# Thoughts on Threat Modelling

John DiLeo (@gr4ybeard)

Gallagher Security  and OWASP NZ

September 2025

# Thank You to Our Sponsors and Hosts!











**Without them, this conference couldn't happen.**

# About Me

- Past lives
  - LONG-time student (4.9 degrees)
  - Simulation developer / analyst
  - University lecturer
  - Web developer and architect (J2EE)
- Doing Application Security (AppSec) since 2014
- Moved to New Zealand in 2017

# About My Day Job

## Application Security Lead

- Lead Cybersecurity Services Team
- Threat Modelling Program
- Product Security Risk Management
- AppSec Maturity Uplift
- Cybersecurity & Privacy Impact Assessments (CPIAs)
- In-House AppSec Training
- AppSec Evangelist

# About My *Other* 'Job'

**Chapter Leader, OWASP New Zealand**

- Hamilton Meetup
- Regional Training Days

**Chair, OWASP New Zealand Day Conference, 2019-2025**

**OWASP SAMM Project – Core Team**

**Launched SAMMwise and State of AppSec Survey Projects**

# Software Assurance

"Level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at anytime during its lifecycle and that the software functions in the intended manner."

*- [US] National Information Assurance (IA) Glossary, April 2010*

# A Software Assurance Program

## Purpose

To provide confidence to all stakeholders that software products are free from vulnerabilities – intentional or unintentional – and that those products reliably function as intended

## Goals

- Foster “Secure by Design” culture
- Improve code-level security of delivered software
- Focus on threats and risks in defining requirements
- Increase development efficiency
- Educate developers in best practices
- Assess and improve program maturity

# Threat Modelling as Part of a Software Assurance Program

- **OWASP Software Assurance Maturity Model** (SAMM) 2.0
  - **Design** Business Function
    - **Threat Assessment** Practice
      - **Threat Modeling** Stream
        - Level 1: Best-effort/*ad hoc* modeling
        - Level 2: Standard processes and tools
        - Level 3: Optimization and Automation
- **Microsoft Security Development Lifecycle** (SDL)
  - SDL Practice: Threat Modeling
- **Building Security In Maturity Model** (BSIMM)
  - **Attack Models** Practice
  - **Architecture Analysis** Practice

# How can we find security issues in our applications and systems?

# Some Approaches

- Static analysis of code
- Dynamic testing
- Penetration testing
- Production bug reports
- Incident response

**“Wouldn’t it be better to find security issues before you write or deploy a line of code?”**

*Adam Shostack*

# The Five W's of Threat Modelling

# WHY Threat Model?

- Improve efficiency
  - Think about security issues early
  - Invest effort more wisely
- Understand requirements better
  - Bring security and development together
  - Shared, maintainable, understanding of risks
- Avoid writing security issues into our code
  - Avoid costs of rework
- Improve stakeholder confidence
- And increasingly…because the regulator said so

# SIDEBAR: Terms of Reference

# Asset

Anything we need to protect, such as:

- Customer data
- Intellectual property
- Competitive information
- Reputation
- Compute spend
- System availability

# Threat

Anything that could let someone or something obtain, damage, or destroy an **ASSET**, if we fail to protect against it

# Vulnerability

A weakness or gap in our protection efforts

# Risk

Potential for loss, damage, or destruction of an **ASSET**, due to a **THREAT**'s having successfully exploited a **VULNERABILITY**

# Security Control

A safeguard or countermeasure implemented to avoid, detect, counteract, or minimize one or more security **RISKS**

# Compensating Control

A mechanism put in place to satisfy the requirement for a **SECURITY CONTROL** deemed too difficult or impractical to implement directly

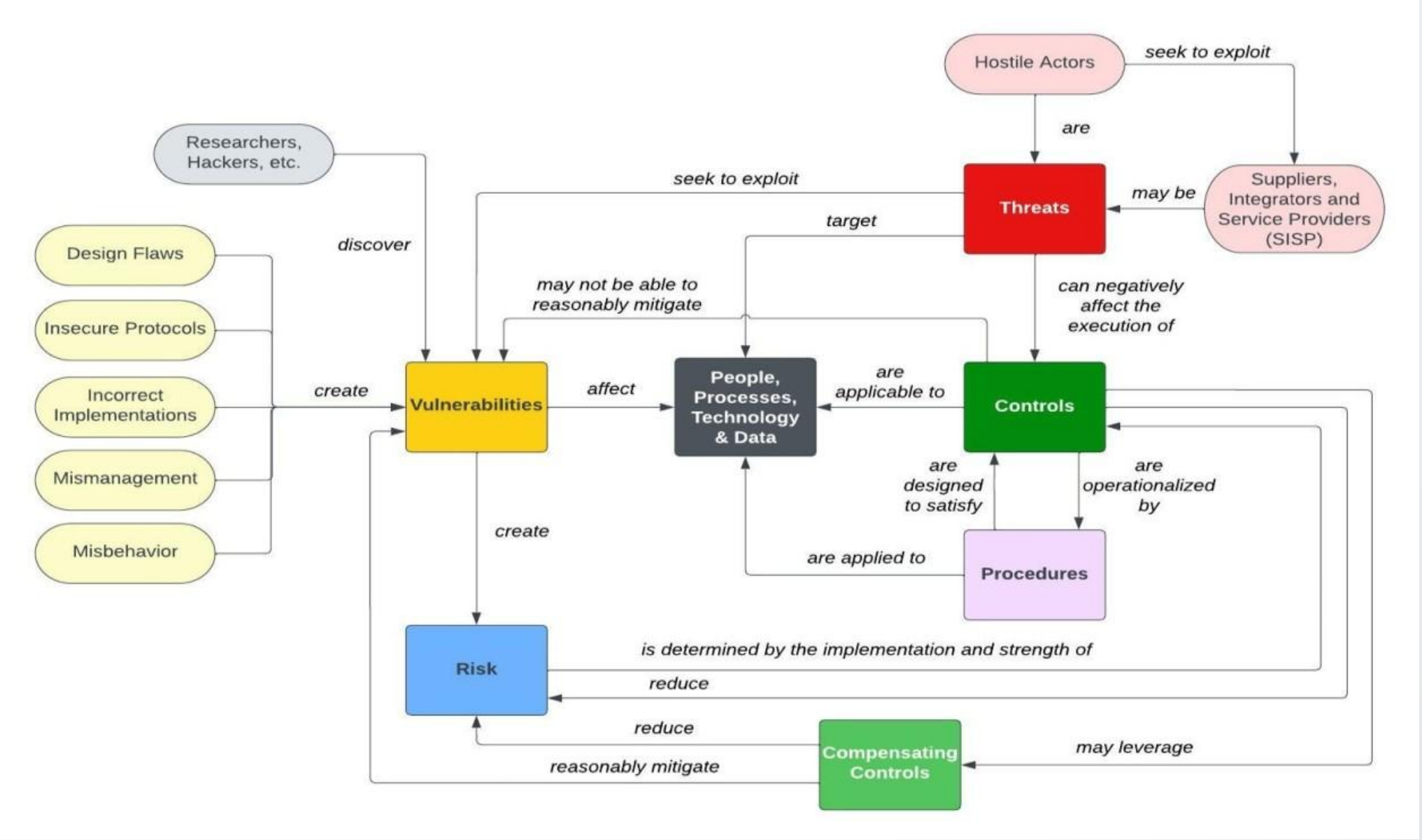
# Control Functions

- Prevention
- Detection
- Recovery

# Control Types

- Technical
- Physical
- Administrative

Multiple controls, from all type and function categories, might be combined to mitigate a **RISK**.




Source: https://www.linkedin.com/pulse/risk-ecosystem-interaction-risks-threats-vulnerabilities-/

# Model

A representation or simplified version of a system. Objectives of a model include:

1. to facilitate understanding by eliminating unnecessary components,
2. to aid in decision making by simulating 'what if' scenarios, and
3. to explain, control, and predict events on the basis of past observations.

A model contains only those features that are of primary importance to the *model maker's* purpose.

All models have a key feature in common: some elements of the actual 'thing' are abstracted.

--*Excerpted from* businessdictionary.com

# WHAT Is a Threat Model?

A **Threat Model** is a conceptual representation of a system, accompanied by a compilation of:

- ***Assumptions*** made in building the model;
- Identified ***threats*** to the system;
- ***Countermeasures*** (controls, mitigations) selected to reduce/eliminate risks arising from the threats;
- Countermeasure ***verification*** procedures; and
- Model ***validation*** approach

# WHAT Is a Threat Model?

Key considerations:

- To be useful ***to more than one person***, the model must be captured in a persistent, shareable form.
- To ***remain*** useful, the model must be kept up-to-date and aligned with the real system.

# WHO Are Our Stakeholders?

- Customers / End Users
- Data Subjects
- Collaborating Enterprises
- **Certification Bodies / Auditors**
- **Government Regulators**
- General Public
- **Cyber Insurance Providers**

# WHO Should Create the Threat Model?

- All stakeholders should be represented
  - Security ”experts” should advise *only*
- Assign lifecycle roles:
  - Owner (Accountable)
  - Maintainer (Responsible)

# WHEN to Create the Threat Model?

**“The best time to plant a tree was 20 years ago.
The second-best time is now.”**

- Start as early as possible
- Existing system, without a Threat Model?
  - Start NOW
  - Use Incremental Threat Modelling approach (Irene Michlin)

# WHEN to Update the Threat Model?

My recommendation:

- Review Threat Model every update cycle
  - Do the proposed changes affect the model?
  - If ‘yes,’ include model update efforts *in the cycle*
- OK…but what’s an “update cycle”?
  - Agile/iterative: Each Sprint, or each Release
  - Waterfall: Each change order

# WHERE Should the Threat Model Live?

- With other project/product documentation
  - Well-known location, with reliable backups
  - Ideally, place under revision control
  - Align model versions with product

# HOW Do I Build a Threat Model?

# Threat Modelling Approach

## DiLeo's "Seven Questions"

1. What are we building? (DFD)
2. What can go wrong? (STRIDE, Risk Patterns)

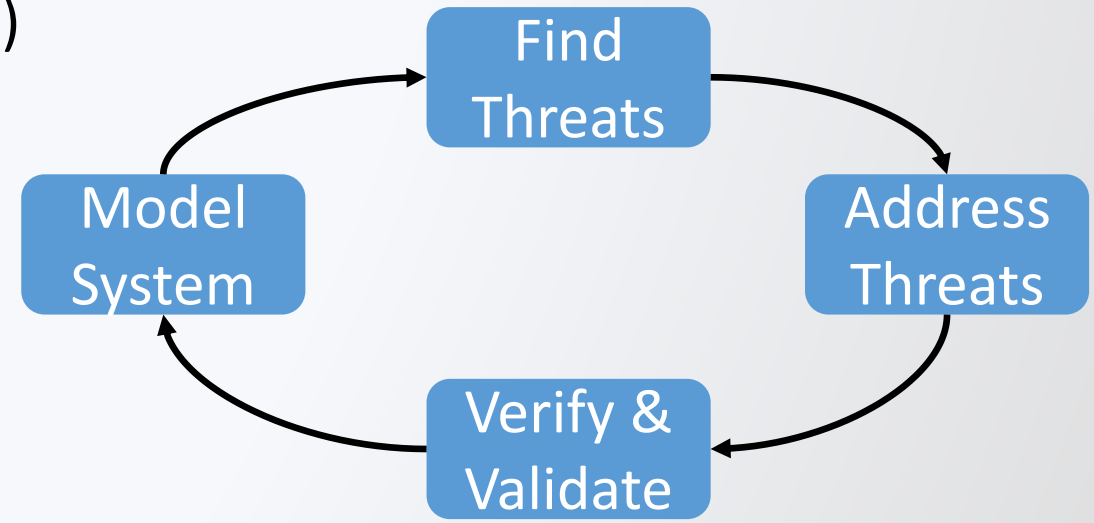
3a. What *might* we do about it? (Identify)

3b. What *will* we do about it? (Select)

3c. Have all ***residual risks*** been accepted?
If "No," repeat #3b, until "Yes"

4a. How will we know it works? (Verification)

4b. Is our model correct? (Validation)



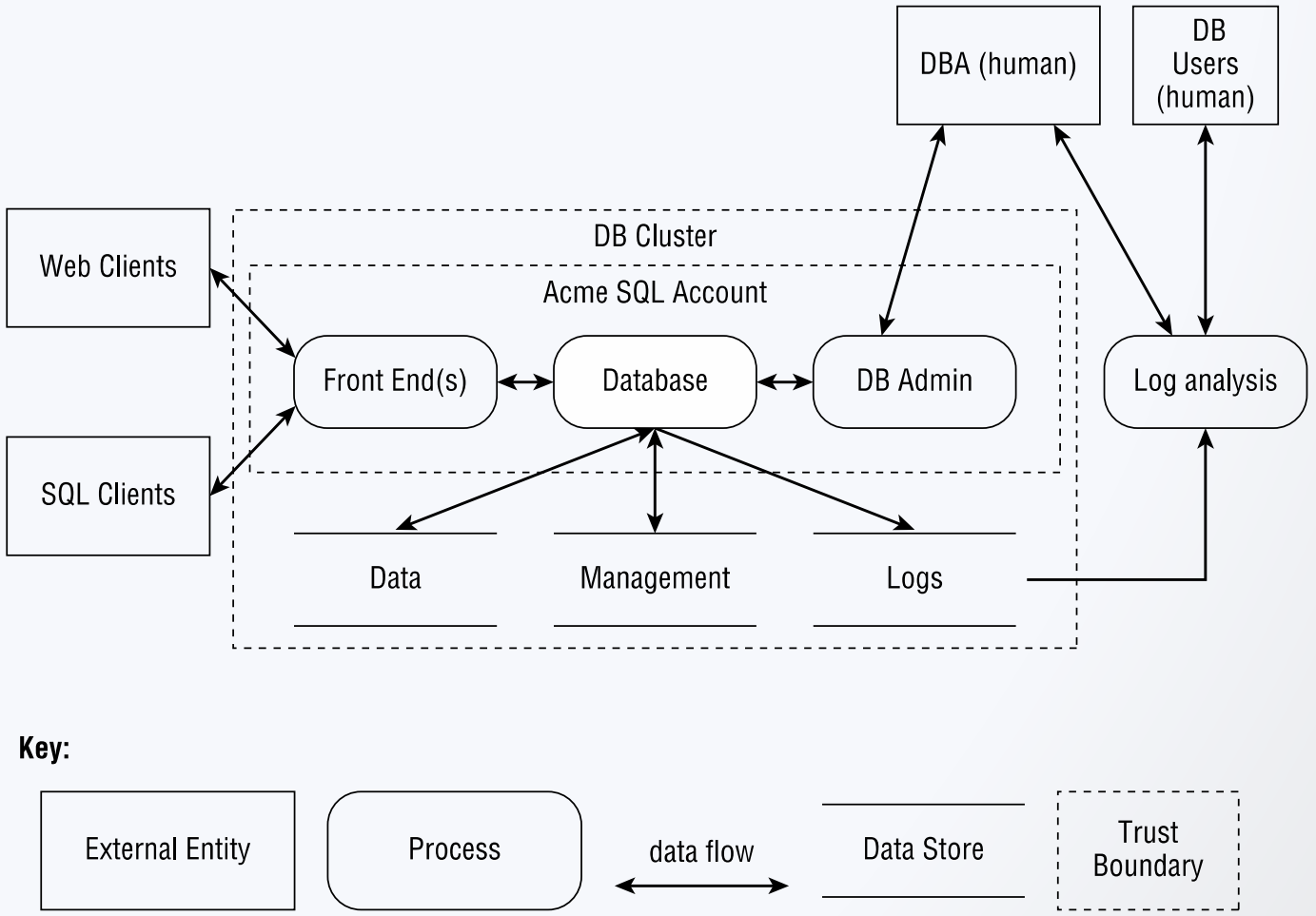
# What Are We Building?

- Create a model of the system
  - Technology used
  - Data stored and processed
  - Software created or used
- A model abstracts away the details so you can look at the whole
  - Diagramming is a key approach
  - Whiteboard diagrams are a great way to start

# Data Flow Diagram (DFD)

- Around since the early 1970s
  - Simple: easy to learn, easy to draw
  - Threats often follow data
- Abstracts programs into:
  - Processes: Your code
  - Data Stores: Files, databases, shared memory
  - Data Flows: Connect processes to other elements
  - External Entities: Everything but your code & data
    Includes people and cloud software
  - Trust Boundaries

# Data Flow Diagram (Example)



# What Can Go Wrong?
## Identifying Threats – Option 1

When Threat Modelling 'Manually'

**STRIDE** mnemonic

- Spoofing
- Tampering
- Repudiation
- Information Disclosure
- Denial of Service
- Elevation of Privilege

# What Can Go Wrong?

## Identifying Threats – Option 2

When Using ‘Automated’ Threat Modelling Tools

- Built-in Component Libraries
- Pre-identified Threats, associated with each Component
- Review identified threats, confirm applicability

# What *COULD* We Do about It?
## Identifying Possible Mitigations

For each identified threat, we could:

- Remove it (Avoid the risk)
- Implement countermeasures (Mitigate the risk)
  - Technical
    - Preferred: Well-known commercial/open-source solutions
    - ***If you must***, Custom mitigations – "roll your own" security
  - Non-technical
    - Physical protections
    - Administrative processes
- Do nothing (Accept the risk)
- Make it someone else's problem (Transfer the risk)

# What *WILL* We Do about It?

## Selecting Mitigations to Implement

Two-stage process:

1. For all mitigations that are easy, mandatory, and/or standard, *just do them*
   - Mark all relevant threats as mitigated
2. For all *remaining* threats:
   - Assess risk to system if *not* mitigated
   - Review candidate mitigations – cost vs. benefit
   - Select mitigation(s) to apply…or accept risk

# Verifying Mitigations

Every selected countermeasure constitutes a *consequential requirement*, which can be tested or verified

- Functional security features:
  - Positive and negative test cases
  - Regression tests
- Security Specifications: Verification checklists

Use threat model as a source for test cases

- Automate wherever possible
  Test manually ***only*** if you must

# Validating Our Modelling Work

Does the model accurately represent the *as-built* system?

Have all selected countermeasures been implemented and tested/verified?

Are all assumptions still valid?

# Getting Started

Staged process:

- Awareness and Education
- Add to AppSec policy / standards (not mandatory *yet*)
- *Carefully*-chosen pilot projects
- Just-in-Time training
- Success Managers (Security Champions)
- Celebrate successes, publish lessons learned
- Phased roll-out
- **THEN**…Make Threat Modelling mandatory

# Legacy Systems

Embrace Incremental Threat Modelling

- Irene Michlin
- OWASP AppSec EU-Belfast, 2017 (YouTube)

TL;DR:
Threat model only elements of the legacy system within scope of proposed changes

- And the rest? We're not making it *worse*.
- Do this for *every* change, and coverage will grow

# Threat Modelling Tools

You don't *necessarily* need a tool, when starting out

- Whiteboards and sticky notes
- Visio, Lucid Charts, Draw.io

Free tools (e.g., OWASP Threat Dragon) are often enough for small portfolios and pilots

Commercial tools provide economic *benefits* for medium-to-large portfolios (> 15 models)

- Forrester Total Economic Impact Study (IriusRisk)

# Questions?

## Connect / Reach out

- Email:
  - Day job: *john.dileo@gallagher.com*
  - “Other job”: *john.dileo@owasp.org*
- Twitter: *@gr4ybeard*
- LinkedIn: *john-dileo*
- OWASP Slack *https://owasp.org/slack/invite*

# OWASP NEW ZEALAND DAY 2025

owasp.org.nz