Cloud-Native Security for Java

Billy Korando
Developer Advocate
@BillyKorando

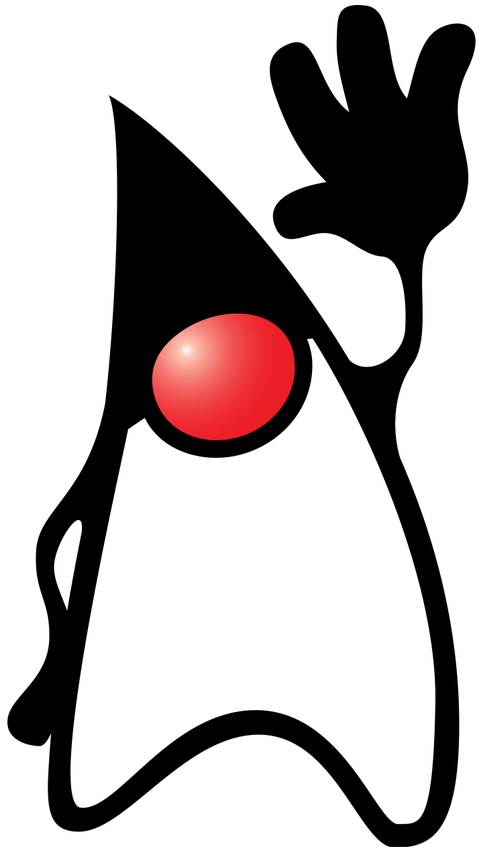
Erin Schnabel
STSM, Cloud Native architectures
@ebullientworks

Code ⇄ think

IBM

https://cloud.ibm.com/docs/java

IBM **Cloud**



http://ibm.biz/javanewsletter

Email addresses

Passwords

Every application

Birthday

Phone number

has valuable data

Full names

Addresses

Order history

Security design must be proactive

Security-in-Depth

Good authentication/authorization practices

Application

Securely storing configuration and credentials

Operations

Encrypting sensitive data

Data storage

Application Security

Identity

Authentication/Authorization

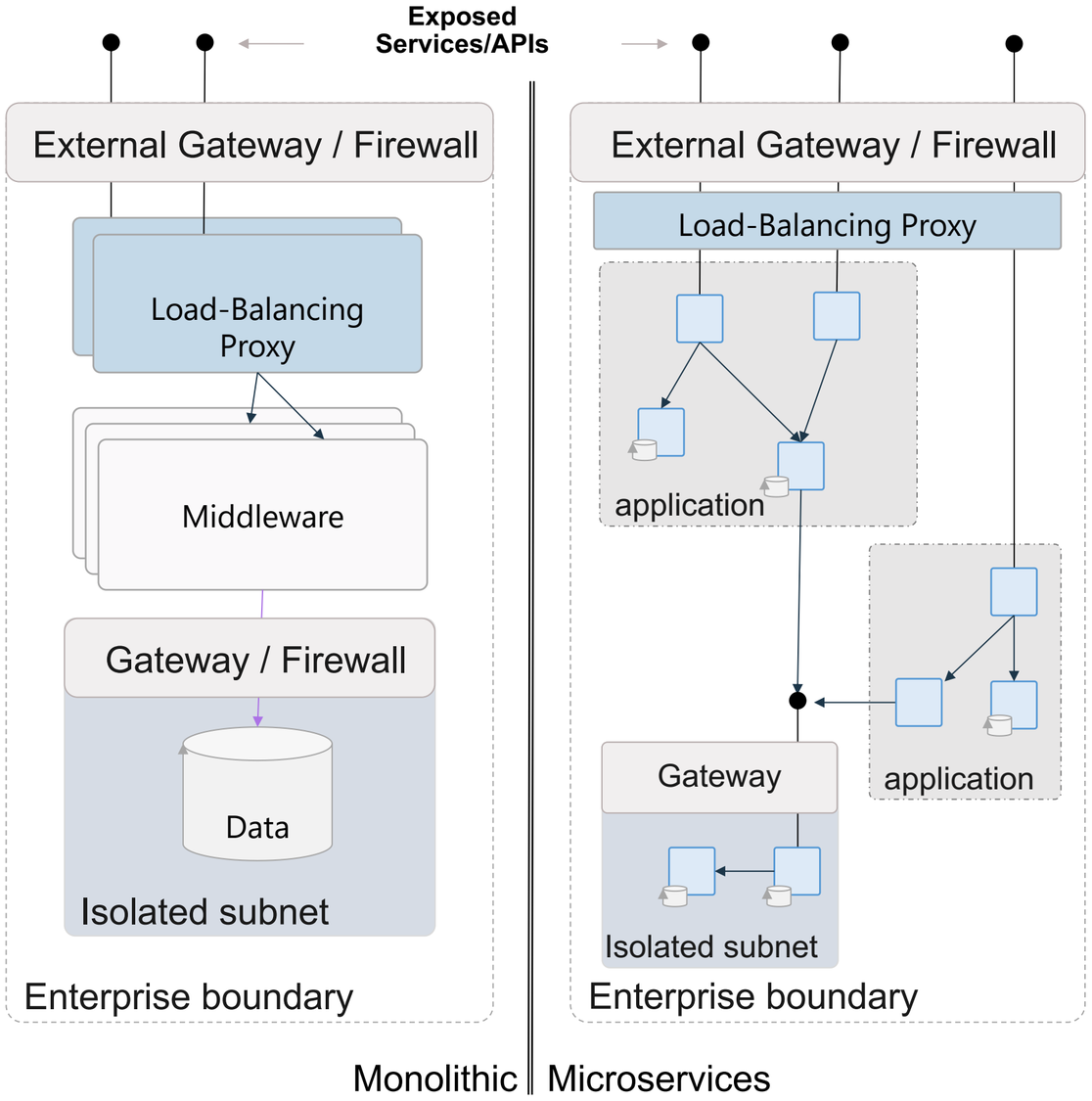
Token propagation

Session management

Accidental Leaking

Application Security is way more complicated

- Services care about originating ID
- Services need to establish trust
  - Who is invoker?
- Avoid bottlenecks
  - E.g. calls to centralized service



Microservices Best Practices for Java, an IBM Redbooks publication
http://www.redbooks.ibm.com/redpieces/abstracts/sg248357.html?Open

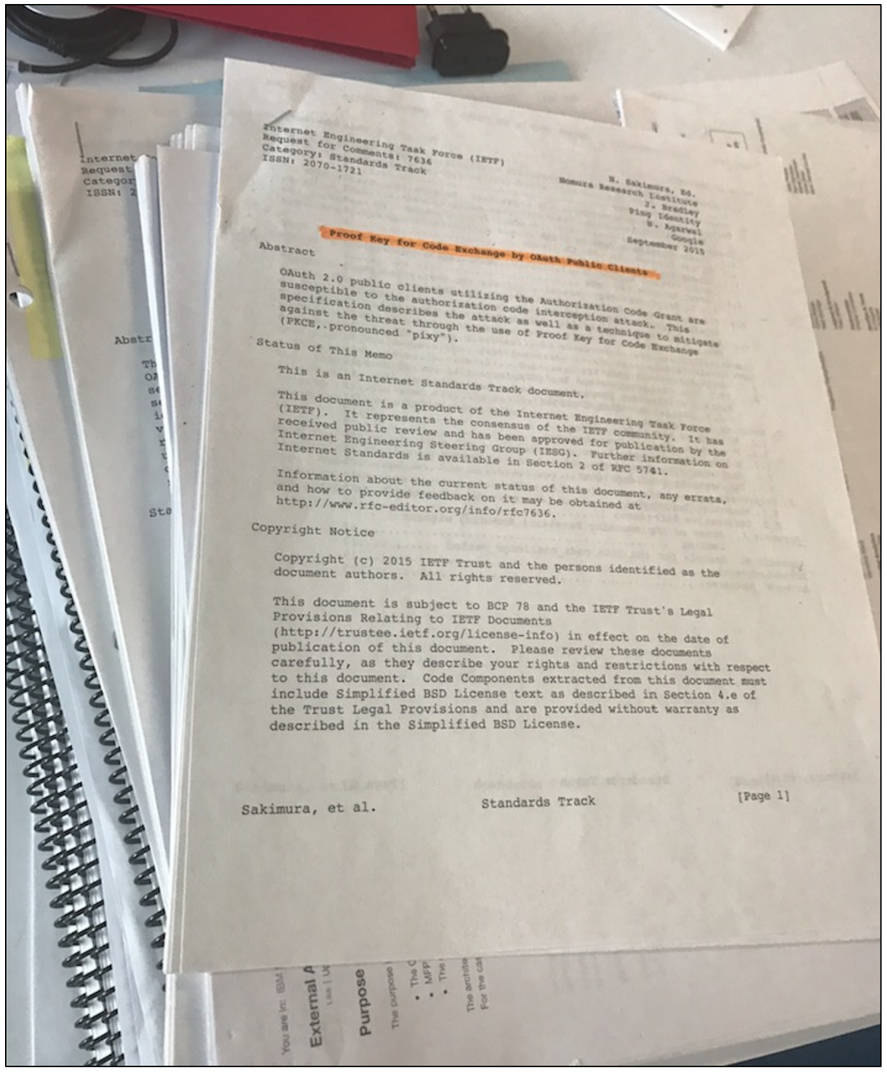
Application security is not piece of cake

!=

Cake photo copyrights - http://www.pieceofcakeinc.com/

Care to read some RFCs?

1. RFC 6749 - The OAuth 2.0 Authorization Framework
2. RFC 6750 - The OAuth 2.0 Authorization Framework: Bearer Token Usage
3. Open ID Connect
    1. Core
    2. Discovery
    3. Dynamic registration
4. RFC 7591 - OAuth 2.0 Dynamic Client Registration Protocol
5. RFC 7592 - OAuth 2.0 Dynamic Client Registration Management Protocol
6. RFC 7518 - JSON Web Algorithm (JWA)
7. RFC 7519 - JSON Web Token (JWT)
8. RFC 7515 - JSON Web Signature (JWS)
9. RFC 7636 - Proof Key for Code Exchange by OAuth Public Clients
10. RFC 7662 - OAuth 2.0 Token Introspection
11. RFC 7521 - Assertion Framework for OAuth 2.0 Client Authentication and Authorization Grants
12. RFC 7523 - JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants
13. RFC 7522 - Security Assertion Markup Language (SAML) 2.0 Profile for OAuth 2.0 Client Authentication and Authorization Grants
14. RFC 7642 - System for Cross-domain Identity Management: Definitions, Overview, Concepts, and Requirements
15. RFC 7643 - System for Cross-domain Identity Management: Core Schema
16. RFC 7644 - System for Cross-domain Identity Management: Protocol



This doesn’t touch Java EE or Spring security behaviors, either…

Cloud Native security:

Open Authorization framework (OAuth)

Open ID Connect (ODIC)

OAuth: Authorization

- access delegation
- ***grant*** access without sharing passwords

Social sign-on

Authorization

Authentication

OIDC: identity

- REST/JSON API
- Identity atop OAuth

Token exchange protocol

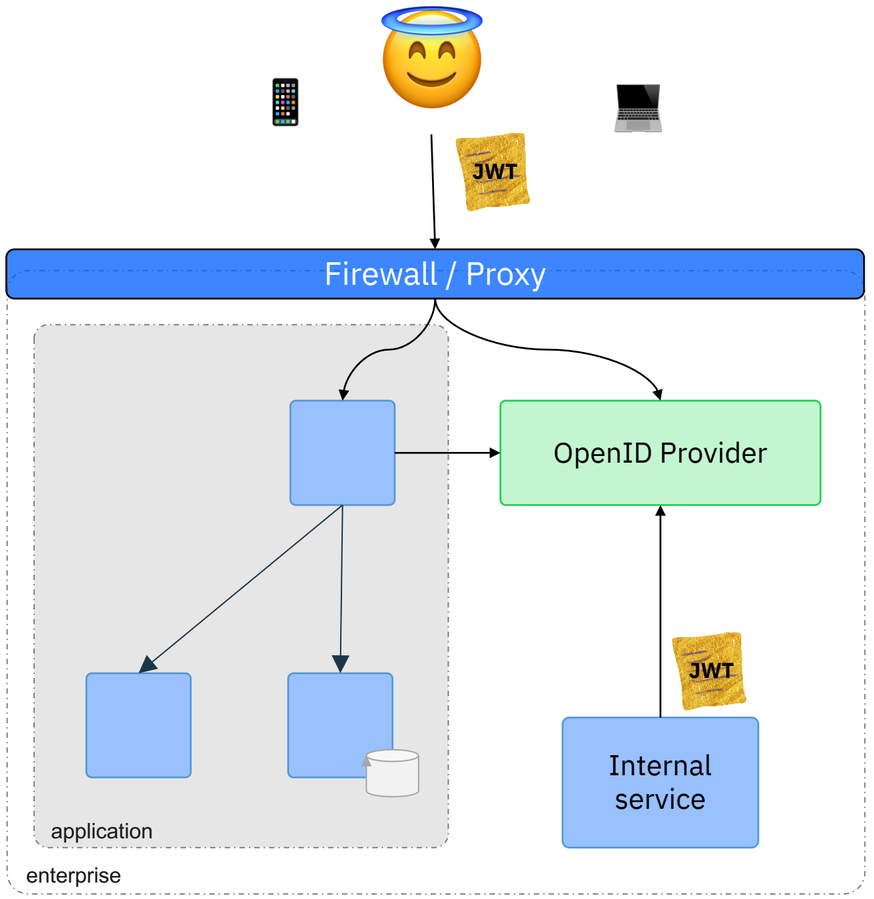
Identity, Authentication, Authorization

OpenID Provider

- Credential management #ftw
- Replaces custom user repositories

Federated identity

- Social sign-on
- Enterprise user repositories
- Backend services



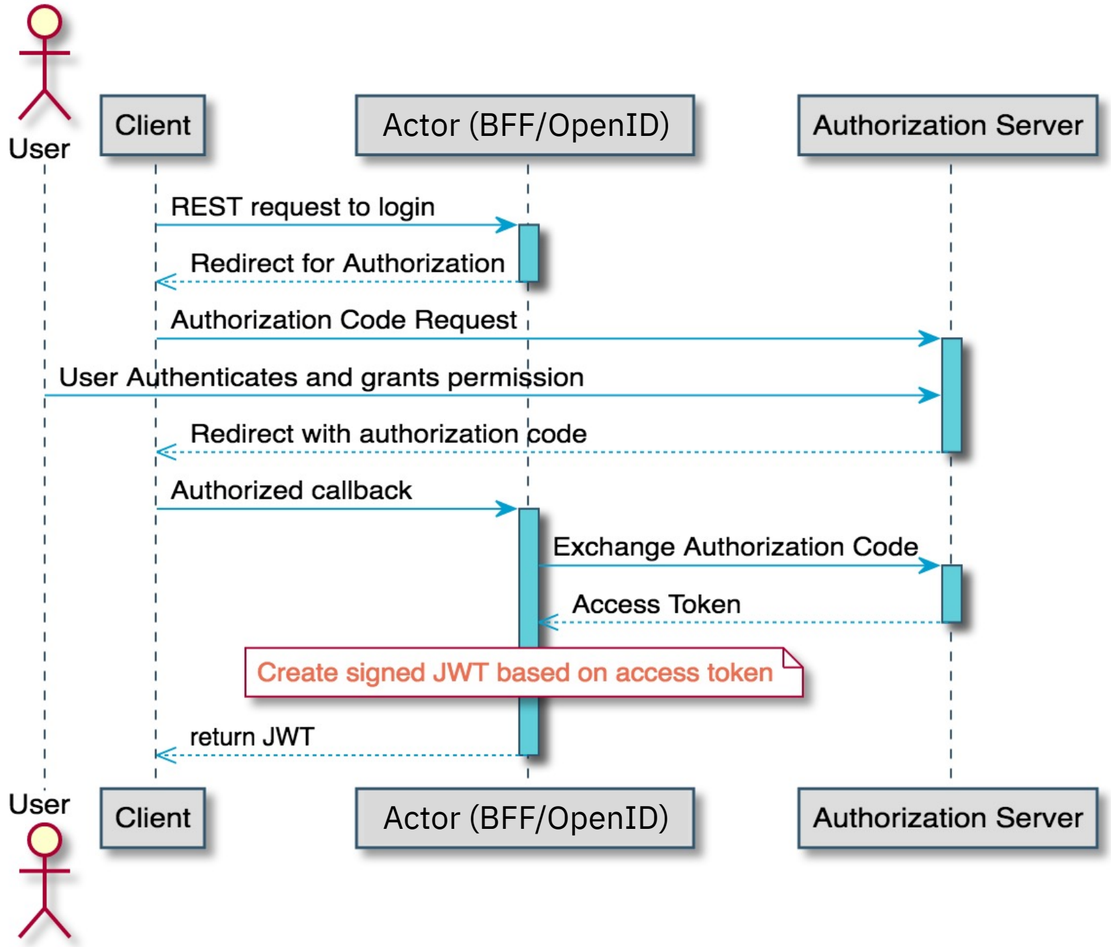
Result of OAuth flow is a JWT

- Browser redirects for users
- Shared secret for internal services

ALL backing services can use JWTs

- Simplify, remove special cases

OAuth flow to grant access to information



JSON Web Tokens / JSON Web Keys

JSON Web Token (JWT)

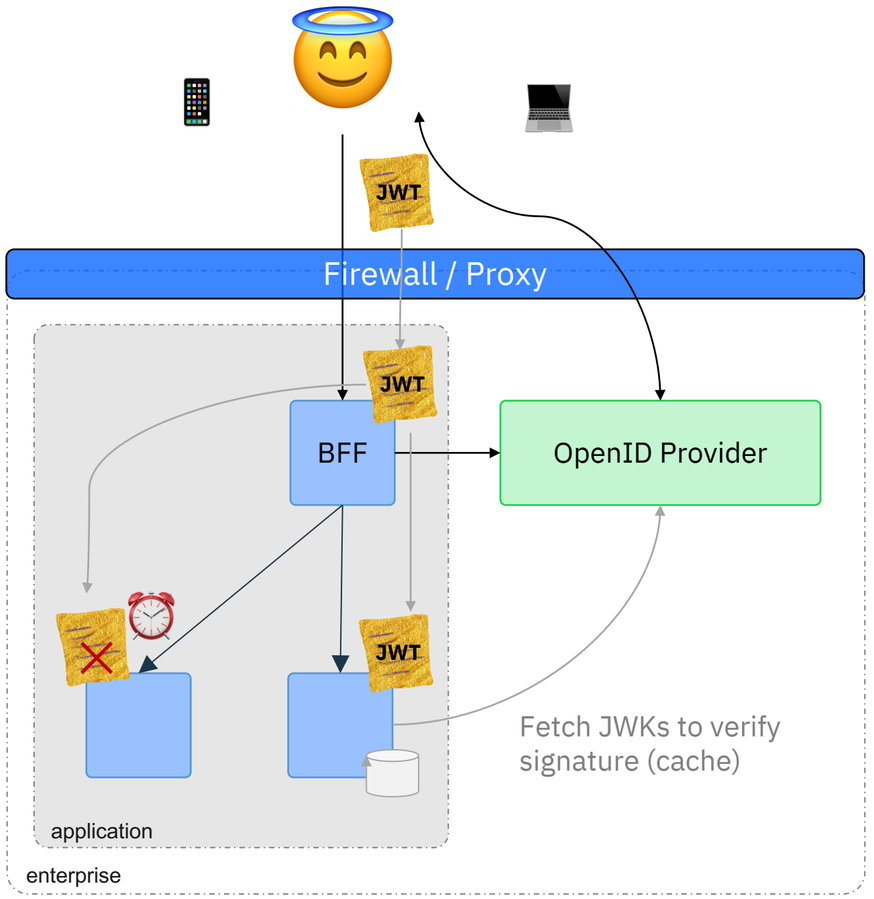
- Claims for identity / role
- **Signed for trust**
  - NOT encrypted

JWT can be validated

- Signature: modification
- Expiration time

JWT can be hashed with request

- Prevent replay attacks



Maintaining Trust with JWT

OpenID Provider:

- *Only* signer of JWT
- Key rotation / revocation

- Asymmetric keys (e.g. SSL / SSH)
  - OpenID provider uses private key
  - JSON Web Key (JWKs) endpoints
    - discover public keys

```
public String createJwt(Key key) throws Exception {
    // create and sign the JWT, including a hint
    // for the key used to sign the request (kid)
    String newJwt = Jwts.builder()
    .setHeaderParam("kid", "meaningfulName")
    .setSubject("user-12345")
    .setAudience("user")
    .setIssuedAt(Date.from(Instant.now()))
    .setExpiration(Date.from(Instant.now().plus(15, ChronoUnit.MINUTES)))
    .signWith(SignatureAlgorithm.RS256, key)
    .compact();
    return newJwt;
}

public void validateJwt(String jwtParameter, Key key) throws Exception {
    // Validate the Signed JWT!
    // Exceptions thrown if not valid
    Jws<Claims> jwt = Jwts.parser()
    .setSigningKey(key)
    .parseClaimsJws(jwtParameter);
    // Inspect the claims, like make a new JWT
    // (need a signing key for this)
    Claims jwtClaims = jwt.getBody();
    System.out.println(jwtClaims.getAudience(), jwtClaims.getIssuer());
}
```

Claims

Signed with private key

Validate against public key

Token and Session Management

- Use JWT for identity propagation instead of sticky sessions

- JWT tokens are the golden ticket!
  - Assume they will escape (e.g. browser back button)
  - Be concise in JWT claims

- ***Make sure tokens expire***
  - Use application libraries for token renewal flows
  - Create new system-tokens when user interaction is no longer possible

Information Leakage

Login

Username not found!

Username

SomeUserName

Password

••••••••••

Login

Password is incorrect!

Username

ACorrectUserName

Password

•••••••••

Login

Username or Password is incorrect!

Username

ABadUserName

Password

••••••••••

Information Leakage

Forgot Login

Email Address Not Found!

Email Address

WrongEmail@mail.com

Forgot Login

Email sent check your inbox!

Email Address

WrongEmail@mail.com

Information Leakage – Final Notes

- Use SSL
  - Prevent spoofing of your website
  - Encrypts data being sent in requests and responses
- Don’t put sensitive info in URL address
- Ensure client UI isn’t storing sensitive information in the clear

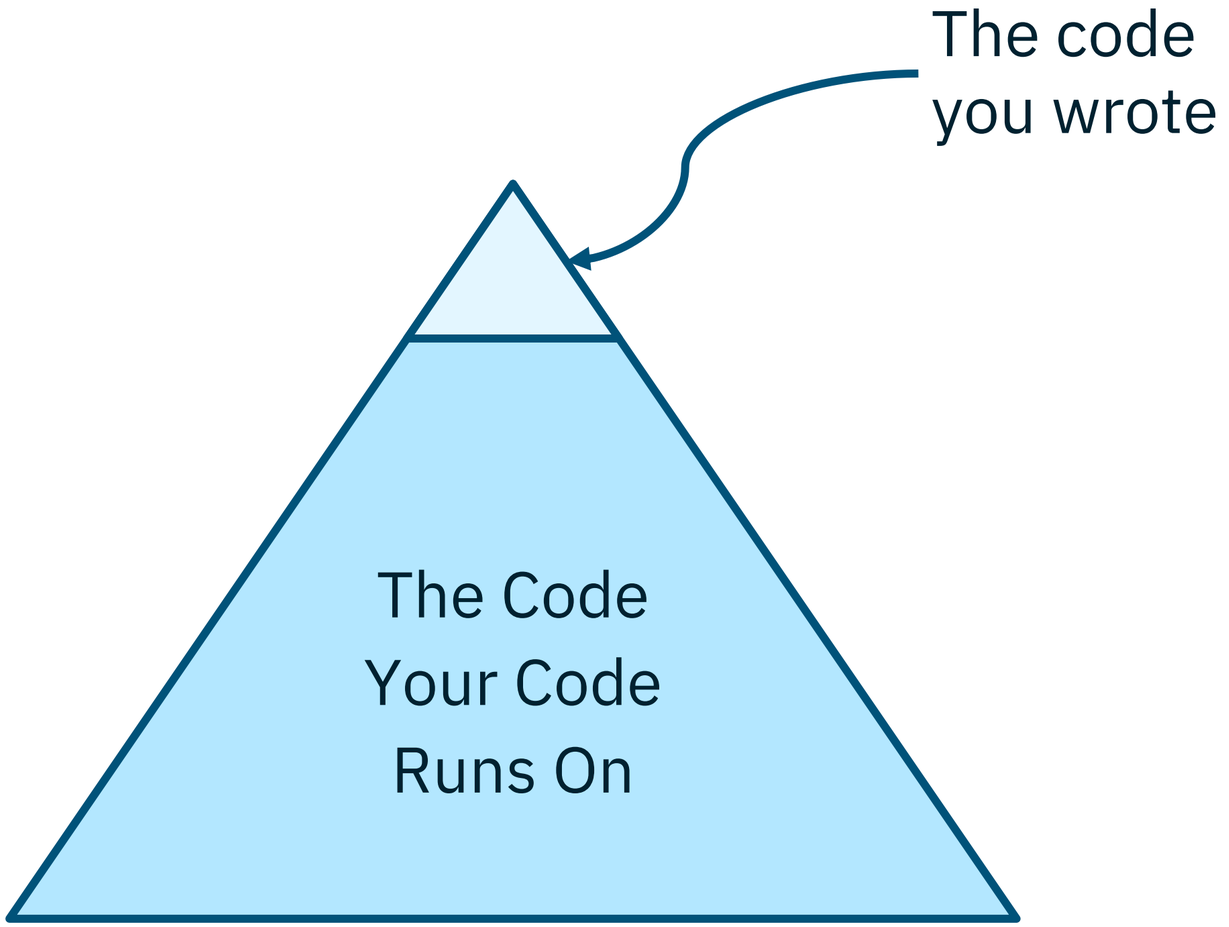
Infrastructure Security

Keep dependencies up-to-date

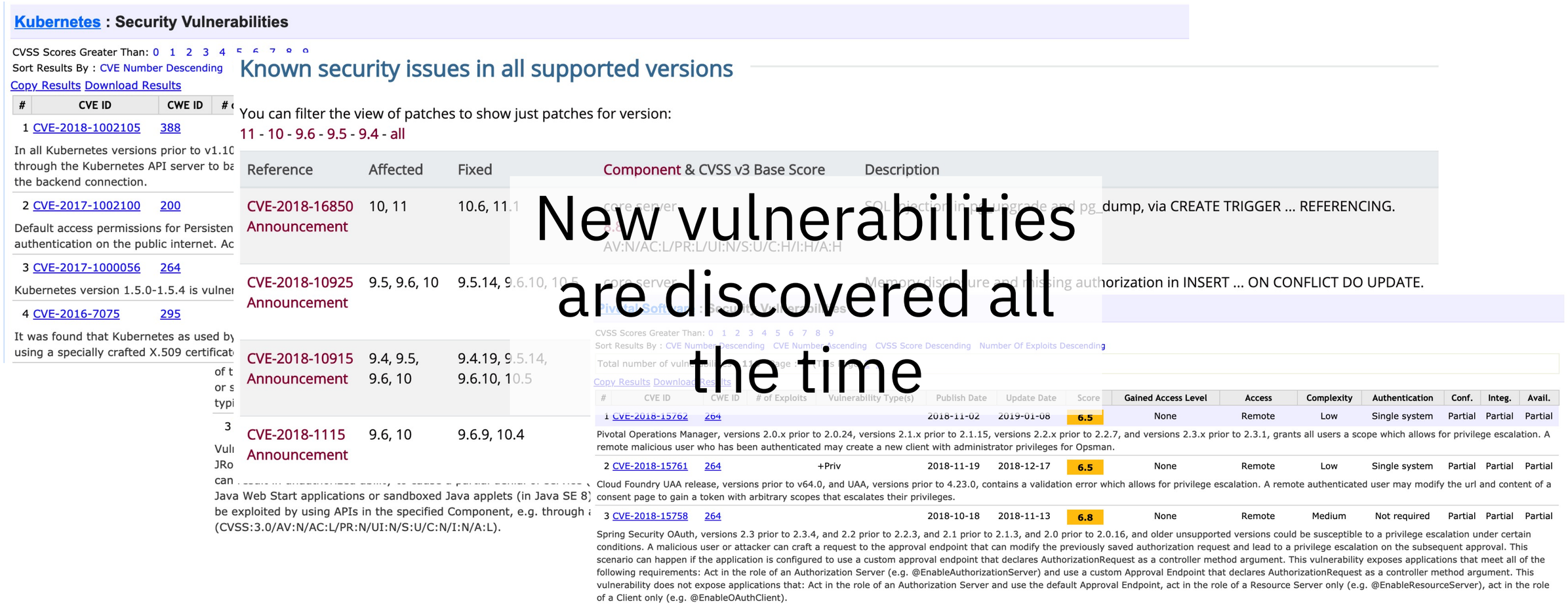
Securely store config info

Logging/Observability

Keep Dependencies Up-to-Date



Keep Dependencies Up-to-Date



Kubernetes : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending
Copy Results Download Results

#	CVE ID	CWE ID
1	CVE-2018-1002105	388

In all Kubernetes versions prior to v1.10 through the Kubernetes API server to ba the backend connection.

2	CVE-2017-1002100	200

Default access permissions for Persisten authentication on the public internet. Ac

3	CVE-2017-1000056	264

Kubernetes version 1.5.0-1.5.4 is vulner

4	CVE-2016-7075	295

It was found that Kubernetes as used by using a specially crafted X.509 certificat

of t
or s
typi

3
Vul
JRo
can
Java Web Start applications or sandboxed Java applets (in Java SE 8) be exploited by using APIs in the specified Component, e.g. through a (CVSS:3.0/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:N/A:L).

Known security issues in all supported versions

You can filter the view of patches to show just patches for version:
11 - 10 - 9.6 - 9.5 - 9.4 - all

Reference	Affected	Fixed	Component & CVSS v3 Base Score	Description
CVE-2018-16850 Announcement	10, 11	10.6, 11.1	core server AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H	SQL injection in pg_upgrade and pg_dump, via CREATE TRIGGER ... REFERENCING.
CVE-2018-10925 Announcement	9.5, 9.6, 10	9.5.14, 9.6.10, 10.5	core server	Memory disclosure and missing authorization in INSERT ... ON CONFLICT DO UPDATE.
CVE-2018-10915 Announcement	9.4, 9.5, 9.6, 10	9.4.19, 9.5.14, 9.6.10, 10.5		
CVE-2018-1115 Announcement	9.6, 10	9.6.9, 10.4		

Pivotal Software : Security Vulnerabilities

CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending
Total number of vulnerabilities : 11 Page : 1 (This Page)
Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2018-15762	264			2018-11-02	2019-01-08	6.5	None	Remote	Low	Single system	Partial	Partial	Partial

Pivotal Operations Manager, versions 2.0.x prior to 2.0.24, versions 2.1.x prior to 2.1.15, versions 2.2.x prior to 2.2.7, and versions 2.3.x prior to 2.3.1, grants all users a scope which allows for privilege escalation. A remote malicious user who has been authenticated may create a new client with administrator privileges for Opsman.

2	CVE-2018-15761	264		+Priv	2018-11-19	2018-12-17	6.5	None	Remote	Low	Single system	Partial	Partial	Partial

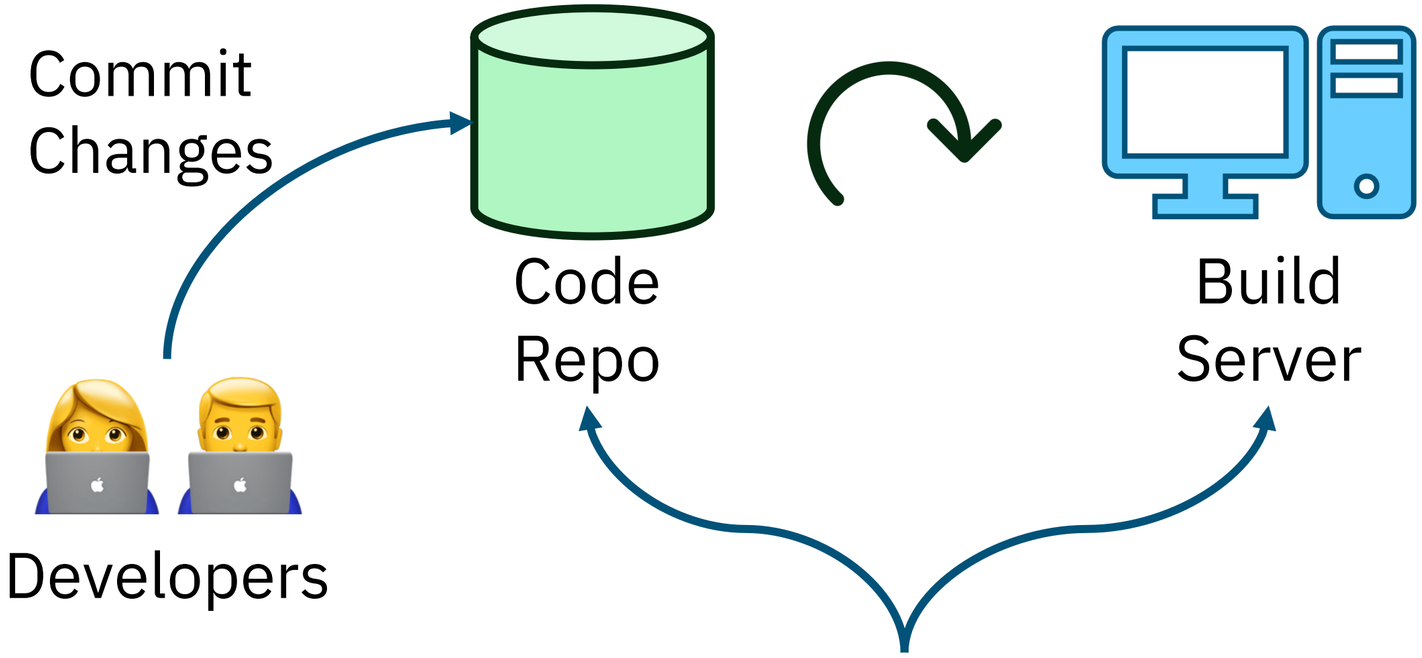
Cloud Foundry UAA release, versions prior to v64.0, and UAA, versions prior to 4.23.0, contains a validation error which allows for privilege escalation. A remote authenticated user may modify the url and content of a consent page to gain a token with arbitrary scopes that escalates their privileges.

3	CVE-2018-15758	264			2018-10-18	2018-11-13	6.8	None	Remote	Medium	Not required	Partial	Partial	Partial

Spring Security OAuth, versions 2.3 prior to 2.3.4, and 2.2 prior to 2.2.3, and 2.1 prior to 2.1.3, and 2.0 prior to 2.0.16, and older unsupported versions could be susceptible to a privilege escalation under certain conditions. A malicious user or attacker can craft a request to the approval endpoint that can modify the previously saved authorization request and lead to a privilege escalation on the subsequent approval. This scenario can happen if the application is configured to use a custom approval endpoint that declares AuthorizationRequest as a controller method argument. This vulnerability exposes applications that meet all of the following requirements: Act in the role of an Authorization Server (e.g. @EnableAuthorizationServer) and use a custom Approval Endpoint that declares AuthorizationRequest as a controller method argument. This vulnerability does not expose applications that: Act in the role of an Authorization Server and use the default Approval Endpoint, act in the role of a Resource Server only (e.g. @EnableResourceServer), act in the role of a Client only (e.g. @EnableOAuthClient).

New vulnerabilities are discovered all the time

Keep Dependencies Up-to-Date



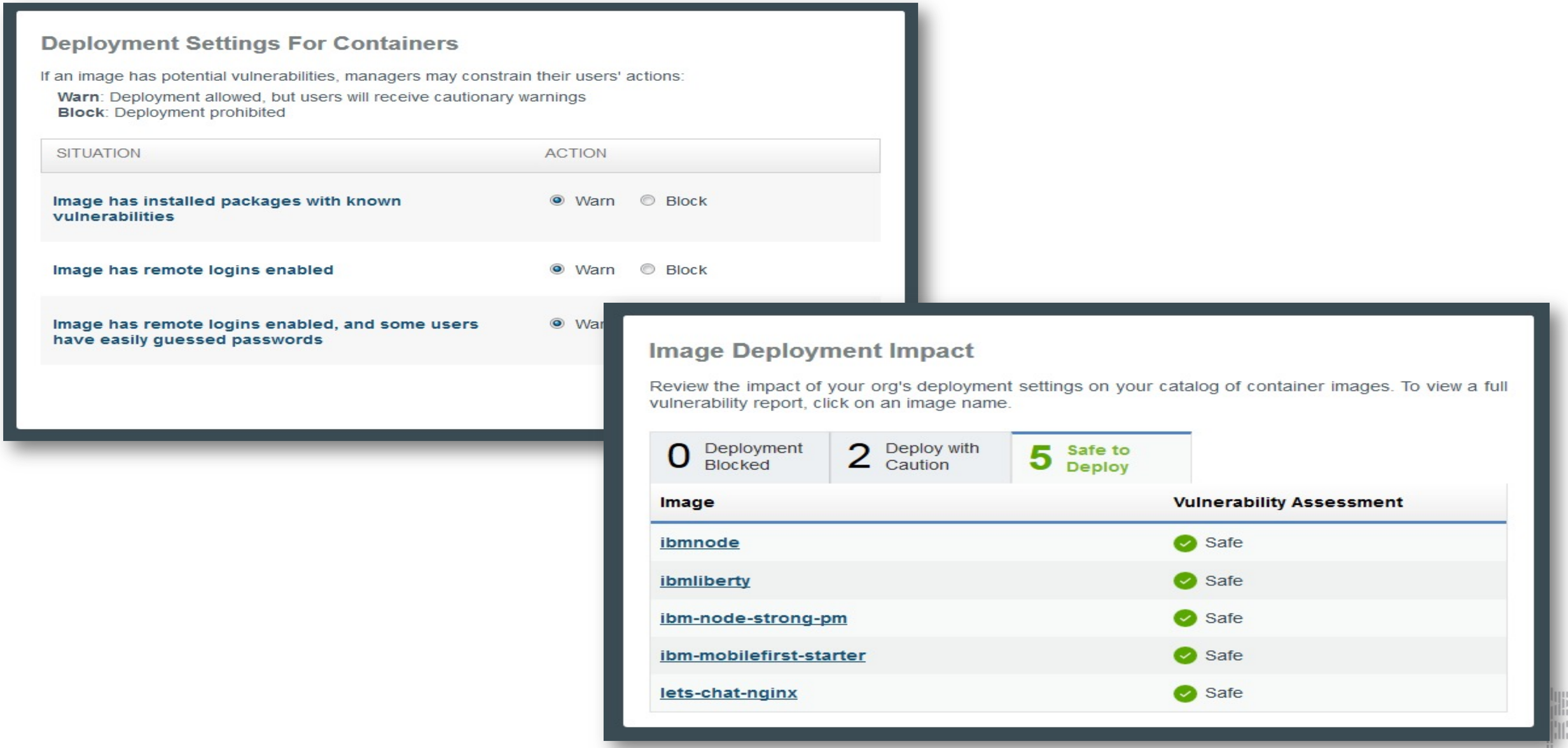
snyk

OWASP
Open Web Application
Security Project

IBM Cloud

Automate Scanning
of Vulnerabilities

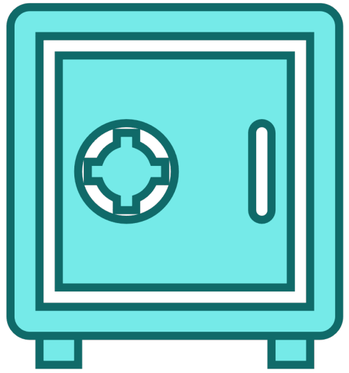
Vulnerability Advisor – Process Manager



Securely Storing Config

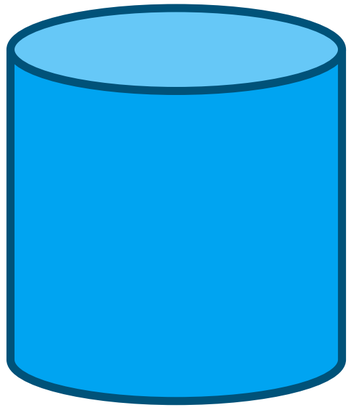
Don’t store sensitive config info in code repositories

Securely Storing Config



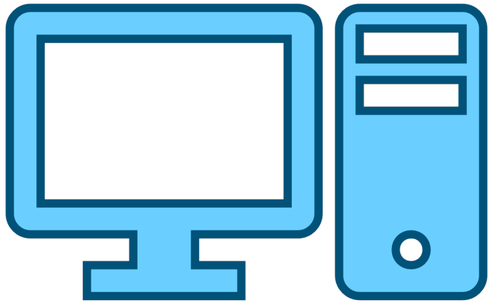
Secured Vault

Hashicorp
Spring Vault



Secured Repository

Spring Cloud Config



Build Server

Travis CI Config

Kubernetes Secrets

Logging/Observability

```
org.springframework.boot.test.autoconfigure.filter.TypeExcludeFiltersContextCustomizer@351584c0,
org.springframework.boot.test.autoconfigure.properties.PropertyMappingContextCustomizer@f83937ce,
org.springframework.boot.test.autoconfigure.web.servlet.WebDriverContextCustomizerFactory$Customizer@6356695f],
contextLoader = 'org.springframework.boot.test.context.SpringBootContextLoader', parent = [null]], attributes =
map[[empty]]]; transaction manager [org.springframework.orm.jpa.JpaTransactionManager@78b7f805]; rollback [true]
2019-02-12 11:36:51.340  INFO 36356 --- [           main] com.bk.hotel.repo.PersonRepo             : Added new
customer:
2019-02-12 11:36:51.340  INFO 36356 --- [           main] com.bk.hotel.repo.PersonRepo             : Customer
[id=1, firstName=John, lastName=Doe, middleName=Middle, suffix=Jr., ssn=333-22-4444]
2019-02-12 11:36:51.347  INFO 36356 --- [           main] o.s.t.c.transaction.TransactionContext   : Rolled back
transaction for test: [DefaultTestContext@424e1977 testClass = PersonRepo, testInstance =
com.bk.hotel.repo.PersonRepo@3571b748, testMethod = savePersons@PersonRepo, testException = [null],
mergedContextConfiguration = [MergedContextConfiguration@10d68fcd testClass = PersonRepo, locations = '{}',
classes = '{class com.bk.hotel.HotelApplication}', contextInitializerClasses = '[]', activeProfiles = '{}',
propertySourceLocations = '{}', propertySourceProperties =
'{org.springframework.boot.test.autoconfigure.orm.jpa.DataJpaTestContextBootstrapper=true}', contextCustomizers =
set[[ImportsContextCustomizer@117e949d key = [org.springframework.boot.autoconfigure.cache.CacheAutoConfiguration,
org.springframework.boot.autoconfigure.data.jpa.JpaRepositoriesAutoConfiguration,
```

Logging/Observability

```
HTTP/1.1 400 Bad Request
Date: Tue, 12 Feb 2019 23:51:06 GMT
Server: Apache/2.4.6 (CentOS) mod_wsgi/4.5.16 Python/3.4
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=UTF-8

1f0c

<!DOCTYPE html>
<html lang="en">
<head>
  <meta http-equiv="content-type" conten...
```

https://www.shodan.io/

Logging/Observability

Restrict information your web server provides to clients

Be very discretionary in the information you log

Encrypt potentially sensitive information

Securely store logging information

Limit how long logging information is kept

Be sure sensitive information isn’t shown in observability metrics

Last Tips

- Automated testing essential in security design
  - Ensure security practices are being properly applied
  - Ensure changes don’t undermine security
  - Gives confidence to upgrade dependencies

Sources

https://www.troyhunt.com/
https://shodan.io
https://www.owasp.org/

Q&A

Slides:
@BillyKorando
@ebullientworks

Sign up for the newsletter:
http://ibm.biz/javanewsletter