RSAConference2016

San Francisco | February 29 – March 4 | Moscone Center

Connect to Protect

SESSION ID: ASD-R03

Introducing a Security Program to Large Scale Legacy Products

Millard Taylor (Tad)

Security Architect

@tad_taylor

#RSAC

Is This Your Situation?

- Establish a comprehensive security program:
  - Legacy product
  - Millions of line of code
    - Some over 20 years old
  - Still under development
  - Customers complaining
  - Significant revenue stream

What You'll Learn in This Talk

- Unique challenges to developing a security program for Legacy Systems
- What Worked Well
  - And What Didn't Work Well
- Running the Security Program once you have it

What's a Large Scale Legacy System?

- Mid-Range SAN & NAS Devices
  - Decades of development (read "old code")
  - Hundreds of developers (on the order of 1000)
    - Multiple locations, time zones, languages, etc.
  - Millions of lines of code
    - Much of it Linux®/Open Sourced-based

The Beginning: In Which a Grassroots Effort to Enhance Product Security Begins

Not Overnight

- I'd like to tell you that I immediately saw everything that needed to be done and went about whipping the program into shape
  - But too many people know the truth…..
- Things evolved over time (years)
  - Overall environment
  - Growing experience, study, and corporate initiatives
  - QA group's recognition that testing security was special
    - And in responding to customer requirements

Views of the Time….

- Attitudes that had to be overcome:
  - “We’re in the data center. We don’t need more (any) protection.”
  - ”They can use a private network for the connections.”
  - “Who would do that?”
  - “If we let the customer change the default password, how will we get in to fix anything?”

Grassroots Beginnings

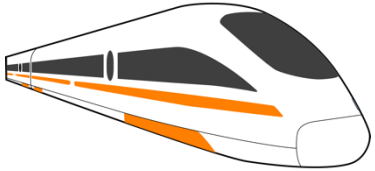
- A grassroots and customer-driven effort during the early years
- Taking the title “Security Gadfly”
  - Where no one is happy to see you at the review
- Customers to the rescue
  - Customers began to ask questions & make demands

self-discovery of a gadfly

Challenges

Challenges in a Legacy Product

- The architecture is a mystery
- Moving target



- Significant revenue stream
  - That you can’t screw up!

More Challenges

- Scale
  - With 1000 developers and too few security experts, you've got to figure out ways to leverage your efforts
- Security Features vs. Software Security
  - Can be difficult concept for some
  - Both are probably necessary

Challenges in Attitude

- Security groups always just say “No”
- Everybody has responsibility for security
- Who else does this?
- Is this a management priority?
- We can’t find enough people

What Worked Well: Tackling the Challenges

Capture the Architecture

- In the process of documenting/learning services, you address the mystery of the architecture
- Start with open ports
  - Then how are they protected, what do they do
- People are usually surprised by how complex things have become

Threat Model

@TAD_TAYLOR

Belief vs. Reality

HTTPS
FTP
FTPS
SFTP
SSH

AUTH SERVER

Service Processor

Service Processor

Admin

SSH
FTP
FTPS
SFTP
HTTP

HTTPS

Integrate Rather than Change

- Rather than add new processes/procedures, integrate into the existing process as much as possible.
  - Big Win: Adding a security section to the functional spec template
- Enhancing the bug tracking system to easily identify security bugs
  - Standardizing on CVS and CVSS also helps when communicating to others

Functional Spec Checklist

- Authentication
- Authorization
- Auditing
- Encryption usage
- New Network Traffic/Network Ports

Find the Like Minded

- With ≅ 1000 developers and too few security experts, finding allies and those with an interest was critical
  - Most developers want to do the right thing
  - Find Security Champions
  - Create an easy path for anyone to get help on a security issue
- Engage with QA

Security Assessment & Executive Sign Off

- Every release had to be assessed against a security checklist before going out the door
  - The list came from the company PSO
- Product executives had to sign off on any shortcomings
- This raised visibility and helped to get things addressed in a future release

What Didn’t Work Well

Resource Competition

- Security Work competing against Revenue Features
  - Getting on the short list is difficult
- Cost/Benefit analysis difficult because metrics are skewed
  - Cost of a bad choice isn't borne by the person making the choice
  - Customers generally don't leave because of a bug

Scaling

- Transition from the security gadfly stage to a more process-driven, well-integrated stage dragged out too long
- At first, just doing what seemed right and necessary and that kept things moving along
  - Effective, but <u>not</u> a mature process
- Would have been better to formalize program sooner, get recognition of requirements, resources, etc.

Slow to Introduce Threat Modeling

- I was slow to come around to threat modeling
  - Not on the idea of it, but….
- After years on the product, it would take a huge effort to produce a threat model that would tell me anything
- Forgot that I'm not the only consumer of the Threat Model
  - System Architects
  - Security Test Team

Responsibility: From Ad Hoc to First Class Process

Ad Hoc Security → Security Program

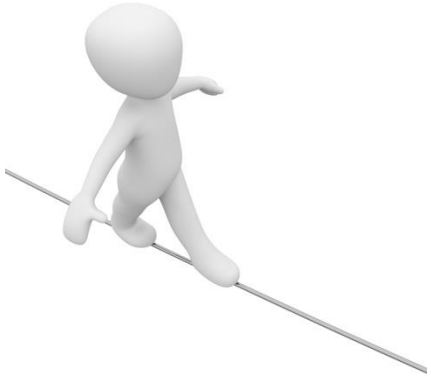
- Transitioning from ad hoc or feature-driven activities to fully integrated activities
  - Vulnerability Response (and attendant patching)
  - Scanning & Testing
  - Documentation & Customer Communications
  - Managing Security Bugs
  - Architecture & Code Review
  - Release Requirements

Responsibility & Authority

- Once you are the official and recognized “security person”, life changes
  - No longer the rare voice speaking out/up
  - Now you have to deal with real responsibility
- Recognizing security requirements vs. business cases
  - Authentication is a requirement
  - Common Criteria is a business decision
- No one buys a device just to have it be secure!

Shipping With Known Vulnerabilities

- You are going to ship with known vulnerabilities
- Security is always a balancing act
- Once you've established a real program:
  - You're the one to say if something has to be fixed or not
  - Objective criteria is essential



@TAD_TAYLOR

Not Shipping with Known Vulnerabilities

- Sometimes, you have to bring the security hammer down
- Compare the security issue to other types of serious, stop ship issues
- For storage, DU/DL is Really Bad™
  - Does the security issue potentially lead to a DU/DL?
- What would the news coverage look like?

Apply

- Find Your Allies
  - QA
  - Customers
  - Like-minded developers and Product Management

- Capture the Security Architecture & Do a Threat Model
  - You have to be able to show what needs attention and why
  - A Living Document

Apply

- Find the best integration touch points
  - Requirements, specs, code review, use cases
- Find and identify security champions
- Provide help and insight
  - Not obstructions
  - Be reachable, approachable
- You're all on the same team!

Some Resources

BSIMM

- Build Security in Maturity Model
- https://www.bsimm.com
- See what others are doing, both in general or in your vertical

SAFECode

- Software Assurance Forum for Excellence in Code (SAFECode)
- http://www.safecode.org
- Some helpful introductions and training

Books

- *Enterprise Software Security: A Confluence of Disciplines*
  - Kenneth van Wyk, Mark Graff, Dan Peters, Diana Burley
  - I like the non-adversarial stance taken
- *Software Security: Building Security In*
  - Gary McGraw
  - Helping to show the distinction between security features and software security

Questions?

Millard Taylor
@tad_taylor
tad_taylor@ieee.org