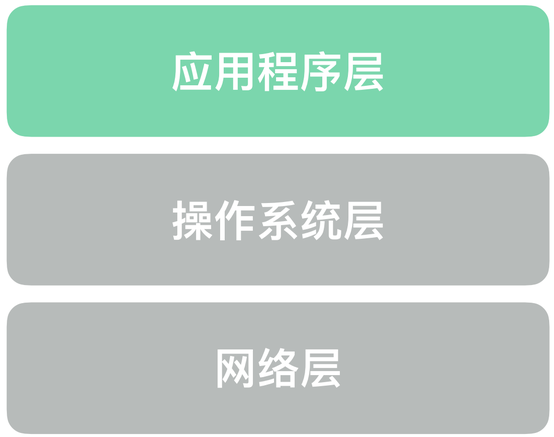
ThoughtWorks®

BQCONF CD

BSI 安全质量内建

马伟

当我们谈安全的时候，我们是在谈什么？



~ 80% 的安全攻击发生在应用程序层

World Quality Report 2015/16

这是一个真实的故事...

发现异常

订单总金额和实际产品的价格不一致

调查结果

某个影响订单总额计算的参数，在订单等待支付的时候，
可以被恶意篡改，从而导致金额不一致

我们的疑惑

为何防火墙没有拦截下攻击？也没有警告信息?

为何渗透测试没有报告这个问题?

原因:

- **漏洞和业务强相关**, 防火墙没有能力识别出这个安全问题，也就没有将其拦截下来
- **深深的隐藏在购买和支付流程中**, 渗透测试很难发现这个问题

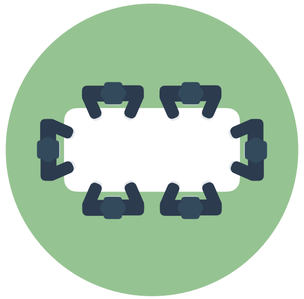
多么痛的领悟

防火墙

渗透测试 或者 漏洞扫描

严重的依赖于防火墙和渗透测试来给我们的应用提供安全保护，

然而它们都并非完美，有各自的优缺点。



反思回顾会议

反思回顾

#1 渗透测试总是在很晚的时候才进行，
团队在修复安全问题的时候面临着很大的交付压力

耗时长

报告漏洞数量多

修复时间有限

#2 传统安全活动有流于形式的趋势

- 我们做威胁建模，但是在产品上线前才做，原因是为了满足安全部门的要求
- 我们有一大堆安全规范，但是太过于冗长，没人真正清楚其中的细节
- 团队应该对自己开发的应用的安全质量负责，但实际上还是严重依赖外部帮助

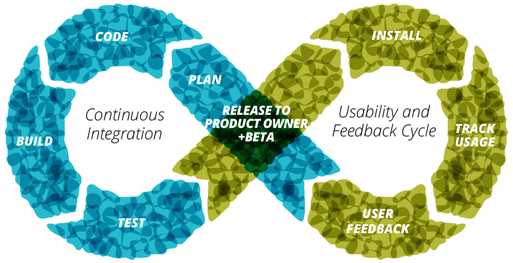
#3 检查第三方依赖是否包含已知安全漏洞相当耗时

```
dependencies {
    compile 'org.springframework.boot:spring-boot-starter-web'
    compile 'org.springframework.boot:spring-boot-starter-data-jpa'
    compile 'org.postgresql:postgresql:9.4.1212'
    compile 'org.apache.poi:poi:3.15'
    compile 'org.apache.poi:poi-ooxml:3.15'
    compile 'io.springfox:springfox-swagger2:2.6.1'
    compile 'io.springfox:springfox-swagger-ui:2.6.1'
    compile 'commons-io:commons-io:2.4'
    compile 'org.apache.commons:commons-lang3:3.5'
    compile 'com.yunpian.sdk:yunpian-java-sdk:1.2.2'
    compile 'org.springframework.boot:spring-boot-starter-security:1.3.0.RC1'
    compile 'io.jsonwebtoken:jjwt:0.7.0'
    compile 'org.json:json:20160810'
    compile 'org.quartz-scheduler:quartz:2.2.3'
    compile 'org.springframework:spring-context-support:4.1.6.RELEASE'

    testCompile 'org.springframework.boot:spring-boot-starter-test'
    testCompile 'com.h2database:h2'
    testCompile 'org.springframework.security:spring-security-test'
    testCompile 'org.powermock:powermock-module-junit4:1.6.5'
    testCompile 'org.powermock:powermock-api-mockito:1.6.5'

}
```

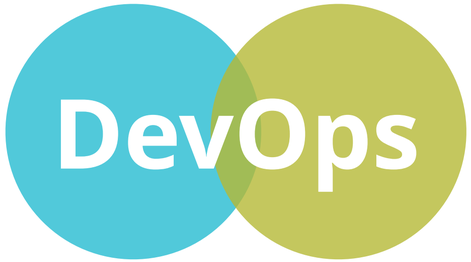
```
:dependencies

------------------------------------------------------------
Root project
------------------------------------------------------------

archives - Configuration for archive artifacts.
No dependencies

checkstyle - The Checkstyle libraries to be used for this project.
\--- com.puppycrawl.tools:checkstyle:7.6
     +--- antlr:antlr:2.7.7
     +--- org.antlr:antlr4-runtime:4.6
     +--- commons-beanutils:commons-beanutils:1.9.3
     |    \--- commons-collections:commons-collections:3.2.2
     +--- commons-cli:commons-cli:1.3.1
     \--- com.google.guava:guava:19.0

compile - Dependencies for source set 'main'.
+--- org.springframework.boot:spring-boot-starter-web: -> 1.5.1.RELEASE
|    +--- org.springframework.boot:spring-boot-starter:1.5.1.RELEASE
|    |    +--- org.springframework.boot:spring-boot:1.5.1.RELEASE
|    |    |    +--- org.springframework:spring-core:4.3.6.RELEASE
|    |    |    |    \--- commons-logging:commons-logging:1.2
|    |    |    \--- org.springframework:spring-context:4.3.6.RELEASE (*)
|    |    +--- org.springframework.boot:spring-boot-autoconfigure:1.5.1.RELEASE
|    |    |    \--- org.springframework.boot:spring-boot:1.5.1.RELEASE (*)
|    |    +--- org.springframework.boot:spring-boot-starter-logging:1.5.1.RELEASE
|    |    |    +--- ch.qos.logback:logback-classic:1.1.9
|    |    |    |    +--- ch.qos.logback:logback-core:1.1.9
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    +--- org.slf4j:jcl-over-slf4j:1.7.22
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    +--- org.slf4j:jul-to-slf4j:1.7.22
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    \--- org.slf4j:log4j-over-slf4j:1.7.22
|    |    |         \--- org.slf4j:slf4j-api:1.7.22
|    |    +--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    \--- org.yaml:snakeyaml:1.17
|    +--- org.springframework.boot:spring-boot-starter-tomcat:1.5.1.RELEASE
|    |    +--- org.apache.tomcat.embed:tomcat-embed-core:8.5.11
|    |    +--- org.apache.tomcat.embed:tomcat-embed-el:8.5.11
|    |    \--- org.apache.tomcat.embed:tomcat-embed-websocket:8.5.11
|    |         \--- org.apache.tomcat.embed:tomcat-embed-core:8.5.11
|    +--- org.hibernate:hibernate-validator:5.3.4.Final
|    |    +--- javax.validation:validation-api:1.1.0.Final
|    |    +--- org.jboss.logging:jboss-logging:3.3.0.Final
|    |    \--- com.fasterxml:classmate:1.3.1 -> 1.3.3
|    +--- com.fasterxml.jackson.core:jackson-databind:2.8.6
|    |    +--- com.fasterxml.jackson.core:jackson-annotations:2.8.0
|    |    \--- com.fasterxml.jackson.core:jackson-core:2.8.6
|    +--- org.springframework:spring-web:4.3.6.RELEASE
```

接近100个外部依赖

软件开发领域里的改变和趋势



持续交付
Continuous Delivery



开发自运维
DevOps

云
Cloud

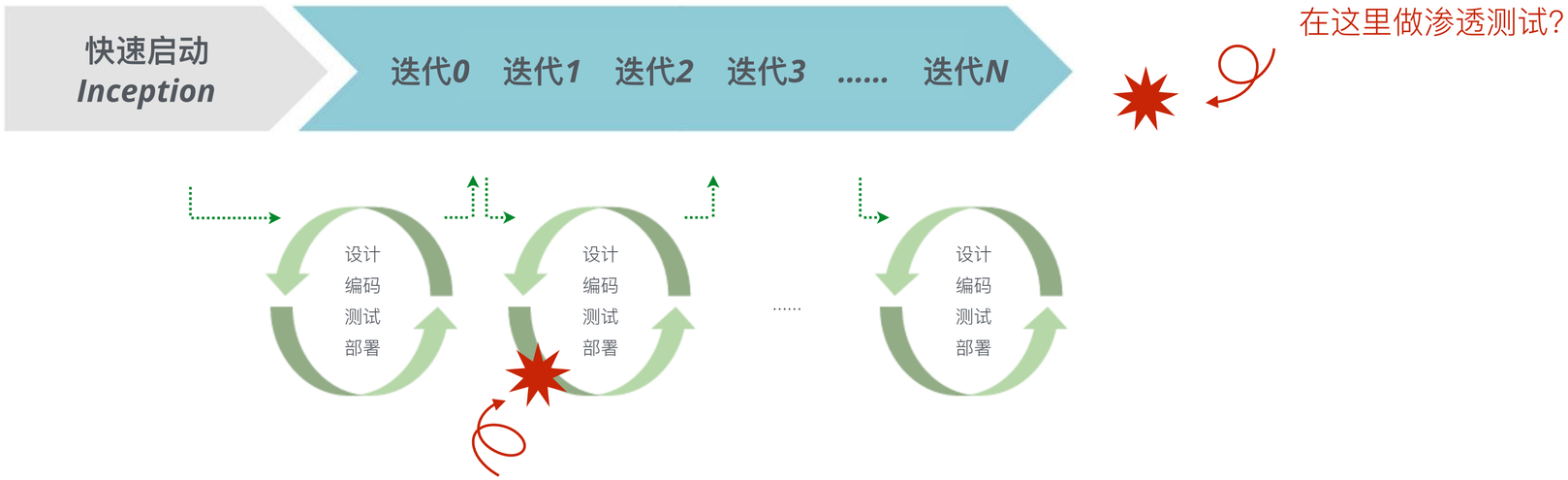
持续性的、快速的完成应用开发

开发团队正在持续性的，以更快的速度完成应用的设计、开发、测试、部署、发布等一些列流程，以更快的速度响应市场需求的变更，而不再采用传统的瀑布开发模式，在几个月的时间里闭门造车，然后推出到市场并期望获得成功。

带来了更多的安全挑战

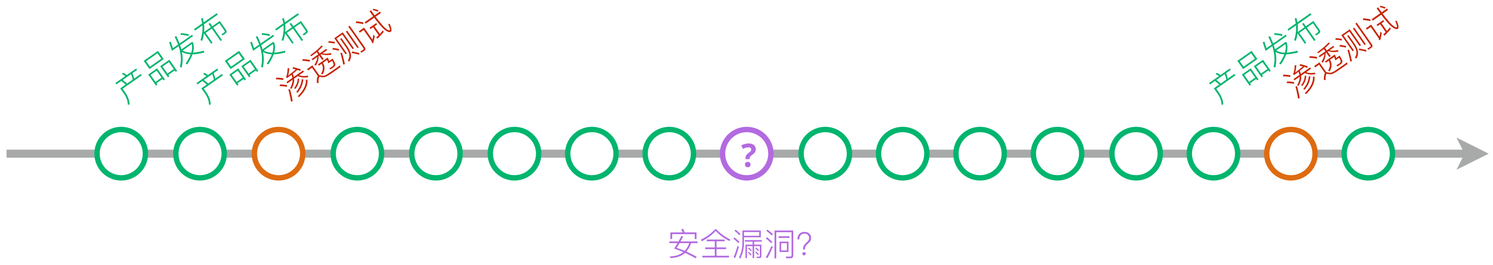
一次性的安全漏洞扫描、渗透测试无法很好的融入到持续交付流程

敏捷开发团队持续性的每一到两个迭代就发布一次产品



带来了更多的安全挑战

一次性的安全漏洞扫描、渗透测试无法很好的融入到持续交付流程



真实案例: 在两次渗透测试之间，产品团队已经进行了***18***次产品发布，这个过程中存在安全隐患

带来了更多的安全挑战

缺乏自动化安全测试的辅助

开发团队倾向于跳过某些耗时长、工作量大的安全活动

我们希望的

我们希望达到的状态

每个迭代在对应用进行发布的时候，都能够对应用进行自动化安全测试，以确保应用没有安全问题。

应对挑战

内建安全应用开发

BuildSecurityIn®

内建安全的应用开发是一系列安全原则、最佳安全实践以及安全工具的综合体，它使得企业或者团队在保证交付速度的同时，开发出具备更高安全质量的应用、服务。

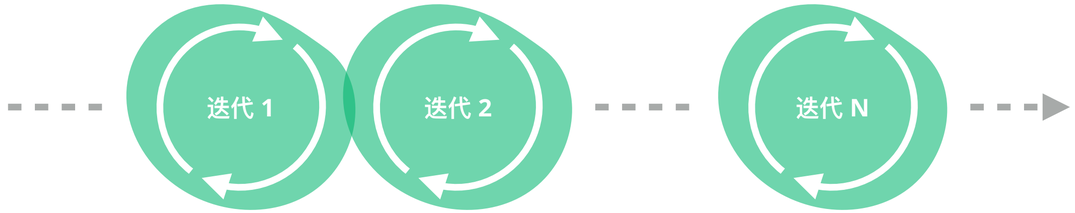
尽早识别安全需求、尽早获取安全反馈

充分利用自动化进行安全测试

将安全实践融入到持续交付流程中

共同承担安全职责

每个迭代内的安全实践



识别安全需求、
安全威胁

将自动化安全测试工具
集成在CI/CD中

自动化的
安全测试用例

工具辅助的
安全测试

实践：识别安全需求、安全威胁

明确识别应当保护什么，怎么去保护

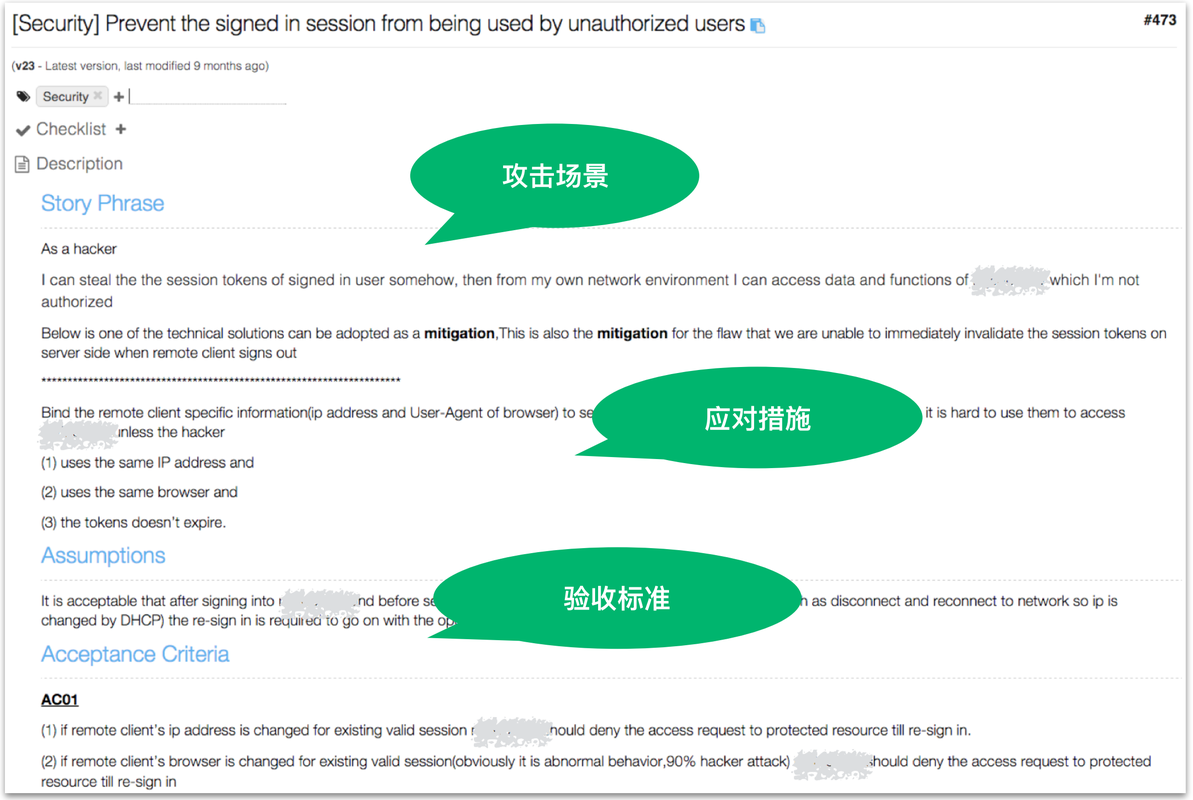
通过使用STRIDE模型来系统化的识别应用可能会面临的安全威胁，并利用DREAD模型来对威胁进行风险评估。安全威胁、安全需求越早被识别出来，修复或者针对安全问题的防御成本、难度都会越低。

STRIDE

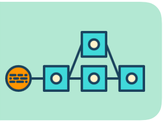
欺骗 *Spoofing*
篡改 *Tampering*
否认 *Repudiation*
数据泄露 *Information disclosure*
拒绝服务 *Denial of service*
权限提升 *Elevation privilege*

DREAD

破坏力 *Damage potential*
可重现 *Reproducibility*
利用难度 *Exploitability*
影响用户 *Affected users*
隐蔽性 *Discoverability*



实践：将自动化安全测试工具集成到CI/CD中



静态代码安全扫描

通过利用自动化的工具来对应用程序的源代码进行安全扫描，使得团队能够在源代码级别发现安全问题并可以开始快速进行问题修复。

动态应用安全扫描

通过在*CI*持续集成流水线中集成自动化的动态应用安全扫描工具，一旦发现有安全问题，持续集成流水线就会自动构建失败，触发报告，从而使得团队能在第一时间获知应用安全质量反馈。

第三方依赖安全检查

通过使用自动化的工具来对第三方依赖进行安全检查，使得团队能够以较少的时间资源投入，以更高的效率获知这些依赖是否含有已知安全漏洞，并安排相应的修复计划。

实践：自动化的安全测试用例

安全需求

订单 只能被 **订单拥有者** 查看，
除非当前用户具有以下角色：'**订单管理员**'， '**审计员**'.

安全测试伪代码

```
bob                 =  given_a_logged_in_user(“bob”);
bobs_order          =  given_an_existing_order_of_user(bob);
result              =  bob.request_order_details(bobs_order)
assert_that ( result, is ( APPROVED) );
```

```
bob                 =  given_a_logged_in_user(“bob”);
johns_order         =  given_an_existing_order_of_user(john);
result              =  bob.request_order_details(johns_order)
assert_that ( result, is ( REJECTED ) );
```

实践：工具辅助的安全测试

开发团队 (尤其是团队中的测试人员) 是最熟悉当前应用的群体，他们知道应用中哪些地方可能存在安全隐患，哪些地方需要特别留意。这些天生的优势是进行安全测试的有利因素，他们非常有潜力去发现应用中的潜在安全问题。

自动化的安全测试工具不是安全团队、安全专家的专用工具，而是可以提供给开发团队使用，并且配合上相应的安全培训，使得开发团队能够自助式的，自己动手检测一些基本的安全问题。

OWASP ZAP

...

知己知彼百战不殆

为了设计出更好的安全解决方案、避开安全问题、检测潜在安全风险，开发团队需要清楚的了解常见安全漏洞的根本性原因、防御策略、检测技巧等。通过给团队提供基于*OWASP TOP 10*的培训和工作坊，使得开发团队能够更好的达到这一目标。

- *A1 Injection*
- *A2 Broken Authentication and Session Management*
- *A3 Cross-Site Scripting (XSS)*
- *A4 Insecure Direct Object References*
- *A5 Security Misconfiguration*
- *A6 Sensitive Data Exposure*
- *A7 Missing Function Level Access Control*
- *A8 Cross-Site Request Forgery (CSRF)*
- *A9 Using Components with Known Vulnerabilities*
- *A10 Unvalidated Redirects and Forwards*

内建安全应用开发的收益

安全问题可以更早的被发现、修复，甚至是直接从源头避免掉。

自动化安全测试使得开发团队能够在每次产品发布时都能确保应用经过了安全测试，安全质量更有保证。

开发团队安全意识和技能得到提升，具备充足的能力以交付更具安全质量的应用

沟通协作

拆掉思维里的墙

ThoughtWorks®

部分内建安全实践分享

Practices of Build Security In

威胁建模

威胁建模是一个结构化的思考框架，它可以用来梳理应用程序中潜在的安全威胁，并且针对这些威胁进行风险级别评估，并制定出相应的安全应对措施。

实践分享：威胁建模

识别安全威胁、安全需求

- 什么东西可能出问题?
- 我们如何应对这些问题?

什么时候应该做威胁建模?

- 越早越好，通常是在快速启动阶段*(Inception)*
- 尽量做的轻量级一些，以便在每个迭代启动前做

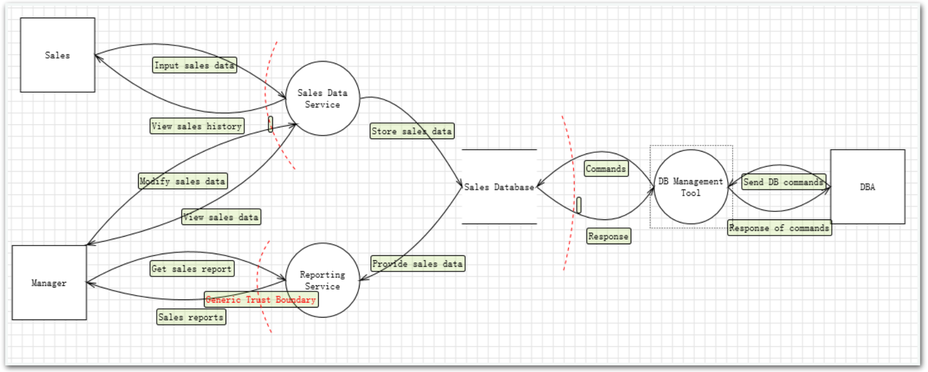
谁来做威胁建模?

- 团队共同协作完成，通常是*BA/DEV/QA*共同参与

实践分享：威胁建模-识别

Option 1 - RECOMMENDED

数据流图 Data Flow Diagram

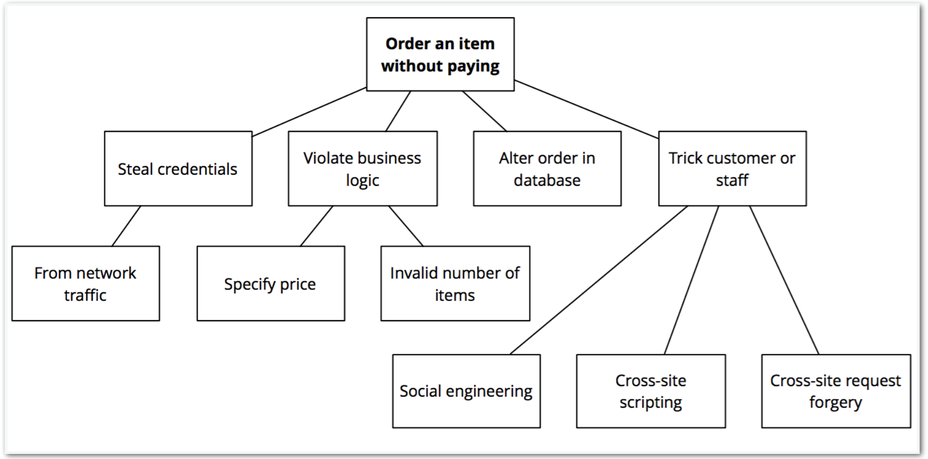


STRIDE

***S**poofing*	*Information disclosure*
***T**ampering*	***D**enial of service*
***R**epudiation*	***E**levation privilege*

Option 2

攻击树 Attack Tree



实践分享：威胁建模-评估

Option 1

DREAD

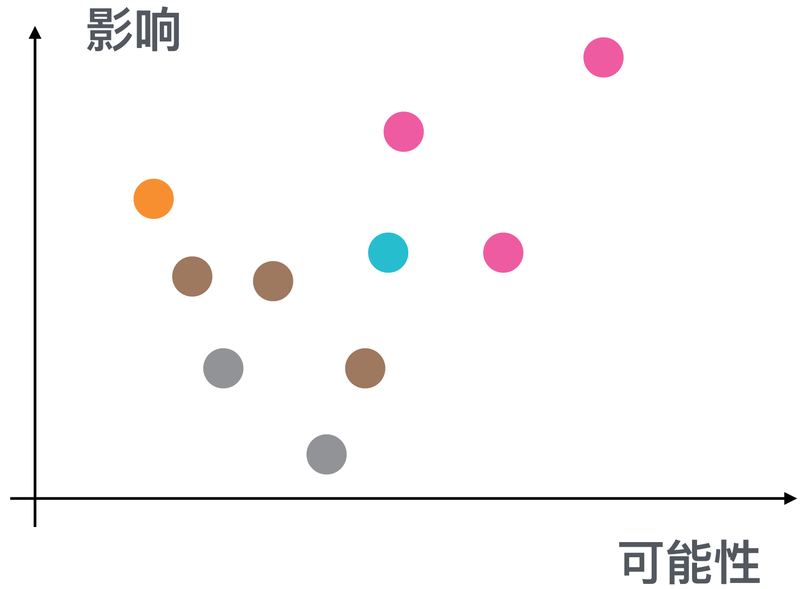
Category / Rating	3	2	1
Damage potential	Critical	Medium	Low
Reproducibility	Easy & Stable	Pre-condition required	One-off
Exploitability	Easy	Medium	Hard
Affected users	All / Most of	Partial	Very limited
Discoverability	Easy/Obvious	Medium	Hard

风险 = *(D+R+E+A+D)*

高: *15 ~ 12,* **中**: *11 ~ 8,* **低**: *7 ~ 0*

Example: 12 (High) = D:3 + R: 3 + E: 1 + A: 3 + D: 2

Option 2 - RECOMMENDED



实践分享：威胁建模-产出

Option 1

经过优先级排序后的威胁列表及其应对措施

Data lack of protection in transit

Action: use TLS / HTTPS for all the connections

Application has multiple roles, EOP issue may exists

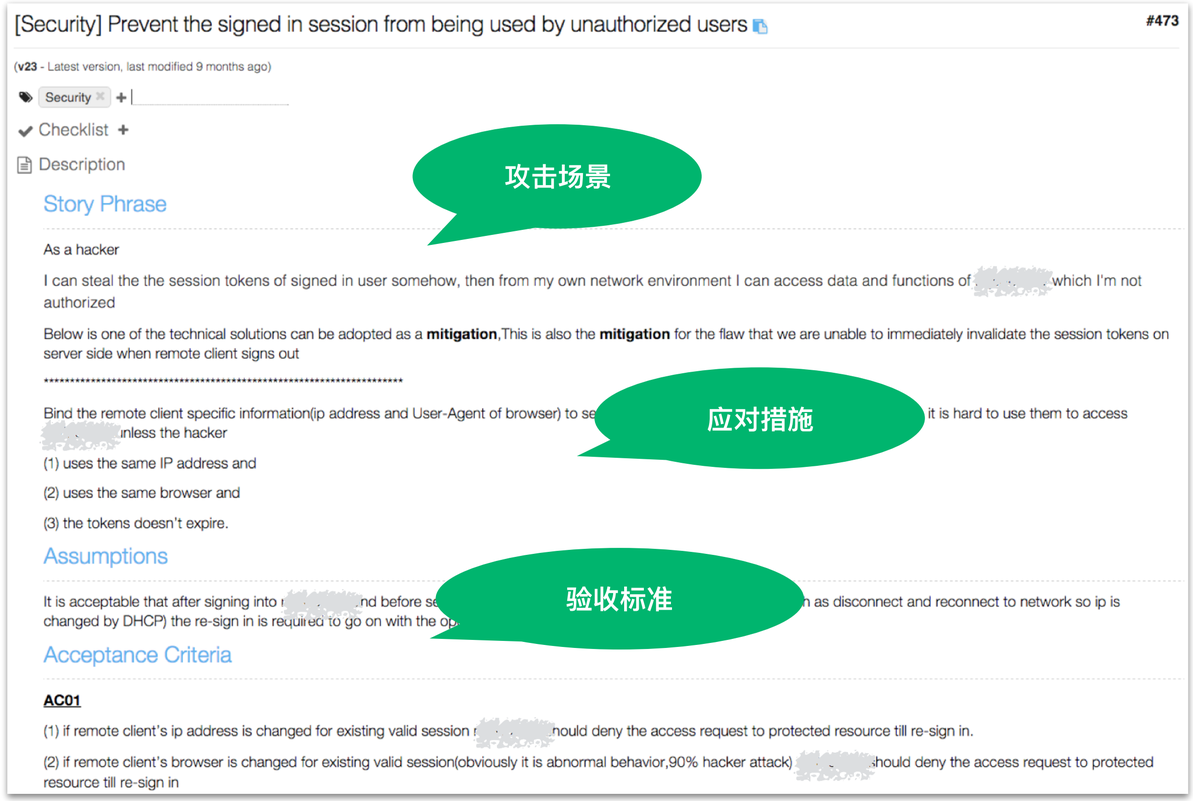
Action: (1) Strong authentication and authorization
(2) Least privilege principle
(3) Fail securely
(4) Exception warning
(5) Comprehensive dedicated security test cases

XSS issue in front web site

Action: (1) use front framework
(2) output encoding
(3) input validation
(4) Content Security Policy

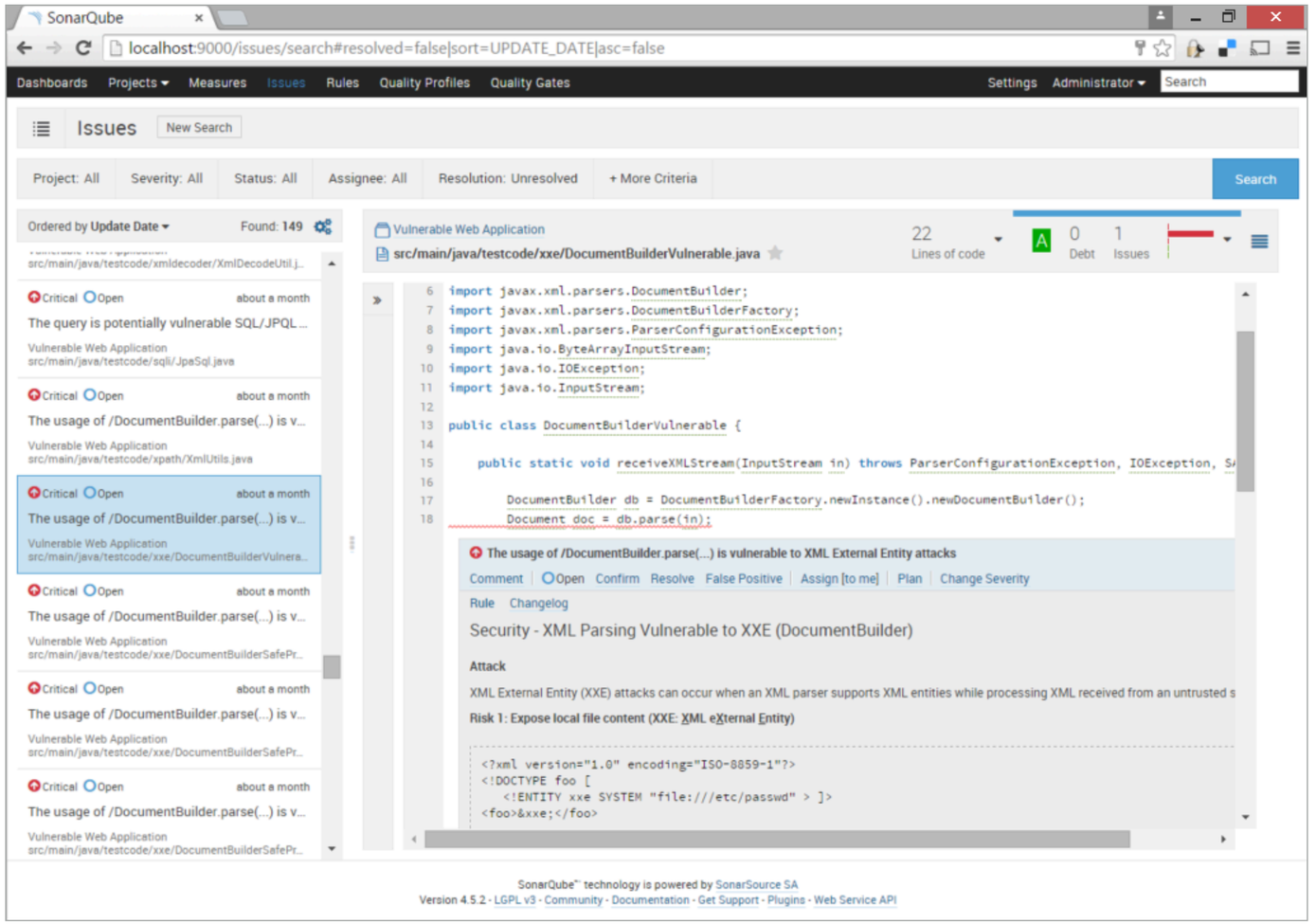
Option 2 - RECOMMENDED

将威胁转化为故事卡中的安全需求、验收标准，或者创建独立的安全故事卡



实践分享：安全测试策略

实践分享：静态代码安全扫描



实践分享：在CI/CD中集成自动化安全扫描工具

ThoughtWorks® | IT大咖说 知识共享平台

配置 >> 运行 >> 获取报告

```
apply plugin: "java"
apply plugin: "idea"

apply plugin: "security-zap"

buildscript {
    repositories {
        mavenCentral()
    }
    dependencies {
        classpath(
                'com.thoughtworks.tools:security-zap:1.0.5'
        )
    }
}
```

```
➜  customer-api git:(master) ✗ gradle zapStart build -Dzap.proxy=localhost:7070 zapReport
:zapStop
stopping zap
Warning: failed to stop ZAP due to connection refused or ZAP already stopped.
:zapStart
Starting ZAP [apikey: 2LZ77g9YVi]
waiting ZAP
waiting ZAP
waiting ZAP
ZAP started
exclusion rules are removed
urls match following regex will be excluded:
.*/css/.*
.*/js/.*
.*/fonts/.*
.*\.css
.*\.js
```

2. Security Alerts Summary

Number of alerts in total: 541

Alerts by severity	Amount
High	2
Medium	5
Low	360
Informational	174

3. Security Alerts By Classification

Classification	Amount
Cross Site Scripting (Reflected)	1
SQL Injection	1
Session ID in URL rewrite	4

实践分享：第三方依赖安全检查


```
:dependencies

------------------------------------------------------------
Root project
------------------------------------------------------------

archives - Configuration for archive artifacts.
No dependencies

checkstyle - The Checkstyle libraries to be used for this project.
\--- com.puppycrawl.tools:checkstyle:7.6
     +--- antlr:antlr:2.7.7
     +--- org.antlr:antlr4-runtime:4.6
     +--- commons-beanutils:commons-beanutils:1.9.3
     |    \--- commons-collections:commons-collections:3.2.2
     +--- commons-cli:commons-cli:1.3.1
     \--- com.google.guava:guava:19.0

compile - Dependencies for source set 'main'.
+--- org.springframework.boot:spring-boot-starter-web: -> 1.5.1.RELEASE
|    +--- org.springframework.boot:spring-boot-starter:1.5.1.RELEASE
|    |    +--- org.springframework.boot:spring-boot:1.5.1.RELEASE
|    |    |    +--- org.springframework:spring-core:4.3.6.RELEASE
|    |    |    |    \--- commons-logging:commons-logging:1.2
|    |    |    \--- org.springframework:spring-context:4.3.6.RELEASE
|    |    |         +--- org.springframework:spring-aop:4.3.6.RELEASE
|    |    |         |    +--- org.springframework:spring-beans:4.3.6.RELEASE
|    |    |         |    |    \--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    |         |    \--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    |         +--- org.springframework:spring-beans:4.3.6.RELEASE (*)
|    |    |         +--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    |         \--- org.springframework:spring-expression:4.3.6.RELEASE
|    |    |              \--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    +--- org.springframework.boot:spring-boot-autoconfigure:1.5.1.RELEASE
|    |    |    \--- org.springframework.boot:spring-boot:1.5.1.RELEASE (*)
|    |    +--- org.springframework.boot:spring-boot-starter-logging:1.5.1.RELEASE
|    |    |    +--- ch.qos.logback:logback-classic:1.1.9
|    |    |    |    +--- ch.qos.logback:logback-core:1.1.9
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    +--- org.slf4j:jcl-over-slf4j:1.7.22
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    +--- org.slf4j:jul-to-slf4j:1.7.22
|    |    |    |    \--- org.slf4j:slf4j-api:1.7.22
|    |    |    \--- org.slf4j:log4j-over-slf4j:1.7.22
|    |    |         \--- org.slf4j:slf4j-api:1.7.22
|    |    +--- org.springframework:spring-core:4.3.6.RELEASE (*)
|    |    \--- org.yaml:snakeyaml:1.17
|    +--- org.springframework.boot:spring-boot-starter-tomcat:1.5.1.RELEASE
|    |    +--- org.apache.tomcat.embed:tomcat-embed-core:8.5.11
|    |    +--- org.apache.tomcat.embed:tomcat-embed-el:8.5.11
|    |    \--- org.apache.tomcat.embed:tomcat-embed-websocket:8.5.11
|    |         \--- org.apache.tomcat.embed:tomcat-embed-core:8.5.11
|    +--- org.hibernate:hibernate-validator:5.3.4.Final
|    |    +--- javax.validation:validation-api:1.1.0.Final
|    |    +--- org.jboss.logging:jboss-logging:3.3.0.Final
|    |    \--- com.fasterxml:classmate:1.3.1 -> 1.3.3
|    +--- com.fasterxml.jackson.core:jackson-databind:2.8.6
|    |    +--- com.fasterxml.jackson.core:jackson-annotations:2.8.0
|    |    \--- com.fasterxml.jackson.core:jackson-core:2.8.6
|    +--- org.springframework:spring-web:4.3.6.RELEASE
```

以前

通过媒体被动获取漏洞信息

人工审查

耗时长

&

不可持续

现在

全自动化的工具

迅速获取安全质量

&

持续监控

实践分享：自动化安全测试用例

自动化安全测试用例

基于安全需求制定安全测试用例

Given an anonymous visitor
When I try to access report page without authentication
Then I was been redirected to login page

Given a user without report access permission
When I try to access report page with authentication
Then I was been redirected to error page

Given a system manager
When I try to access report page with authentication
Then I can access report page successfully

将安全测试用例通过普通的测试来实现

```
public void anonymousVisitorCanNotAccessReportPage() {
    Page currentPage = accessReportPage();
    assertThat(currentPage, is(LOGIN_PAGE));
}
```

```
public void userWithoutProperPermissionCanNotAccessReportPage() {
    loginAsMember();
    Page currentPage = accessReportPage();
    assertThat(currentPage, is(PERMISSION_REQUIRED_ERROR_PAGE));
}
```

```
public void managerCanAccessReportPage() {
    loginAsManager();
    Page currentPage = accessReportPage();
    assertThat(currentPage, is(REPORT_PAGE));
}
```

实践分享：将自动化的安全测试集成到CI/CD中

将自动化的安全测试集成到CI/CD流程中

使得团队能够持续性的对应用的安全质量进行监控

小结

内建安全应用开发

BuildSecurityIn®

内建安全的应用开发是一系列安全原则、最佳安全实践以及安全工具的综合体，它使得企业或者团队在保证交付速度的同时，开发出具备更高安全质量的应用、服务。

尽早识别安全需求、尽早获取安全反馈

充分利用自动化进行安全测试

将安全实践融入到持续交付流程中

共同承担安全职责

谢谢

ThoughtWorks®