Threat Modeling at Scale

From One to Many

WINNISEC, April 2025

Today’s Agenda

01
Threat Modeling?

02
Once

03
Many

04
Takeaways

01
Threat Modeling?

Seeks to identify security deficiencies in a system

System can be an application, environment, or ecosystem

Prioritization of threats based on decompressing the system

Commonly applied to developer workflows, but can apply elsewhere

With enough information, anything can be threat modeled

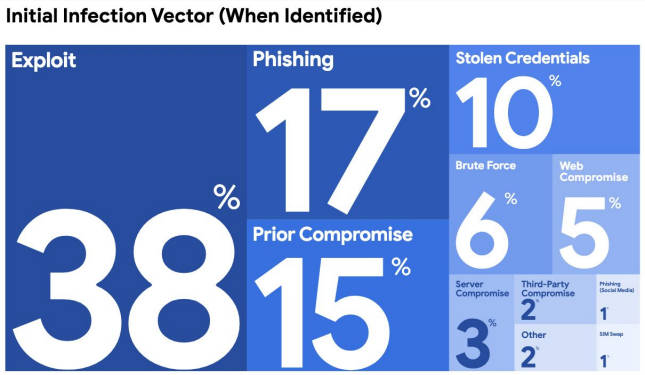
Why?

Prevent issues from reaching production

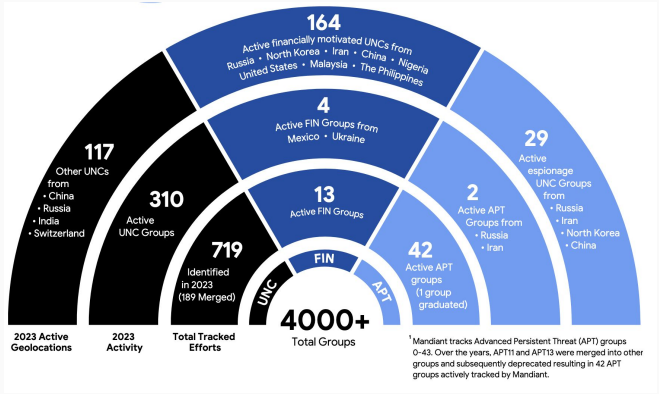
Analyze the full scope and intent of the target of the threat model

Prioritize security controls and resources that will have the most impact

Optimize usage of internal resources, skills, and capabilities



[1]



[2]

[1][2]: https://services.google.com/fh/files/misc/m-trends-2024.pdf

Sample Findings

Business Logic Issues

- Payment processing orchestration on client-side
- Lack of data validation on APIs
- Authentication flow issues
- Lack of controls applied to business process

Operational Issues

- Lack of monitoring, use cases, response planning
- Unauthenticated access to infrastructure
- Lack of understanding of how system operates
- Lack of segmentation in network architecture

Development and DevOps Issues

- Overly permissive version control system and repositories
- Lack of enforced code reviews and testing
- Hardcoded secrets
- Vulnerable third-party dependencies in use
- Lack of CI/CD pipeline segmentation

Key Takeaway

Threat modeling is a tool that can help prevent or identify security issues within our business systems.

But wait, there’s more!

Testing and Validation

Threat model outputs can be used when scoping or executing validation (e.g., penetration testing).

Threat Hunts

Threat scenarios can feed into threat hunt programs and adopted as the basis for threat hunt hypothesis.

Threat Detection

Threat scenarios and reference architecture can be used to inform or define threat detection use cases.

Threat Response

Threat modeling can be used to identify incident response playbooks or ‘plays’ to be developed for common scenarios.

Timing Considerations

Early, and often!

Where it fits:

- During the development process, based on design or development artifacts - be cautious
- Before production deployments during the development lifecycle, based on design or development artifacts
- After production deployments, based on architectural diagrams or configuration data
- Within other organizational processes (e.g., procurement, architecture reviews, risk assessments, etc.)

When to revisit:

- Significant architecture or technology changes
- Significant shift in threat actor tactics or motivations
- An incident impacting the system

02
How to Threat Model Once

Many common methodologies exist, all with pros and cons:

- STRIDE
- PASTA
- VAST
- LIDDUN
- MAESTRO
- Adam Shostack's 4-Question Framework
- NIST SP 800-154

The high-level flow outlined leverages multiple frameworks/methods

High-Level Flow

Learn and Build Reference Architecture

Understand the target system's components, connections, and workflows

Collect Threat Intelligence

For our target system, understand what threat actors are motivated by and their TTPs

Create Threat Scenarios and Attack Paths

Using the reference architecture, determine how threat actors can achieve their objectives

Prioritization and Remediation

Determine which threats are likely, and how to prevent, detect, or respond to them

03
How to Threat Model at Scale

Centralized Model

“All for one…”

- Dedicated threat modeling team
- All requests for exercises flow through a static team
- Excels when threat modeling functionalities are purpose-built
- E.g., performed by security operations team for use case development

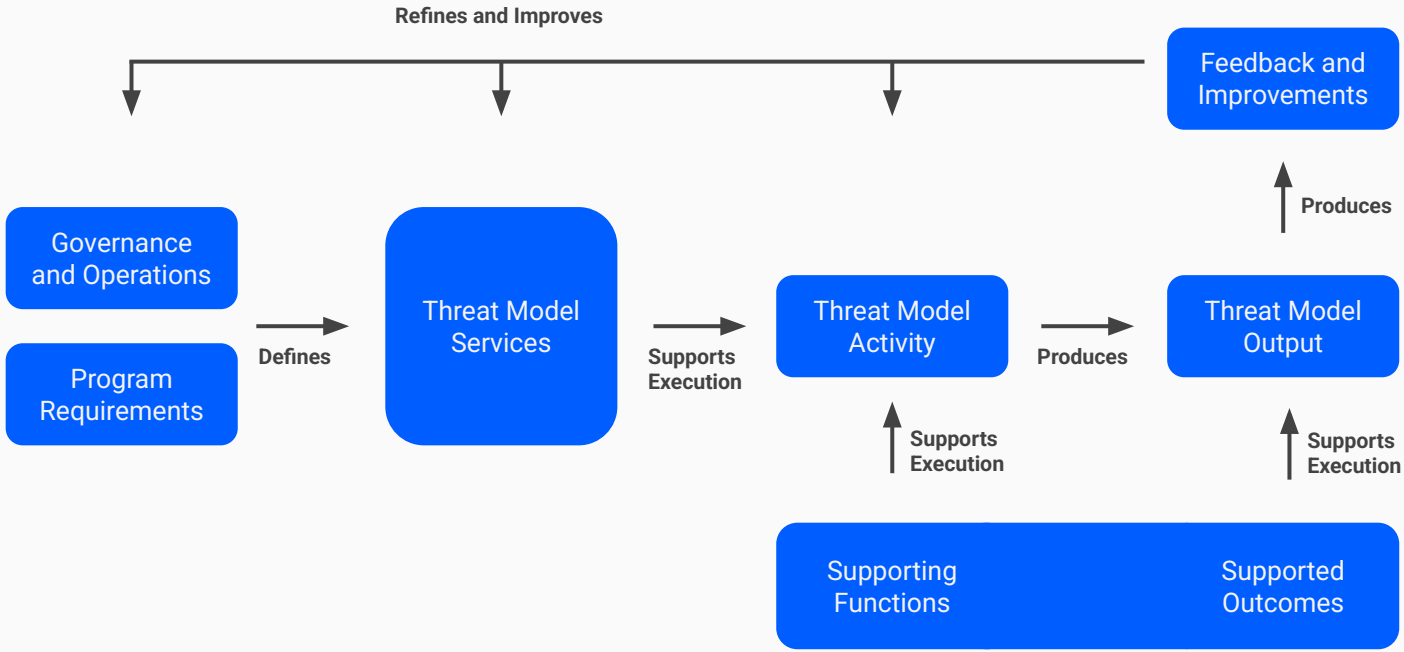
Centre of Excellence Model

“...And one for all”

- Dedicated multidisciplinary threat modeling function
- All requests for exercises flow through a dynamic team
- Excels at providing scalability and subject matter expertise
- Encourages development of Threat Model Champions across organizational units and teams

Developing a Program: Establishing the Program Constructs

“Repeatable Threat Modeling Framework (RTMF)” - github.com/dylanam204/rtmf - Soon™



Developing a Program: Operationalization Considerations

"Repeatable Threat Modeling Framework (RTMF)" - github.com/dylanam204/rtmf - Soon™

Developing a Program: Continual Improvement

“Repeatable Threat Modeling Framework (RTMF)” - github.com/dylanam204/rtmf - Soon™

Methods to assist with continual improvement of the exercises and program:

- **Program governance**
  - Program maintainers and members
- **Metrics**
  - Definition of metrics to capture
  - Collection of metrics (manual and automated)
- **Feedback forms**
  - Two layers of feedback:
    - Activity team members
    - Stakeholders/participants
- **Considerations**
  - Continually adjust and change
  - Measure against a maturity model

04
Key Takeaways

Questions?

Want to speak at a future event?

talk@winnisec.life

- **Threat Modeling…**
  - Is a method to determine where security can be introduced or enhanced in a given system
  - Outputs can help support the organization from many perspectives
  - Can help us focus where to spend our resources
  - Can help us optimize the usage of our tools, technologies, or capabilities
  - STRIDE, 4-Question Framework, NIST SP 800-154
- **A Threat Modeling Program…**
  - Can help us consistently and regularly conduct threat modeling
  - Can be implemented using a centralized or decentralized model
  - Provides a framework to conduct activities, scale, and continually improve
- **To get started…**
  - Adopt a methodology
  - Conduct one threat model
  - Continue to add-on, scale, and grow - start slow and build for purpose