Carnegie Mellon University
Software Engineering Institute

Using Threat Modeling to Guide Everything in DevSecOps

December 2021

Kenneth R. van Wyk
KRvW Associates, LLC

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Carnegie Mellon University Software Engineering Institute

DEVSECOPS DAYS

Contents

Carnegie Mellon University Software Engineering Institute

Using Threat Modeling to Guide Everything

Introduction

What's So Special About Threat Modeling?

We use threat modeling to assess the security of a design, whether proposed or in production

- Done well, it's a pretty good process step for just that

Necessary ingredients

- Knowledgeable and motivated team of people
- Big whiteboard (and the means to record its content)
- Spreadsheet or tracking tool

That's it? What about tools?

Carnegie Mellon University Software Engineering Institute

DEVSECOPS DAYS

Results

Most threat modeling processes generate a prioritized list of potential design flaws

- Tools focus on tabulating and tracking those flaws
- Some integrate with existing bug tracking systems

Nothing wrong with that, but…

Singular Focus Misses Opportunities

A threat model and the knowledge built during the process has so much more intrinsic value

After all, DevSecOps embraces building and reviewing code once, and then building systems on accepted code building blocks

Let's take that concept and push it ever further left as well as benefiting rightward

- Code reviews, testing, and even production deployments can all benefit from the threat model

Carnegie Mellon University Software Engineering Institute

DEVSECOPS DAYS

Using Threat Modeling to Guide Everything

Coding Activities

Guide code reviews

It’s great to use static analysis tools and such to scan piles of code

Let’s also laser focus our human attention on the highest risk code blocks

- New functionality
- Handling of most sensitive data
- Access control around administrative processes

Focus manual reviews and/or human intervention around those code areas

Example: iPhone app with payment data

Threat model of popular iPhone/Android app with payment data persisted locally

- Storage needed to be portable across platforms
- Server could not be used for key management
- Threat model quickly spotlighted the sensitive functionality
- Manual code review of a few hundred (vs. 10s of thousands) LOC

Carnegie Mellon University Software Engineering Institute

DEVSECOPS DAYS

Using Threat Modeling to Guide Everything

Testing Activities

Guide testing

Similarly, guide test processes to areas of highest risk

- Threat scenarios from threat model (even the ones you dismissed)

Test approaches

- Validation testing of security-critical assumptions
  - Encryption, key management, TLS algorithms
- Instrument and observe

Fuzzing

- For particularly high risk areas, fuzz APIs and network targets
  - Key management, session management, access control of admin functions

Example: Hotel payment system

Threat model produced several threat scenarios, including malicious actor at hotel branch using API to steal payment data *en masse* from central database

- Testing led to addition of ”velocity checker” function to throttle high volume queries

Carnegie Mellon University Software Engineering Institute

Using Threat Modeling to Guide Everything

Production

Yes, even production

Threat model spotlights most dangerous functionality

- Production configuration of containers and servers focuses hardening, access control, etc., on those areas
  - Encrypted disk volumes
  - Data encryption at rest and in transit
  - Multi-factor authentication on most sensitive functions

Example: Cloud based business analytics application

Threat model found numerous high risk storage buckets

- Vulnerability management team validated volume encryption and other data protection methods functioning in production as planned
- Threat hunt team developed scripts to verify configurations and access logs for specific concerns

Carnegie Mellon University Software Engineering Institute

Sounds risky

It does, doesn't it?

Threat modeling is a great input to building a deep understanding of a system's intrinsic risks, at a technical level

Using Threat Modeling to Guide Everything

Threat Modeling Resources

Further reading

If you haven't already, check out

- *Threat Modeling: Designing for Security*, Adam Shostack, 2014
- *Software Security: Building Security In*, Gary McGraw, 2006
- *Enterprise Software Security*, Kenneth R. van Wyk (et al), 2014