

# CS 4530: Fundamentals of Software Engineering

## Module 12.2: Threat Modeling

Adeel Bhutta, Joydeep Mitra and Mitch Wand

Khoury College of Computer Sciences

© 2025, released under [CC BY-SA](#)

# Learning Objectives for this Module

- By the end of this module, you should be able to:
  - Appreciate the need for threat modelling
  - Understand the process of threat modeling
  - Understand the STRIDE framework
  - Integrate threat modelling with requirements

# Outline of this lecture

1. The need for Threat Modelling
2. The Threat Modelling Method
3. The STRIDE Framework
4. How to Apply Threat Modelling in an Agile Process Model

# Secure By Design

- Traditional Software Process Models do not bake security into the software development lifecycle.
  - Security issues are usually an after thought.
- The modern approach to secure software engineering is to **consider security during the design phase.**
- A useful method to do secure by design is through **threat modeling.**

# What is Threat Modeling?

- **Identification:** Recognizing potential threats to your system.
- **Assessment:** Evaluating the likelihood and impact of each threat.
- **Countermeasures:** Implementing strategies to mitigate or prevent identified threats.
- **Review:** Regularly reviewing and updating threat models to adapt to evolving risks.

# Getting Started with Threat Modelling

- Concentrate on technical risks rather than broad threats.
  - Are there any missing controls?
  - Is there a data flow that can be abused?
  - Technical threats combine to create broad threats.
  - Focusing on vague nation-state attacks and zero-day exploits can overshadow essential application security details.

# Getting Started with Threat Modelling

- Collaborate with stakeholders
  - Diverse views on security are necessary to define a comprehensive model.
  - Product owners or clients must be involved to gain perspective on user behavior and prioritizing risks.

# Getting Started with Threat Modelling

- Frequent and small iterations
  - Start with the thinnest slice of the system. E.g.,
    - User registration flow.
    - A microservice and its collaborating services.
    - Current iteration
  - Repeat and refine them.
- Defining a threat model upfront for the entire system is counter productive.



# Basic Structure of Threat Modeling in Agile

An effective threat modeling session must deal with the three primary questions

Activity	Question	Outcome
Explain and explore	What are you building?	A technical diagram
Brainstorm threats	What can go wrong?	A list of technical threats
Prioritize and fix	What are you going to do?	Add prioritized fixes to backlog (todo list)

## An Example

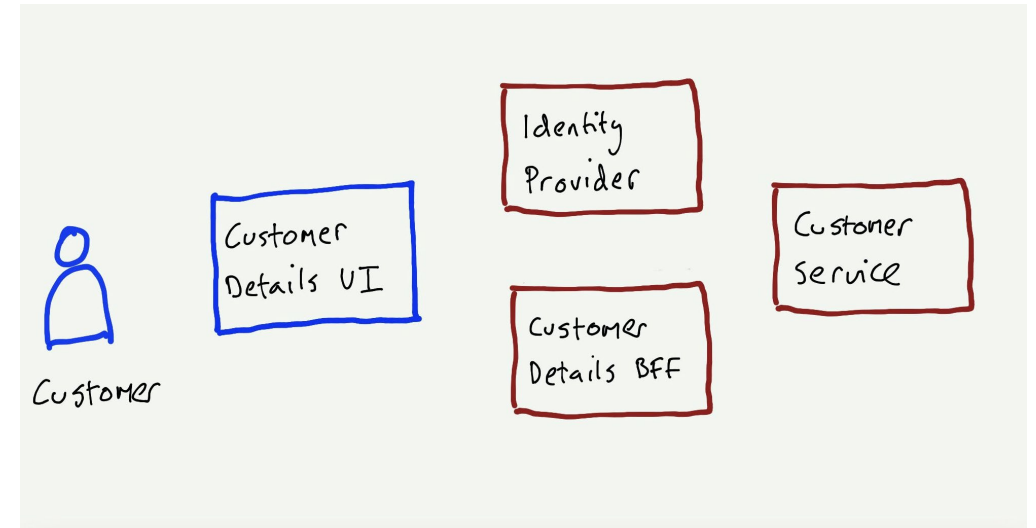
“As a customer, I need a page where I can see my customer details so I can confirm they are correct”

*An Epic (high-level requirement) in Agile*

- *Epics are specified as user stories*
- *and broken down into sprints.*

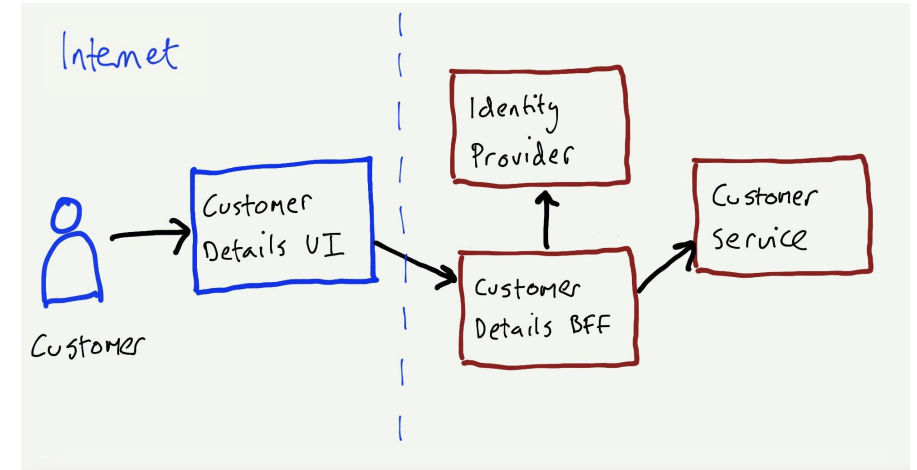
# An Example

- What are we building?
- Use a sketch to represent
  - relevant components
  - users that interact with a component
  - collaborative components



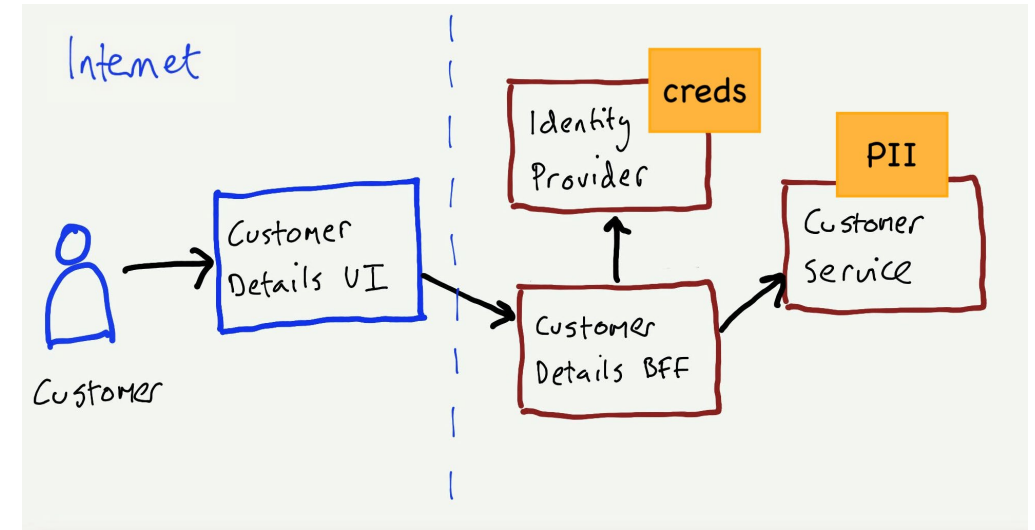
# An Example

- What are we building?
- Explicitly model data flows in the sketch.
- Data flows help show where requests originate (source).
- Label networks and show boundaries between them.
  - Collaborate with DevOps if you need to include firewalls and load balancers in the analysis.



# An Example

- What are we building?
- Identify and show assets e.g., personally identifiable information (PII), your application has access to.
  - Often derived from business requirements and the operating environment.



# Brainstorming Threats

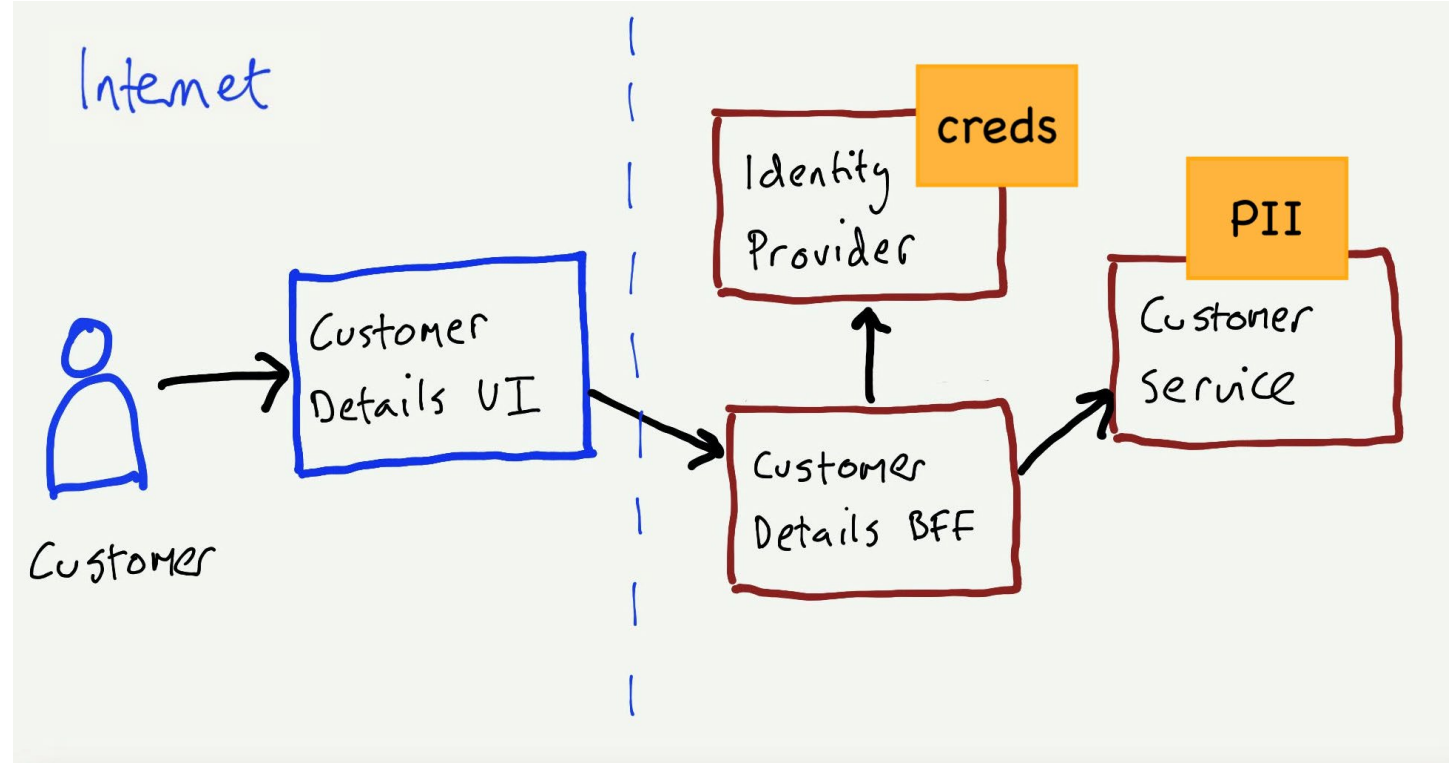
- The **STRIDE** framework is useful to reason about potential threats.
  - **S**poofing.
  - **T**ampering.
  - **R**epudiation.
  - **I**nformation Disclosure.
  - **D**enial of Service.
  - **E**levation of Privilege.

# Security Properties

- A *threat* is a potential danger or risk that could compromise the CIA properties of an application.
  - **C**onfidentiality: Ensuring that sensitive information is accessible only to authorized individuals or entities.
  - **I**ntegrity: Guaranteeing that data remains unchanged and uncorrupted during storage, transmission, or processing.
  - **A**vailability: Ensuring that information and resources are accessible and usable when needed by authorized users.
- Addressing the STRIDE threats helps meet the CIA properties of a system

# Back to The Example

- Using the STRIDE model, we should identify possible threats that can happen on each flow.
  - Customer -> UI
  - Customer -> Identity Service
  - UI -> BFF
  - BFF -> Customer Service

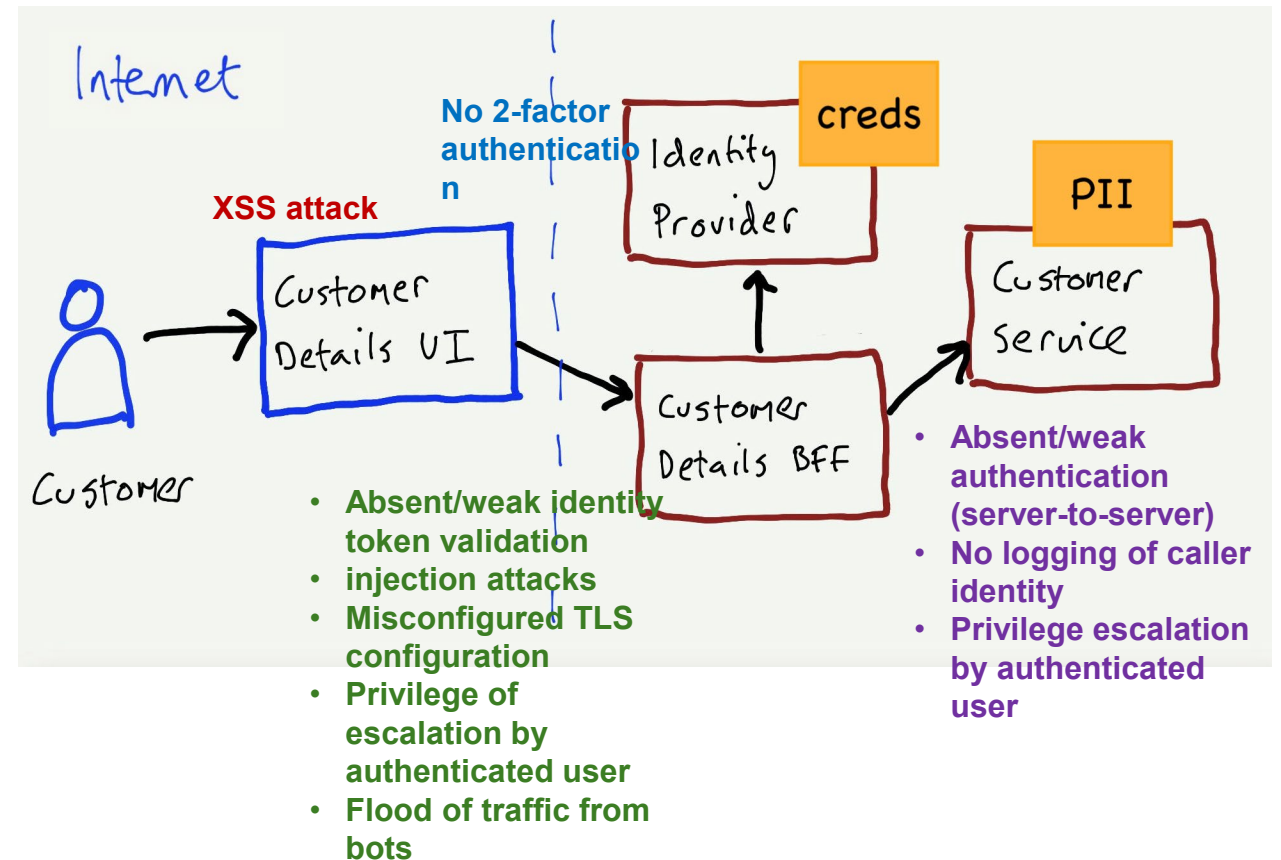




# Back to The Example

- Using the STRIDE model, we should identify possible threats that can happen on each flow.

- Customer -> UI
- Customer -> Identity Service
- UI -> BFF
- BFF -> Customer Service



# Prioritize and Fix

- Threats need to be prioritized as teams don't have unlimited bandwidth and some threats may not be significant.
- Prioritization criteria:
  - Will the threats jeopardize organization objectives?
  - Opinion of product owners and security teams.
  - What does everyone on the team think? Vote!
- Aim to address at least three threats. Could be more but three is a manageable number.



# Prioritize and Fix

- Threats of the highest priority must be added to the requirements (or backlog for agile).
- Few ways to concretely document.
  - **Conditions of Satisfaction:** Extend an extended scenario with additional (security) constraints in a way that is testable.
  - **Story:** The identified threat might need a story of its own.
  - **Definition of Done (DoD):** If all features need to be extended then specify it as a DoD applicable to all features.
  - **Epics:** A significant architecture specification to address a threat. E.g., adding an identity provider or configuring a network gateway.

# Example Specification

- Suppose the team identified the following threats to be addressed:
  - Authorization bypass when accessing an API.
  - XSS attack via user input.
  - Denial of service from the internet.

Given the user is logged in  
When they request to view their profile page  
And they have a valid token  
Then their profile page is displayed

Given the user is logged in  
When they request to view their profile page  
But they do not have a valid token  
Then they are asked to login or signup

**Extend the existing view user profile page scenario with *conditions of satisfaction* for authorization bypass.**

All API changes tested for sanitization of XSS and SQL injection attacks.

***This applies to all features. Hence, expressed as Definition of Done.***

all Internet facing UI and API requests to pass through the Content Delivery Network to prevent DDoS attacks.

***An epic that requires architectural changes in collaboration with a security expert and the DevOps team.***

# Additional Reading

- A Guide to Threat Modeling by Jim Gumbley.
  - <https://martinfowler.com/articles/agile-threat-modelling.html>