Many attacks happen on the application layer where infrastructure security solutions are not effective.
Application Security is important, however, still rare in application development. Let's change that !

Three useful notions for proactive Application Security !

「Shift Left」

「Security by Design」

「Continuous Hardening」

A security story so that「bottom-up」meets「top-down」

Shift Left
Mindset for increasing efficiency and controlling cost during the whole application lifecycle.

「Shift Left !」

Find and fix defects early for reducing cost and risk
Handle it like Product Quality or Safety !

Top-Down Support
Efficient Application Security is not feasible without strong support from top management!

Application Lifecycle

Application Lifecycle

Dev | System Design — User Requirements | Coding & Testing — Mindset, Culture

Release | Run IT — Production, IT Operations | End of Release | End of Support & Updates Security Patches | EoL — End of Life might not be End of Use!

Shift Left ! Shift Left ! Shift Left !

Years ahead

Time when finding a defect: .....

Damage and cost for fixing it: ..... + $ ..... + $ $ $ ..... + $ $ $ $ ..... + $ $ $ $ $

Rule of Thumb
The later it gets, the more expensive it is to change an application (getting significantly costlier!)

Top Management Involvement Required
Keep an eye on the realities in your organization regarding,「Shift Left」,「Security by Design」and「Continuous Hardening」!

Years . . . . . Months . . . . . ← Delivery Time — The Universe of Development Methods — Delivery Time → . . . . . Weeks . . . . . Days . . . . . Hours

Traditional Dev. Methods

Waterfall approach at one extreme end in the universe of development methods.

Security? Leave it to IT...

Business — Business department, customer, sponsor role, product owner, etc.

Project Start → Requirements Analysis → Functional Specification → High Level Design (Architecture) → Detail Design (Components) → Coding (Dev.) → Unit Test → Integration Test → System Test → Acceptance Test (Testing/ Staging) → End of Project → Release

Ops

Security? We just provide platform and the security basics. Do you use them? actually

Production Environment — internal/external and/or cloud...

Multiple Test Env.

Development Env.

Security? Leave it to IT Ops...

V-Model (similar to Waterfall)

Dev and/or IT Ops → internal IT department or external IT service provider etc.

Tragedy in the IT Trenches

Still an often seen reality!

Agile Dev. & DevOps

Agile Dev. & DevOps combo at the other extreme end in the method universe.

Security? Leave it to IT...

Business — Product owner (from business department, customer, sponsor, etc.)

Wow! It's really fast now!

Agile / DevOps

Dev. Sprint | Dev. Sprint | Dev. Sprint | Dev. Sprint | Dev. Sprint

Dev — Dev. Environment

Security? Leave it to IT Ops...

Testing/ Staging: Autom. Testing | Autom. Testing | Autom. Testing | Autom. Testing

DB (Continuous Build) | CI (Continuous Integration) | CD (Continuous Delivery/Deploy)

Multiple Test Env. — Pipeline

Ops

Release

Production Environment — internal/external and/or cloud...

Security? We just provide platform and the security basics. Do you use them? actually

Dev and/or IT Ops → internal IT department or ext...

Add「Security by Design」&「Continuous Hardening」!

Security: Does it work for you?

Require to Include Security Requirements!

- Analyze Threats
- Create Security Requirements
- Create Security Specification
- Design Security Architecture
- Design Security Details

「Security by Design」

Yes!

Business — Business department, customer, sponsor role, product owner, etc.

Project Start → Requirements Analysis → Functional Specification → High Level Design (Architecture) → Detail Design (Components) → Coding (Dev. Env.) → Unit Test → Integration Test → System Test → Acceptance Test (Testing/ Staging) → End of Project → Release

Ops

Yes!

Production Environment — internal/external and/or cloud...

Multiple Test Env.

- Secure Coding
- SAST ・ Reviews
- Assisted Coding
- Platforms
- APIs/Security as Code
- DevSecOps WAF
- IAST
- RASP
- Pen Testing
- DAST
- Security Patching
- Conventional
- "Ops only" WAF

Support for「Continuous Hardening」

Better Approach

1. Business requires a proactive approach and provides budget.
2. Development project creates a security design. Suggested starting point: Zero Trust.
3. Development and other teams work together for efficiently
   a) implementing Continuous Hardening.
   b) maintaining the intended security level afterwards in order to stay consistent over time.

Security: Does it work for you?

Require to Include Security Requirements!

- Analyze Threats
- Create Security Requirements
- Create Security Specification
- Design Security Architecture
- Design Security Details

「Security by Design」

Yes!

Business — Product owner (from business department, customer, sponsor, etc.)

Yes!

Agile / DevOps

Dev. Sprint | Dev. Sprint | Dev. Sprint | Dev. Sprint | Dev. Sprint

Dev — Dev. Environment

Testing/ Staging: Autom. Testing | Autom. Testing | Autom. Testing | Autom. Testing

DB (Continuous Build) | CI (Continuous Integration) | CD (Continuous Delivery/Deploy)

Multiple Test Env. — Pipeline

Ops

Release

Production Environment — internal/external and/or cloud...

- Secure Coding
- SAST ・ Reviews
- Assisted Coding
- Platforms
- APIs/Security as Code
- DevSecOps WAF
- IAST
- RASP
- Pen Testing
- DAST
- Security Patching
- Conventional
- "Ops only" WAF

「DevSecOps」support for「Continuous Hardening」

ACROSEC 株式会社 アクロセック