Its All About Making the Right Choice:
How to choose the right Open Source for your DevOps projects

Presented By:
Meetali Sharma

Statistics…

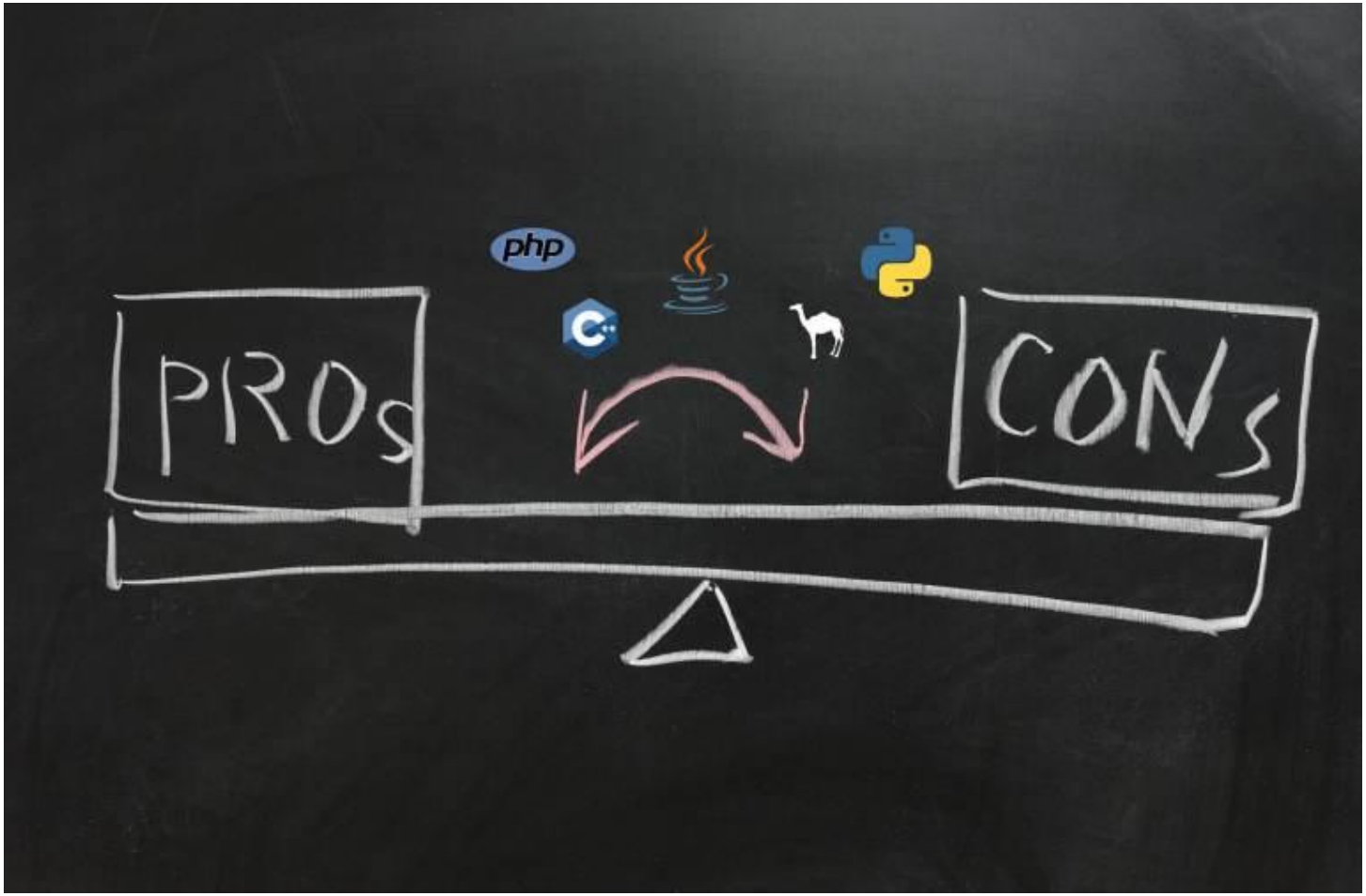
- As many as 93 percent of organizations use open source software and 78 percent run part or all of their operations on it, according to *The Tenth Annual Future of Open Source Survey*

- As applications continue to move online, more companies and development teams are adopting a process of continuous software development and deployment, such as DevOps

Can **TRADITIONAL** web application security controls fit in…

NO

… a DevOps & Open Source environment?!


php
PROs
CONs

Open Source – Advantages

- Generally free
- Continuously evolving and improving – fast fixes, integration
- Scalable and can be changed easily
- Less costly, low licensing fees
- Agile and customizable with real time improvements
- Easy to manage

Open Source – Disadvantages

- Software support/Orphan software
- Warranties
- Training and skill set requirement for support
- Susceptible to vulnerabilities and attacks
- May not be too user-friendly

How to choose – Questions to ask?

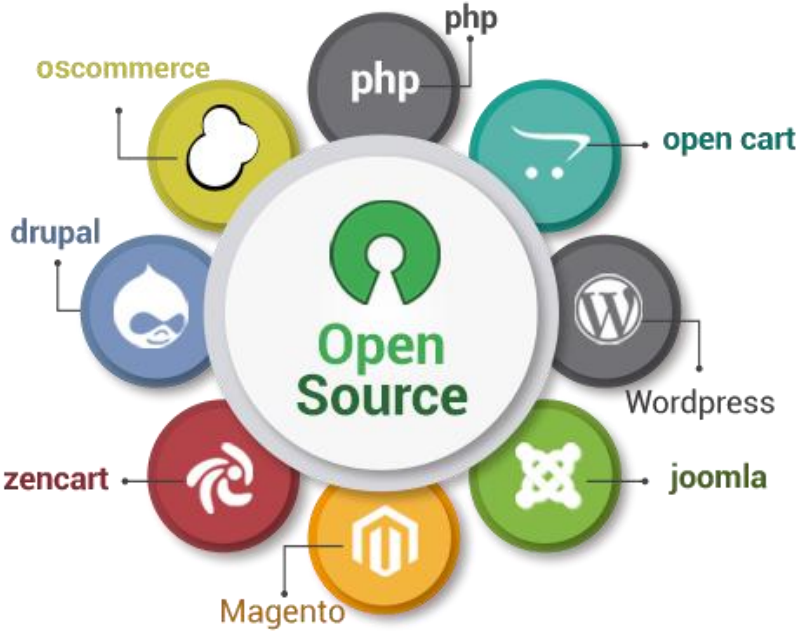
- Does it do what you want?
  - Does the software do what you want it to do? What are your requirements?
- Is the software good for its role?
  - Investigate prior uses of the software. Has anyone used it in the manner you want to use it?
- Is the software actively used, developed and supported?
  - The importance of an active community around the software shouldn't be underestimated. Check out the mechanisms for how support is provided: support forums, direct email support and issue trackers, and investigate whether they are actively used and have a good level of response to queries and issues.
- Does the software have a future?
- How is the software provided?
  - In most cases, good user and developer documentation is a must.
  - Are the prerequisites of the software well defined and straightforward to obtain and deploy, and do they fit your own requirements?
- Choosing the right version

Way Forward…

Secure SDLC Approach

Conduct Risk Assessment

- Conduct a thorough Risk assessment of your environment to understand the inherent risks, need for the open source software and where it needs to be used



Plan for Security

- Identify unsecured APIs and frameworks
- Map security sensitive code portions. E.g. password changes mechanism, user authentication mechanism.
- Anticipate regulatory problems, plan for it.

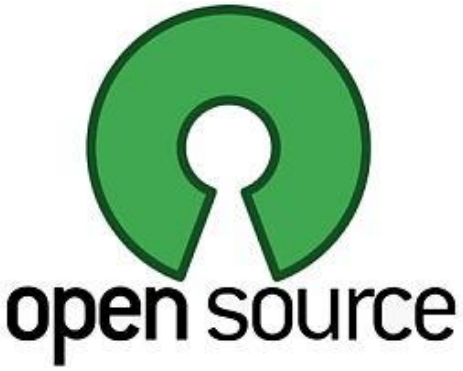
Community engagement

- Connect developers to security
- Train your developers
- Have an open door approach on emerging trends and technologies
- Set up an online collaboration platform

  E.g. Jive, Confluence etc.

Establish Secure Framework

- Secure frameworks:
  - Use a secure framework such as Spring Security, JAAS, Apache Shiro, Symfony2
  - ESAPI is a very useful OWASP security framework
- SCA (Service component architecture) tools that can provide security feedback on pre-commit stage.
  - Rapid response
  - Small chunks



Automation

- Integrate security within your build (Jenkins, Bamboo, TeamCity, etc.)
  - SAST
  - DAST
- Create a test framework to automate checks
- Constantly check code
- Fail the build if security does not pass the bar.

Use a mix of traditional and new

- Periodic penetration testing
- WAF (Web Application Firewall) on main functions
- Code review for security sensitive code portions.
- Documents everything

Open Source has its own pros and cons but with the emerging threat landscape and agile environment, it is the need of the hour.

Although, whether to choose from open source or proprietary software is entirely based on an organization's business needs – **Security, Scalability and Stability** of the tool/software should be analyzed with due diligence. This would ensure a secure environment.

Thank You

Contact Details –

Meetali Sharma
+91-9971393639
meetali.arora@sdgc.com, meetalisharma81@gmail.com
www.meetalisharma.com
https://www.linkedin.com/in/meetali-sharma/