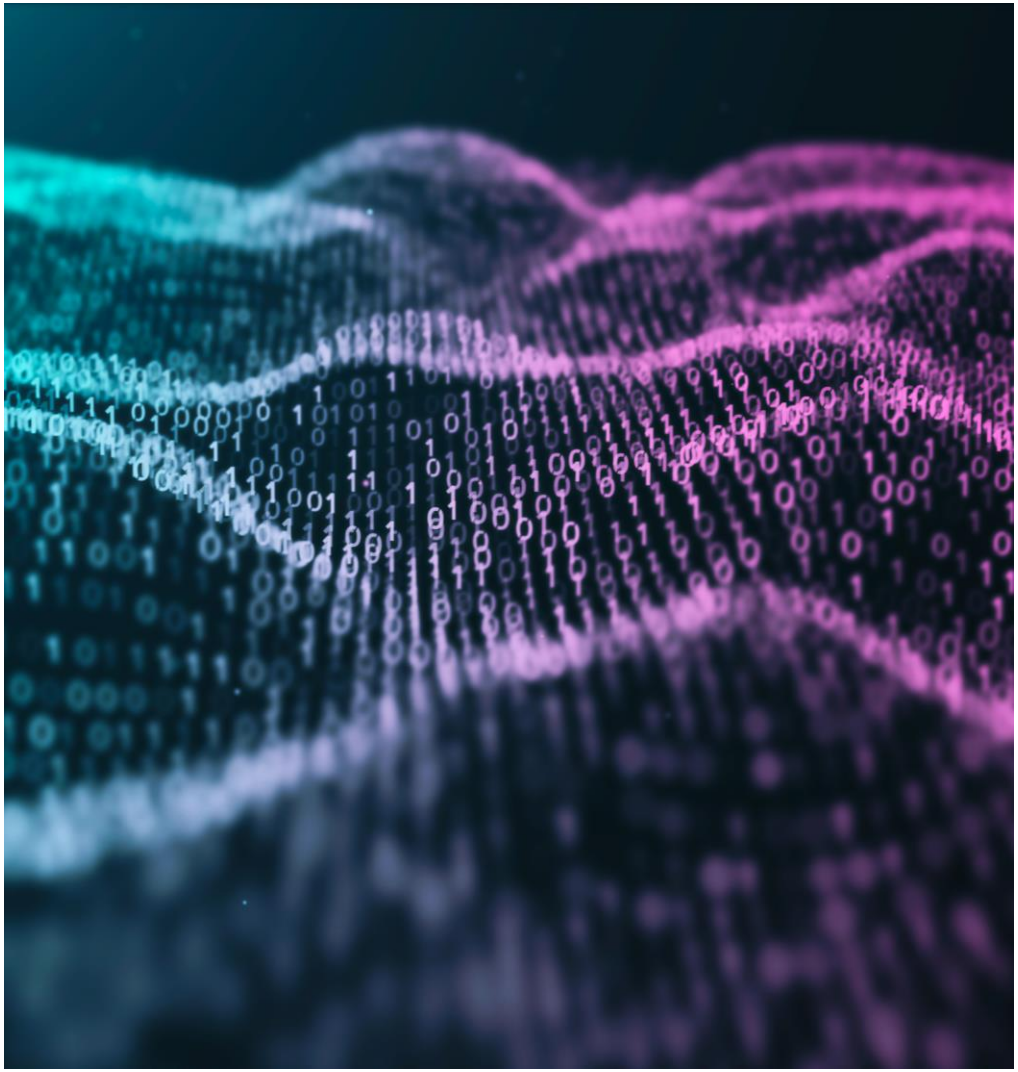
Threat Modeling in DevOps: A Practical Guide for Developers

Krishna Chaitanya Rudraraju



Why Threat Modeling in DevOps?

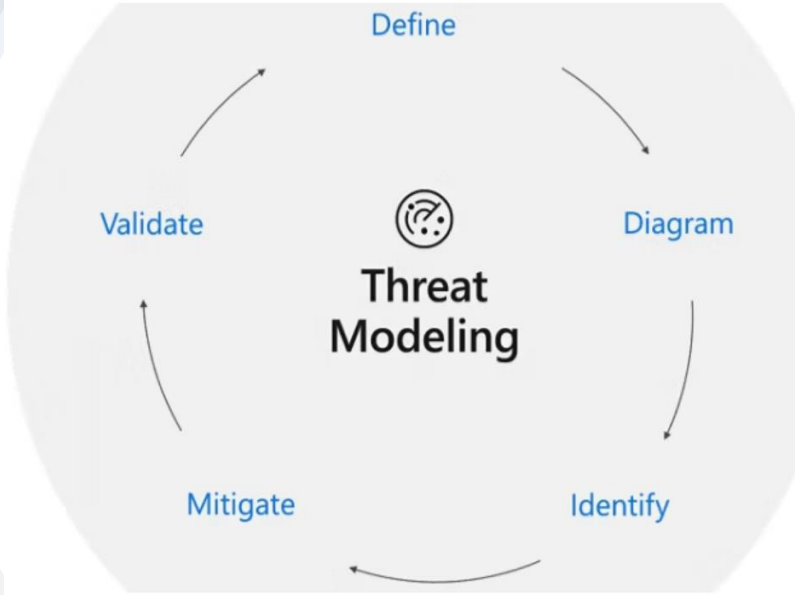
- DevOps prioritizes speed and agility, but security can feel like a blocker.
- Threat modeling integrates seamlessly into DevOps workflows to:
  - Identify vulnerabilities early.
  - Reduce remediation costs.
  - Enhance the overall security posture.

What is Threat Modeling?

A structured process to identify, evaluate, address potential threats and enable proactive security measures.

Core Questions:

- What are we building?
- What can go wrong?
- What are we going to do about it?
- Did we do a good job?

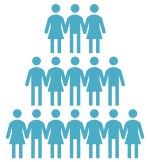


Key Benefits of Threat Modeling

- Early risk identification in SDLC.

- Seamless integration into CI/CD pipelines.

- Reduces vulnerabilities in production.



- Encourages collaboration between developers, operations, and security teams.

STRIDE Framework Overview

Spoofing: Fake identity or authentication.

Tampering: Data modification.

Repudiation: Denying actions.

Information Disclosure: Data leaks.

Denial of Service: Disrupting services.

Elevation of Privilege: Gaining unauthorized access.

Threat Modeling in a DevOps Workflow

1. Plan: Identify potential risks during feature planning.
2. Develop: Automate code analysis and dependency checks.
3. Build: Secure your CI/CD pipelines (e.g., signed artifacts).
4. Release: Perform penetration testing and verify controls.
5. Deploy: Monitor runtime environments for anomalies.
6. Operate: Continuously assess and respond to new threats.

Case Study: E-Commerce Application

Application Overview:

- React Frontend, Node.js API, MongoDB, Kubernetes on Azure.
- CI/CD with GitHub Actions and Azure DevOps.

Threats Identified:

- **Spoofing:** Fake tokens.
- **Tampering:** Altered Docker images.
- **DoS:** API flooding.

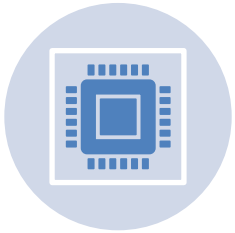
Mitigations Implemented:

- OAuth2 for authentication.
- Signed Docker images.
- Rate limiting and auto-scaling in Kubernetes.

Tools for Threat Modeling

Microsoft Threat Modeling Tool: Visualize and assess threats.

OWASP Threat Dragon: Open-source modeling tool.



Snyk & Checkmarx: Dependency and code analysis.

Azure Monitor & Sentinel: Real-time monitoring and threat detection.

Best Practices for Success

1. Collaborate early and often.

2. Automate wherever possible.

3. Regularly update models with new threats.

4. Train your team on security principles.

5. Monitor and iterate on security measures.

Call to Action

- Embrace threat modeling as a DevOps enabler, not a blocker.

- Start small, focus on critical areas, and grow your practices.

- Build secure, resilient applications without compromising speed.

Q&A

Let’s discuss: How can threat modeling fit into your DevOps workflows ?

Contact:
Krishna Chaitanya Rudraraju
Senior Software Engineer - Microsoft
LinkedIn: /krishnachaitanyarudraraju