CYBERARK®

EBOOK

# 6 Core Principles for Establishing DevOps Security at Scale

# INTRODUCTION

## DevOps Requires a Fresh Approach to Security

**Businesses all over the world are looking to development and operations teams to adopt DevOps methodologies to deliver applications more quickly and cost-effectively. Businesses increasingly recognize that by eliminating independent functional and administrative silos, and automating change management, configuration management and deployment processes, DevOps helps organizations accelerate time-to-market and improve product quality and customer satisfaction. Continuous integration and delivery (CI/CD) methodologies allow engineers to incorporate user feedback quickly and efficiently, enabling businesses to be more flexible and responsive to customer needs.**

DevOps can help businesses increase the pace of innovation and accelerate digital transformation. But disjointed DevOps security systems and practices can slow down CI/CD pipelines, frustrate developers, and lead to risky workarounds. Developers often hard-code credentials into applications or take other shortcuts, exposing the organization to cyberattacks and data breaches that can disrupt business, damage a company's reputation, and result in revenue loss, compliance violations, and legal settlements.

Most conventional security management solutions and practices, designed to support traditional software applications and development methodologies are too slow and complex for the fast-paced world of continuous integration and delivery. DevOps and security leaders recognize that DevOps requires a fresh approach to security that mitigates risk and uncertainty without impairing velocity. This e-book presents six guiding principles for enabling DevOps security at scale.

**Agile Firms Grow Revenue 37% Faster and Generate 30% Higher Profits**

**Source:** How business can survive and thrive in turbulent times. The Economist Intelligence Unit Ltd.

## Contents


**Guiding Principle 1:** Instantiate Security Policy as Code ..... 3

**Guiding Principle 2:** Instill Separation of Duties ..... 6

**Guiding Principle 3:** Focus on Flow and Velocity ..... 8

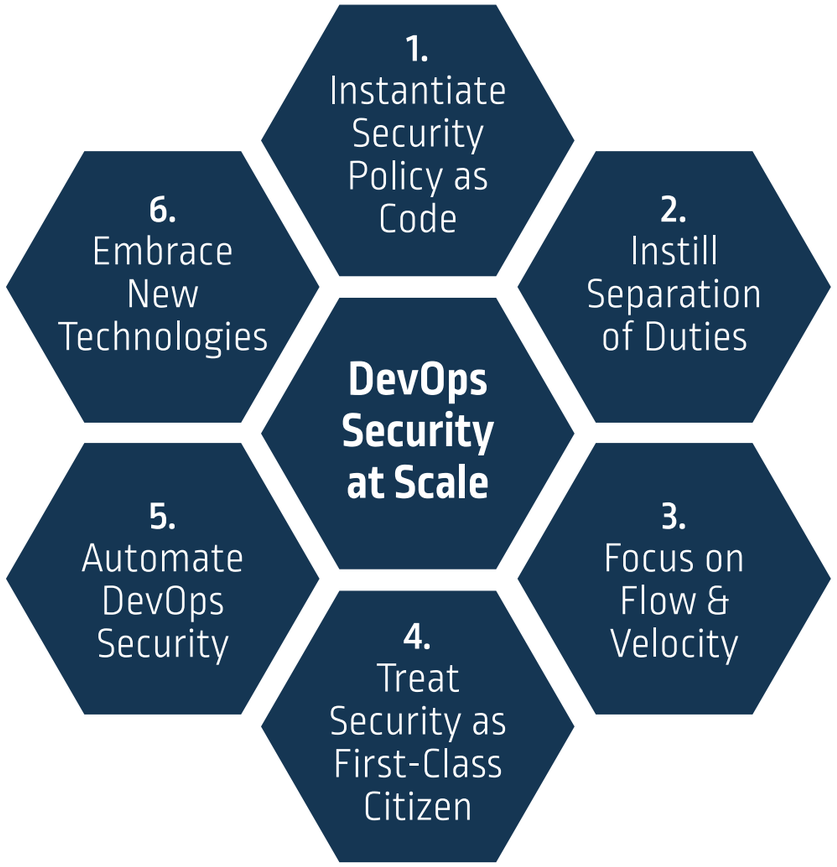
**Guiding Principle 4:** Treat Security as a First-Class Citizen ..... 10

**Guiding Principle 5:** Automate DevOps Security ..... 12

**Guiding Principle 6:** Embrace New Technologies ..... 14

**Delivering Security that Works at DevOps Velocity** ..... 17

**CyberArk Secrets Manager** ..... 18




# GUIDING PRINCIPLE 1: INSTANTIATE SECURITY POLICY AS CODE

**Conventional approaches to controlling access to critical systems are notoriously inefficient. Most organizations rely on manually intensive, error prone processes—configuring permissions and managing passwords by hand across physical and virtual machines scattered across the enterprise. By extending DevOps concepts to security—implementing security policy as code—organizations can improve their security posture, while maintaining DevOps velocity and scalability.**

A cornerstone of DevOps is the concept of "Infrastructure as Code" (sometimes referred to as immutable infrastructure), which supplants the traditional model of manually administering and configuring servers and software. Infrastructure as Code enables automatic scaling, and fosters predictability as new features roll into different test environments and into production. The code that represents this infrastructure is checked into source control just as the application code is, and can be efficiently versioned, compared and protected.

**By applying the Infrastructure as Code approach to security—instantiating and managing security policy as code—organizations can eliminate manually intensive, error-prone configuration processes and accelerate the pace of innovation. With a Security Policy as Code approach, applications and services declare their security policy requirements within code.**

Conjur Secrets Manager Open Source is an open-source security service for controlling access to critical systems via a policy language. A simple example of a security policy for a currency conversion microservice, written using Conjur's policy language is shown here.

```
- !policy
  id: currency-converter
  body:
  - &variables
    - !variable
      id: currency-database/password
      annotations:
        description: Currency values DB password

  - !group secrets-users

  - !permit
    resource: *variables
    privileges: [ read, execute ]
    role: !group secrets-users

- !grant
  role: !group currency-converter/secrets-users
  member: !host currency-app-host
```

## DECLARING SECURITY POLICY AS CODE

**A** Apply Infrastructure as Code approach to security

**B** Eliminate manually intensive, error-prone configuration processes

**C** Accelerate the pace of innovation and reduce risks

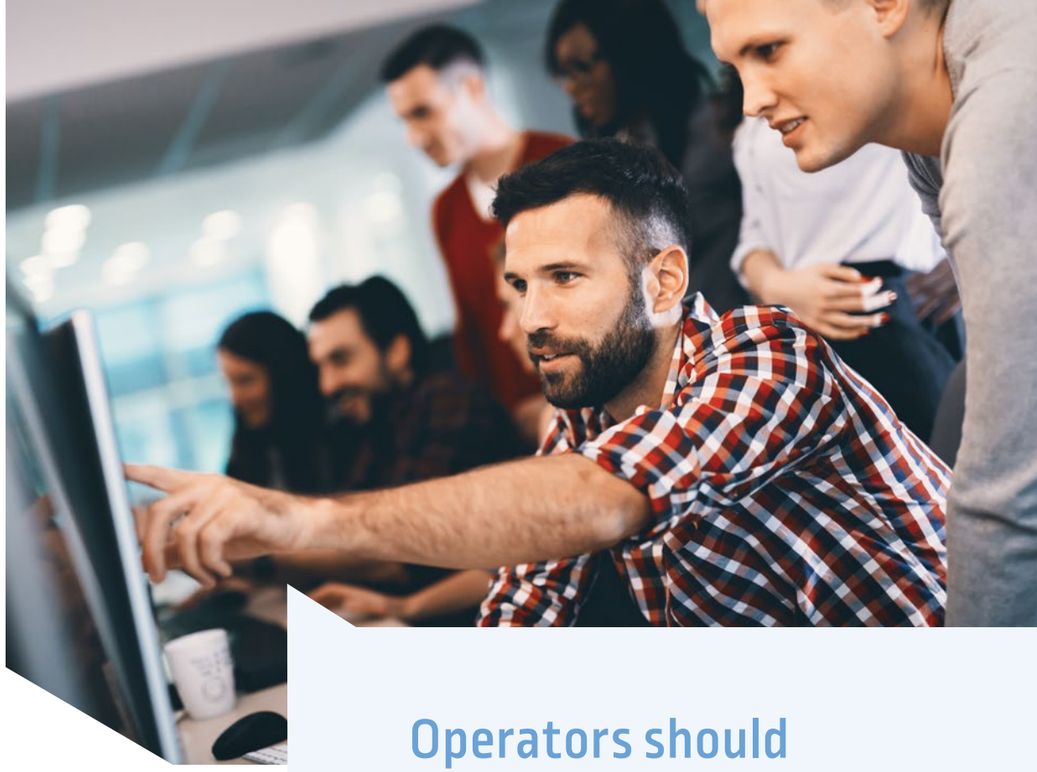
The syntax is human-readable and simple to understand: the code grants the **currency-converter** service the right to access the password value for the database that stores currency values. The policy is completely self-contained, reproducible, and depends on nothing external that might change how it is interpreted in the future. It gets checked into source control, and is just as much a part of the application's code as its internal implementation.

Security Policy as Code is critical for enabling a highly agile, scalable and secure DevOps environment. The security staff responsible for managing permissions of thousands of applications can use this Security Policy as Code model to scale their team's capacity to new levels—in a way that brings much more predictability to their work. Compare this approach to configuring security permissions for applications manually as they move across dev, QA, staging and production systems.

# **GUIDING PRINCIPLE 2:** INSTILL SEPARATION OF DUTIES

**It's not rocket science. Operators should be operators and focus on operations. Developers should be developers and focus on development. Sadly, too often in many organizations, developers and operators are also tasked with security duties: maintaining service accounts, defining security controls, controlling access to sensitive data, etc. Doing so places additional burdens on already over-worked staff. It also introduces security risks. It is unreasonable to expect a developer or system admin to become an expert in an entirely new domain—security—when they're already expected to be experts in software engineering or IT operations and administration.**

By instilling separation of duties—clearly defining distinct roles and responsibilities within a DevOps team—businesses can optimize operations, minimize risks and accelerate the pace of development.  Security and oversight should be handled by dedicated, experienced security staff, so developers can concentrate on writing code, and operators can focus on maintaining and running infrastructure.



Operators should be operators and focus on operations. Developers should be developers and focus on development.

## INSTILLING SEPARATION OF DUTIES

| | |
|---|---|
| A | Let developers focus on creating applications to drive business results |
| B | Let the operations team focus on delivering reliable and scalable infrastructure |
| C | Let the security organization focus on safeguarding assets and data, and mitigating risks |

### More Specifically:

- The **Development Team's** primary responsibility is creating applications to support the business. Developers must justify and communicate the infrastructure and security needs for their applications to the Operations and Security teams, respectively.
- The **Operations Team's** primary responsibility is to fulfil the infrastructure needs of the applications being produced and ensure IT systems deliver adequate performance, scalability and availability.
- The **Security Team** is responsible for granting the permissions to applications as requested by Developers, and for ensuring strong security solutions and practices are in place across the enterprise to mitigate risk and meet compliance requirements.

The interactions between each group can be codified in a Security Policy. Developers create the security policy that declares what privileges their application or service requires. Security staff then review and approve the security policy, and operators make sure the application's deployment goes as expected.

# GUIDING PRINCIPLE 3: FOCUS ON FLOW AND VELOCITY

At the core of DevOps is the notion of sustainable, smooth workflows. By introducing CI/CD pipelines, orienting teams around very small but frequent code changes, getting "end-to-end" early, and adopting system architectures (such as those based on microservices), DevOps teams can improve productivity and accelerate the pace of delivery. To ensure success and mitigate risks, organizations must extend DevOps practices to include security. A high-velocity DevOps team should never have to sacrifice security to ship features faster.

## Model and Optimize the Security Workflow

DevOps organizations often fail to integrate security into their CI/CD practices. Instead, security reviews are too often treated as an afterthought and performed too late in the process, if at all. Then the potentially substantial last-minute changes needed to address vulnerabilities result in delayed releases. Forward-looking organizations are using advanced workflow scheduling and management tools like Kanban to model flows and accelerate development. Kanban lets teams visualize workflows to identify bottlenecks and eliminate inefficiencies. By incorporating security into Kanban analysis (shifting security to the left), DevOps teams can detect and address security issues earlier in the development process, avoiding last-minute changes and release delays.

> "A high-velocity DevOps team should never have to sacrifice security to ship features faster."

## FOCUSING ON FLOW AND VELOCITY

**A** Incorporate security early in the CI/CD practices and development processes

**B** Use Kanban systems to visualize flows and identify bottlenecks, and incorporate security reviews into Kanban visualizations

**C** De-construct applications into microservices to streamline development and simplify security reviews and changes

**D** Get end-to-end early to reduce risks and accelerate application delivery

### Secure Microservices for Speed

Microservice architectures offer many benefits to security teams. Monolithic applications are notoriously difficult to deliver, extend and maintain. It can take days or even weeks for a security team to analyze thousands of lines of code to identify potential vulnerabilities. By deconstructing large applications into smaller microservices, each with their own security policy, organizations can simplify code reviews, reduce risks and accelerate the pace of delivery.

### Go End-to-End, and Do So Early

DevOps has embraced the concept of "getting end-to-end early." It is a simple method of de-risking delivery and improving confidence in estimates. The faster a team can get to "connecting the pipes" in a new service, the sooner they'll learn how to build a smooth deployment flow for making frequent upgrades and updates to it.

Many organizations use feature flags to hide features and capabilities that aren't fully ready for production.  A feature can be included in one release, and exposed in a future release after it is fully functional and fully tested. This approach can be applied to security as well, especially when the team has embraced the Security Policy as Code principle.

For example, say a new service must access an internal CRM database and payment gateway. The team would build a CI/CD pipeline for this new service as their first task, and get a v.0.1 of it moving through that pipeline with only basic connectivity established. The service would have its privileges already vetted and would be "pre-enrolled" to access the sensitive resources it needs once deployed in production. Developers can create the remaining features in the service in later versions, safe in the knowledge that the service's permissions and security settings have already been set up.

# **GUIDING PRINCIPLE 4:** TREAT SECURITY AS A FIRST-CLASS CITIZEN

**Most development teams strive to optimize product quality and proactively address application defects, design flaws and performance issues. Yet many organizations only address security vulnerabilities after they fail an audit or worse yet, after a serious breach has occurred.**

To ensure successful DevOps outcomes, development organizations must take a proactive approach to security. By treating security as a first-class citizen—instituting strong security systems and practices throughout the application lifecycle—development teams can reduce vulnerabilities, improve their security posture and mitigate risks.

### Good security hygiene practices include:

- Address security requirements and potential vulnerabilities holistically. For example, just with DevOps it's essential to ensure the administrative consoles for the DevOps tools and cloud providers are secure, that the credentials used in scripts, etc. to automate the CI/CD pipeline and DevOps tools are secure, and of course that the credentials and secrets used by the applications and the deployment environment are secure. This is in addition to securing the privileged access to the basic infrastructure, and so on. An attacker may only need to exploit one vulnerability, while a defender needs to lock down all vulnerabilities.
- Maintain secrets used by machines and people (passwords, certificates, API keys, tokens, and SSH keys) in a **secure, highly available vault** out of source code and off of developer laptops and user-accessible storage systems. Automatically rotate secrets on a regular basis based on policy to limit exposure in the event of a breach.

> "Too many organizations only address security vulnerabilities after they fail an audit or worse yet, after a serious breach has occurred."

## TREATING SECURITY AS A FIRST-CLASS CITIZEN

| | |
|---|---|
| A | Address security needs and potential vulnerabilities holistically |
| B | Apply principles and policies like least privilege and segregation of duties |
| C | Manage secrets in a secure and reliable vault. Rotate secrets regularly to minimize exposure. |
| D | Run vulnerability scans and conduct penetration tests to improve security posture |
| E | Educate developers on security threats and best practices. |
| F | Foster close cooperation and collaboration between security and development teams. |

- Apply the principles of **least privilege** wherever possible. Ensure machines and people only get access to the resources they absolutely require. Promptly revoke permissions if an application is decommissioned or if a person changes roles or leaves the company.
- Automatically run **vulnerability scans** on all dependent 3rd-party libraries. Ensure vulnerable libraries are always up-to-date.
- Execute **penetration tests** regularly to scan applications (and CI/CD environments) for potential attack vectors. Many organizations conduct penetration tests as part of a CD pipeline using automated tools, and augmented with regular human-driven, white-hat ethical hacks.
- Ensure all developers are **trained** in application and information security as part of their new hire programs, and in their ongoing education.
- Include explicit checks for information exposure, attack vectors and security policy violations as part **development workflow** for product code changes.
- Ensure development and security teams work closely together to ensure successful outcomes. Make a conscious effort to **"shift security left"** in the workflow. Ensure the security organization is directly involved at the early stages of development of new applications. Have the security team review architectures and technology choices to avoid reworks, delays and budget overruns in the long run.

# **GUIDING PRINCIPLE 5:** AUTOMATE DEVOPS SECURITY

Effective DevOps teams use automation to accelerate application lifecycle management and remove human latency. CI/CD tools help organizations automatically move applications from development to integration to production. DevOps teams should take a similar approach to security, leveraging automation to improve their security posture while avoiding human latency and application development and delivery barriers.

## Minimize Human Interaction and Manual Intervention

As long as an application's security policy hasn't changed and no new privileges are required, there's no reason to involve human security personnel in an upgrade process. By incorporating certain security tasks into the CI pipeline (e.g., scan 3rd-party libraries and runtimes for common vulnerabilities and exposures, check application code for security flaws, run penetration tests, etc.) you can automate upgrades while mitigating risks. While they sometimes take time to run, the overall cost of automatically executing these tests with each deployment is drastically smaller when compared with the cost, risk and time spent having humans do the same tasks over and over.

> "Leverage automation to improve the organizations security posture."

## AUTOMATE DEVOPS SECURITY

**A** Minimize human interaction and manual intervention

**B** Guard against breaches with automation, including automatically rotating credentials based on policy

**C** Respond to breaches with automation by establishing policies and workflows to automatically respond to potential vulnerabilities and breaches

### Guard Against Breaches with Automation

Automated security procedures can help DevOps teams take a proactive approach to security, reducing vulnerabilities and risk. For example, by automatically rotating secrets—passwords, keys, certificates—organizations can prevent attackers from gaining access to DevOps tools, access keys, or systems for an extended period of time. Time is a hacker's best friend. With unfettered access a bad actor can expand an attack surface and do more damage, or simply steal access keys for crypto-mining.

For effective security, rotate passwords frequently as part of comprehensive "rotate, repave and repair" strategy. Rotate datacenter credentials every few minutes or hours. Repave every server and application in the datacenter every few hours from a known good state. Repair vulnerable operating systems and application stacks consistently within hours of patch availability.

### Respond to Breaches with Automation

Automated security procedures can also be used reactively if a breach is detected. To help contain threats and minimize exposure, automatically rotate credentials the moment a security breach is identified. Immediate credential rotation is essential for containing attacks and restoring infrastructure (VMs, containers, etc.) to a pristine state.

# GUIDING PRINCIPLE 6: EMBRACE NEW TECHNOLOGIES

DevOps organizations must leverage new security technologies and practices to support the dynamic world of on-demand applications and cloud-based services. The more traditional approaches and policies for security designed to protect legacy IT environments often aren't well suited for today's dynamic environments with microservices, containers, orchestrators, and serverless technology. Forward-looking security teams are adapting a DevOps culture, embracing new security technologies and models while leveraging the policies and lessons learned from more traditional environments.

## Introduce New Technology to Improve Security

It may be counterintuitive, but adopting new technology can help organizations become more secure. It just has to be done judiciously and safely. Embracing new technology enables new kinds of security protections to be put in place, it helps security workflows keep pace with IT innovation, and it prevents shadow IT initiatives.

By replacing traditional network access controls with newer identity-based security controls, DevOps organizations can take full advantage of cloud-based IT without sacrificing security. The BeyondCorp security model from Google is a perfect example. It transforms the typical deployment model for internal applications. Instead of placing sensitive applications behind a VPN, Google deploys applications to the public internet. Application access is governed by the accessor's identity, not by which network they are on. The model has helped Google achieve rapid service velocity and massive scalability.

New developer-centric technologies are radically reducing the amount of sensitive assets that engineers need to access. By segregating duties, DevOps teams can improve security while streamlining application development and delivery. For example, container platforms like Red Hat OpenShift and deployment orchestrators like Kubernetes clearly isolate developers and operators for greater security and agility.

## EMBRACING NEW TECHNOLOGIES WHILE LEVERAGING EXISTING PLATFORMS

**A** Embrace change

**B** Introduce new security technologies for DevOps environments

**C** Adopt new security models for DevOps environments

**D** Establish a single control point to enable security access policies to be consistently applied across the entire organization

New serverless technology also has the potential to dramatically improve security. Developers working with VMs or containers are very close to the operating system, where many potentially sensitive or risky assets live. Those assets can be completely inaccessible in a serverless or Function-as-a-Service (FaaS) model. Serverless technology has the potential to be an appealing option to security teams.

### Adopt New Ideas

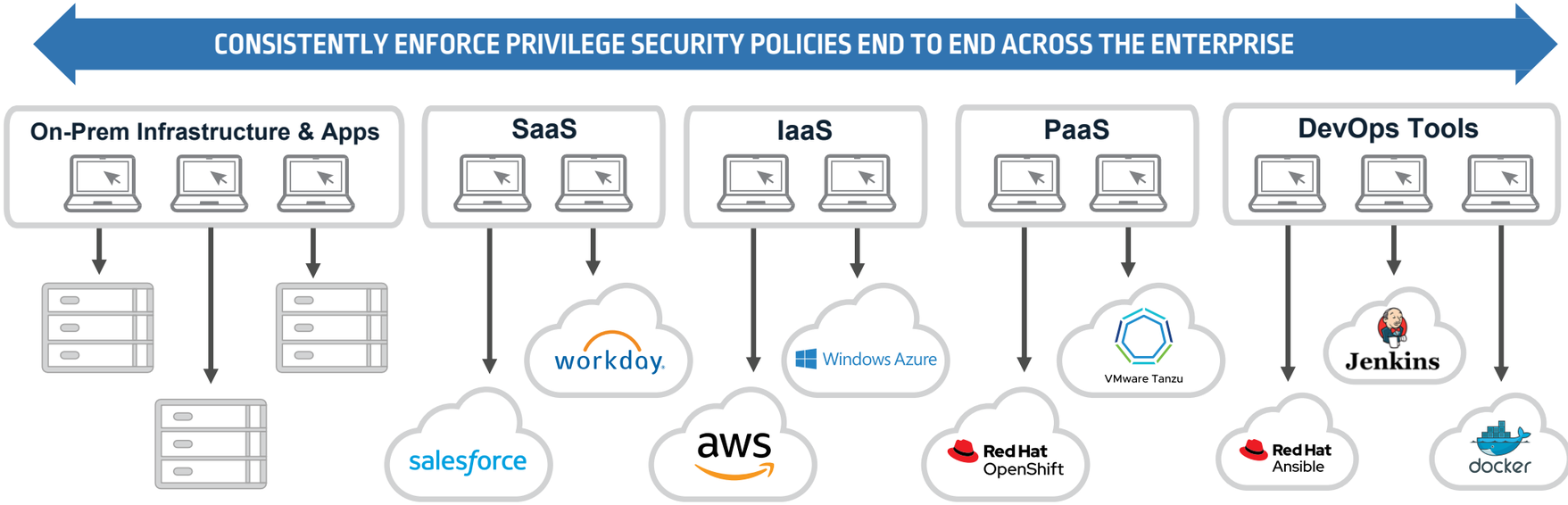
The traditional IT security approach focused on identifying which assets to control, and then building structures to lock them down. All workflows and process layered on top were built with this centralized control model as a fundamental assumption.

Now, security teams must introduce new methodologies, practices and solutions to support the dynamic world of DevOps. In today's world it is impossible to map an organization's set of systems. Instead security systems must be built around identities (both for humans and machines) and policies. In the new world every entity must have an identity, privilege changes must be communicated clearly through policy, duties must be clearly separated, and business velocity must be enabled through automation. With this new philosophy and culture, and solutions like CyberArk Secrets Manager, DevOps teams can accelerate the pace of innovation without sacrificing security.

## Leverage Existing Environments

Many enterprises adopting DevOps methodologies, even if they are undergoing a complete digital transformation, have at least some legacy environments and on-premises infrastructure. For example, while a DevOps team may provide access to a legacy customer database by packaging it into a container, the access to the underlying database application running on the on-premises infrastructure needs to be secured. So in this case both the existing environment and the new DevOps environment need to be secured. As a best practice, CISO and IT Leaders want to consistently enforce privileged security access policies across their complete infrastructure and application environments.

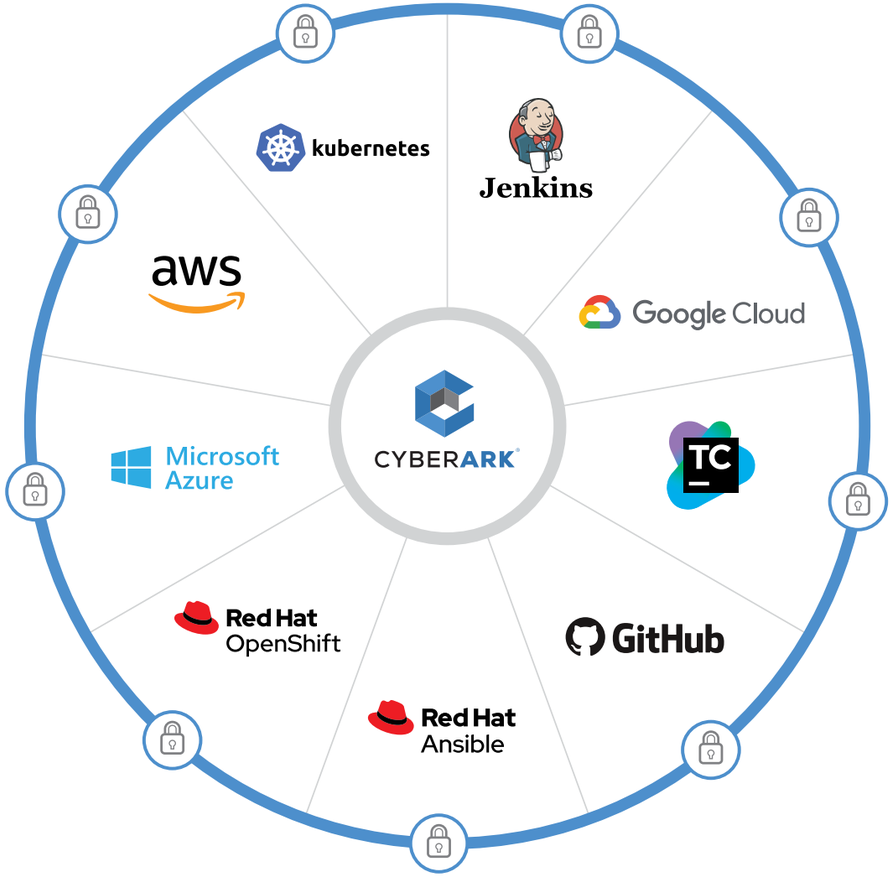
This can be achieved with an integrated solution for managing privileged access, secrets and other credentials, that establishes a single control point for applying consistent policies regardless of the compute or development environment.



# DELIVERING SECURITY THAT WORKS AT DEVOPS VELOCITY

Companies in every industry are turning to DevOps—leveraging agile software development, integration and delivery practices—to accelerate digital transformation and enhance business performance. Development organizations must take a fresh look at security to unleash DevOps agility and scalability. By following the guiding principles in this e-book—taking a proactive approach to security, leveraging automation and programmability, and encouraging cooperation across organizations—development teams can accelerate the pace of innovation while ensuring compliance with corporate security mandates.

CyberArk Secrets Manager is a powerful secrets management solution specifically designed to help developers easily and conveniently meet the security requirements of agile and scalable DevOps environments.




Secrets Manager has many validated integrations with DevOps partners, and is designed to secure the secrets and credentials your DevOps tools and platforms rely on across the entire DevOps tool chain."

# CYBERARK SECRETS MANAGER

CyberArk Secrets Manager enables organizations to centrally secure and manage secrets and credentials used by the broadest range of applications, including COTS, RPA, automation platforms and CI/CD tools, running in hybrid, cloud-native and containerized environments. Mission-critical applications running at scale can securely access high-value resources, including databases and IT infrastructure, to improve business agility while reducing operational complexity.

Loved by security teams and developers, Secrets Manager offers the most out-of-the-box integrations for easily securing applications and DevOps environments.

Conjur Secrets Manager Enterprise is a fully featured, enterprise-class solution, backed by CyberArk's world-class support and services organization. An open source version is available as Conjur Secrets Manager Open Source at www.conjur.org. Conjur Open Source can easily be upgraded for full enterprise class capabilities.

Conjur Enterprise integrates with CyberArk Privilege Access Manager and CyberArk Privilege Cloud enabling organizations to use a centralized policy-based approach to consistently manage the credentials used by human users as well as applications and other non-human identities.

## Secrets Manager enables organizations to:

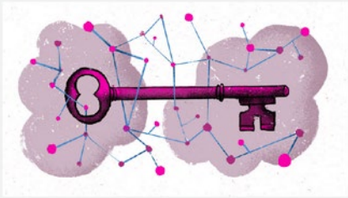
- Secure secrets used by machines and users throughout the DevOps pipeline. API keys, certificates, passwords, SSH keys and tokens are securely stored and managed in an encrypted and access-controlled container and can be automatically rotated based on policy.
- Institute role-based access controls (RBAC). Administrators can define various roles (e.g. development, test, operations, administration) and grant each role unique privileges (e.g. read, write, delete) for specific resources (e.g. database password, VM or server, web service endpoint.)
- Protect DevOps toolchains and platforms. The secrets management solution secures and manages secrets used by CI/CD pipelines such Jenkins and Electric Cloud, automation tools such as Ansible and Puppet and PaaS/container orchestration software such as Kubernetes, Red Hat OpenShift, VMware Tanzu, and Cloud Foundry.

To learn more about how CyberArk can help your organization achieve DevOps security at scale, visit conjur.org for Conjur Open Source, or cyberark.com/devops for more information.



## LEARN MORE

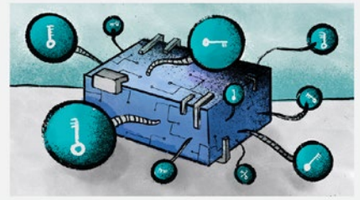
Visit conjur.org/blog and cyberark.com/resources to learn which version is right for you.



How a Stateless Cloud Native Application can Access Vaulted Secrets with IAM Authentication

Secrets Management RBAC Policy Example


Security Considerations for Data Stream Processing



Secrets Management for Hybrid Applications

TRY CONJUR OPEN SOURCE TODAY!