Getting your Ducks in a Row

Contents

1 Getting your Ducks in a Row

An Introduction to Managing Components in your Software Supply Chain

Manfred Moser - #simpligility - www.simpligility.com

Sonatype - www.sonatype.com

2 What Are You Gonna Learn Today?

- Interesting facts and figures
- A new perspective about your development efforts
- Lots of simple steps for improvements

3 Data Sources

- Central Repository usage statistics
  - A.k.a. Maven Central
  - Largest binary repository
  - Running for 10+ years
  - Sponsored and managed by Sonatype
- Sonatype Nexus usage statistics
  - Most commonly use repository manager
  - >50k installations

4 Research Reports

- 2011-2014 Open Source Software Development Survey
  - >11k responses
- 2015 State of the Software Supply Chain Report

5 How We Develop Today

- Use components as building blocks
- Enjoy power of reuse and collaboration
- Software stack doesn’t matter - all use components
- Open source FTW everywhere!

6 Guess...

What is the percentage of components in a typical application?

Tip
80-90%

7 Estimate...

How many components can be found in an average application?

Tip
More than 100 !

8 So What is a Component?

> Frameworks, Libraries, ... Plumbing you don't want to write yourself. Like Logging, IoC, persistence layer, ORM, widgets,...

You get them easily by declaring dependencies with Maven, Gradle, nuget, npm...

9 What is a Component?

Third party AND everything you create

- JARs, WARs, EARs..
- rpm, deb
- npm, nupkg, gem packages
- zip, tar.gz files
- installer packages, docker images
- ...

10 Software Supply Chain

Just like for traditional manufacturing

- Understand complete inventory of your product
- Including origin (suppliers)
- And usage

11 Software Supply Chain Management

Components are integral part of Software Development Life Cycle (SDLC)

It is an endless circle of activities:

- Research what components to use
- Implement usage of components in development
- Check component usage in QA and release process
- Monitor applications and components used in production
- Go back to the start and change versions or entire components

12 So what do you think. . .

When do software supply chain management efforts stop?

Tip
When all production deployments are turned off!

Warning
Not when development stops.

13 Supply Chain Best Practices

- Better and fewer suppliers
- Higher quality parts
- Improved visibility and traceability

14 Best Practice: Select Projects

Tip
Open source projects are your suppliers!

- Large vs small project
- Active vs inactive in terms of commits
- Foundation backed or stand alone project
- Commercial company backing or not
- Active community and support

15 Best Practice: Communicate with Suppliers

- Most often open source project
- Report bugs and feature requests
- Help with documentation
- Be present on mailing lists, forums, IRC, . . .
- Attend events

16 Best Practice: Support Suppliers

- Promote project via presentations, . . .
- Become a committer
- Sponsor a committer
- Provide infrastructure
- Sponsor foundations
- Pay for support

17 Best Practice: Reduce Vendors

Each additional vendor

- Adds integration complexity
- Adds communication channels
- Add need for tracking
- Add new APIs to learn
- Adds license terms to understand
- . . .

Tip
So having less is easier.

---

18 Now..

That we know that we use **lots** of components

Where do we get them from?

19 Public Repositories as Warehouses/Distributors

Very important in their eco-systems

- JVM - Central Repository - 17B downloads in 2014
  - up from 500 M in 2007
- JavaScript/Node - npmjs.org - 15B downloads in 2014
- Ruby - rubygems.org - 5B downloads since inception
- .Net - NuGet Gallery - 300m downloads in 2014

20 Example: Central Repository

- Approx 1 Million open source components
- Approx 11 Million users
- 1000 new components added daily
- Exponential growth

Tip
Growth of other repositories is similar

---

21 Best Practice: Control Component Source

Tip
Run your own local warehouse!

- Reduced bandwidth usage and costs
- Improve performance and stability
- Internal caching and storing of components → enables collaboration
- Reduced dependency on external repositories
- One component storage location for backup, audit, control...
- Store your own components centrally

→ Use a **repository manager**!

22 Guess...

Are people following this easily implemented best practice?

> **Tip**
> No!

- 95% of downloads from Central Repository → build tools,...
- Only 5% via repository manager
- 18 % of respondents to component survey use **no** repository manager

23 Sonatype Nexus as Central Hub

→ Nexus is a key component of your enterprise development infrastructure

24 Repository Management with Nexus OSS

- Used by 64% of repository manager users
- Formats include Maven, NuGet, NPM, site, Yum and Gems

→ Way better than manual management or ignoring the need

> **Important**
> Yet easy to implement... and open source!

25 Best Practice: Know Your Components

- Look at your build files
- Crack open the deployment archive
- Identify with checksum search

> **Warning**
> You will be surprised what you find!

26 Best Practice: Know Their Dependencies

- `mvn dependency:tree` or similar analysis
- Use Dependency Management or BOM POM
- Dependency Hierarchy in M2Eclipse or Nexus Pro
- Challenge yourself to produce a Bill of Materials

Tip
Demo time!

27 Best Practice: Avoid Duplication

- Multiple logging frameworks
- Multiple web frameworks
- Multiple technology stacks

Tip
But still don't be afraid of using what is best for the job. Find the right balance.

28 Best Practice: Reduce Their Numbers

- KISS[https://en.wikipedia.org/wiki/KISS_principle]
- Less complexity
- Less learning effort
- Less tracking updates, issues, communication, ...

Warning
You are responsible for **all** components used in your application!

29 Room for Improvement

Sonatype Application Health Check analysis of 1500+ applications

- On average 106 components
- 24 with known vulnerabilities
- 9 with restrictive licenses

30 What About Component Versions . . .

From the Top 100 components downloaded from the Central Repository - how many are old?

Tip
27 - so about a third is out of date!

Not too surprising, since a typical component has 3-4 releases per year.

31 Guess. . .

How many versions of each library are used at Google?

Tip
One or two are mandated in most cases!

32 Excursion to DevOps

One critical part of DevOps - Release Early, Release Often!

Why?

- Bring benefits of new features to users as soon as possible
- Enable tighter user involvement
- Fix bugs as soon as possible
- Reduce complexity of change

33 Best Practice: Upgrade Component Versions Regularly

Just like release often - upgrade often!

- Reduces complexity of updates
- Access latest features
  - Open source projects work on master
  - Latest release - latest features and fixes
  - Sometimes you will get burned with regressions
- Access latest security fixes
  - Back ports are very rare
- Easiest to report issues and receive fixes

And just like in DevOps

→ The more often you release(upgrade), the better you get at it.

34 Guess . . .

An average large enterprise downloads about 250k components from the Central Repository per year.

How many have known security vulnerabilities?

Tip
Approximately 15k!

Some of them are running in production right now. . .

35 How do Companies React?

- About 50k components with known security vulnerabilities in Central Repository
- 46 million vulnerable components were downloaded in 2014.
- 16% must prove they are not using known security vulnerabilities
- New vulnerabilities are found regularly
- Yet 63 % do NOT monitor for changes in vulnerability data

→ Lip service mostly or struggling.

36 Best Practice: Know Component Security Characteristics

Very difficult, laborious task

- Follow mailing lists
- Monitor security databases
- Figure out specific versions affected
- Assess impact

37 Security Tools

Some free:

- OWASP dependency check

Various commercial

- E.g. Nexus Pro

Tip
Demo time!

38 Similar Problems with Licenses

- 63 % of respondents have incomplete view of license risk from components
- Only 32 % examine all open source components for license compliance
- 58 % say they are not concerned about license risk

BUT

Approx 280k components in Central Repository have restrictive licenses.

39 Best Practice: Understand License Implications

Similar to security issue - laborious and difficult task

Tools to the rescue

40 What now?

Follow DevOps ideas again..

- Define what we want to do
- Automate
- Monitor

41 Define What We Want

Define policies e.g.

- No components older than 5 years
- No components with know security vulnerabilities of score > 8
- No GPL licensed components

42 Problems with Policies

- Only 56 % have policies
- Of these only 68% follow policy
- Often manual, slow
- But 78% say they have never banned a component

→ Things do not add up, too painful to work with.

43 Let's automate this!

- Add tools to automate the process
- Configure tools with policies

44 Aim of Tools

1. Empower developers with the right information at the right time
2. Design frictionless, automated approach for continuous DevOps processes
3. Create, manage and monitor component bill of materials for each application

→ More and more tools for different stacks are emerging!

45 Bill of Material

- Tracking productions applications BOM
  - 40% including dependencies
  - 23% NOT including dependencies

46 Sonatype Nexus Pro

- component info
- RHC
- search

47 Nexus Pro+

In a nutshell:

- Configurable component policies - **very** powerful
- Managed on the Sonatype CLM server
- Tied into Nexus staging

48 Sonatype Nexus Lifecycle

Expands Nexus Pro+

- Manual analysis via web interface upload
- Eclipse IDE integration
- Continuous Integration Server Jenkins/Hudson/Bamboo support
- SonarQube support
- Command line scanning

49 Summary

- Your code is only part of your application
- Components are important
- Apply software supply chain thinking
- Easy to start with
- Powerful tools available

Tip
Don't wait!

---

50 The End

Questions, Remarks & Discussion

Tip
Slides on OSCON site or email manfred@sonatype.com now

---

51 Resources

- 2014 Open Source Software Development Survey Results
- 2015 State of the Software Supply Chain Report
- Sonatype slides
- The Nexus Community
- www.sonatype.com
- Repository Management with Nexus
- Application Health Check
- modulecounts.com
- Java Tools and Technologies Landscape for 2014
- Nexus releated slides including this one...

52 Disclaimers

Image sources:

- Ducks in a Row from wikimedia