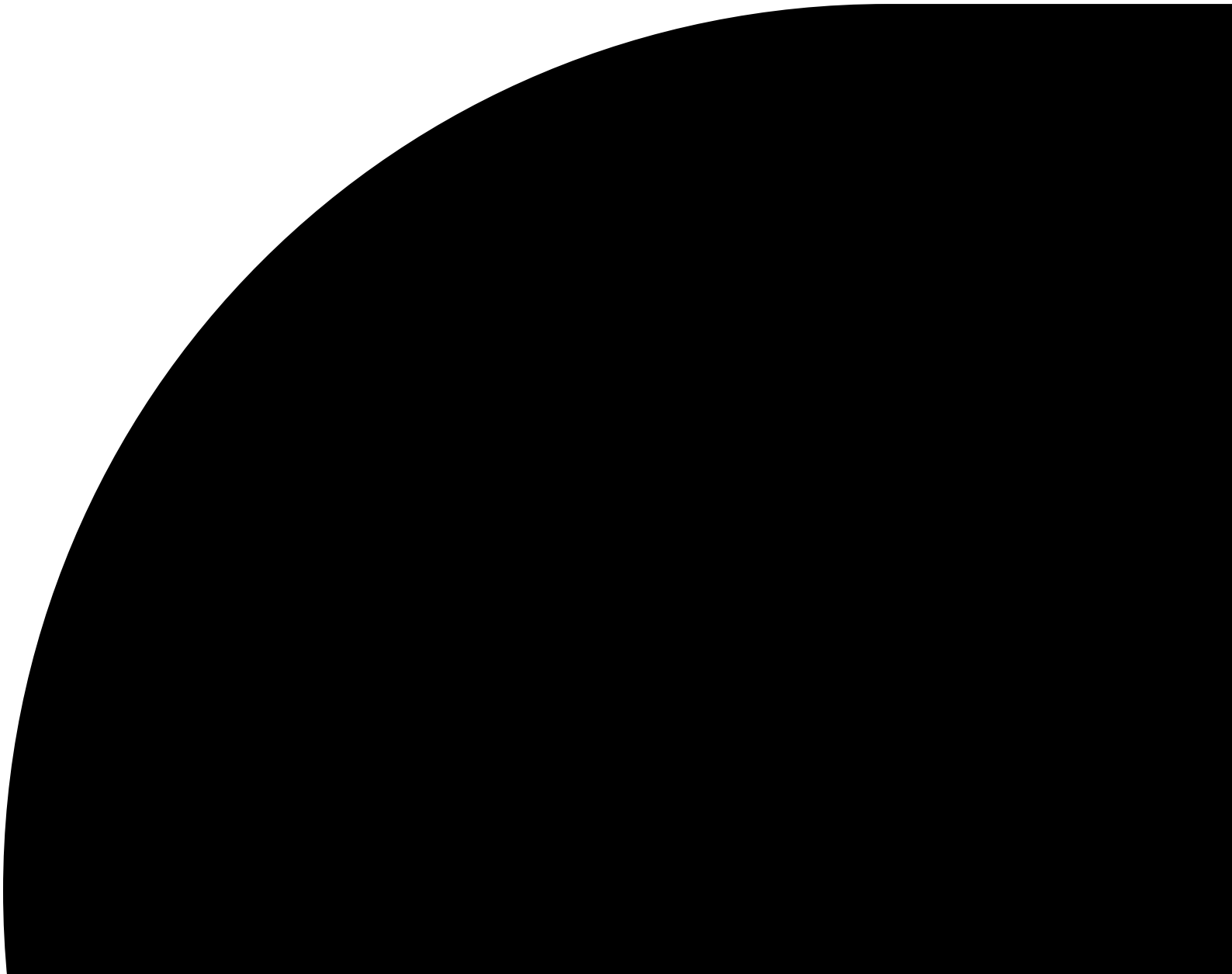
# DevSecOps: A comprehensive approach to secure applications



Enterprises have shifted their attitudes about security in recent years, recognizing security as core to business success and not just a necessary evil for IT. The most prominent factor contributing to this shift toward application and systems security is the widening adoption of cloud and digital services, which have eroded the traditional corporate network boundary. Data processing and storage now span internal and external resources.

This move to the cloud has led to increasing legislation about data privacy, since cyberattacks often exploit the distributed nature of these environments. "Privacy by design" and "latest state of the art" are phrases that now appear regularly in legislation, meaning organizations must pay specific attention to security and privacy in design and maintain currency in controls, countermeasures and approaches. Legislation and protection of citizen rights are paramount in many regions of the world, and penalties for noncompliance can be stiff.

Organizations also recognize the scope of new threats. Attackers are often organized criminal networks or hostile foreign powers, and the sophistication of the attacks can be staggering. The visibility and costs of security breaches warrant board-level attention.

At the same time, enterprises are increasingly recognizing security as not just a top issue but also a key value stream to safeguard digital business. As companies further digitize their operations, the risk of security breaches presents a real danger of disruption and destruction of business value and intellectual property.

To be effective Dev, Ops and Sec must be more than just peer partners. Dev and Ops must be wrapped in Security.

As systems become more diverse and dispersed, opportunities for disruption increase. The response to these security challenges must be one of continual vigilance and countermeasures. Security is never finished.

This paper details the role security must play in the development of applications and systems, and key capabilities and concepts that can help an organization ensure security is a natural outcome of the development process, not an afterthought.

To begin, let's look at the relationship between development, operations and security.

## The Dev-Sec-Ops triumvirate

A natural tension exists between application development groups and business operations. Traditional application development teams are focused on completing applications quickly to deliver value as soon as possible. Operations teams are concerned with matters such as protecting the live system estate from disruption at all costs. In an environment where these two units operate separately, releasing a

**DXC Labs | Security**

DXC Labs delivers thought leadership and technology prototypes to enable enterprises to thrive in the digital age.

DXC Labs | Security brings together our world-class advisors to develop strategic and architectural insights to reduce digital risk. DXC's Cyber Reference Architecture is at the heart of our research, providing clients with detailed guidance on methods to efficiently resolve the most challenging security problems. We help clients minimize risk while taking maximum advantage of the digital commons.

Learn more at **www.dxc.technology/securitylabs**

new application or major upgrade has traditionally been a large-scale and infrequent activity that restricts the flow of business value along with the IT capabilities.

DevOps overcomes the historic boundary between development of a system and the handover and acceptance into operations. DevOps focuses on the rapid release of small updates that can be easily deployed. This improves workflow and delivers value with lower operational risk. But it does not resolve concerns about security.

To be effective Dev, Ops and Sec must be more than just peer partners. Dev and Ops must be wrapped in Security (see **Figure 1**). This is more than a matter of token steps, such as automating security testing. It means taking a deeper look at the underlying areas of concern. In some ways we must return to first principles: What are the wider drivers for some of these changes? How does DevOps as a concept relate to security?

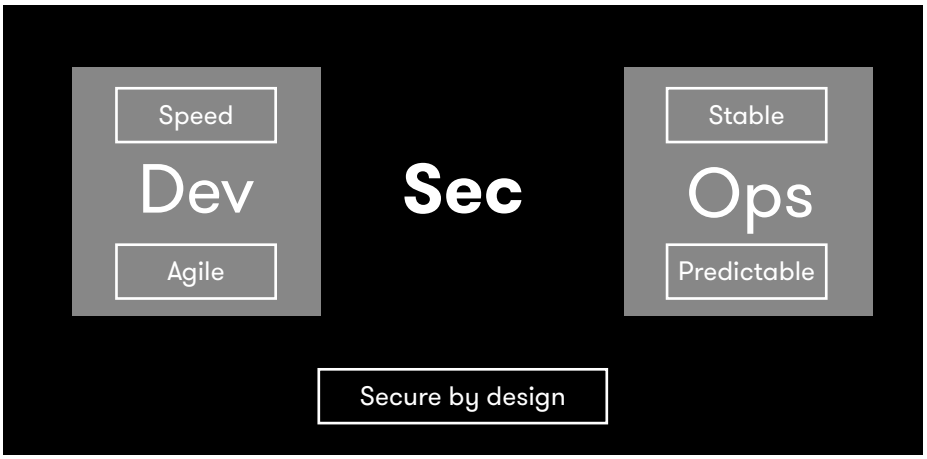



**Figure 1.** DevOps helps resolve the need for speed versus stability

## DevOps in application development

In certain spheres of application development, DevOps is a mature discipline with continuous delivery pipelines that propagate changes through development to live system operations in a seamless manner. The maturity, however, is linked to the type of system. One might accept this pace for systems of engagement, i.e., a customer banking application, but may not for systems of record like a core financial ledger application.

As DevOps widens its scope of applicability, we must consider wider system design aspects such as performance, scalability and, of course, security. Security is perhaps the least interesting for most application developers, who often just see one dimension of security (authentication and authorization) and not the range of controls and countermeasures that collaborate to keep systems and data secure from attack and compromise.

As we seek to introduce security controls, we must ensure they are devised so as not to be seen as an obstruction to the DevOps pipeline. Such obstructions are likely to result in developers avoiding security concerns, with consequent risk to the business. Security must be an enabler, not an obstacle.

## Cloud and everything as code

Traditionally, security concerns have been addressed by infrastructure controls and countermeasures. Applications could operate, safe in the knowledge that controls were in place. Indeed, many advanced IT organizations provided security functions in common middleware for developers to ease consumption. With the rise of cloud computing and the ability to rapidly provision new runtime services and application services, the security boundary has shifted to a virtual boundary concept. Security controls are often complex, applied at abstract levels and overlapping to provide comprehensive protection.

Security is a systems engineering concept that must now be comprehended by, and featured in, application development in its wider context. Application development teams can no longer rely on security features to be available and faultlessly configured in runtime platforms. Moreover, platforms and services are now configured as code and frequently controlled by policy, whether at the network layer or at the operating system, hypervisor or container layer.

Security is a systems engineering concept that must now be comprehended by, and featured in, application development in its wider context.

With the opening of external platform services and the increasing use of open source code, there is an inherent risk of insertion of malware and an increased responsibility on applications development to ensure security concerns are addressed. Given these shifting boundaries, it may be unclear who is responsible for security of the cloud platform. This concern needs to be addressed along with the wider issues exposed by deployment of internet of things technologies and connected devices.

Then there is the ever-growing range of automated processes possible with cloud platforms. While these may offer many business benefits, the exploitation of automation and automated activities can give a false sense of security. Security needs to be just as much a concern in systems managed by artificial intelligence and machine learning as those without. Each of these new and exciting capabilities changes the way systems are constructed, but it does not eliminate the fact that they must be secured (see **Figure 2**).

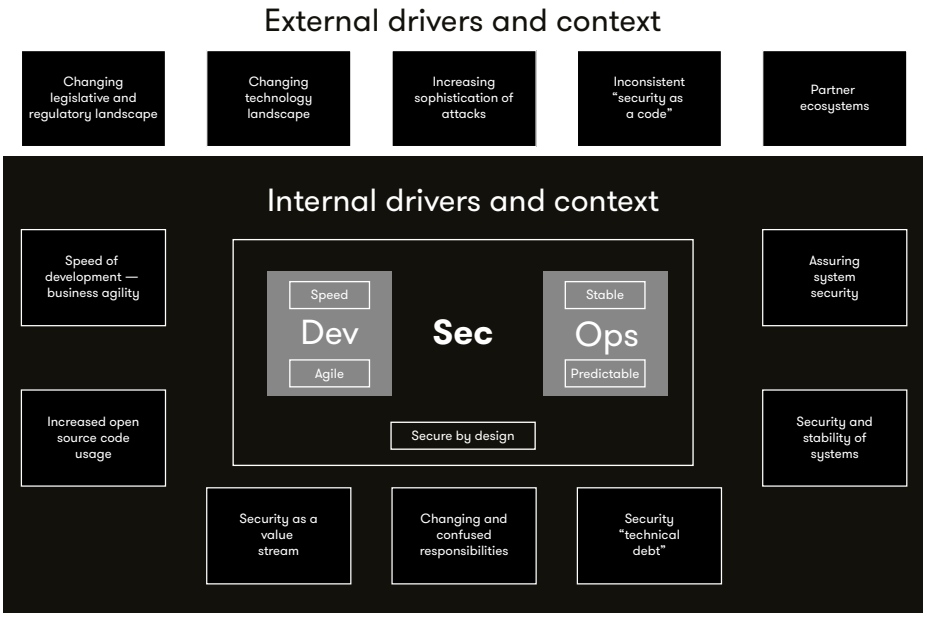



**Figure 2.** Drivers shaping the DevSecOps environment

## “Sec” defined

Historically, security management within the boundaries of the corporate network has been complex but well understood and relatively stable. Security risks are assessed, the data and applications classified, and controls and countermeasures devised to reduce the risks to an acceptable level and cost.

This level of security management is often the realm of early systems design and infrastructure implementation. Application development is frequently a consumer of the platforms that are a product of this activity. Security concerns may include features such as:

- Identity mechanisms, including authentication, authorization and audit
- Data privacy, confidentiality and integrity — e.g., encryption at rest and in transit and digital signatures
- Service availability
- Network security for protection against distributed denial of service (DDoS) attacks and boundary penetration, and control of point-to-point connections
- Infrastructure security such as control of server and storage security, delegation of administration privileges, divisions of responsibility, and database and middleware security
- Applications security, including safeguarding source code, using identity and access management (IDAM) services, and ensuring good warning, diagnostic and failure design
- Business security — i.e., trust in staff, auditing checks
- Data loss prevention through import and export controls
- Physical and environmental security using access controls, building safety and security, physical security policies, etc.

In traditional boundary-controlled systems, the careful construction of overlaying sets of these controls and countermeasures is often assumed by application development. However, as we embrace more rapid development and delivery techniques the security response can be diluted or dispersed, especially if traditional Ops boundaries are overridden by DevOps and the velocity of continuous integration and continuous delivery (CI/CD).

Some mature IT systems already have a defense-in-depth concept with varying zones of trust, where differing degrees of data classification are managed within the same corporate network. Where this is the case, extending beyond the corporate network may be simpler than in the case where the outer network boundary is the major control point, because the skills and mechanisms to devise these controls already exist in the organization.

## DevSecOps in context: Security by design

To give you a more complete picture of the end-to-end role of security in DevOps, let's consider some of the key concepts, beginning with the precepts of security by design.

"Security by design" implies a continual and diligent level of attention to security concerns. This means that security must be embedded as a core discipline in the development of any IT system. Designers and developers all need to consider elements of security by design, including:

"Security by design" implies a continual and diligent level of attention to security concerns. This means that security must be embedded as a core discipline in the development of any IT system.

- Data classification and domains of trust
- Data encryption by default at rest and in transit, and a strength of encryption commensurate with the data security classification
- Data tokenization to reduce privacy exposures
- Micro service architectures (OAuth) to improve security in depth and segment security concerns
- Runtime container security hardening and policies
- Delivery pipeline automated policy checking and testing regimes
- Internal and external gateway and interfaces, including compliance checking, protocol mapping and policy enforcement
- Security information and event management (SIEM) by default
- Collection of security and audit events and persistence following failure or reboot, for forensic purposes
- Compliance-driven design and test
- Security best practice in security by design
- Automation and autonomic non-human controls, especially for mundane tasks

A **cyber reference architecture** (CRA) brings a rigorous and structured approach to developing and maintaining security by design. A CRA can be applied at any level of abstraction and provides rigor in approach and outcome. Having a CRA also allows the development of security blueprints to simplify commonly occurring patterns of controls and countermeasures.

A key concept of security by design is the idea of **security assurance that is built in**. All the features of security by design should be constructed, operated and maintained with continual vigilance to assert that a system meets these requirements. That includes securing the development environment, application code and software. Runtime code and containers must be locked down as well as the end-to-end pipeline. Security compliance assurance in DevOps must be included.

**Security-enhancing tools and techniques** are part of the requirement as well. When considering security-enhancing tools and techniques, consider issues like encapsulating security policy and configurations as code for the platform. Determine how configurations and policies can be protected and verified and which build/configure/deploy activities can be automated. You'll also need to assess if checking and testing can be automated and if the pipeline itself can be checked and

assured. Enabling any configuration as code simplifies deployment and improves confidence in configuration and repeatability. But consider how those assets themselves are going to be managed and protected.

**Moving from small-scale applications development** in the cloud to industrialized usage of cloud and SaaS services for critical business purposes requires a change in security response from the business itself. As you look to move from small application-focused projects to larger programs, consider the need to create an environment of security automation. You'll need to work to a broader set of global security policies. Develop a proactive security architecture, policies and standards, for example, for containers and testing. You'll need to continually revisit the architecture, policies and standards through iterations as landscape, scope, complexity and system criticality change. Adopt and reuse trusted policies and code in preconfigured elements.

A security runway accounts for a continually moving landscape in which security controls and countermeasures must move in multiple dimensions. Developing a security runway enables planning security architecture, controls and countermeasures just in time for application, platforms and pipeline interception.

**Security education** follows a parallel track with agile development and scaling. Security education applies to everyone in the organization, across all business units. Security is an equal requirement, not an afterthought. Security education is important, and your team needs to know why they are doing things, not just what they need to do. This is a function addressed by the CRA and blueprints.

As your **partner ecosystem** expands you should not assume all partners have the experience or pedigree of the old-style large IT product partners. We recommend you ensure the security maturity of any partner you're considering as a prerequisite. You should share and instill security by design practices in developing offerings. Further, you should develop shared security contexts — for example, establishing and using a cyber reference architecture or common trust domains and control sets.

**Situational awareness** from start to finish is needed to improve security in such a rapid and diverse landscape. This means monitoring and managing the pipeline and environments in the same way as the live operational systems. It includes collecting and assessing instrumentation and metrics throughout the DevOps life cycle from platforms and activities. In this way, trust in streams of development activity can be established early and remediation built in before any live system exposures.

It is important to **understand the overall governance, risk and compliance regime** of the environment. This includes understanding what data is where, data characteristics and what domains of trust exist in the system (what can be controlled and what level of trust and compliance is built in). It demands a broad security view (e.g., a CRA) and that proportionate security risk management be built into the DevOps life cycle.

The **technical security capabilities** landscape moves as fast as the threat landscape. DevSecOps should continually look for and apply new techniques to maintain currency in control technologies. This often drives the need for a micro view of security to complement the broad security view. It includes steps such as improving runtime and process protection and generally reducing the scope for aberrant behavior of rogue code.

A **security runway** accounts for a continually moving landscape in which security controls and countermeasures must move in multiple dimensions. Developing a security runway enables planning security architecture, controls and countermeasures just in time for application, platforms and pipeline interception. It also allows effective prioritization of the overall backlog of work to ensure that key enablers are in place at the right point in time.

Where the **code pipeline** spans from Dev to Ops, we must seriously consider the security of the pipeline itself. We must secure the pipeline of delivery for both the application and the platform elements of any system. The strength and rigor of controls may vary across the two areas. We can make pipelines secure by providing:

- Automation of security functions and controls
- Infrastructure and platform vulnerability scanning
- Deployment, release and change controls and automation

To reduce the chance of avoidance we must make security easy. Security functions should be transparent to developers, including elements such as application security testing, platform compliance with configuration specifications and preconfigured gateways. When you make security functions transparent, application developers are more likely to embrace them than avoid them. We can further ease take-up and ensure consistency of built-in security features by providing common services like IDAM for people, machines and interfaces.

## Securing the code supply chain

Next, let's look at an example focused on the code supply chain (see **Figure 3**).

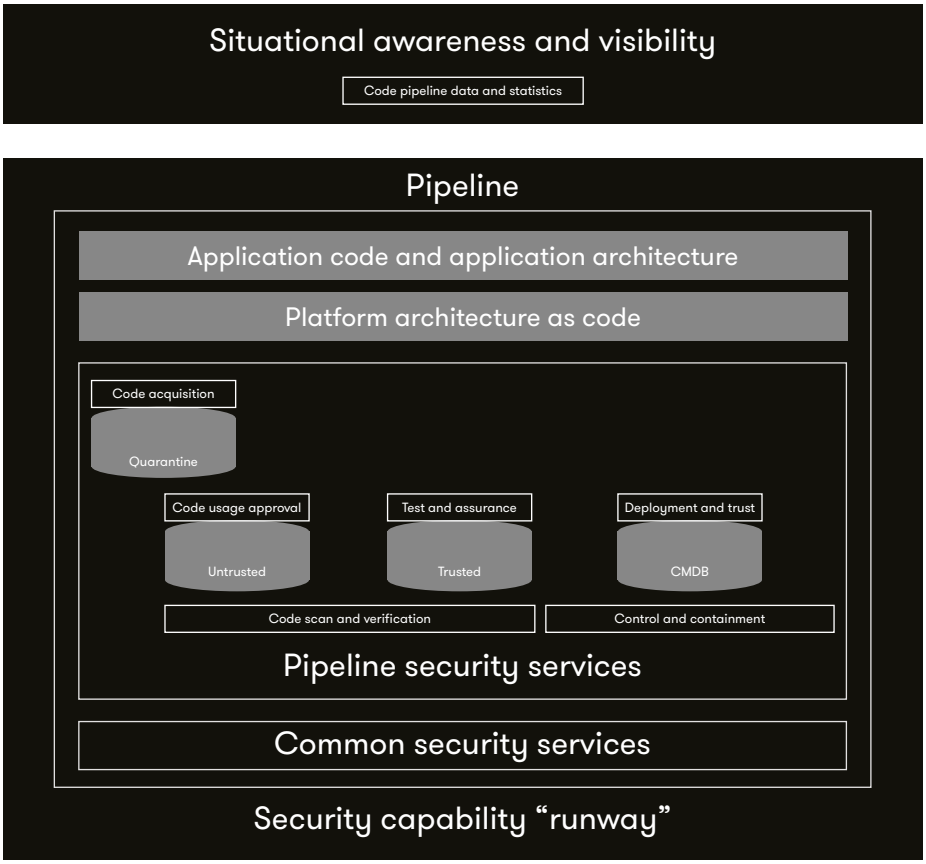



**Figure 3.** Components of the code supply chain

Everything is software now, and the code supply chain is no different than any other source. Supply chain security and due diligence are critical. Consider how you can reduce risk by implementing components like these:

- **Code acquisition mechanisms** to verify the source and ensure the code is free of known vulnerabilities. These can identify where code came from or who introduced it, citing whether code was open source, home written or sourced from a bona fide partner.
- **Code approval and governance** to ensure only verified people make endorsed changes to the assets and maintain code hygiene, avoiding cross-contamination between trusted and untrusted code.
- **Software asset tracking** to identify where code has been used. If a flaw is found in code, developers want to know where it was deployed.
- **Continuous review** of vulnerabilities identified, bad behavior, etc., employing code-scanning mechanisms to root out known vulnerabilities and inspect code veracity.

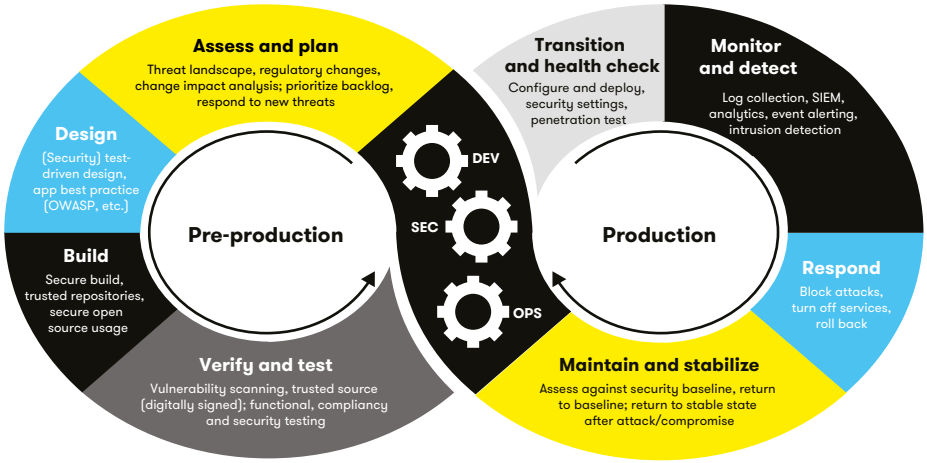



**Figure 4.** DXC DevSecOps life cycle

The DXC DevSecOps life cycle, depicted in **Figure 4** above explores these elements and others.

Part of code security is containment. We first try to prevent the incursion of malware, but we must plan for failure. To do this, we seek to restrict and contain the potential impact area should the worst occur. We can also consider additional features like microservice architectures, secure containers and discrete network and process containment.

These and other containment mechanisms reduce the blast radius of any incursion. However, the landscape is fast moving, and there are implications to application architecture. This makes the security architecture runway so critical.

## Where to start

So, if you are intent on strengthening your DevSecOps posture, where should you begin? Here are several actions to get started:

**Extend situational awareness.** Extend the reach of your security operation into the Dev domain and the pipeline. Identify and collect security-related metrics from these domains.

**Establish pipeline security.** Automate your platform security in areas such as setup and configuration confirmation, then place the code and policy for this under secure control. Automate the mundane and automatable security-related tasks in the development pipeline. Enhance your design for DevSecOps — release packages, deploy, undo, redo — to enable rapid regressions in case an exposure is identified.

Adopt and develop a cyber reference architecture and consult it regularly — it should continually change, mature and improve.

**Leverage services.** Consider how you can limit platform subversion for environments, servers/VMs, containers and storage through separation, encryption and virtualization. Consider autonomic security testing for platform and pipeline. Consider how you secure cross-boundary interconnects, including IDAM, API gateways and endpoint detection and response (EDR). Consider how you easily provide access to pervasive trust services and how you control privileged access.

**Clarify responsibility.** Consider how you clarify responsibility for overall security and individual responsibilities for securing elements that include:

- Big-picture architecture
- Security controls and countermeasures
- Pipeline security for apps
- Pipeline security for platforms
- Pipeline security services
- Common security services

**Apply security by design.** Consider how you strengthen and apply security by design. Adopt and develop a CRA (see next page) and consult it regularly — it should continually change, mature and improve. Consider how you can link risks to controls and countermeasures to aid change impact and traceability.

**Use state-of-the-art controls.** Consider how you monitor the security landscape and state of the art. How can you constrain subversion to the control processes and process interactions of the application only? Consider workload segmentation and process-level protection using code and policy. Also, consider micro segmentation with code and policy. Microservice architectures can improve security posture through defense-in-depth and isolation. Deception technologies are another viable alternative. And consider using behavioral analytics in both the live operational domain and the pipeline domain.

**Adopt a CRA.** It's clear that integrated, proactive, comprehensive security results from extensive planning and preparation. What's less clear is how to ensure your company has developed a plan that clearly identifies your needs in all security domains and that the right prescriptive steps have been defined to address them. The best way to do this is to adopt a CRA, as shown in **Figure 5**.

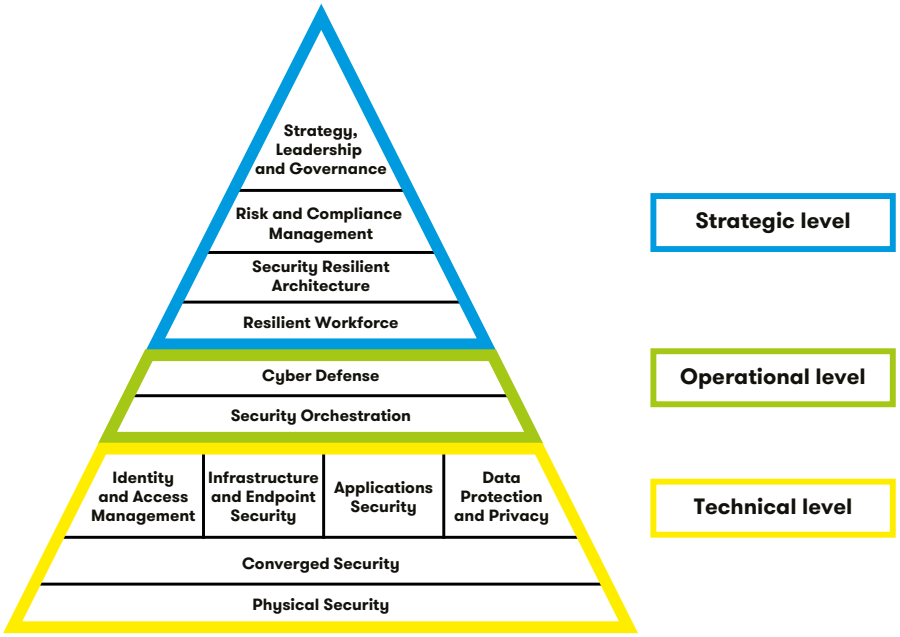



**Figure 5.** DXC Cyber Reference Architecture

The CRA is a framework of strategies, tactics and capabilities that provides a common language, a consistent approach and long-term vision to help your company align security strategies with the business and accelerate your digital transformation. This approach helps you understand what objectives matter most, define the security requirements needed to achieve those objectives and map out the best approach for implementation.

From this governing framework, you can create blueprints that accelerate the process of defining a detailed plan for deploying security capabilities (see **Figure 6**). This is done through a story-board approach that defines discreet statements of work, called work packages, that address each business objective.

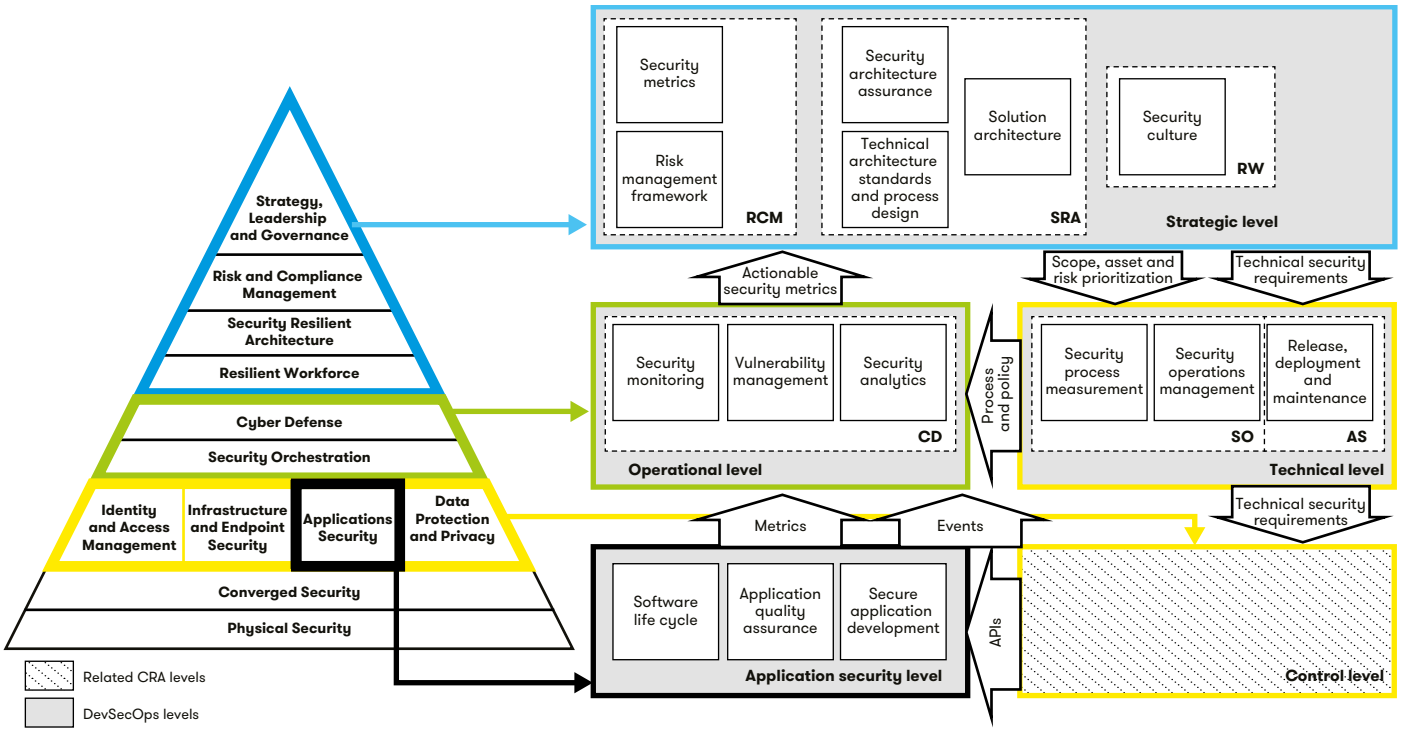



**Figure 6.** Layers and key capabilities in the DXC Cyber Reference Architecture

The **cyber reference architecture** created by DXC is based on decades of experience monitoring billions of threats globally and responding to some of the world's largest attacks. We applied that experience to create a security framework and methodology that helps organizations in all industries move from a reactive mode to higher levels of **cyber maturity**. Organizations are becoming better equipped to visualize their future state and develop a roadmap of short- and long-term timelines for getting there.

## Securing the future

The security of applications and platforms is so critical to the health and performance of the enterprise, it can no longer be bolted on as an afterthought. Security must be an integral part of the software development life cycle.

Anything less than an all-out focus on security is sure to put the enterprise at risk. With the measures and means available today, that's a risk that no one needs to take.

## About the authors

**Nolan O'Dwyer** is lead technologist at DXC Technology's Digital Transformation Center in Newcastle, UK. He is an expert in strategic planning, designing, building and delivery of IT infrastructure and applications in large, complex business modernization and transformation programs.

**Chris Rogers** works in the Global Architecture team at DXC Technology. He focuses on digital transformation capabilities for business solutions that help clients address the real problems of digitization in a rapidly changing business ecosystem.

**Mark Evans** is the chief security architect for the Security Consulting, Integration & Compliance team at DXC Technology for its UK, Ireland, Israel, Middle East and Africa (UKIIMEA) region. He focuses on cloud security, information- and business-centric security and cyber security in the digital world.

## Learn more at DevSecOps and digital applications



CP_1053a-20. April 2019