RSAConference2020
Asia Pacific & Japan
A Virtual Learning Experience | 15–17 July

HUMAN ELEMENT

SESSION ID: SPS-W03V

The AppSec Error Loop

How We Can Break the Cycle and Stop Making the Same Mistakes

Pieter Danhieux

CEO and Co-Founder, Secure Code Warrior

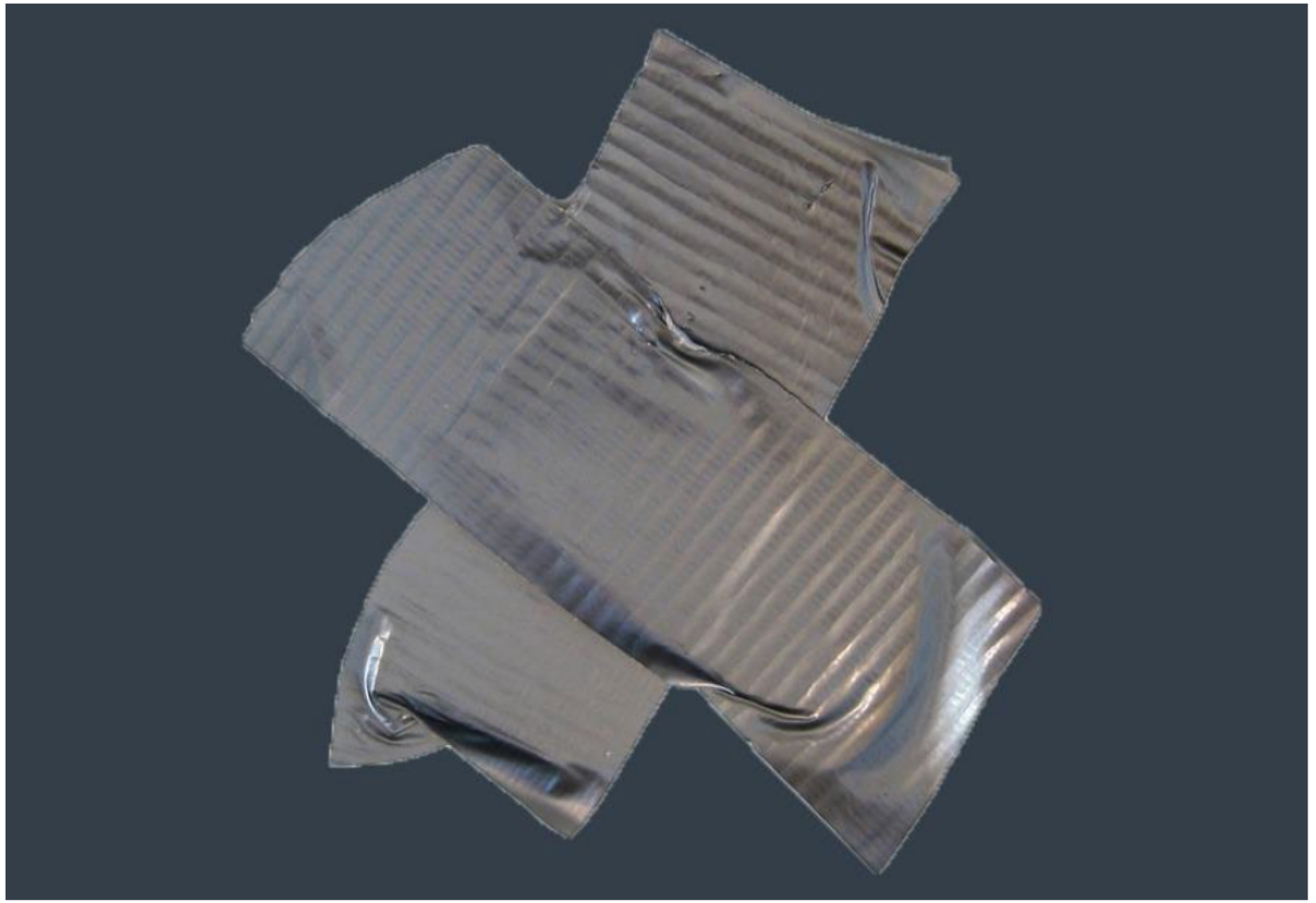
Another day, another data breach

2020 (So far)

- EasyJet - 9M customer records, including 2208 complete credit card numbe
  cause under investigation
- Nintendo - 300,000 customer records stolen by **exploiting a legacy login system**
- Marriott (again) - 5.8M customer records stolen due to **misconfiguration**
- ??? - An unprotected Google Cloud server had 200M US residents' **se**
  exposed for one month
- Microsoft - 250M customer service records stolen that were **stored in plain**
  due to **misconfiguration**
- Virgin Media - 900,000 customer records stolen from an **unsecured d base**

… and many more

We've used duct tape for too long.



RSAConference2020 APJ

A Virtual Learning Experience

>>> AppSec by the numbers

24M

Software developers around the world ~

EvansData

Source: https://evansdata.com/press/viewRelease.php?pressID=278

111BN

Lines of code written by developers every year ~ CSO Online

Source: https://www.csoonline.com/article/3151003/application-development/world-will-need-to-secure-111-billion-lines-of-new-software-code-in-2017.html

1 to 4

Exploitable Security Bugs in every 50,000 Lines of Code

SECURE CODE WARRIOR

Source: StackOverflow

90%

Security incidents result from defects in the design or code ~ DHS

Source: https://www.us-cert.gov/sites/default/files/publications/infosheet_SoftwareAssurance.pdf

22%

Of data breaches caused by security misconfiguration ~ Verizon

Source: Verizon, Data Breach Investigations Report, 2020

20%

Of data breaches exploit known web application vulnerabilities ~ Verizon

Source: Verizon, Data Breach Report, 2020

RSAConference2020 APJ

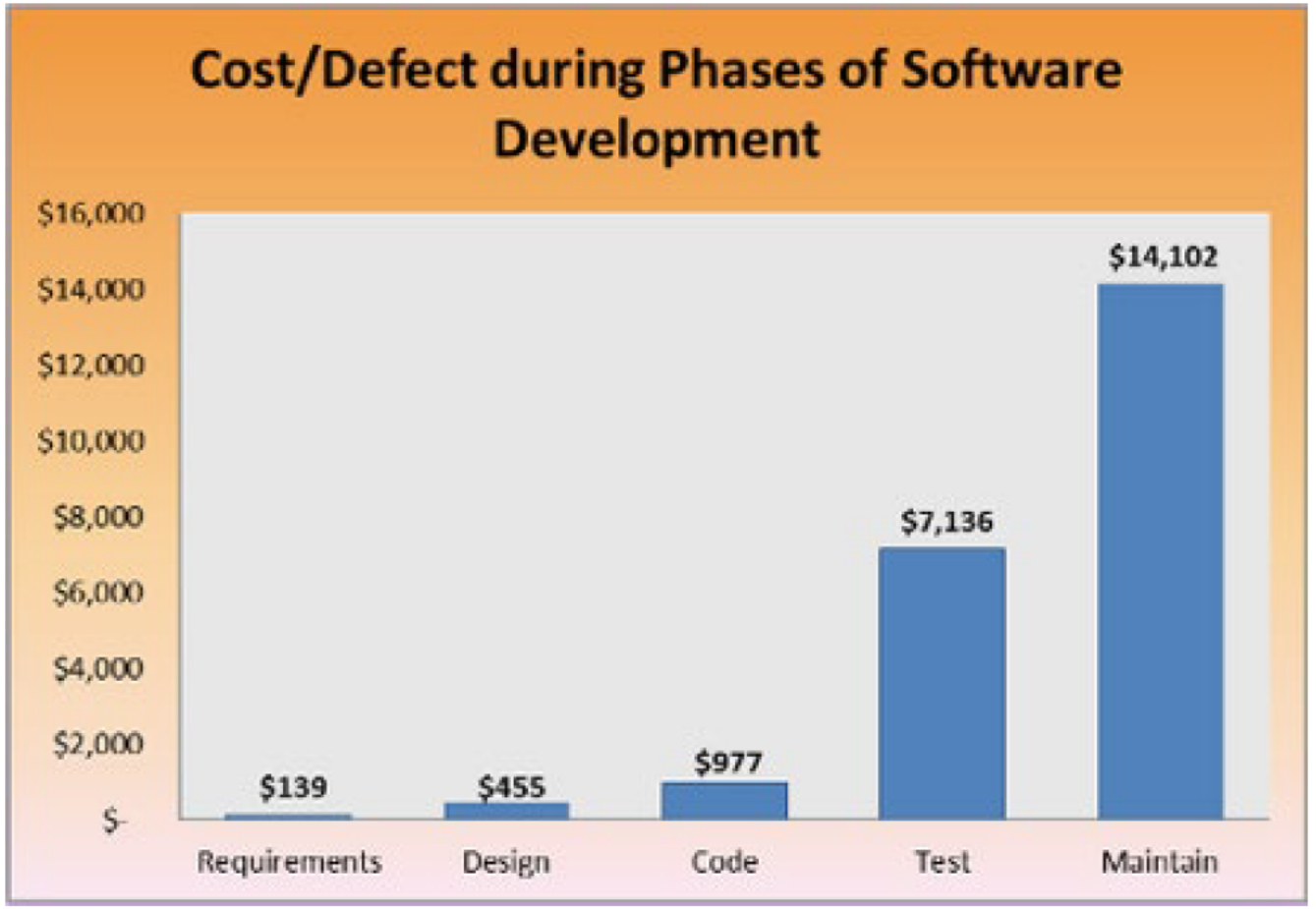
A Virtual Learning Experience

>>> How did we end up here?

Building code is like building a house.

This is how we used to build houses before 2000 years of engineering experience...

Discovering a late flaw in your construction is costly (and can be downright **dangerous**)



Actual data

We’re doing it wrong.

- Society’s demand for software is growing faster than **security knowledge distribution** and skills
- There is no emphasis on building a **culture of security awareness**
- Many vulnerabilities (and their remedies) have existed for decades, but coders are **still introducing them into modern software**.

SECURE CODE WARRIOR

AppSec in 2020

Everything runs on software. Cybersecurity & AppSec are hot topics.

> SAST is still here…

> Runtime Application Security Protection (RASP)

> Dynamic Application Security Testing (DAST)

> Interactive Application Security Testing (IAST)

> Crowd-Sourced Security Testing

> **DevSecOps**
- Containerisation
- Integrating security and ops into dev
- Security pipelining

> **SHIFT LEFT**

SECURE CODE WARRIOR

AppSec in 2020

Challenge - Tools mostly suck

> SAST - Expertise, false positives, slow, framework support

> I/DAST - Expertise, false negatives, slo

> RASP - WAF++, nobody uses block mode, tech specific

> Testing tools spit out long, mostly inac e reports with often useless advice

One tool to rule them all?

No such thing.

- No singular tool can scan for every vulnerability, in every situation, and they are **s l o w**
- Stopping common vulnerabilities before they're committed greatly reduces reliance on tools
- Failing to balance tools and people (i.e. investing in specialists, plus training developers) is a rod for your own back.

AppSec in 2020

Challenge - Pen-testing mostly s[illegible]cks

Develop[illegible]

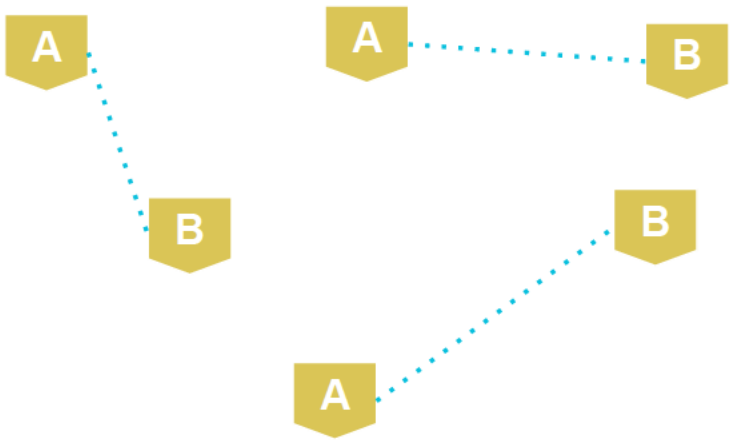
The AppSec/Dev “Language Barrier” is real.

- JAVA Spring
- Constructors
- SWIFT
- Angular.JS

- SQL Injections
- Object Deserialization
- XSS
- IDOR

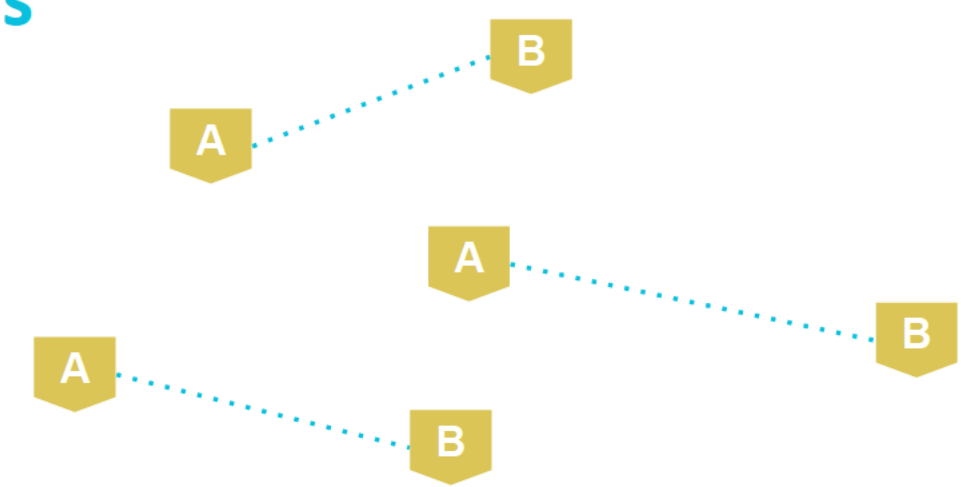
Challenge - AppSec is often a bott

AppSec in 2020



Software Developers (Agile)

200



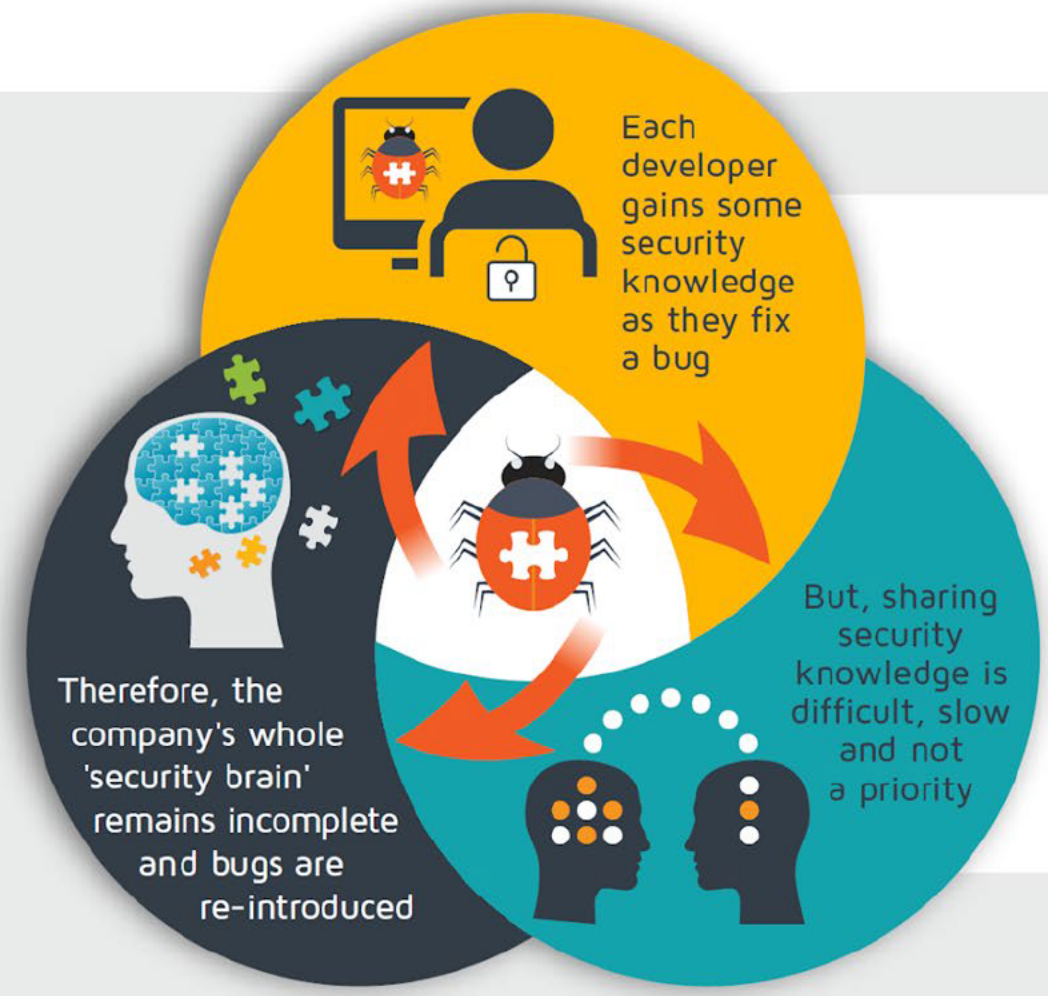
Application Security Experts

1

We will never have enough AppSec specialists to deal with the amount of code being produced.

- Stop trying to win an impossible game
- Security-aware developers = your secret security weapon
- Ignite their passion for problem-solving with hands-on training
- Impart a sense of responsibility

We're Failing in Learning from Our Mistakes



- 125+ frequently occurring vulnerability types
- Same vulnerabilities continually re-introduced
- New vulnerabilities also introduced into code
- Today's apps ripe with vulnerabilities

There is a pattern emerging...

Check out some of the data we analysed from 1.2 MILLION challenges played on the Secure Code Warrior platform

- from 400 enterprise organizations
- mainly in Java, C#, Python, NodeJS and Ruby

> SQL Injection was correct 58% of the time **... that's 42% had it WRONG**

> Identifying usage of vulnerable components was correct 64% of the time **... that's 1 in 3 WRONG**

> Poor authorization (IDOR, or Insecure Direct Object Reference) was **incorrect 79%** of the time

> CSRF was **incorrect 63%** of the time

AppSec @ Work

This is what everyone thinks AppSec is doing

AppSec @ Work

This is what AppSec does

RSAConference2020 APJ

A Virtual Learning Experience

~~SHIFT~~ START left

Scale and Make an Impact as an AppSec Pro

The “blue skies” remedies

AppSec in a perfect world looks like:

- A positive security culture that is lead from the top of the organization
- Security as a shared responsibility from the start of the SDLC; everyone knows and performs their role in software secu best practice
- Developers are given the **time** and **training** to code securely and take pressure off security specialists.

Negative focus = negative results.

> Demand better pen-testing!

> Bobby'; DROP TABLE pentesting_attitude;

> Provide a FIX more than input_validation();

> Create a JIRA ticket with advice/fix

> Create a pull request (wishful thinking)

> Lessons learned by dev teams should enter a knowledge distribution cycle

Less finding problems, more security engineering.

The right type of training

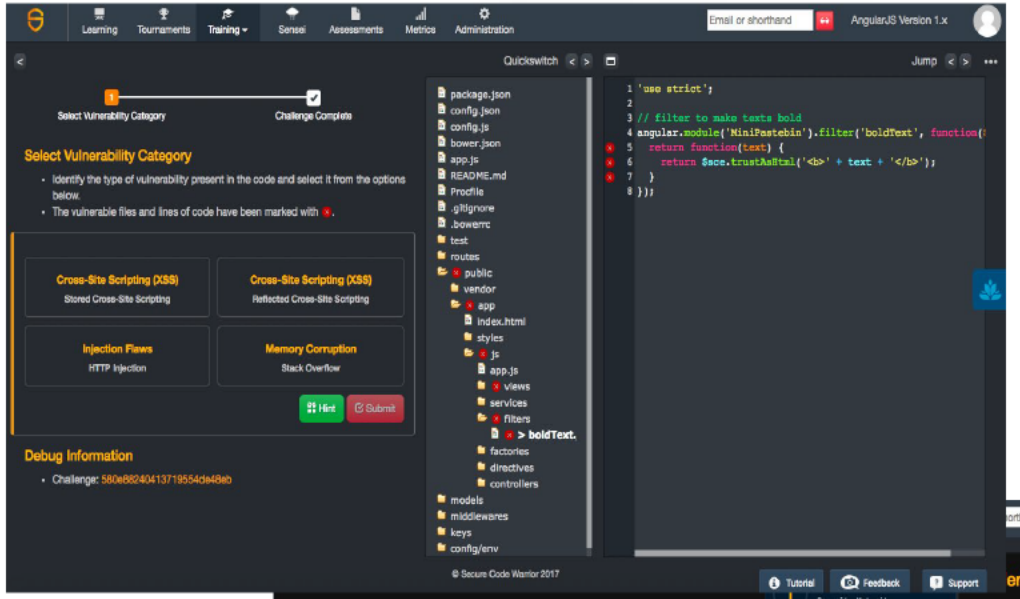
EMPOWER DEVELOPERS WITH THE KNOWLEDGE TO SUCCEED IN DAY-TO-DAY WORK

> Hands-on, bite-sized and contextual

> Language/framework specific (yes, even COBOL!)

> Incentive-based, with assessable outcomes

> **Doesn't bore everyone senseless**



SECURE CODE WARRIOR

In the relevant languages and frameworks

FOCUS on the TOP 3 weaknesses. Don't boil the ocean - your developers need to be security-aware, not security experts.

Weaknesses vs Controls

OWASP

OWASP Top 10 - 2017

The Ten Most Critical Web Application Security Risks

OWASP Open Web Application Security Project

OWASP PROActive CONTROLS

FOR DEVELOPERS

2018 **v 3.0**

10 Critical Security Areas That Software Developers Must Be Aware Of

PROJECT LEADERS

KATY ANTON
JIM MANICO
JIM BIRD

OWASP Open Web Application Security Project

Application Security Verification Standard 3.0

October 2015

Distribute Knowledge

Secure Coding Guidelines

1. *Ensure application logging (Where, What, When, Who, Why)*
2. *Use context encoding on untrusted user input*

Project X - Secure Coding rules for *<insert your favourite coding framework>*

1. Use SecureLogger log_object;
2. Don't use GetParameter(), Use LibSafe_GetParam()

Upon Commit

1. Your code violates security rules: You shall not pass!
2. Your code violates security rules: Fill in your get out of jail card (JIRA ticket)
3. Points++ for delivering secure code

Atlassian JIRA

Visual Studio

GitHub

IJ

eclipse

Culture of Security Awareness

With the right tools, training and support, security becomes part of an organization's DNA.

> Developers remain agile

> A higher, safer standard of software security is possible

> AppSec stops losing their hair

It's faster to achieve than you think.

Security in the DNA @ DBS

A deep focus on developer engagement and organization-wide security awareness.

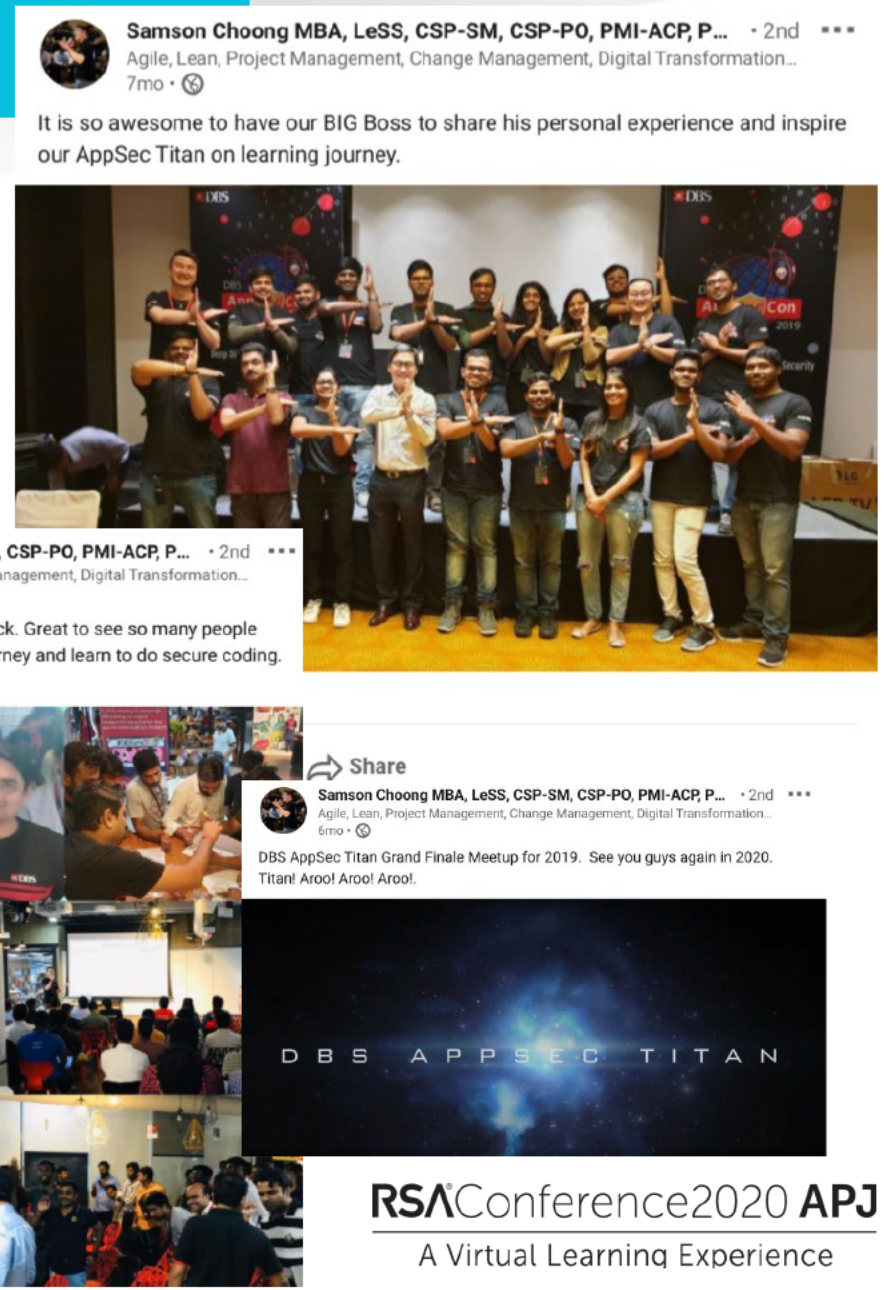
> Training to build a security-conscious culture

> **STRONG** support from management

> Belting/grading program targeting key SAST vulnerabilities

> AppSec Titan Program tailored to the most common security weaknesses

> "No developer left behind" policy - every developer must be security-certified.



SECURE CODE WARRIOR

RSAConference2020 APJ
A Virtual Learning Experience

Let's apply this knowledge.

- Next week you should:
  - Speak to your developers and ask them what you can do to help them write secure code
  - Ask them how effective (or non-existent) their knowledge is around security
  - Ask them if they know what tools exist to support them
  - Find your security champions/coaches; they have **active interest** and **people skills**
- In the first three months following this presentation you should:
  - Identify your **TOP 3 weaknesses** + main coding languages, **and** provide **sample fixes in those languages** to your developers
  - Consider testing potential new hires: Do they know about this top 3?
  - Do the above by working with your security coaches within the development groups
- Within six months you should:
  - Measure the impact of Top 3 training on new code written
  - **Celebrate**. You just saved everyone a lot of frustration and pain.