I don't speak on behalf of my employer.

The information and perspective that I present are personal and don't represent those of my employer.

This presentation is the result of my personal researches and experimentation.
CHAPTER 1

THE REVELATION
BOB

- Works for Cash Register Unlimited Company (since 2003)
- In charge of Application Security
- 70 projects per year
- 200 applications
SOFTWARE DEVELOPMENT LIFE CYCLE

- Requirements
- Design
- Code
- Test
- Deploy
EVERYTHING ALL RIGHT!
- **Alice**: Hi Bob! Are there security issues regarding email sending?

- **Bob**: Maybe, what is data?

- **Alice**: For instance, banking data (PAN, IBAN, ...)

- **Bob**: In this case, I must analyse your project. What is the deadline?

- **Alice**: This is already in production 2 weeks ago!

- **Bob**: Oh! How is it possible? without security validation? Without security acceptance testing?

- **Alice**: Well ... we use Agile Methodology!
AGILE ???

- Manifesto for Agile Software Development (February 2001)
- Scrum / Kanban
- Cash Register Unlimited Company has implemented Agile SDLC since 2013
PRODUCT THINKING

No Project

No Application
PO is the only person responsible for managing the Product Backlog.

PO have a lot of stakeholders to take into account.

Security Officer should become a major stakeholder.

Alice is the Product Owner of "Cash Register 2.0"
PRODUCT BACKLOG

• It is a prioritized inventory of work to be done.

• Type of Product Backlog Item (PBI):
  ▪ Features (User Stories)
  ▪ Non-Functional Requirement
  ▪ Defects (Bug Stories)
  ▪ Refactoring
  ▪ ...

Security Officer should include security topics in the Product Backlog.
USER STORIES

Security features:

As seller, I want to change my password on the Cash Register

Acceptance Criteria:

The password is at least 8 characters. The password contains a character from each of the following groups: Lower case alphabet, Upper case alphabet, Numbers Special Characters (!,@,#,$,%,^,&,*
SECURITY-FOCUSED STORIES

• Approach introduced by Safe Code
• A way to include non-functional requirement in the backlog

Example: As developer, I want to verify that sensitive data is kept restricted to actors authorized to access it.
EVIL USER STORIES

aka "Abuser Stories"

- Approach introduced by OWASP
- Using Personas: Insider Hacker, Professional hackers, Script kiddie, ...

Example: As a hacker, I can modify the price of an article.
SECURITY IN PRODUCT BACKLOG

- User Stories with acceptance criteria
- Security-focused stories (NFR)
- Evil user Stories
DEFINITION OF DONE

List of activities to validate each item in the Product Backlog.

Security Officer should include security in DoD.

Example of secure activity: *There should be no open critical and high vulnerability identified by Source Code Analysis*
Security Officer should take part in sprint meeting.
MINIMAL PRODUCT

Minimal Viable Product (MVP)
Product which allows a team to test an idea with the least effort.

Minimal Marketable Product (MMP)
Product with the smallest possible feature set that addresses the needs of the initial users.

Security Officer should define the Minimal Viable Security (MVS) for the product.
CHAPTER 3
SECURITY IN SPRINT PHASES
SPRINT PHASES

1. Code
2. Test
3. Deploy
PHASE 1 # CODE
COWBOY CODING

- Prevent "cowboy" development:
  - Define allowed frameworks
  - Define security guideline for each framework
  - Change management

- Identify framework with known vulnerabilities:
  - Artifact repository: JFrog X-Ray, BlackDuck Hub, ...
  - Build: Dependency Check / RetireJS
MICROSERVICE

- Best agile software architecture
- 2 parts:
  - Front End (WebApp / MobileApp)
  - Back End (MicroService)
SECURING MICROSERVICE

• Front End (WebApp):
  ▪ Linter (ESlint Security)
  ▪ Minify and Obfuscate (UglifyJS)

• Back End (Microservice):
  ▪ Stateless & Autoscaling
  ▪ Authentication Token (OAuth / JWT)
  ▪ HTTPS
  ▪ Privileged Orchestratioin pattern to Choreography
MICROSERVICE
PHASE 2 # TEST
TEST-DRIVEN DEVELOPMENT

1. Start by writing an automated test case.
2. Run the test which should fail.
3. Write the minimum amount of code required to make the test pass.
4. Run the tests to check the new test passes.
5. Refactor the new code.

Positive testing (Valid data) and **Negative testing** (Invalid data)
BEHAVIOR-DRIVEN DEVELOPMENT

- Integration test
- Test are written with DSL (Domain-specific language) like Gherkin

Feature: Account Holder withdraws cash

Scenario: Account has sufficient funds
  Given the account balance is $100
  And the card is valid
  And the machine contains enough money
  When the Account Holder requests $20
  Then the ATM should dispense $20
  And the account balance should be $80
  And the card should be returned
COMMON TESTING TOOLS

- Fitness
- Mockito
- Cucumber
- Selenium
- JBehave (Java)
- Behat (PHP)
- Hiptest
SECURITY TESTING TOOLS

- ZAP
- Gauntlt (Be mean to your code and like it)
- BDD Security
PHASE 3 # DEPLOY
STRATEGY DEPLOYMENT

New environment for each deployment

"Blue/green" or Canary release
INFRASTRUCTURE AS CODE (IAC)

- Tools: Chef, Puppet, Ansible, ...

- Test Driven Infrastructure:
  - Linter: puppet-lint, Ansible lint, Foodcritic, RuboCop, ...
  - Unit testing: RSpec-Puppet, ChefSpec, ...
  - Acceptance testing: Beaker for puppet, Test kitchen for Chef, ...

- Network As Code (LaaS and FaaS): Neutron from RedHat
CONTAINER

- Software:
  - Container runtime: runC, Docker, Rocket, Garden, ...
  - PAAS: OpenShift, CloudFoundry, Bluemix, ...

- Secret storage: Vault from Hashicorp, Barbican from RedHat, ...
- Network Overlay & Micro Segmentation
- Segregate Containers by host (CoreOS)
- Container vulnerabilities Scanner (Clair)
CHAPTER 4
SECURE SOFTWARE SUPPLY CHAIN
SOFTWARE SUPPLY CHAIN

- Source Code Repository
- Framework Repository
- Linter Source Code Audit
- Build Repository
- Configuration Management
- Staging VM
- Automated Testing Tools
- Production VM
- Health Check
TOOLS

PERIODIC TABLE OF DEVOPS TOOLS (V2)
SOME BEST PRACTICES

For securing your Software Supply Chain

- HTTPS
- Authentication
- Access Management
ANY QUESTIONS?

View online at https://git.io/vNtqD