Capgemini engineering

DEVSECOPS IN REGULATED INDUSTRIES

ACCELERATING SOFTWARE RELIABILITY & COMPLIANCE

TABLE OF CONTENTS

EXECUTIVE SUMMARY

DevOps practices enable rapid product engineering delivery and operations, particularly by agile teams using lean practices. There is an evolution from DevOps to DevSecOps, which is at the intersection of development, operations, and security. Security cannot be added after product development is complete and security testing cannot be done as a once-per-release cycle activity. Shifting security Left implies integration of security at all stages of the Software Development Life Cycle (SDLC). Adoption of DevSecOps practices enables faster, more reliable and more secure software.

While DevSecOps emerged from Internet and software companies, it can benefit other industries, including regulated and high security environments. This whitepaper covers how incorporating DevSecOps in regulated Industries can accelerate software delivery, reducing the time from code change to production deployment or release while reducing security risks.

This whitepaper defines a playbook for DevSecOps goals, addresses challenges, and discusses evolving workflows in DevSecOps, including cloud, agile, application modernization and digital transformation. Bi-directional requirement traceability, document generation and security tests should be part of the CI/CD pipeline. Regulated industries can securely move away from traditional V cycle software development to a more agile approach by taking advantage of DevSecOps.

INTRODUCTION

The DevOps movement has its origins in the seminal talk given by John Allspaw and Paul Hammond of Flickr at the O'Reilly Velocity 2009 conference.[1] They described how they were able to perform ten deployments per day and their approach became popular as "DevOps." DevOps can be described as a "business driven approach to deliver solutions using agile methods, collaboration, and automation."

DevOps fosters the integration of development and operations and creates a collaborative professional culture that breaks down traditional business process bottlenecks. As per a McKinsey report, the adoption of DevOps practices leads to rapid product engineering delivery and operations, particularly by agile teams using Lean practices.[2]

As per Accelerate State of DevOps Report, 2019 adoption of DevOps practices create elite, integrated teams and leads to improvement in productivity, e.g., faster time from code change to production, and can improve time to market for new products or features.[3] It also leads to faster recovery from production incidents and fewer occurrences of such incidents. The result is an increase in velocity that can have a quantifiable and positive business impact.

[1.] https://www.youtube.com/watch?v=LdOe18KhtT4

[2.] https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/Beyond%20agile%20reorganizing%20IT%20for%20faster%20software%20delivery/Beyond%20agile%20Reorganizing%20IT%20for%20 faster%20software%20delivery.pdf

[3.] https://cloud.google.com/devops/state-of-devops/

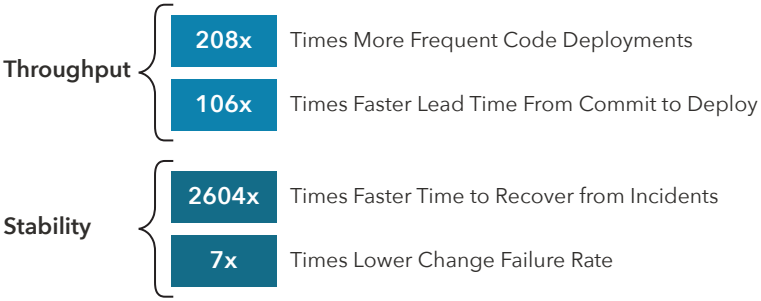

Figure 1: Elite Performers in Comparison with Low Performer

Nearly a decade ago, famous venture capitalist Marc Andreessen had stated that "Software is Eating the World," meaning that all types of businesses will ultimately be digital enterprises.[4] This trend is now visible in all sectors from fighter jets to military drones, medical equipment like Magnetic Resonance Imaging (MRI) scanners, energy control systems, and modern automobiles. A modern car can be considered a computer on wheels with 100 million code lines. In the future, self-driving cars are expected to run 300 million code lines.

The evolution from DevOps to DevSecOps is at the intersection of development, operations and security. Cloud and mobile technologies have created a hyper-connected world that is increasingly vulnerable to cybersecurity attacks. In practice, security cannot be added after product development is complete and security testing cannot be done on a once-per-release cycle basis. Folding security into the DevOps ethos of incrementally improving software in smaller, faster builds helps fix defects and security issues much earlier in the release cycle. DevSecOps addresses the critical areas of making reliable software, maintaining system integrity and enabling efficient incident management, governance and compliance.

[4.] https://a16z.com/2011/08/20/why-software-is-eating-the-world/

Regulated industries from automotive energy, life sciences, and industrial manufacturing can successfully adopt DevSecOps best practices and strategies. While DevOps and DevSecOps practices emerged from Internet and software companies, they can be used in other industries, including regulated industries and high security environments, such as aerospace and defense, government and military.

IMPEDIMENTS TO DEVSECOPS ADOPTION

Regulated industries have been slow in the adoption of agile, DevOps, and DevSecOps practices due to their unique, industry specific challenges.

Safety and Security Compliance

Regulated Industries have to meet complex compliance requirements with multiple regulations and standards. This is the biggest challenge, directly affecting the product release velocity and productivity.

Based on a survey done by DZone across a global audience of developers, architects, and software engineers, the perceived factors impacting the organization's security assurance approach include regulatory requirements, customer requirements, including security requirements and security awareness across the organization.[5]

Organization Security Assurance Factors

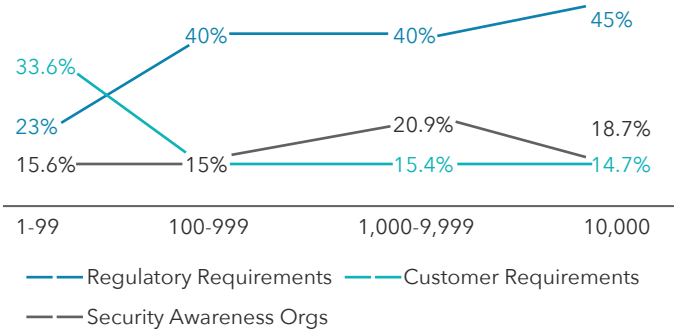

Figure 2: Organization Security Assurance Factors

[5.] https://dzone.com/trendreports/devsecops-1

Regulatory requirements top the list. Regardless of organization size (100+ to 10,000+), the focus on regulatory requirements (37.2%) is higher; however, the security awareness focus still remains low (17.5%).

Aerospace and Defense companies, in particular, need to comply with DO-178C, ED-12B safety standards. Similarly, Automotive companies need to comply with safety standards like Motor Industry Software Reliability Association (MISRA) and ISO 26262. Life sciences companies in the United States need to comply with the US Food and Drug Administration (FDA) Code of Federal Regulations (CFR) and other regulatory standards like IEC 60601, ISO 13485, and ISO 14791.

Any compromise in safety standards can lead to drastic consequences with respect to regulatory bodies or customers. For example, Boeing tried to circumvent the problems in the airframe of 737 MAX 8 by relying on Maneuvering Characteristics Augmentation System (MCAS) software. However, the 737 MAX 8 product suffered two major air crashes in the recent past.[6] New technologies or practices that are adopted must not only make safety and security top business priorities but must also incorporate end to end security and regulatory requirements along with traceability.

Keeping software and systems secure is the highest priority for any organization. As companies connect enormous numbers of devices and develop ever more complex data structures, cybersecurity becomes increasingly important.

The proliferation of mobile and cloud computing technologies has expanded cybersecurity attack surfaces and has increased security risks. By 2022, the mobile attack surface width for attackers will be 6 billion. Further, the use of third party and opensource software requires that these components are not vulnerable.

Traditional Development Practices

Across industries, software complexity is increasing exponentially, while development productivity is stagnating. McKinsey expects that by 2030 software complexity in the automotive sector will increase 5x, while software development productivity will only be 2x.[7]

[6.] https://www.bbc.com/news/business-50177788
[7.] https://www.mckinsey.com/industries/automotive-and-assembly/ourinsights/the-case-for-an-end-to-end-automotive-software-platform

The aerospace and automotive industries typically use Model Based Software Engineering in V-cycle SDLC. V-cycle provides simplicity and ease of use but is not well suited for agile processes. It works well for small projects, but there is an increased risk that defects will not be found early sufficient for massive projects. Thus V-cycle is inflexible and encourages a rigid and linear view of software development similar to a legacy waterfall SDLC. The long development cycles and low level of test automation make the process less flexible in the face of change, setting aside the implicit, slower release cadence.

Similarly, in life sciences, there is a new desire to leverage advanced analytics, automation, and the cloud to increase productivity and improve the quality of decision making.

Platform Dependency

The same aircraft, vehicles, or medical devices need to be supported for many years, leading to a huge legacy codebase and complex branching strategy. Lack of visibility into real world production environments leads to development environments being inconsistent with runtime environments. Moreover, there are resource constraints with less room to compensate for hardware (CPU or memory) variations. Reducing hardware dependencies, accelerating test automation and providing production like test environments repeatedly and reliably, without compromising security, is difficult, if not impossible, to achieve without adopting a DevSecOps approach.

PLAYBOOK FOR DEVSECOPS ADOPTION

For an organization to successfully introduce and adopt DevSecOps practices, it is important to be aware of the challenges and have a playbook of potential solutions for various scenarios. There is a set of proven, repeatable plays for adopting DevSecOps at enterprise scale in regulated industries.

Agile Workflow with W-Model

Adoption of agile planning and development processes is the first step in the road to DevSecOps. DevSecOps can be done with waterfall or V-cycle but will not deliver full benefits under a V-cycle approach.[8] Moving from a traditional V-model to a hybrid W-model, which embodies the Agile spirit of working on small increments of requirements, is a key enabler for DevSecOps in regulated industries.

[8.] https://en.wikipedia.org/wiki/V-Model

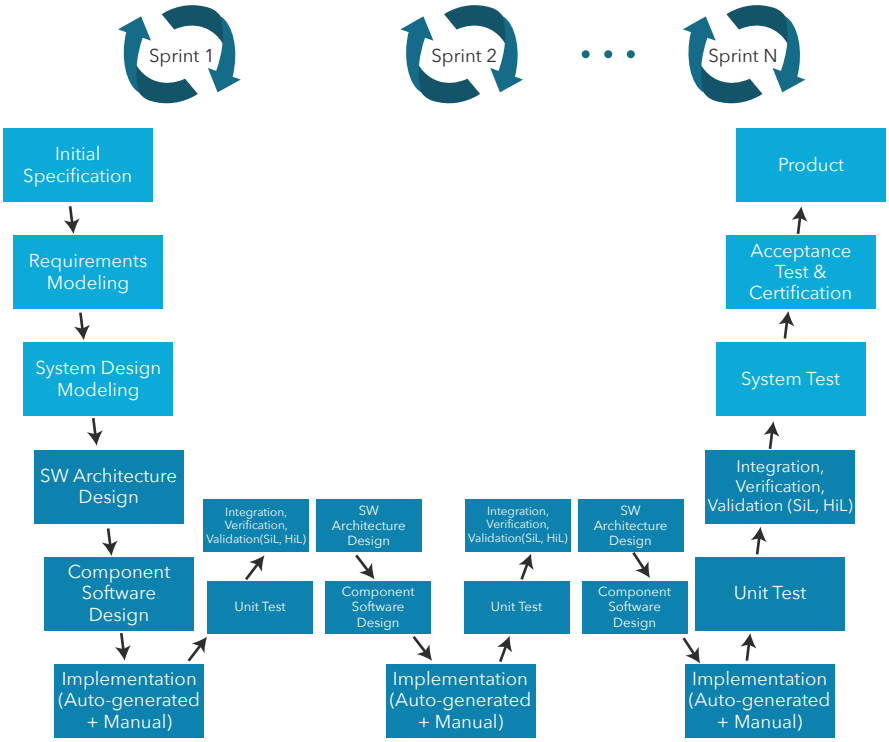

Figure 3: W-Model for Agile Workflow

Backlog planning can be done using traditional tools like Rational Dynamic Object Oriented Requirements System (DOORS) or using modern Agile project management tools, such as Jira. Along with user stories for features, non-functional safety and security requirements can be managed and tracked using agile project management tools. Instead of having longer iterations following a V-model SDLC, a DevSecOps approach enables organizations to plan shorter iterations, i.e., agile sprints, and to build software incrementally, including automated safety, security and compliance verification.

Initial sprints in a DevSecOps approach focus on requirement modeling and system design modeling which are iteratively refined based on feedback from automated tests. In each sprint, implementation of particular components or features includes component design, implementation, unit and integration testing, security and compliance verification.

Continuous Integration Enabled Version Control System

Legacy version control systems like Clearcase are not well suited to Continuous Integration/Continuous Delivery (CI/CD) pipelines and have limited integration capabilities with respect to third party tools. Migrating to modern version control systems like Git or Bitbucket enables integrating with modern CI/CD pipelines. Using DevSecOps tools, code reviews can seamlessly integrate with version control. Code that is autogenerated from requirements or design models can be compiled and stored in or retrieved from artifact repositories using simple Representational State Transfer (REST) APIs.

The long duration of support for different platforms leads to legacy codebases and complicated branching models. Delays in merging code changes from a feature or release branch to main or version-specific branches can lead to frequent merge conflicts resulting in unnecessary effort to resolve the conflicts. Branching strategies that leverage modern source control systems reduce merge conflicts and save time.

For example, in a typical DevSecOps approach, a GitFlow strategy is used to prevent conflicts in the master branch.[9] Feature or release branches are frequently merged back into the master branch, which is used to build release artifacts. GitFlow branching model defines a central master branch and a parallel develop branch representing the latest code changes, used for nightly or weekly builds that serve as gates for building promotion and release. After stabilization, the development branch's changes are merged into the release branch and tagged with a release number. The develop branch for the next version is then forked from the release branch.

Continuous Testing

Automated testing across different CI/CD pipeline stages is crucial to obtain immediate feedback on the risks associated with a software release candidate. Since automated testing is executed continuously, safety, security and compliance risks can be mitigated from progressing to the next stage. Security and compliance requirements are integrated directly into the build promotion criteria. Different stages of automated verification lead to better quality, test coverage, faster release cadence and provide traceability to safety, security and compliance requirements:

[9.] https://nvie.com/posts/a-successful-git-branching-model/

Static Code Analysis: Static code analysis reveals potential vulnerabilities such as null pointer problems, buffer overflows, memory leaks, division by zero, etc. Compliance with standards like MISRA and DO 178C can be verified as part of static analysis. In a DevSecOps approach, automated scripts are used to run static analysis from the CI orchestrator based on a trigger from the source control system, e.g., the result of a merge. If threshold levels are set for the number and severity of warnings, the pipeline can be halted if the threshold is reached.

Software Composition Analysis: Increased use of third-party and opensource software requires a thorough assessment of any associated vulnerabilities. Software Composition Analysis (SCA) tools evaluate third-party and opensource components to create a package bill of materials, which provide insights into third-party software and show vulnerable areas of code along with risk severity levels.

Unit Testing: A CI orchestrator can be used to automatically trigger unit tests when source code changes are committed to verifying that the individual components perform as designed, including verification of safety, security and compliance requirements at the component level. Stubbing or mocking can be used to simulate complex scenarios in virtualized environments independent of the target hardware platform.

Code Coverage: For regulated software, the aim should be to get 100% code coverage since there are often mission-critical safety and security concerns. Test cases can be correlated with code coverage report to assess the completeness of testing. Advanced code coverage reporting at the unit test level supports KPIs like line, function, statement, MC/DC and other empirical measures of code quality, security and compliance.

Integration and Functional Testing: Automated integration and functional testing verify the integration of different units of a module within the flow of CI/CD pipeline, along with upwards or downwards compatibility of components, e.g., at the API level. Test plans can be automatically generated from a system model and can be further augmented by additional test plans or AI/ML based test case selection and prioritization. Virtual prototypes and simulations like software-in-loop and hardware-in-loop can also be used to verify CI/CD pipeline flows.

Continuous Security Testing

Integrating security into DevOps can speed up software releases without compromising controls or increasing risk. Continuous security testing verifies that systems and applications are analyzed for vulnerabilities in a continuous cycle.

A DevSecOps approach integrates safety, security and compliance verification at appropriate points in the SDLC. Security requirements and best practices are factored into all elements of product development, from the code itself to the infrastructure it runs on.

01 Threat Modeling

Risk based software development approach to uncover external risks and identify potential vulnerabilities

02 Secure Coding Guidelines

Use industry coding standards like CWE, OWASP, CERT, MISRA for complying with established best practices

03 Software Composition Analysis

Identify all the open source in codebase and map that inventory to a list of current known vulnerabilities

04 Static Analysis

Scan source code to reveal potential vulnerabilities and ensure compliance with standards like DO 178B/C, MISRA

05 Vulnerability Testing

Identify potential vulnerabilities to evaluate the quantum of risk

06 Penetration Testing

Simulated attack to identify exploitable weaknesses as well as strengths, enabling a full risk assessment to be completed

Figure 4: Automated Security Checks in CI/CD Pipeline

Threat Modeling: In the planning phase, what security issues should be addressed in the next few sprints should be decided. Secure by design principles in the design phase uses threat modeling, data flow diagrams, and trust boundaries to uncover risks and identify potential vulnerabilities. Threat modeling can be done using techniques like Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of privilege (STRIDE), and is used to document key assets and risks thoroughly and systematically.

Secure Coding Guidelines: Select tools can integrate within the development environment, IDE and help in complying with different coding conventions prescribed by industry forums like OWASP, CWE, CERT, MISRA.

Software Composition Analysis (SCA): SCA can be used to enforce the organization's opensource software policies and to create a list of third-party software, e.g., libraries or SDKs, which must be monitored for vulnerabilities. For example, an Apache Struts vulnerability was responsible for the Equifax data breach in 2017.

Static Analysis: IDEs can be configured with static analysis tools that can immediately run during compilation to discover potential vulnerabilities and error-prone codes.

Vulnerability Testing: During the integration testing phase, vulnerability testing can find potential vulnerabilities and estimate the magnitude of risk.

Penetration Testing: During functional testing, attacks can be simulated to identify exploitable weaknesses and understand strengths better, facilitating comprehensive risk assessments.

Integrating security checks throughout CI/CD pipeline helps in fixing flaws as you write code. DevSecOps follows a fail fast principle so that, at any stage of the CI/CD pipeline, testing, or build promotion process, various kinds of security vulnerabilities can be found and fixed before vulnerabilities appear in a release.

Requirements Traceability

Bi-directional requirements traceability is essential for safety, security and compliance. Requirements traceability describes the life of a requirement both forwards and backward. Bi-directional requirements traceability ensures coverage of requirements in design and code. Similarly, there is coverage of design and code in the requirements.

While traditionally it is either done manually, e.g., using Microsoft Excel spreadsheets, or using tools like DOORS, in a DevSecOps environment, requirements traceability can be automated such that all stakeholders can view and collaborate on changes across the SDLC.

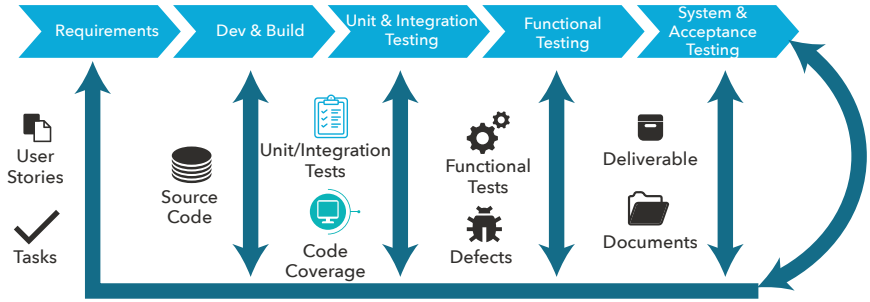

Figure 5: Bi-directional Requirement Traceability

Requirements are maintained in agile backlog as user stories and tasks. While creating system design and component design, links object models, data models and sequence diagrams to user stories and requirements.

Automatic code generators provide the ability to trace back directly to the individual parts of the design model from which the code was generated. Source code and unit test code can contain user story IDs or tags and references to design models. Automated scripts can link user story to code changes on code commit.

Test automation tools can be integrated with a common test asset management tool or repository where all test scenarios, test scripts and associated results can be stored so that traceability can be established for code changes, requirements and defects.

After deploying to production, defect IDs can be mapped back to requirements in case of a defect or incident. The generated documentation can contain a reference to the requirements and design sections.

Documentation Generation

Rather than creating comprehensive documentation up-front, incremental documentation can be generated in every sprint and documentation tasks can be automated in the CI/CD pipeline. Inputs can include requirements models, system models, component design models, test plans, or source code. Document generation tools like Doxygen can extract comments from source code to create documentation. In this manner, documentation remains consistent with source code even when high velocity code changes are made. Additional documentation can always be created manually.

Compliance with DO-178C / ED-12C

DevSecOps can help in compliance with regulatory standards, such as the DO 178C or ED 12C safety standards in the Aerospace industry. Compliance as code brings team, auditors, project management office on the same table, and the development and operations team.

Manufacturing processes contain policies to ensure that end products exhibit the required quality and security characteristics. The software should be developed in accordance with policies that clearly state quality, safety, security and compliance goals. System and component architectures should be evaluated automatically for compliance with policies.

Test automation and source code comments can be used to enforce bi-directional requirement traceability. Performing data flow analysis, static analysis, unit testing and peer review ensures that source code complies with specifications and requirements. Automated test cases can be used to check compliance and test cases can also be used to demonstrate safety, security and compliance to auditors. Performing extensive code coverage ensures that requirements have corresponding implementations and helps to map code to requirements.

Requirements, architecture, design and software components should be verified using simulations and prototypes wherever possible. Simulations like requirement-in-loop, model-in-loop, software-in-loop and hardware-in-loop accelerate verification and validation for safety, security and compliance implementations.

Continuous Delivery Pipeline

A central Continuous Delivery pipeline enables software deployment along with continuous security verification through automates testing.

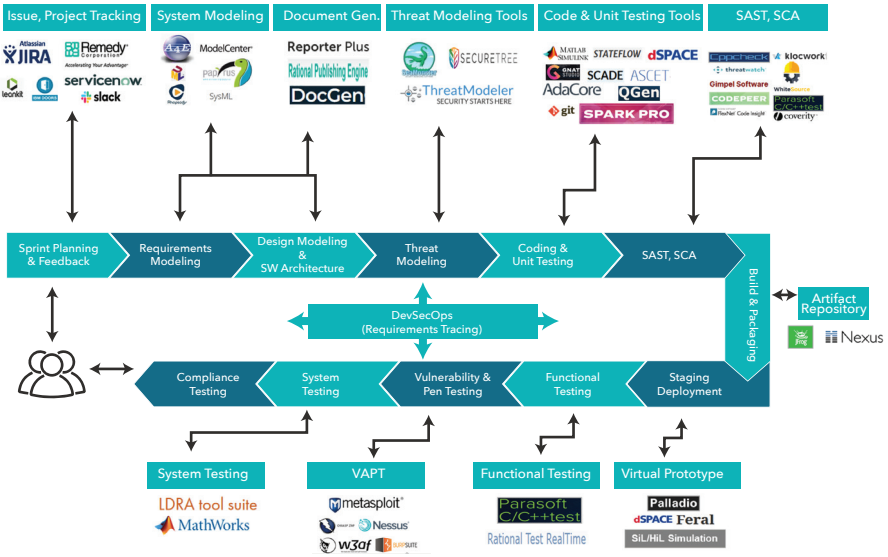

Figure 6: DevSecOps CI/CD Pipeline

The pipeline shown above is enabled by a CI/CD orchestrator like Jenkins on code commit. After performing automated unit testing, generated binaries are versioned, tagged and stored in an artifact repository, such as JFrog Artifactory.

Deploying binaries in test or staging environments leveraging simulations like software-in-loop, hardware-in-loop accelerates automated functional testing. Vulnerability testing and Penetration testing are then followed by system testing to verify end-to-end test scenarios.

DevSecOps implies not only shift-left of testing but also shift-right of feedback. After performing different testing types, any defects, suggestions, or enhancements can be automatically recorded in tools like Jira, e.g., stories or tasks can be created automatically based on test results.

CONCLUSION

Regulated industries like Aerospace, Defense, and Automotive, typically follow traditional development practices like V cycle with manual testing and low software reusability. Hardware dependencies and long support contracts have led to a huge legacy codebase with siloed software development teams. Regulated software requires compliance to safety standards like DO-178B/C, ED-12B, MISRA, ISO 26262 and security standards like DO-326A, ED-202A, SAE J3061, etc.

Adoption of DevSecOps practices in regulated industries can accelerate software delivery, reducing the time from code change to production deployment or release while reducing security risks. Rigorous, automated security testing, which is key to adopting DevSecOps, can also validate compliance requirements. Bi-directional requirement traceability, document generation and security tests can be done as part of the CI/CD pipeline.

With this in mind, Capgemini Engineering has been developing a range of software frameworks that can help clients adopt and improve DevSecOps implementation by performing a number of activities related to CI/CD pipelines: automated testing, simulation, safety, security and compliance verification. Combined with our managed DevSecOps platform, we can free up clients' resources and bandwidth from routine tasks to focus instead on creating high value features and new, innovative capabilities.

Capgemini engineering

About Capgemini

Capgemini is a global leader in partnering with companies to transform and manage their business by harnessing the power of technology. The Group is guided everyday by its purpose of unleashing human energy through technology for an inclusive and sustainable future. It is a responsible and diverse organization of 270,000 team members in nearly 50 countries. With its strong 50 year heritage and deep industry expertise, Capgemini is trusted by its clients to address the entire breadth of their business needs, from strategy and design to operations, fueled by the fast evolving and innovative world of cloud, data, AI, connectivity, software, digital engineering and platforms. The Group reported in 2020 global revenues of €16 billion.

Learn more about us at

www.capgemini.com

For more details contact:

engineering@capgemini.com

People matter, results count.

Marketing_24/03/2021_Shivani Sood