters Diagram URL & Context into Cell B4
        v
[ 📊 Google Sheets UI ]
        |
        | 2. Clicks "Run Threat Model" button
        v
[ 📜 Google Apps Script ]
        |   - Fetches image from Google Drive
        |   - Converts image to Base64
        |   - Assembles JSON payload (Prompt + Image + Context)
        |
        | 3. POST Request via UrlFetchApp
        |    URL: https://{REGION}-aiplatform.googleapis.com/...
        v
```

```
[ 🛡️ Vertex AI (The API Gateway) ]
        |   - Validates Project ID & IAM permissions
        |   - Routes request to the correct model
        |
        v
[ 🧠 Gemini 2.5 Flash (The Foundation Model) ]
          - Analyzes the image & text prompt
          - Generates the JSON Threat Model array

|
        | 4. Returns JSON Response
        v
[ 📜 Google Apps Script ]
        |   - Parses JSON
        |   - Maps components to STRIDE threats
        |   - Applies Red/Green traffic light logic
        v
[ 📊 Google Sheets (Output) ]
        |   - Renders Threat Model rows
        |   - (Optional) Exports to Doc/PDF
        v
[ 👤 User reviews the generated Threat Model ]
```

# Tool Demo

# Q/A