Securing your Forge App: Do It Like Autodesk

Tony Arous

Director, Security Architecture & Engineering – Autodesk

Izar Tarandach

Lead Security Architect – Autodesk

FORGE DEVCON

AUTODESK. UNIVERSITY

About the speaker

Tony Arous

@tonyarous

Tony Arous is the Director of Security Architecture & Engineering at Autodesk. He is responsible for overseeing the security of the company's application development, including the strategy and execution of its security program for over 3,000 engineers. Prior to joining Autodesk, he spent over 10 years at Dell EMC (formerly EMC Corporation) where he most recently led Product Security Engineering and provided technical engineering security services. Tony is a Certified Secure Software Lifecycle Professional (CSSLP). He also has an MBA degree from Northeastern University and an M.S. degree in computer engineering from Tufts University.

About the speaker

Izar Tarandach

@izar_t

Izar Tarandach is a Lead Security Architect at Autodesk. Prior, he was the Security Architect for Enterprise Hybrid Cloud at Dell EMC, for long before a Security Consultant at the EMC Product Security Office. With more years than he's willing to admit to in the information security arena, he is a core contributor to the SAFECode training effort and a founding member of the IEEE Center for Security Design. He holds a Masters degree in Computer Science/Security from Boston University and has served as an instructor in Digital Forensics at Boston University and in Secure Development at the University of Oregon.

By show of hands, who are you?

Raise your hand if you…

- Are an innocent bystander interested in security
- Are a security expert in your daily job
- Support more than one product team in their security needs
- Are an architect, product manager or similar
- Are a developer
- Are curious, excited and sometimes overwhelmed by the expression “secure development”
- Are in the wrong room

Why are we here today?

WHAT IS SOFTWARE SECURITY?

A piece of software can be considered secure when it performs the tasks it is intended to in a way that is continuously reliable and does not expose the data and intellectual property it is tasked with protecting.

WHAT IS SDL/S-SDL/APPSEC?

The set of activities, policies, standards and training that ensures that the software development lifecycle generates secure software. It is sometimes called the Security Development Lifecycle (SDL), Secure Software Development Lifecycle (S-SDL) or simply Application Security (AppSec).

HOW DOES AUTODESK DO SECURITY?

Autodesk is constantly improving its security program in order to better serve its developers and customers. We follow the industry's best practices and adapt and improve on state-of-the-art techniques to safeguard ourselves and our customers from risk.

HOW CAN IT HELP YOU?

We hope you can use our experience to inform your own approach to security, reducing your learning curve and giving you tools to secure your own development processes.

But the real reason… why are *really* we here?

Why do we need to do anything?

- The Forge platform gives you a lot of flexibility – and with that flexibility, comes great responsibility.
- There are a few basics that every software developer can use to improve the security of their apps.
- You can reuse these same principles on any of your software projects.

What is Software Security?

It is critical to engineer software and systems so that the confidentiality and integrity of user data plus the availability of the system are maximized.

- Systems' complexity constantly go up, even though their building blocks' complexity goes down
- User's expectation of privacy and confidentiality go up as a function of their reliance on these systems
- Vendors are expected to have robust security processes in place that can support secure engineering and compliance to established guidelines

The Security Program At Autodesk

GUIDANCE

We establish clear ways of developing and fielding systems, and constantly research upcoming development technologies for their security impact. We interact with other companies to keep abreast of the constantly moving best practices available and to exchange experiences

SUPPORT

We provide tools and workflows to support the guidance and expectations, research issues brought up by product teams and educate towards secure development practices. We help product teams navigate communication with external parties.

VERIFICATION

We help make sure that secure development was correctly executed by using best practices of testing and investigation on our own products before they’re released

Secure Software Development Lifecycle

THREAT MODELING

DESIGN

We continuously examine our designs for flaws that can be corrected before coding.

STATIC & SOFTWARE COMPOSITION ANALYSIS

CODE

Teach developers to code properly by giving them the tools to verify their work as close to commit as possible. Integrate tooling in CI/CD and make reporting clear and accessible.

DYNAMIC ANALYSIS

TEST

Use best-of-breed tools to perform security validation of your software. Integrate and automate where possible and use manual testing when necessary.

PATCHING

RELEASE

Engage the security research community and customers about any rising issues; lead a responsible-disclosure procedure.

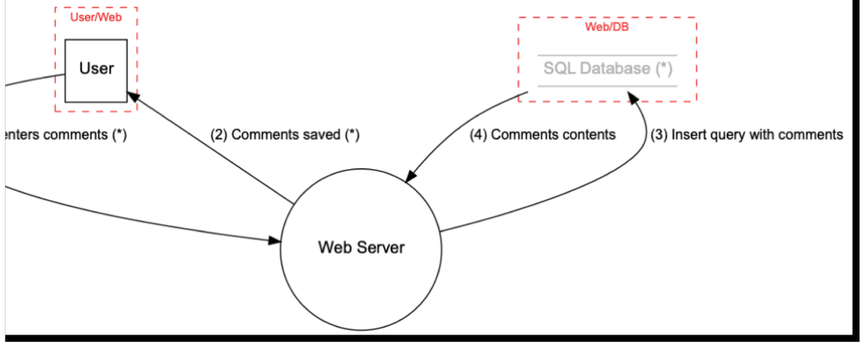
Establish clear SLAs and stick to them.

How Do We Do It?

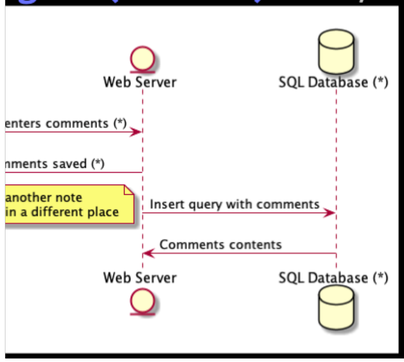
Threat Modeling

- A conceptual exercise that looks for flaws into the design of a piece of software; usually performed at design time
- Diagrams and flows help understand the way elements interconnect and relate to each other
- Many possible flaw/threat-identification techniques. Autodesk uses an experience-based focus approach

```
ytm git:(master) ✗ ./tm.py --dfd | dot -Tpng
```




```
git:(master) ✗ ./tm.py --seq | plantuml
```




```
eats here '''


tion": "Dataflow not authenticated",
: Dataflow,
on": "target.authenticatedWith is False",
s": "instead of looking at both sides of a flow
ides authenticate",
```

Threat Modeling: What Is It Good For?

“What is the use of a technique that centers on design flaws?”

- Identify threats
- Identify issues that can be solved at the design level by using the correct framework or structure
- Extend findings and documentation beyond design!

Threat Modeling: Does it scale?

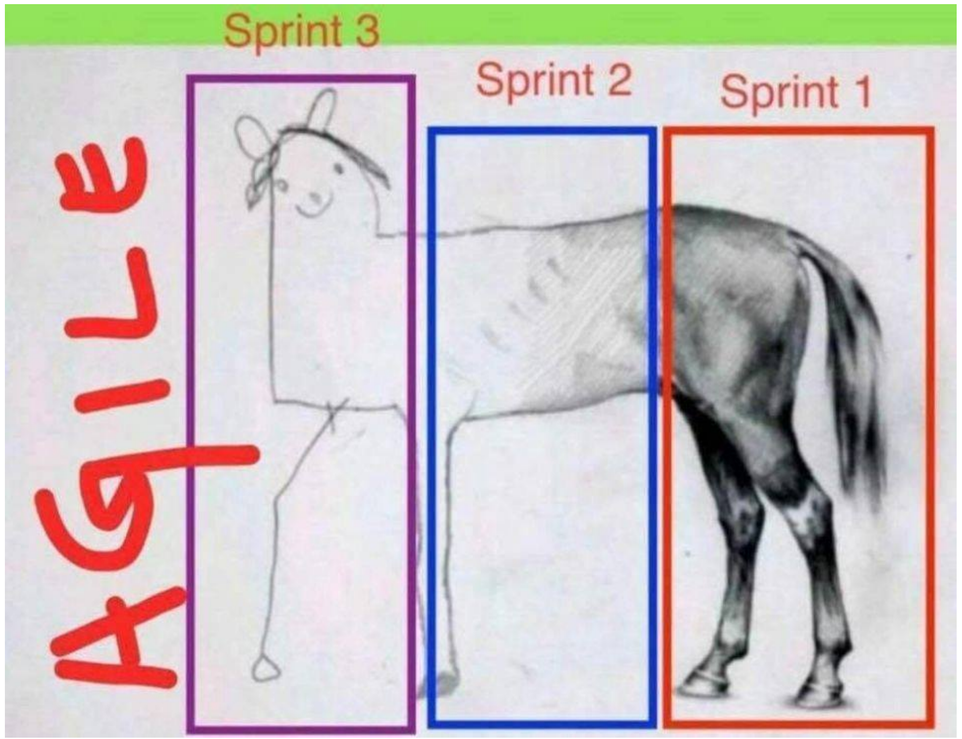
“It sounds like something a security expert would have to do. I don’t have security experts.”

- Transfer security knowledge to teams
  - Scale the AppSec function
  - The team knows their product better than AppSec ever will

Threat Modeling: We are an Agile shop

“We don’t do waterfall anymore. We are super agile. This will slow us down.”

- The Autodesk Threat Modeling Handbook
  - Methodology
  - Areas of focus
  - Security-notable events and checklist
- Evolutionary process
  - The perfect is the enemy of the good
  - Tomorrow’s TM will be better than today’s; today’s is better than none
  - Everybody learns!
- Continuous threat modeling
  - “Threat Model Every Story”



It’s all good – but how can we actually verify code?

Static and Software Composition Analysis to the rescue!

- Know your targets – what tools are available for your language(s) of choice?
- Do you need a complete static code analysis solution? Can you get by with guidelines, secure code reviews and checklists?
- Let’s talk checklists for a minute: Secure Development Checklist

Open Source and The Inheritance Tax

We use a library. That is written in a framework. That is written in a language that has an interpreter. And they're all Open Source. Everything depends on everything else.

- Who is responsible for this thing?
  - Patching
  - Source integrity (code-signing, anyone?)
  - Dependencies
- It is *your* responsibility to get on top of things. 3rd party tools are great, but nothing beats staying on top of your dependencies – so teach your teams to stay on top of theirs

DevSecOps – Avoiding the Oops

MAKE IT REPEATABLE

- Like any other testing methodology, your tests should be easy to add, and once added, easy to repeat
- No dependencies on human interaction
- Fire-and-forget – even if asynchronous

MAKE IT TRANSPARENT

- You don’t need the security expert to come in to start the security part of your cycle
- Make the reports and results part of the build deliverable
- Generate simple pass/fail results with drillable-down details

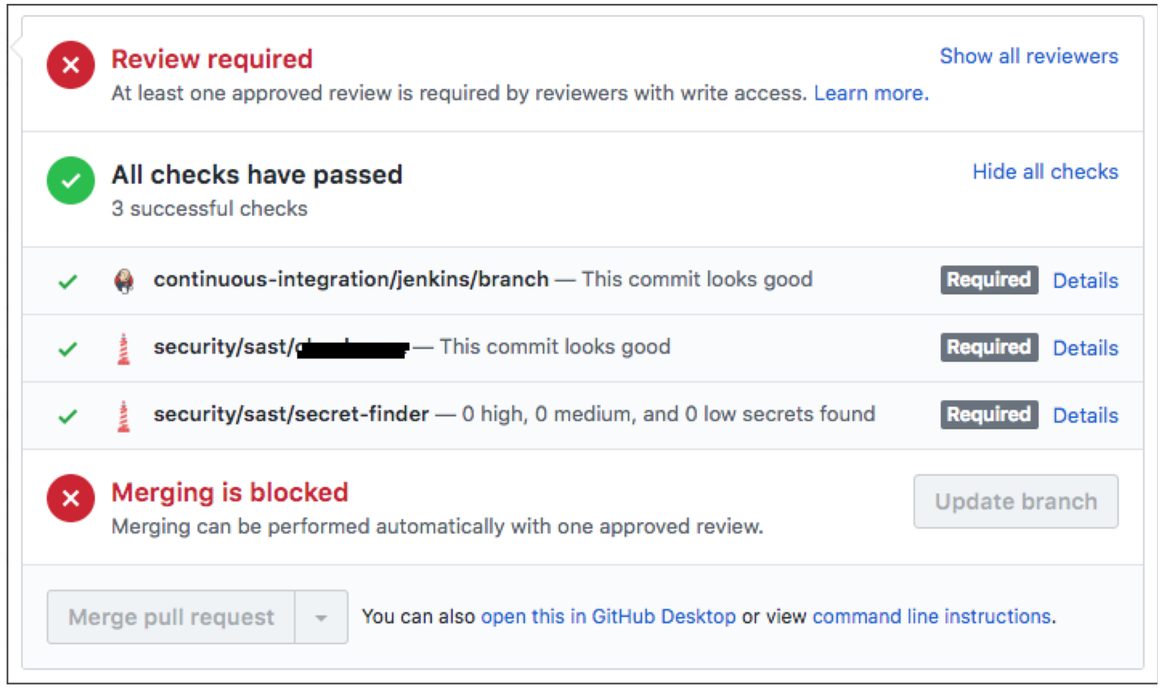
MAKE IT USEFUL

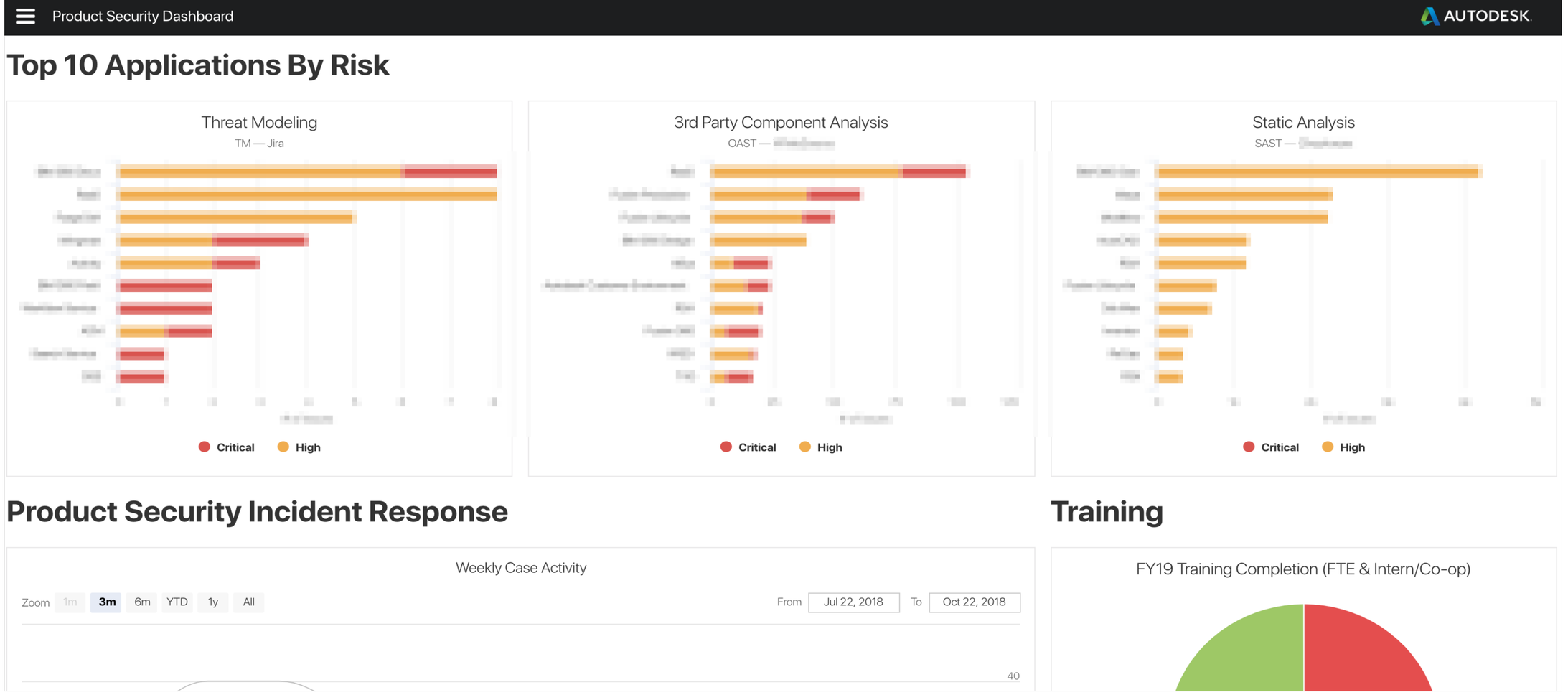
- Aim for minimum false positives, then go hunt the false negatives
- Report useful information
- Express risk, but also express how you’re mitigating that risk

How we use the tools we use

Integration is a great thing – provided it works.

- Source control is based on GitHub – great workflows, simple interface, accountability, and GitHub apps!
- GitHub app developed in-house:
  - Works on a single repo or a full organization
  - Runs selected security tools on every single push event (SAST, sensitive content finders, etc.)
  - Asynchronous – builds are not stopped
  - Reports linked back to specific commit
  - OPTIONALLY blocks merges based on status
- Concentrate results – have a dashboard that brings it all together





The importance of dashboarding

It is hard to fix what you can't see.

Thank You!

Questions?

Please let us know how we did!

AUTODESK®

Make anything™