VIEW POINT

DevSecOps

Speed with security: Setting up a new goal for DevOps teams.

Infosys®
Navigate your next

DevOps to DevSecOps

Stories of faster time to market and other organizational benefits accompany every DevOps implementation. DevOps helps achieve goals setup by Agile for faster time to market by automating engineering activities, removing manual interventions, improving quality and speed of releases. However, DevOps also falls short when security testing is required as part of application lifecycle. In such cases, like Agile-fall, the tail-end of DevOps also seems to be lagging and impacting speed of releases. Important and unavoidable security checks delay the release by days or weeks when security scans are done towards end-of-cycle.

DevSecOps which extends DevOps concepts to make security an integral part of DevOps, instead of keeping it at the tail-end of the release, is definitely a solution to speed up such application releases.

Challenges while adopting DevSecOps

Teams trying to adopt DevSecOps face more challenges than typical DevOps adoption,

- **People challenge:** Forming cohesive teams of Dev (and test) and Ops was already difficult for DevOps teams. Adding security experts as part of the team is an additional challenge since in most cases dedicated bandwidth of security experts is not available to be added as a part of DevOps team
- **Culture challenge:** Making Dev and Ops collaborate and break the barrier was a cultural challenge which DevOps teams tried to overcome. Today, security experts work in siloes and are fully responsible for security gaps and its impacts. It is definitely a challenge as how security SMEs can trust any security testing done by DevOps team and approve the release based on it
- **Technology challenge:** Just adding the security tools in CICD pipeline is not going to help implement DevSecOps. 1. Configuring security tools in pipeline so as to cover adequate testing and analyzing outcomes from these tools itself needs security expertise. 2. Not all security testing tools are ready-to-be-integrated in CICD pipelines and need custom code development to achieve automation

DevSecOps- the right way

To overcome these challenges, like DevOps, DevSecOps too needs to focus on people, processes, and technology to remove roadblocks and achieve speed.

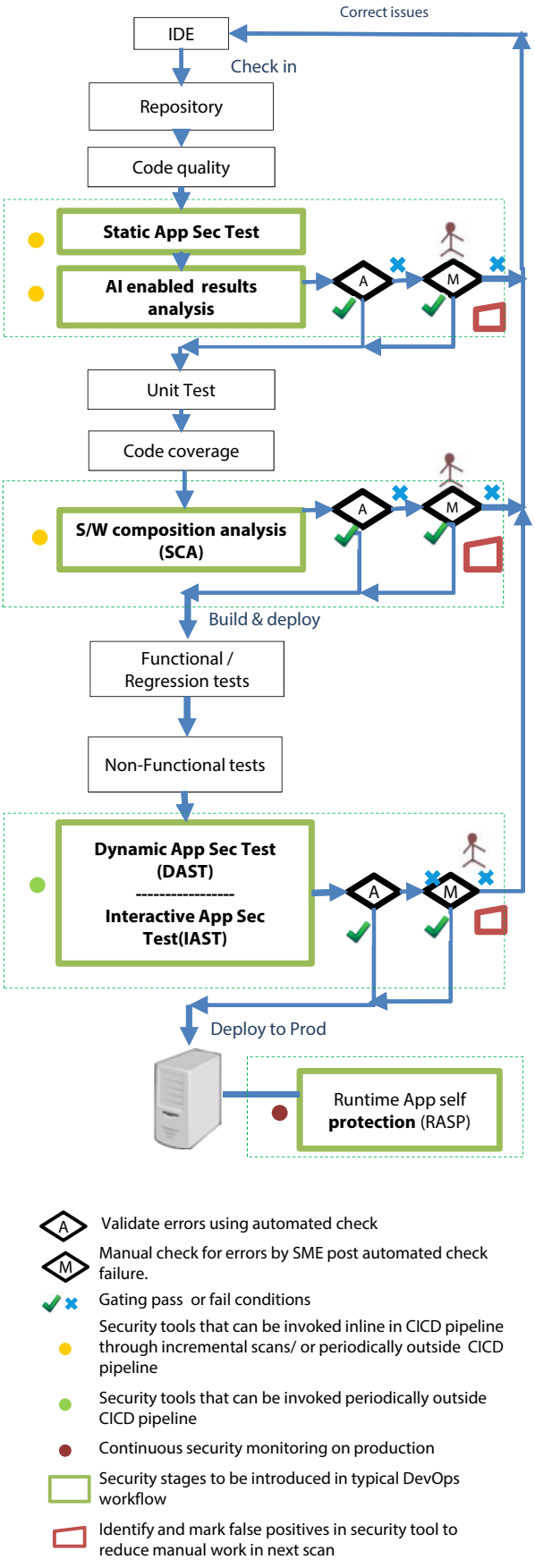
- **People:** Ever increasing demand on security SME bandwidth due to frequent releases must be reduced not only through automation but also with enablement of DevOps team on security aspects and increased collaboration between DevOps team and security SMEs
- **Process:** Starting from requirements to deployments, every stage of SDLC lifecycle must include security standards' compliance. Unless it is an integral part of process, security checks will always remain a post facto activity towards end of release leading to rework and delay
- **Technology:** Security testing tools and their integration in CI-CD pipeline is a must-have for DevSecOps success. Shifting left the security checks, using tools to cover all possible security tests and attempting as much no-touch automation as possible along by using AI capabilities will be important for DevSecOps success

Before drilling down into the above aspects, review the picture below which depicts the various security phases which can be introduced in typical DevOps workflow.

Types of security testing to be considered for DevSecOps

- Static Application Security Testing (SAST) – Detecting security issues in application code in build phase
- Software Composition Analysis (SCA) – Finding vulnerabilities of any open source software components in application
- Dynamic application security testing (DAST) – Identifying application security issues in its 'running' state
- Interactive Application Security Testing (IAST) - Finding potential security vulnerabilities from within the running application in real time through usage of software instrumentation
- Runtime application self-protection (RASP) – Monitoring and protecting running application from runtime attacks

Security stages within typical DevOps workflow



Technology transformation for DevSecOps

Automated security tests are repeatable, improve the speed of testing, reduce manual work and provide early feedback on application security. But, such automated security tests are unachievable unless security testing tools are used and those are configured correctly in the CI-CD pipeline. Key factors for technology transformation are

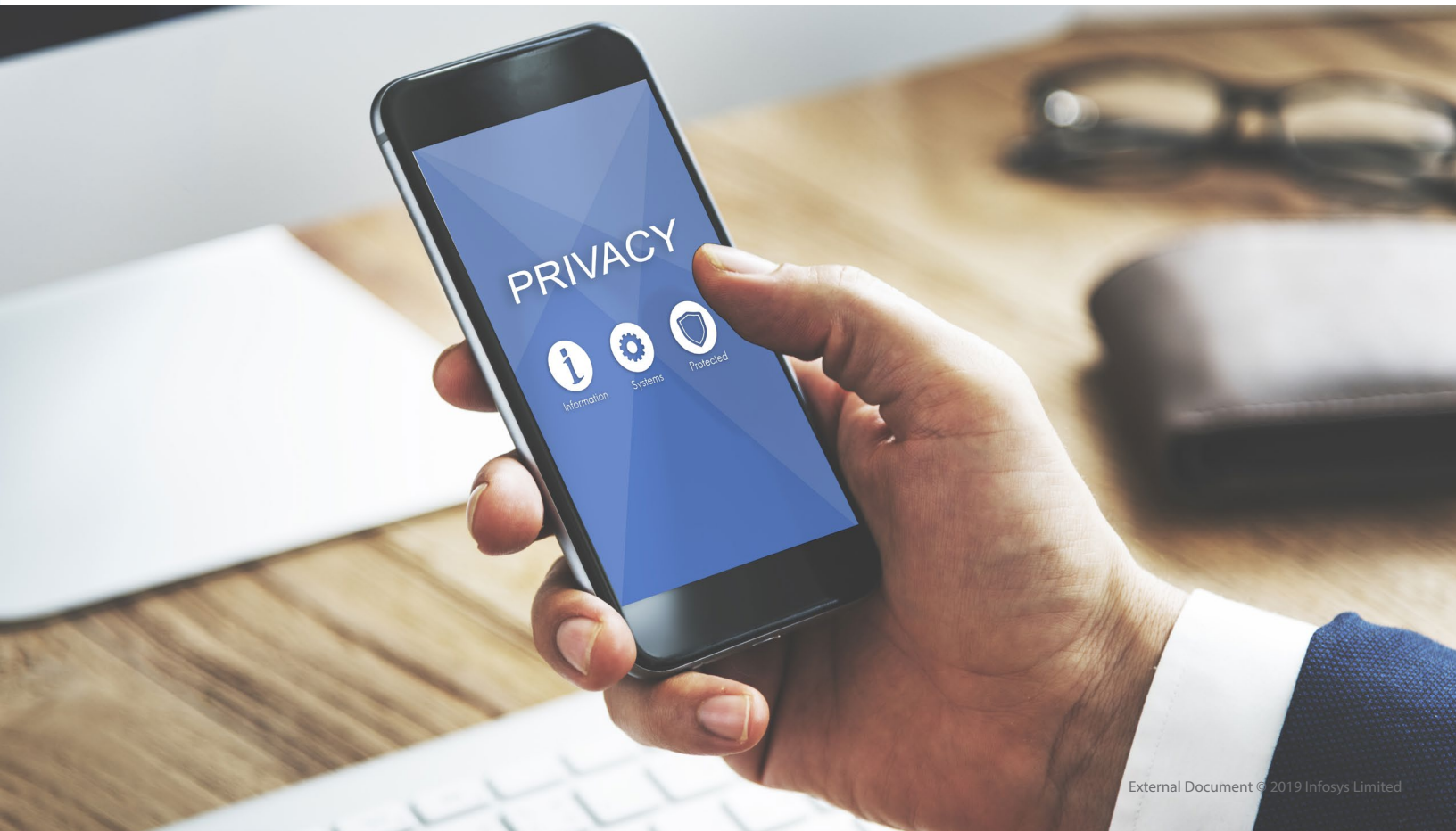
- **Shift left security**
  - » Including security checks early in SDLC rather than at the end of the release cycle
  - » Use correct security tools to achieve increased security coverage by choosing tools which can be integrated in DevOps pipelines, which can be automated to invoke the tool and perform gating of results with no-touch
  - » Configure tools optimally to suit business and application security needs rather than just using default rule sets provided by the tool
- **Integrate tools in CI-CD pipeline** and don't depend on teams to invoke tools as and when required. Integration with CI-CD pipeline ensures that security checks are not bypassed and are executed for every build
  - » SAST tool can be integrated in CI pipeline or with developer IDE if it is configured for incremental scan
  - » SCA tool can be in CI pipeline if it can take incremental scan else it can be setup for periodic run
  - » Integrate DAST and IAST tool in continuous delivery pipeline or alternately these can be setup for periodic run or parallel run based on time taken for scan
- **Follow DevOps principles** for DevSecOps tooling
  - » Aim for no touch automation so as to not miss any security checks
  - » Try built-in-quality through automation of gating of build against desired security standards and shift-left as much as possible
- **Optimize tool usage**
  - » Create own enhanced rule sets for tools to ensure optimum, faster scans and reliable outcomes from tools
  - » Plan incremental scans to reduce time taken for full scans
  - » Use artificial intelligence(AI) capability given by few vendors to reduce time spent on analyzing errors reported from tools

People aspect

Since there is not enough bandwidth to make security experts an integral part of every DevOps team; it is necessary to define effective collaboration for DevSecOps to succeed.

No collaboration – No high quality code: DevOps' built-in-quality principle expects that we add high quality code to build and it fails early if it has any issues. To build high quality code with no security issues, developers need understanding of security best practices. Today many developers do not have this awareness as security tests run as independent check at the end of cycle and they don't know against what their code is getting tested. If developers know the Dos and Don'ts of security practices, then high quality code will get added to build in first place with lesser defects and lesser rework. So collaborate. Let developers get enabled by security SMEs about security standards, security testing rules, commonly found errors, best practices. Let developers discuss the upcoming architecture, design changes with security experts upfront to validate that those are as per desired security standard

- **Tools too demand security SME time:** Only automation with security tools cannot solve the security testing challenge in DevSecOps. Security tools may be integrated in CI-CD pipeline but security SME knowledge is still required and here is why. Tools check application against a set of rules to find vulnerability however does the selected rules set cover desired security standards and compliance checks? Does the application under consideration needs any additional rules? Does a particular business or organization need any specific rules? All of these can be answered only by security SME and creates heavy dependency on them. By effective enablement of DevOps team and continuous collaboration this dependency can be reduced. Developers need to be enabled on security standards, rules setup in tools and analyzing outcomes from tools
- **Results from tools too demand security SME time:** Further even after the rules are appropriately configured in security tools; security testing tools may throw hundreds (literally) of security issues when the scan runs. Which of these issues are real issues and which are 'false positive' issues and hence can be ignored? A developer does not always understand what is permitted and what is not as per security standards. More the developers are enabled by security SMEs, lesser is the dependency on security experts. Further, usage of AI based tools will reduce the work required in removing 'false positives' manually
- **Responsibility issue:** Whose responsibility is security? In traditional world it is security experts' responsibility. Goals are set for security teams and impact of security failure are severe. Unless this is changed, security experts will never trust the scans performed by DevOps teams and signoff the release. They would always want to run their own scans and see the results themselves which adds delays. It is most important to make security as a responsibility of whole team and set it as team goal. With shared goal the collaboration between the two teams will be more effective and will result in improved cycle time.



Process changes

Traditionally teams which adopt agile methodology for incremental development still end up in Agile-Fall as security checks are performed on final deliverable. This results in late feedback and even costlier remediation. Teams trying to adopt DevSecOps should consider some of the below aspects.

- Treat security as requirement to be worked on in every phase of lifecycle and not at the end of release. Like any other nonfunctional requirements; security requirements must be noted at the very beginning of application and release life cycle
- Security requirements must be part of coding standards for developers so as to avoid introducing security errors
- Process must include check points on security testing. Similar to how coding and testing is part of 'definition of done'; security testing must also be included in DoD. Alternately security items can be placed in acceptance criteria. Aim is to include security as part of typical dev and test activities rather than totally ignoring it till the time it moves to security expert's bucket
- Create more intervention points for security SME to participate in agile ceremonies
- Build incrementally, test progressively and enhance feedback loops for security
- Depending on greenfield or legacy applications, security guidelines and considerations can be setup during onboarding or adopted gradually to improve security coverage
- Plan continuous improvement through time to time review of security aspects

Conclusion

DevSecOps is a must have for fast moving applications. Unless security is an integral part of the DevOps cycle it will result in delay of release by waiting for security experts to test and approve the release.

Adopting shift left policy for security, using tools for various types of security testing, integrating tools as part of CI-CD pipeline in no-touch manner, configuring tools for optimum usage can result in lesser time required for security testing every cycle.

But, adopting DevSecOps is not just adding security testing tools as part of CI-CD pipeline. Security expertise is required for using these tools effectively and that creates dependency. Security experts cannot be dedicated for all agile teams or for all their frequent releases. Effective enablement of DevOps team and continuous collaboration between DevOps teams and security teams can reduce dependency on security teams and improve build quality.

Making security as integral part of processes can reduce the delay in feedback loop and reduce rework; finally resulting in faster releases.

About the authors:

Anupama Rathi
Head of DevOps Practice, Infosys

Prasanna Ghanekar
Engineering Head, DevOps Practice, Infosys

Gayatri Bhat
Lead, DevOps Practice, Infosys

Request copy of complete Infosys DevSecOps handbook here: DevOpsPractice@infosys.com

For more information, contact askus@infosys.com

Infosys®
Navigate your next