



build | integrate | secure

# The Permanent Campaign – Driving Software Security Initiatives in the Enterprise

John B. Dickson, CISSP

## Personal Background

- *15-year information security consultant background*
- *Principal at Denim Group*
- *Ex-Air Force security analyst at AFCERT*
- *Trident Data Systems, KPMG, SecureLogix, and Denim Group information security consultant*
- *Works with CIO's and CSO's to build successful software security initiatives*
- *CISSP since 1998*

## Denim Group Background

- *Professional services firm that builds & secures enterprise applications*
- *Secure development services:*
  - Secure .NET and Java application development
  - Post-assessment remediation
  - Secure web services
- *Application security services include:*
  - External application assessment
  - Code review
  - Classroom and CBT instruction for developers
  - Software development lifecycle development (SDLC) consulting

## Presentation Overview

- Background – the Current State of Affairs
- Characterizing the Landscape
- Securing Champions
- Defining Strategy & Initial Standards
- Executing the Strategy
- Sustaining the Initiative

## Background – the Current State of Affairs

- Creating meaningful enterprise-wide software security initiatives is hard
  - *Organizational culture or politics can present more daunting challenges than anything in the technical world*
- The vast majority of info software security focuses on means to write more secure code or strategies for putting controls around the software development process
  - *Very little is written about how one goes about “winning hearts and minds”*
  - *The body of literature assumes corporate buy-in*
- Building custom software on time, on budget, and without bugs is difficult
  - *Building secure software on time, on budget, and without bugs is even harder!*

## Background – the Current State of Affairs

- Large and complex organizations have various groups and cultures – this complicates things...
  - *They have groups with cultures that span the spectrum of openness*
  - *Mergers and acquisitions and geographic distribution increase this diversity*
- Large organizations are also likely to have a variety of software development approaches that mirror their organizational diversity
  - *Waterfall, XP, MSF, Agile.... Errr... Fragile*
- Understand how something called “status quo bias” does not help
  - *The argument of “it hasn’t happened to us” is particularly compelling to C-level executives and senior leaders*
  - *Couple this with the current “do more with less” business strategy in the worst recession of our lives – you get the picture*

## So Given These Realities?

- How does one go about leading the effort to build secure software within large and complex organizations?
- How does one reengineer one of the most complex, tedious set of process that an organization possesses?

## Characterizing the Landscape

- Five Critical Components
  - *Compliance framework*
  - *Cultural norms*
  - *SDLCs in place*
  - *Artifacts of software security*
  - *ID Gap between policy and practice*

## Characterizing the Landscape – The Compliance Framework

- Step 1: Understand the organizational compliance framework
- Key Questions:
  - *What are the known regulations or directives that will exert influence on how security should be applied to software?*
  - *Who is responsible for monitoring evolving compliance frameworks?*
  - *Who audits these regulations, both inside the organization and externally?*
  - *How can one use regulations or directives for business justification for a software security initiative?*
  - *Does a relationship with the internal audit manager benefit the leader of the software security initiative?*

## Characterizing the Landscape – Cultural Norms

- Step 2: Analyze and understand the culture of different groups building software in your organization
- Key Questions:
  - *Why and for whom are they building software?*
  - *What are the terms that would best describe each software team?*
  - *How would you characterize their leadership environment?*
    - Is it top-down or collaborative?
  - *How do requirements flow to the software development groups?*
  - *How receptive are they to new ideas?*

## Characterizing the Landscape – SDLC's in place

- Step 3: Catalog and understand the different way software is built by teams within your organization
- Key Questions:
  - *What formal or informal methodology do the different groups use?*
  - *How different are they within each division or operating location?*
  - *Why are their approaches different and what projects/products are they working on that influence the choice of development approach?*
  - *How long have the different approaches been in place?*

## Characterizing the Landscape – Artifacts of software security

- Step 4: Identify and capture what is already being done in the way of application security
- Key Questions:
  - *Who has done what to this point?*
  - *Why haven't they been more successful?*
  - *Are external penetration software tests being conducted in others?*
  - *Are policies in place regarding software and data security?*
  - *Have any of the groups bought tools or put open source tools in place?*

## Characterizing the Landscape – ID Gap between policy and practice

- Step 5: Quantify the gap between what is policy and what is happening in practice with software security
- Key Questions:
  - *Is there a gap between what management portrays and is in place in the development teams?*
  - *Is it a recurring approach?*
  - *Is it consistent across development teams?*
  - *Are they aware of benchmarking tools like OpenSAMM?*

## Securing Champions

- Understand and explicitly focus on winning over champions to build a successful software security initiative
- Three key stakeholders should be brought into the fold:
  - *CIO and one of his/her peers*
  - *A security evangelist in each development team*
  - *Internal audit, if relevant...*
- How does one secure champions?
  - *Lunch and learns*
  - *Attack demonstrations*
  - *Audit opinions*
  - *Telling relevant stories*

## Defining Strategy & Initial Standards

- Conduct a “lightweight” risk assessment of applications to ID the most vulnerable applications for qualitative ranking for decision-making
- Conduct a brief risk analysis and rank order the applications from most critical to least critical
- Pick on application and conduct a security assessment

## Defining Strategy & Initial Standards

- Set reasonable goals and to demonstrate future progress
- Examples of modest goals
  - *Conduct a ½-day classroom overview of application security concepts*
  - *Publish a “Top 10” list of coding standards that reflect fundamental*
  - *Remediate one of the applications you assessed earlier in the process.*
  - *Implement threat modeling at the early stages of a handful of big projects*
  - *Improve one facet of the SDLC to inject a software security “control”*
- Great resources from which to pull:
  - *OpenSAMM*
  - *OWASP Testing Guide*
  - *NIST Publications*

## Executing the Strategy

- Focus: Grab attention of development managers and keep their interest
- Put champions to work!
  - *ID substantive tasks for them to undertake*
  - *Keep them briefed on the software security initiative*
  - *Consider a team wiki*
- Tailor the delivery to the audience
  - *Monitor success*
  - *Adapt and adjust as you see fit*
- Measure goal attainment
  - *The old cliché “what is measured matters” is 100% relevant*
  - *Communicate success to the champions, teams, and management*

## Sustaining the Initiative

- Bottom line: Show quick wins, highlight positive behaviors, and do it over and over again, ratcheting up expectations and software security with each iteration
- Increase the depth and breadth of activities across the different development teams
- Use benchmarks such as OpenSAMM to chart progress
- Regular, disciplined update of the regulatory framework must occur
- Monitor the market: keep an ear to the ground in the commercial tools arena, as rapid innovation is changing the game
- Observe what's occurring within your organization to determine whether there are new risk areas emerging

## Conclusions

- Changing the way organizations build software is difficult. Crafting a software security initiative has to be methodical and tailored to the organization
- Organizational and cultural hurdles are likely to be more challenging than technical – if one understands how to address them, one can have an impact
- Implement a sustained effort that highlights wins, publicly praises positive behaviors, and do it over and over again, ratcheting up expectations and software security with each iteration

## Contact Information

- John B. Dickson, CISSP
  - [john@denimgroup.com](mailto:john@denimgroup.com)
  - *Twitter @johnbdickson*

[www.denimgroup.com](http://www.denimgroup.com)