SPICON 2.0

SDLC Lessons Learned

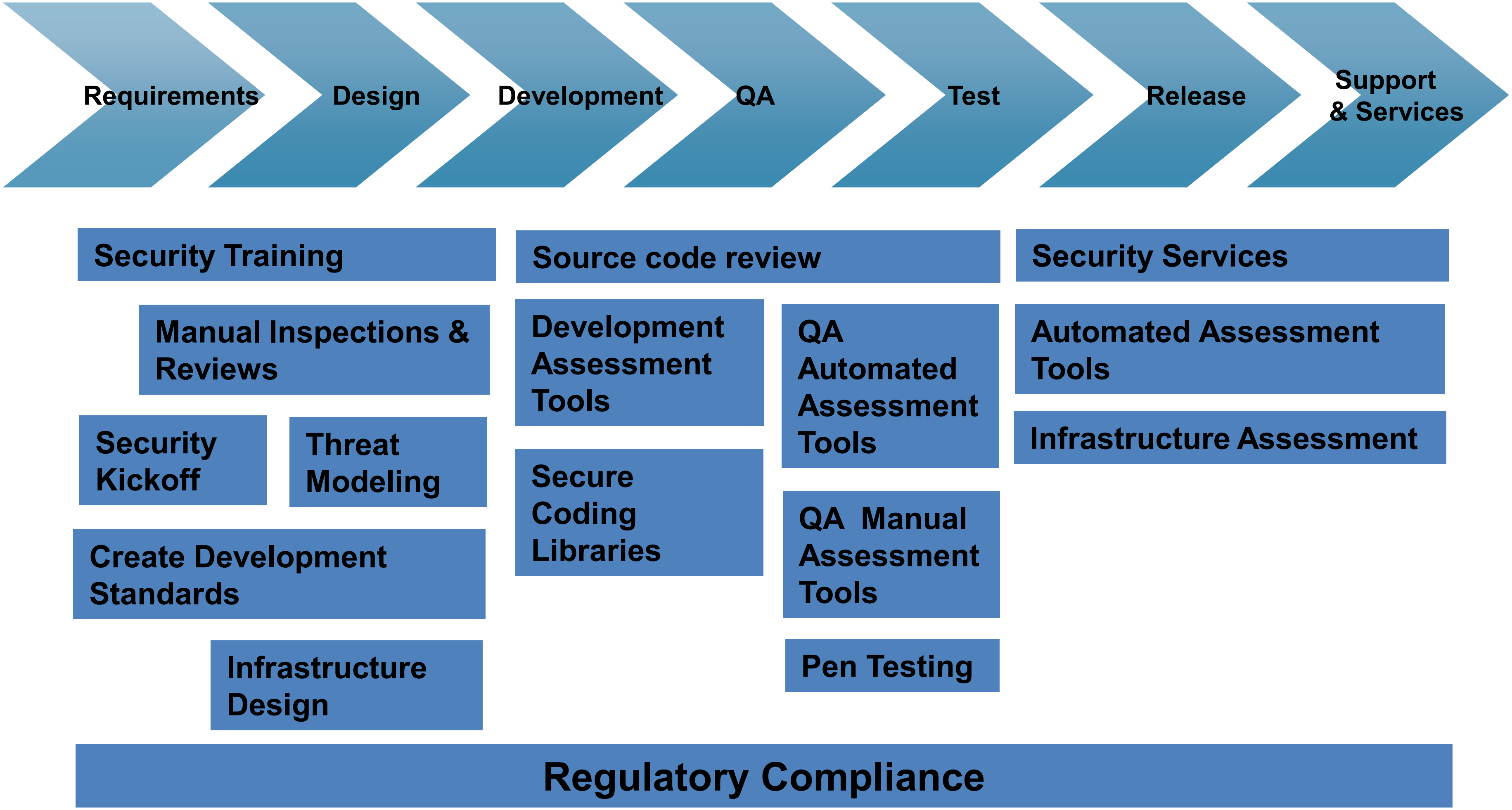
By Vincent Liu

STACH&LIU

Agenda

- Know what to look for…
- Understand what you find…
- And learn from the mistakes of others.

App Sec Assurance Program



Requirements

Design

Development

QA

Test

Release

Support & Services

Security Training

Manual Inspections & Reviews

Security Kickoff

Threat Modeling

Create Development Standards

Infrastructure Design

Source code review

Development Assessment Tools

Secure Coding Libraries

QA Automated Assessment Tools

QA Manual Assessment Tools

Pen Testing

Security Services

Automated Assessment Tools

Infrastructure Assessment

Regulatory Compliance

Get Executive Support.

1

- Not everyone cooperates.
- Takes time and money.
- **Establish an application security policy**.

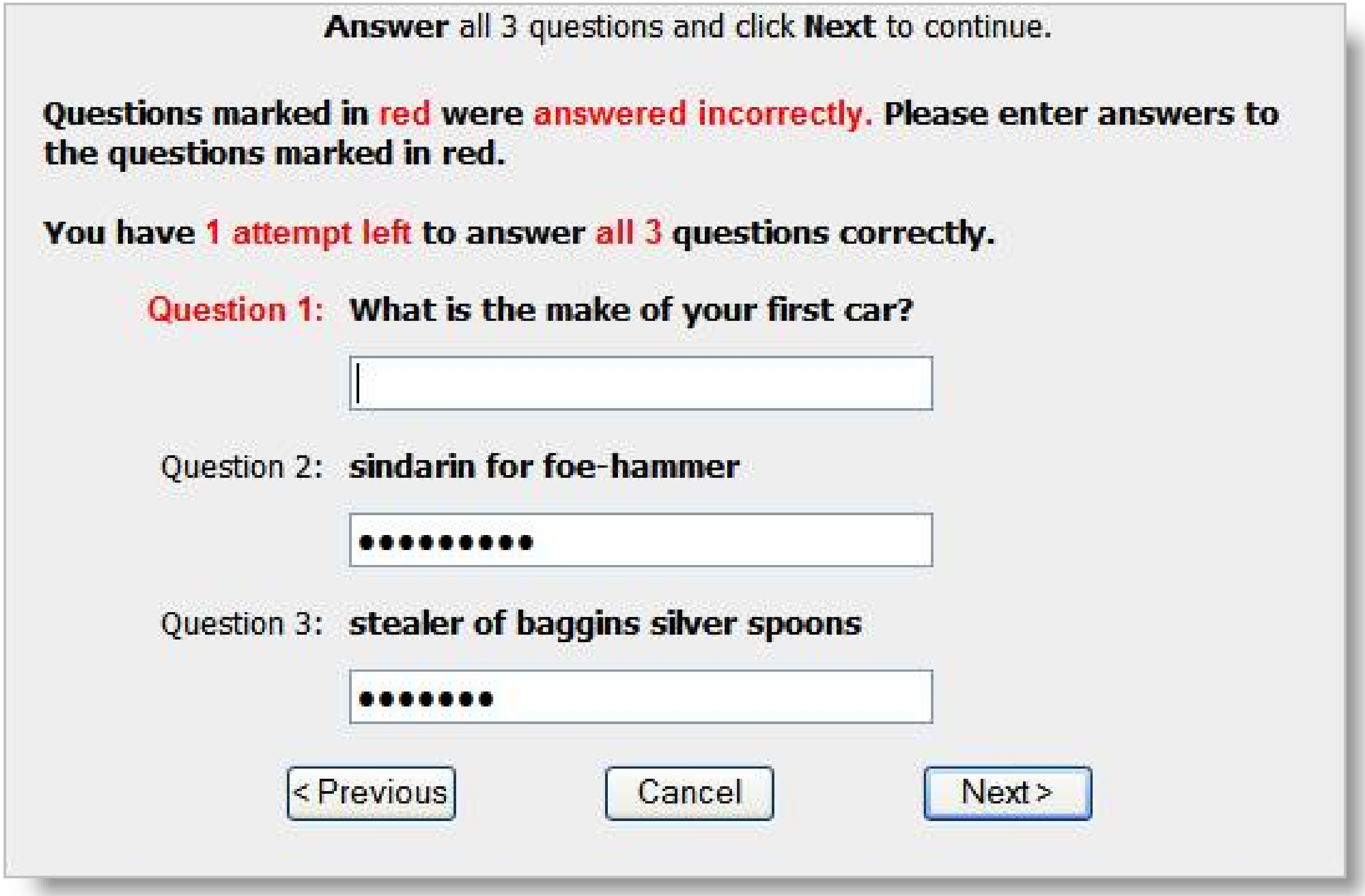
There’s No Silver Bullet.

2

- Get past the marketing.
- The 50 / 50 split.
- **Touch each stage of the application lifecycle.**

SPICON 2.0

Design Issues.



Answer all 3 questions and click **Next** to continue.

Questions marked in red were answered incorrectly. Please enter answers to the questions marked in red.

You have 1 attempt left to answer all 3 questions correctly.

Question 1: **What is the make of your first car?**

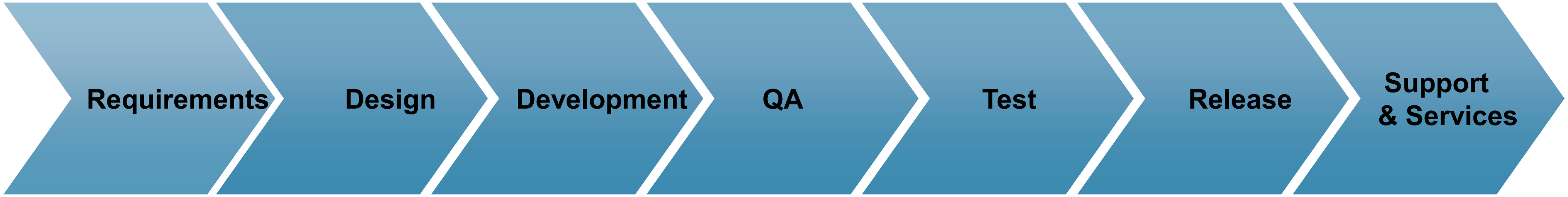
Question 2: **sindarin for foe-hammer**

●●●●●●●●●●

Question 3: **stealer of baggins silver spoons**

●●●●●●●●

Right Tools. Right Place. Right Time.



	Dynamic Analysis	Static Analysis	Expert Analysis
Directory Browsing	X		X
Insecure Function		X	X
Security Questions			X

Measure Twice, Cut Once.

3

- Avoid only doing the fun assessments.
- Money must be applied to more than assessments.
- **Don’t forget the boring work.**

One, Two, Three, Four…

4

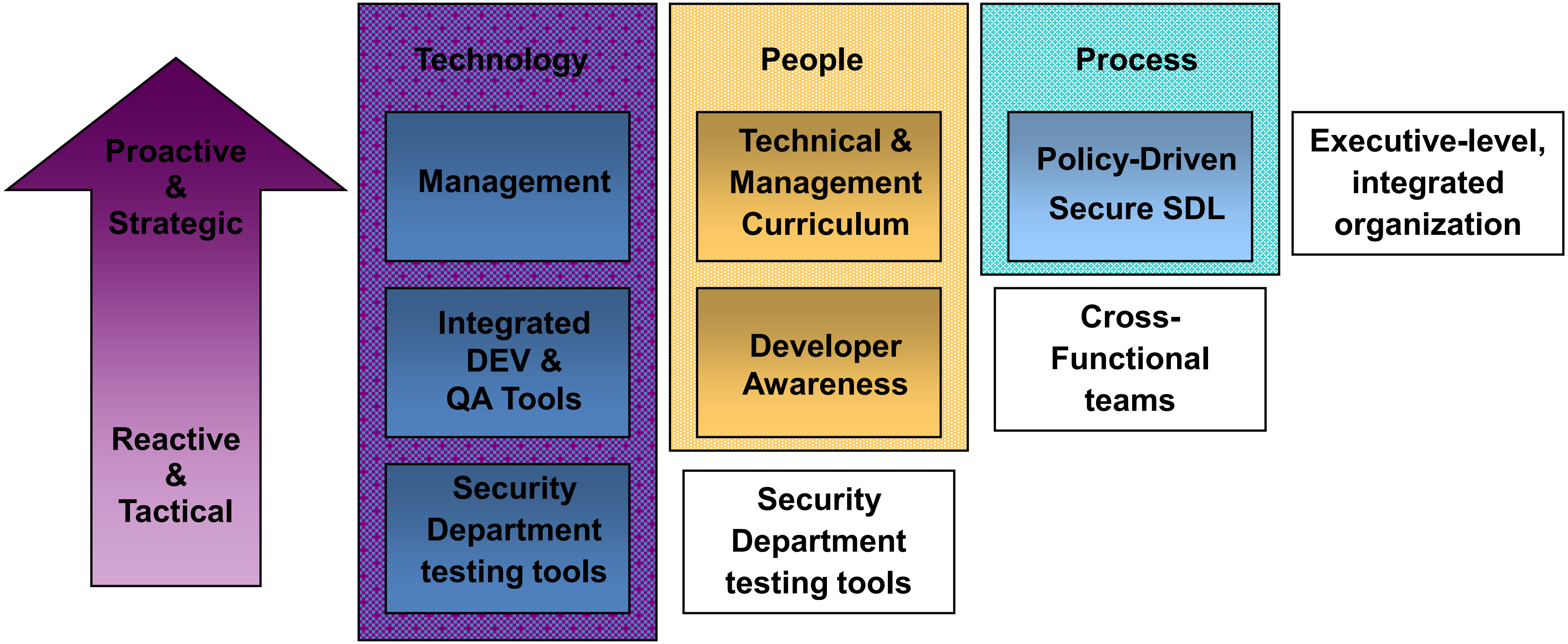
- Nobody pays to “feel” secure.
- Must measure to manage.
- **Establish a metrics model.**

The Best Laid Plans…

5

- Full-scale enterprise deployment is a fool’s quest.
- Too much you don’t know.
- **Pilot first, then evolve.**

ASAP Maturity Model



Proactive & Strategic

Reactive & Tactical

Technology

Management

Integrated DEV & QA Tools

Security Department testing tools

People

Technical & Management Curriculum

Developer Awareness

Security Department testing tools

Process

Policy-Driven Secure SDL

Cross-Functional teams

Executive-level, integrated organization

What does this imply?

“Software quality is cumulative because a number of bugs are acceptable, up to a point, and yet the software is still good enough to ship. Software security is absolute because a single vulnerability left in the application could be the one that ultimately wreaks havoc.”

-Fortify Software

Quality and Security in Software: Cumulative versus Absolute

Security is Not Absolute.

6

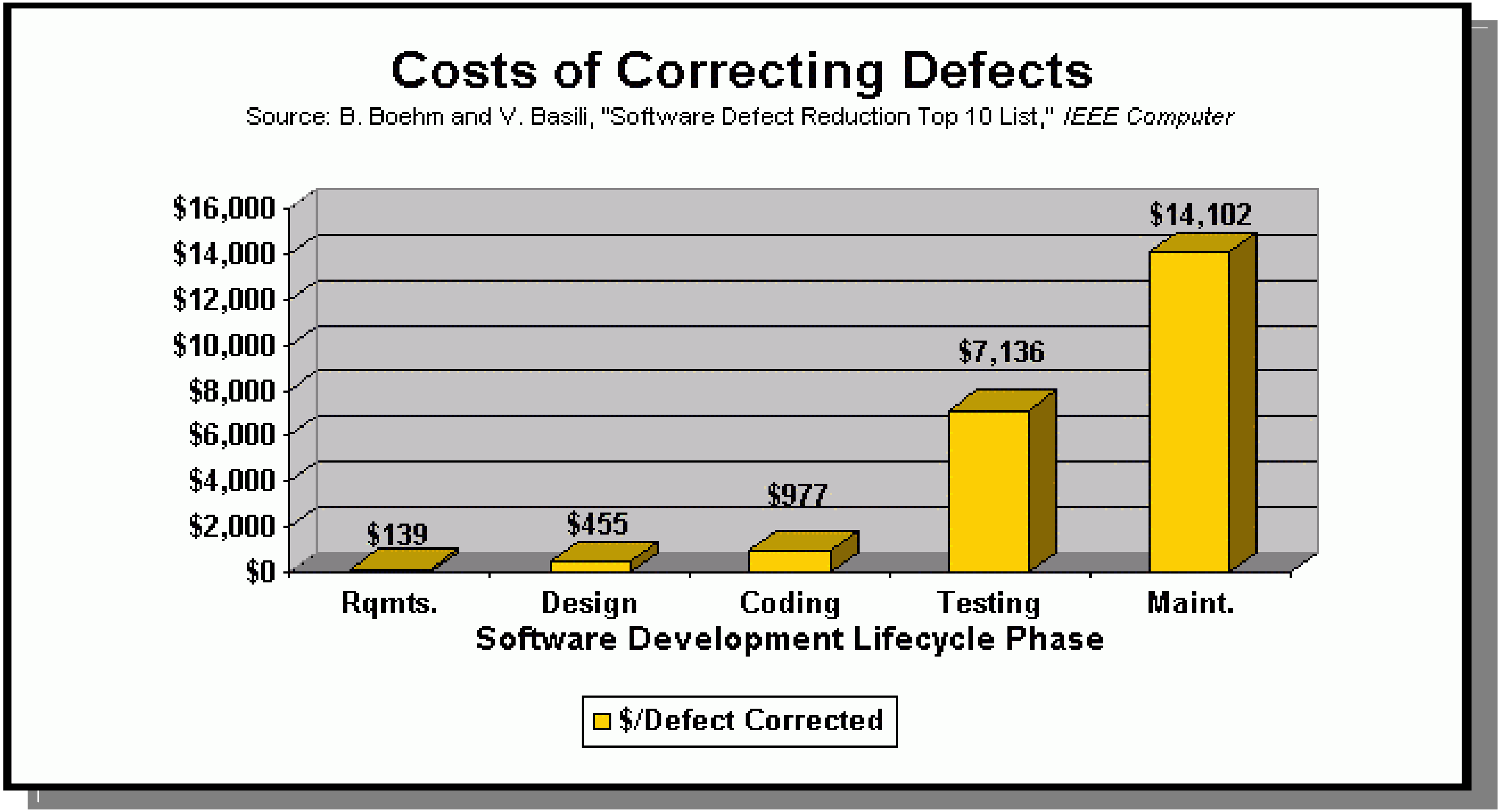
- You will *never identify* every vulnerability.
- You will *never fix* every identified vulnerability.
- **Application security is risk management.**

Penny Wise. Pound Foolish.

7

- Application security is expensive.
- There’s no magic island full of security experts.
- **Spend smarter.**

Be More Effective.



Costs of Correcting Defects

Source: B. Boehm and V. Basili, "Software Defect Reduction Top 10 List," *IEEE Computer*

Train Right. Eat Right.

8

- Don’t turn developers and QA into security experts.
- Security experts get paid more…somewhere else.
- **Train appropriately and provide support.**

Hmm.

“Debugging is at least twice as hard as writing the program in the first place. So if your code is as clever as you can possibly make it, then by definition you're not smart enough to debug it.”

-Brian Kernighan

Department of Computer Science, Princeton University

Get a Second & Third Opinion

9

- It’s difficult to debug your own code or design.
- Finding security bugs is even harder.
- **Get a different perspective.**

Man and Machine

	Cost	Speed	Quality
The Expert	High	Ok	Best
The Tools	Low	Fast	Good

One Step Forward, Two Steps Back.

10

- Introduce operational risk through cost cutting and off-shoring.
- Exposing yourself to a new threat.
- **You get what you pay for.**

The Top 10.

1. Establish an application security policy.
2. Touch each stage of the application lifecycle.
3. Don’t forget the boring work.
4. Establish a metrics model.
5. Pilot first, then evolve.
6. Application security is risk management.
7. Spend smarter.
8. Train appropriately and provide support.
9. Get a different perspective.
10. You get what you pay for.

Thank you for your time.

Questions?

STACH&LIU