the GORILLA GUIDE® to...

Securing Devs Within the Cloud

Key Considerations for AWS Developers in Today's Agile World

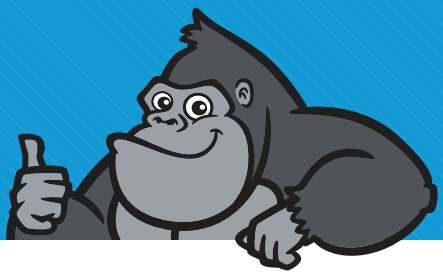
JAMES PANETTI

CYBERARK® The Identity Security Company

powered by aws

POWERED BY ActualTech MEDIA

the GORILLA GUIDE® to...



Securing Devs Within the Cloud

Key Considerations for AWS Developers in Today's Agile World

By James Panetti

POWERED BY ActualTech MEDIA

Copyright © 2024 by Future US LLC
Full 7th Floor
130 West 42nd Street
New York, NY 10036

All rights reserved. This book or any portion thereof may not be reproduced or used in any manner whatsoever without the express written permission of the publisher except for the use of brief quotations in a book review. Printed in the United States of America.

www.actualtechmedia.com

PUBLISHER'S ACKNOWLEDGEMENTS

DIRECTOR OF CONTENT DELIVERY

Wendy Hernandez

GRAPHIC DESIGNER

Olivia Thomson

HEAD OF SMARTSTUDIO

Katie Mohr

WITH SPECIAL CONTRIBUTIONS FROM CYBERARK

Jillian Kane
AWS GLOBAL MARKETING MANAGER

Josh Kirkwood
PRODUCT MARKETING - CLOUD SECURITY

Chris Smith
PRODUCT MARKETING - MACHINE IDENTITY SECURITY

ABOUT THE AUTHOR

James Panetti has over 16 years of tech experience spanning technical support, software performance testing, release automation, and cloud Software-as-a-Service (SaaS) technology. His drive is a love for knowledge sharing and communication. Whether it's writing to help a layman understand a tech solution, an executive understand the work of their IT team, or an IT professional learn a new tool, in the end, it's all about helping people learn, grow, and connect with each other in new ways.

ENTERING THE JUNGLE

CALLOUTS USED IN THIS BOOK

SCHOOL HOUSE

In this callout, you'll gain insight into topics that may be outside the main subject but are still important.

FOOD FOR THOUGHT

This is a special place where you can learn a bit more about ancillary topics presented in the book.

BRIGHT IDEA

When we have a great thought, we express them through a series of grunts in the Bright Idea section.

DEEP DIVE

Takes you into the deep, dark depths of a particular topic.

EXECUTIVE CORNER

Discusses items of strategic interest to business leaders.

DEFINITION

Defines a word, phrase, or concept.

GPS

We'll help you navigate your knowledge to the right place.

KNOWLEDGE CHECK

Tests your knowledge of what you've read.

WATCH OUT!

Make sure you read this so you don't make a critical error!

PAY ATTENTION

We want to make sure you see this!

TIP

A helpful piece of advice based on what you've read.

INTRODUCTION

Changing Security Culture to Meet the Cloud

Cloud architecture serves as critical infrastructure to most modern businesses, from fully online, digital-native businesses to traditional brick-and-mortar enterprises trying to transform their heavy workloads for agility. Meanwhile, digital transformation is lifting developers up to new levels of prominence with elevated autonomy across industries. Organizations across every industry are therefore under constant pressure to evolve and empower developers to create innovative, disruptive applications and features across multiple cloud environments, all the while accounting for the various identities associated with them as they determine their security posture. Who should have access to what, when, and how?

Targeting Developers' Identities

Malicious actors across the globe, be they individual cyber criminals or entire nation states, are constantly and rapidly innovating new attack methods. New threats demand that organizations must adapt and rethink their approach to cybersecurity. The traditional

paradigms of old are not sufficient to address these new threats, nor can they prepare us for what further malicious innovations await us in the future.

Cyber attackers no longer simply target vulnerable machines; rather, they aim for identities in an endeavor to compromise and steal privileged accounts. Every organization has a wide spectrum of identities. Every single member of the workforce has some type of digital identity associated with them. Some belong to highly specialized workers, such as software developers, and others belong to machine identities, also referred to as non-human identities, also referred to as non-human identities. This means that every single identity—human and machine—must have the right level of privilege controls to be truly secure.

Basic security controls are not sufficient to protect all these various identities, particularly because many identities for more specialized roles require complex privileges. The combination of new, innovative attack methods and complex identity needs demand that organizations rethink the very nature of identity security. Cybersecurity measures must be flexible and able to account for the complexity of each identity's role, the complexity of associated environments, and the risks associated with accessing those environments. Furthermore, cybersecurity teams must be able to discern when basic security controls are sufficient and when more heightened, adaptive controls are called for.

Despite this rapidly growing evolution in the threat landscape, 80% of today's digital-native businesses failed security self-assessments. Worse yet, 90% have actually suffered at least one identity-related security breach within the past year. Why is such a massive threat so often neglected?

Why Identity Security Is Difficult for Everyone: Defenders & Attackers

Why do most organizations fail to protect the assets attackers want most? Simply put: Identity security is a constantly moving target. Think of it as a huge building full of multiple doors and every door has multiple locks. There's never simply one key to protect, but many keys and many types of keys that vary widely across a vast sea of unique resources.

This is especially true for software developers. Development environments have evolved rapidly and dramatically. They now include countless virtual machines and cloud service providers, resulting in multiple, varied layers of architecture. Unfortunately, the proliferation of cloud-based tools, all accessed via third-party hosts, introduces additional points of potential breaches. Each cloud service introduces its own user identities and user management mechanisms, which ultimately results in one organization relying on countless identities spread across many platforms, each with their own unique types of secrets to protect. Many organizations leave their development teams responsible for the configuration of cloud services and the accounts run within them, who understandably may not be aware of best security practices.

It's not simply a matter of securing identities, though. Software developers must move fast in today's agile world. Deadlines, milestones, an endless backlog of bugs and stories, and management pressure mean developers can never afford to let obtrusive security measures slow down their productivity.

CHAPTER 1

Securing Developers Without Limiting Autonomy

Balancing developer autonomy while satisfying the security team's requirements poses a major, often paradoxical challenge as cybersecurity team's and software development team's priorities often inevitably clash. Effective cybersecurity necessitates thorough, multi-layered controls while developers are expected to do their work as quickly and with as much autonomy as possible. For example, overly restrictive cybersecurity requirements can end up competing with the development team's need to innovate and meet release deadlines.

Can one only truly be satisfied at the cost of the other?

Meeting the Needs of the Security Team

The cybersecurity team must prioritize securing organizational resources as steadfastly and thoroughly as possible because the threat landscape demands they never cut corners. Now that developers are a bigger target than ever before, cybersecurity experts must take extra measures to protect them.

ADDRESSING OVERLAPPING AUTHORITY

Securing developer resources is easier said than done. As digital transformation advances, development teams are steadily rising in importance and more and more responsibilities are shifting from cybersecurity teams to developers, blurring the boundaries of authority when it comes to who owns the cybersecurity procedures that apply specifically to development environments.

Development environments are far from monolithic; rather they tend to encompass multiple build environments, multiple Continuous Integration/Continuous Delivery (CI/CD) pipelines, multiple cloud environments, and multiple complex tool integrations throughout. Developers require highly elevated privileges so they can oversee the various identities and credentials associated with these platforms and provision and administer compute resources during the various phases of the pipeline—all of which is often done via automation tools. This is especially true in environments where developers must manage multiple cloud platforms (such as AWS), provisioning tools (such as Red Hat Ansible and Terraform), container-orchestration environments (such as Red Hat OpenShift and Kubernetes), and every additional tool and platform that integrates with each of them.

Research from McKinsey & Company discovered that more than 80% of organizations in North America and Europe use two or more cloud service providers (such as AWS, Microsoft Azure, and Google Cloud).

RESTRICTING PRIVILEGES

With all that in mind, the best means of securing developer environments and identities begins with upholding Least Privilege. The principle is simple: only ever grant an individual user the absolute minimum permissions to resources needed for them to perform their duties.

The new best practice regarding Least Privilege is to implement Zero Standing Privileges (ZSPs). Standing privileges are those permanently granted to a user account and used to be commonplace in traditional environments. ZSP was coined by leading analysts to describe the new ideal state of privileged access: no standing privileges for any users whatsoever—no exceptions.

MANAGING CREDENTIALS

Cybersecurity teams should rotate every password regularly and developers should never hard-code credentials under any circumstance so that no application's secrets can be leaked. Maintain continuous credential management and rotate credentials regularly, ideally via a combination of secure Identity and Access Management (IAM) and Privileged Access Management (PAM) solutions.

EMBRACING MULTI-FACTOR AUTHENTICATION

Strong authentication methods are another key safeguard and should be implemented across all development architecture, including virtual and cloud environments. Secure, Multi-Factor Authentication (MFA) best practices ensure appropriate levels of authorization across all cloud-native services.

Every cloud platform login should require MFA. They should also integrate with third-party identity providers (IdPs) so that the organization can leverage its existing Single Sign-On (SSO) and MFA solutions to ensure all cloud-native services fall under the purview of the organization's standardized identity controls.

CENTRALIZED IDENTITY MANAGEMENT

Leveraging multiple providers' identity management systems spreads authentication controls far and wide, introducing countless points of entry. A centralized identity management platform that automatically integrates with third-party services not only maintains control in one place, but also ensures reliable identity lifecycle management and identity compliance mapping. These systems should also vault credentials and implement tight user session management controls.

MONITORING & AUDITING

Always keep an audit trail and stay vigilant in monitoring all developer resources and user identities. Thorough, real-time logging systems can ensure that any security breaches that may possibly occur are immediately detected so that incident response teams can swiftly spring into action.

Various cloud monitoring and logging solutions can maintain an ever-watchful eye over all user activity across cloud applications and consoles. For example, development teams working within AWS environments can leverage Amazon CloudWatch to monitor resources and applications and AWS Identity and Access Management to monitor identities.

AUTOMATING COMPLIANCE

Automate regulatory compliance to ensure developers adhere to best practices without slowing them down or impeding their workflows. Organizations can leverage various automation tools and

processes that ensure compliance with cybersecurity best practices and Identity Access Management (IAM) policies. For example, AWS supports a number of compliance tools, such as AWS Config, AWS Organizations, and AWS CloudFormation.

Meanwhile, review all identities that have been inactive for a certain period on a regular basis to ensure no abandoned identities maintain unused or standing privileges.

SECURING THE PIPELINE

Secure all DevOps practices across all pipelines and secure access to all development tools employed across the pipeline (such as DevOps, automation, provisioning, and other CI/CD tools). Test both identity and access controls at every step along the development lifecycle and secure all applications involved (Chapter 4 will cover this topic in detail).

Meeting the Needs of the Developers

How do you fully protect developers across a wide variety of architecture without sacrificing either their autonomy or agility? Restricting developer access via layers of authorization presents a significant User Experience (UX) challenge. Developers need to move fast, and they require a lot of autonomy to successfully innovate.

SUSTAINING A HIGH-VELOCITY CULTURE

The modern developer culture demands constant velocity impeded by minimal speed bumps. Developers live in a world of intense demands for swift turnaround, so they often must rely on the benefits of resources shared among colleagues and innovative workflow shortcuts. For example, development teams commonly use code repositories to share the codebase and automate daily activities.

They also often perform ad-hoc tooling on the fly and automate everything that can possibly be automated. All these approaches have long been staple practices of their craft.

While these practices may satisfy the need for speed, they unfortunately tend to foster quick solutions with little regard for security, especially where traditional security practices do not integrate well with modern DevOps methods.

AVOIDING CUMBERSOME ROADBLOCKS

An example of security measures hindering a developer's workflow can sometimes be seen in the login process. For example, developers often must first log in to a PAM system to access their credentials, then log in to their native systems via PAM's session management. The result is that, from the developer's perspective, they now must go through one application to access another application or, worse yet, implement their own tooling.

This is a major headache when a developer simply needs to work natively on their own systems with their own credentials. The problem intensifies when the developer needs to work through a CSP's Command Line Interface (CLI) instead of through the web console.

Satisfying Both with Time, Entitlements, and Approvals

These challenges are not insurmountable. The lines of authority need not remain blurred, and cybersecurity teams don't need to become developers themselves to understand how to secure developer environments. Developers need not worry about losing their autonomy or agility, either. It is possible to satisfy both teams' requirements by implementing the next evolution of PAM, an approach known as Time, Entitlements, and Approvals (TEA).

The security team need no longer force developers to use standing credentials intended for "what if" contingencies. Instead, federated identities allow secure, direct access to each CSP's console and CLI.

FEDERATED IDENTITIES

Think of a federated identity as "one identity to rule them all." A federated identity links a user's identity to all identity management systems across all platforms, regardless of how disparate those systems may be, so that the user can access everything with a single set of credentials. Not only does it solve the problem of having so many points of entry across multiple CSPs, it also simplifies each developer's login process so they can easily and quickly access their resources.

Each federated identity has no default entitlements, thus providing ZSP in adherence to Least Privilege. Entitlements are only granted after appropriate approvals and only for the minimum window of time the developer requires at that moment. The session is fully monitored, and all entitlements will be revoked the moment it expires.

TEA also simplifies auditing. Since one federated identity is used regardless of the CSP used, it's easier to attest actions to a user.

TEA BENEFITS ALL

FIGURE 1 demonstrates how TEA can simultaneously improve security while lessening the developer's burdens. TEA can dramatically shorten how long an ephemeral session needs to exist, reduce how many entitlements are granted for a single session, and simplify the approval process for each user.

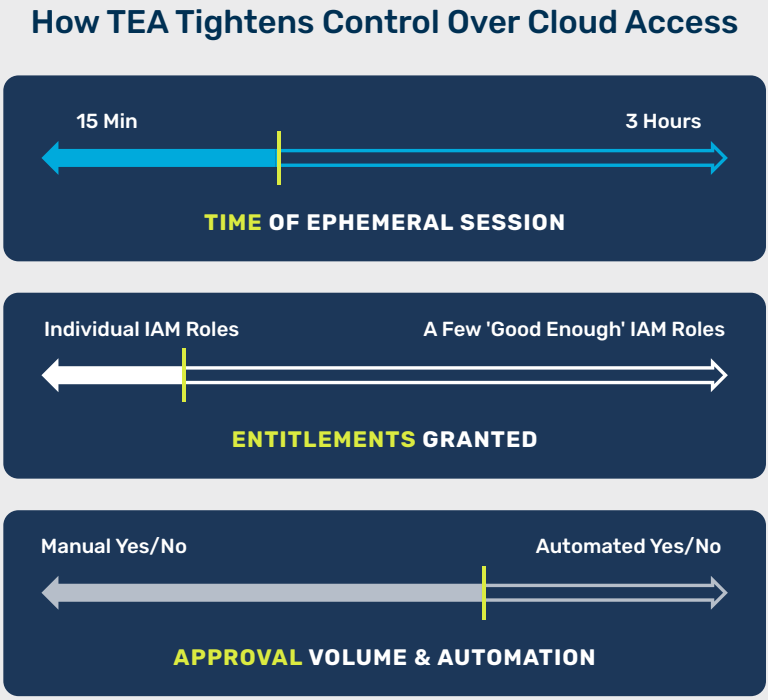


FIGURE 1: TEA results in shorter sessions, more restricted entitlements, and a simplified approval process

Implementing the TEA approach will give developers the UX they need, maximizing the ease-of-use CSPs provide without sacrificing security. Successful implementation strikes the right balance of time, entitlements, and approvals to everyone's benefit.

CHAPTER 2

Addressing the Machine Challenge

Not all user identities belong to humans—especially in production, operations, and developer environments that heavily leverage automation tools. A machine identity, sometimes referred to as a Non-Human Identity (NHI), is a unique identifier representing a non-human entity such as software code, an application, a workload, a virtual machine (VM), and so forth. These identities regularly use various credentials, such as API keys, cloud access keys, private keys, digital certificates, and secrets to authenticate to the other machines, applications, and other IT resources they need to connect and interoperate with. Attributes of the machine's identity are used to authorize the machine to access some or all the resources of another machine.

Adversaries often seek to compromise, steal, spoof, or recreate machine identities to gain the authentication and authorization needed to launch their attack. A lost, stolen, or otherwise compromised machine identity can in turn compromise the authentication and authorization between the machines it interacts with, leading to outages, security incidents, and breaches. Adversaries also move laterally across human and machine identities.

Critical, but Too Often Overlooked

Secure machine identities are critical to cybersecurity, preventing attackers from compromising machine-to-machine authentication and authorization; however, organizations often unintentionally fail to adequately prioritize machine identities and therefore fail to fully address them.

Many cybersecurity teams unfortunately do not always secure and rotate machine identities' credentials in the same manner as human identities' credentials, which inadvertently leaves many machine identities inadequately secured. Attackers far too often take advantage of unprotected machine identities as they increasingly provide a key entry point.

Recent research revealed that many organizational leaders simply don't realize that not all identities belong to humans. For example, 61% of those surveyed strictly defined a privileged user as a human identity. In fact, machine identities are estimated to outnumber human identities by a factor of 45 to 1. Meanwhile, 68% of respondents indicated that 50% of all machine identities have access to sensitive data. The rapid rise in prominence of AI means these findings are more important than ever; indeed, 93% of those surveyed expect AI to have a negative impact on cybersecurity.

Modern digital transformation efforts are only accelerating their growth as organizations expect an 89% increase in SaaS application deployments going forward, which means even more applications, containers, automation tools, and VMs than ever before and a corresponding growth in the secrets associated with them. Despite this explosive growth, only 65% of respondents stated they either plan to or are currently taking steps to protect machine identities within the next 12 months as of the time of the survey.

Securing both human and machine identities is critical to securing the organization. Uber learned this the hard way during the fall of 2022, when an attacker moved between machine and human identities to ultimately gain full administrative access to a solution that granted privileged access across the IT organization. Uber narrowly escaped severe harm from an attack vector accidentally opened by a developer simply embedding a credential in a basic PowerShell script (see **FIGURE 2**).

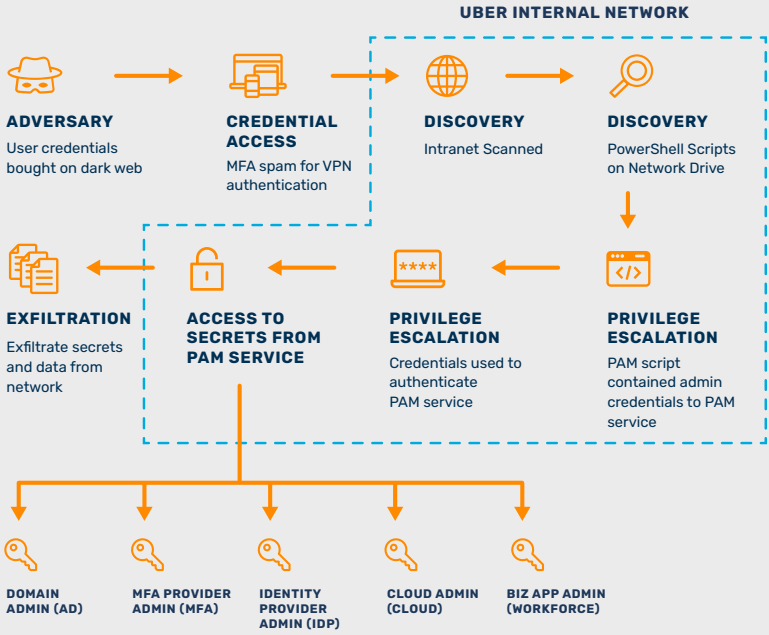


FIGURE 2: A roadmap tracking the attacker's journey through Uber's infrastructure

Uber's attacker purchased stolen employee credentials from a dark web marketplace to gain VPN access, then used social engineering to bypass the platform's MFA mechanism by texting an Uber worker under the guise of corporate IT.

SECURING NON-HUMAN ACCOUNTS

Protecting non-human accounts should begin with implementing identity management policies that not only govern machine identity generation, but also the human users that administer the tools and processes which create them (e.g., DevOps admins, certificate admins). Furthermore, these policies must include mechanisms for both the renewal and revocation of existing machine identities. Also make sure to include machine identities when centralizing identity and secrets management to enhance visibility, which will further secure certificate provisioning across the enterprise's IT infrastructure, including hybrid and multi-cloud architecture.

Unique Considerations

Machine identities merit unique considerations. Security concerns should regard applications, cloud workloads, and all the other resources automated, non-human accounts interact with.

Adopting an identity policy that accounts for machine identities will grant the organization greater visibility into its IT environment and allow the monitoring and tracking of machine activity, including the discovery of managed and unmanaged machine identities.

A secret is a credential used to authenticate and authorize access, such as a password, key, or token, while a certificate is a special file that verifies a digital entity's authenticity via cryptography.

Key Steps for Securing Machine Identities

When the cybersecurity team helps secure machine identities (unless they have fully shifted left—more about that in the next chapter), one may almost assuredly assume machine identities such as secrets and certificates already exist.

There may be many machine identities to review. Some may be in use, others not. The cybersecurity team may manage some while various parties (such as development teams or IT) may manage others, sometimes without cybersecurity's awareness. Some machine identities (perhaps those left over from the application's development) may not be managed at all. The bottom line: It is extremely likely there are machine identities, secrets, and certificates that cybersecurity is unaware of, so how can they assess and manage the associated risks? It's tough being assigned responsibility for something you aren't aware of!

FIGURE 3: Three key phases of lifecycle management

The following three phases of lifecycle management (see **FIGURE 3**) serve to rectify this dilemma and ensure that all types of machine identities are effectively secured.

DISCOVERY AND VISIBILITY

The first step in managing machine identities is to discover the secrets and certificates being used and ensure their visibility. Consider, for example, development teams working in AWS, securing their secrets via AWS Secrets Manager (ASM). The cybersecurity team can only be aware of these secrets if the development team notifies them. What if they aren't notified?

Each development team may establish their own ASM instances, resulting in secrets stored across multiple vaults. Discovery therefore begins by scanning all AWS accounts. Ideally, scanning across the AWS Organization will determine which secrets exist, which are in use, and which are managed or unmanaged. The cybersecurity team can then assess the risk and prioritize actions to improve security accordingly.

A similar approach may be applied to certificates. Though a certificate manager (such as Amazon Certificate Manager [ACM]) may manage certificates within AWS environments, it likely does not actively manage all certificates, nor may the cybersecurity team be aware of them all. Certificate discovery, like secrets discovery, should therefore be used to identity certificates across the entire organization, including those set up by shadow IT functions.

ONBOARDING AND MANAGEMENT

Once security teams have discovered all secrets and certificates, onboard them into a centralized management solution platform that can secure and manage them across the organization's entire compute environment (including multi-cloud, hybrid, and even on-premises environments). New secrets or certificates may also be created by developers or certificate admins, so take care to onboard these as well.

Take care not to just onboard managed secrets. Onboard unmanaged secrets as well.

Note that though some individual secrets and certificates can be onboarded and managed using a UI, manual secrets management processes are inevitably prone to errors and cannot keep up with how rapidly modern, dynamic hybrid and multi-cloud environments evolve. Automated processes are preferred because they scale far better. For example, when users manage and provision cloud infrastructure to run applications, secrets can be automatically onboarded into the secrets manager. Provisioning tools can integrate into the secrets manager to automate onboarding. Some tools even support onboarding secrets for multi-cloud environments. Once all secrets are onboarded, the secrets manager can secure them and enable rotation based on factors like policy, audit data, and so forth.

GOVERNANCE AND LIFECYCLE MANAGEMENT

Rotate all credentials (save for any necessarily dynamic credentials) based on policy to protect and secure them, meet compliance requirements, and minimize risk. Meanwhile, manage all machine identities across their entire lifecycles, from creation to deletion and removal. Delete a secret once it's no longer required or used by any application or tool so that an attacker cannot inadvertently discover it. Remember, the attacker doesn't care that a code snippet or script is no longer used; they just want a valid credential they can exploit (as demonstrated by the aforementioned Uber example). Significant and predictable operational consequences may follow if certificates are not precisely managed across their entire lifecycle. Assume an outage will occur otherwise.

Certificate lifecycle management has recently grown in importance, often reducing certificate lifecycles to as little as 90 days. Take care when determining lifecycle duration. In some cases, browsers will no longer accept certificates from previously accepted Certificate Authorities (as has been the case with Entrust's certificates).

No cybersecurity or operations team wants to "discover" an inadequately managed certificate via a web page failing to load in Chrome or a network losing access due its sudden expiration.

The certificate management solution must ensure adequate time is allowed for certificate renewal processes, given the involvement of external parties. Examples of renewal processes include retrieving a renewed certificate from the respective Certificate Authority, updating applicable infrastructure, and testing, all of which must be completed before the go-live date and the old certificate's expiration.

Paper Trail

Don't forget to continuously monitor and regularly audit all machine identities. You need as much of a "paper trail" for them as you do for human identities.

NO SECRETS IN THE SHADOWS

A reliable, centralized secrets management solution can help plug any potentially leaky holes in the ship. Managing all identities (human and machine), secrets, and certificates in one centralized, enterprise-wide system can shine a light on any lingering shadow IT infrastructure and ensure consistent visibility, security, and compliance.

CHAPTER 3

Securing the CI/CD Pipeline

As digital transformation drives massive, rapid change across industries—established and emerging ones alike—development teams are pressured to create and deploy applications and features faster than ever. Many organizations are adopting Continuous Integration (CI) and Continuous Delivery (CD) methodologies (as introduced in Chapter 2) to keep up with competition by rapidly developing and deploying new applications and features.

CI leverages automation tools and methods to streamline development and testing, the product of which immediately feeds into CD, which similarly implements automation to validate and deliver (deploy) the finished application. The workflows between the two methodologies are largely and potentially entirely seamless, hence we refer to the combined processes as the CI/CD pipeline.

Checks and Balances

Note that many (especially larger) organizations maintain clear separation and segmentation between development and production environments, including checks and balances and steps to force manual review before deploying new code to production.

The Need to Secure Human and Machine Access

Human and machine identities must be secured and their access requirements managed as applications advance from development to production across the development lifecycle (see **FIGURE 4**). Requirements, however, are prone to change. For example, development environments typically require that developers and other humans have access to cloud resources, code repositories, and various other assets. Conversely, applications deployed to production environments require little if any human involvement. Machine identities and secrets are often used to support elastically scaling the application or service. Human access in production environments should therefore only be needed as an exception—one that is extremely, tightly controlled.

Methodologies such as ZSP provide secure access while preventing over-provisioning (such as giving developers and DevOps admins extensive, permanent access to build environments, code repositories, or cloud resources). These controls must be doubly restrictive

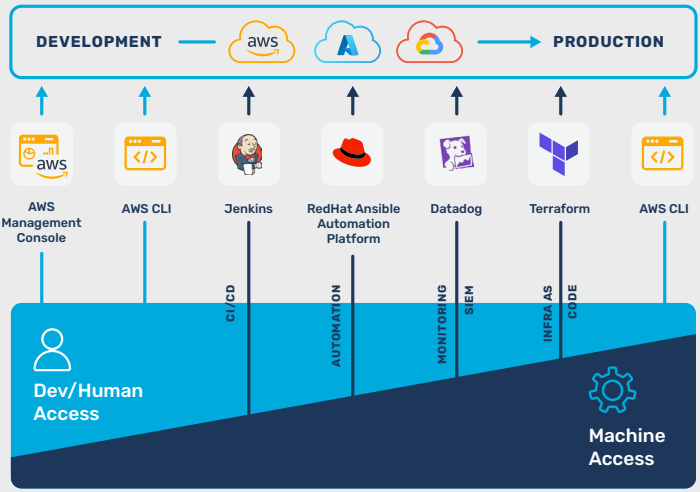


FIGURE 4: An example of how the mix of human and machine identities shifts as code advances to production

and establish privilege controls and governance for access to the build and production environments for both human and machine access. The organization will be vulnerable unless all identities—human and machine—are secure. Integrating identity security into all CI/CD pipelines seamlessly weaves identity integrity directly into the development lifecycle—from beginning to end—so that security isn't an afterthought.

These models grow exponentially more complex when they scale to larger enterprises (see **FIGURE 5**). A single organization can easily have a wide array of development projects spanning multiple development teams, CI/CD pipelines, and complex build and production environments.

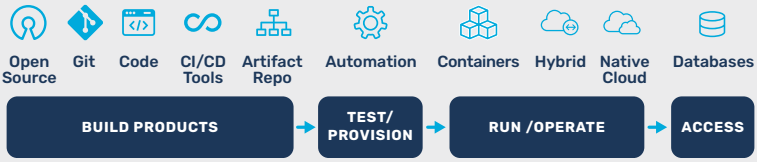


FIGURE 5: An example of a single enterprise-scale development process through complex workflows and environments

Organizations can leverage various automation tools and integrate security into both DevOps processes and CSP's services. However, unless security is built in or requires little to no change in the developer's experience and workflows, achieving a high level of developer adoption can be challenging. If security slows down the development process, then developers, though they want to ensure the security of their code and apps, are nonetheless likely to find workarounds to maintain efficiency. Security processes should therefore be automated or avoid making changes to developer workflows so that they can help developers accelerate the development and deployment of applications and services.

Security Across the Pipeline

Security holes can open at any stage of software development, from the very first snippets of rough code to full pre-release builds. The modern lifecycle also involves many different parties with their own security concerns. For example, developer teams write and test code, DevOps (and Ops) teams build the various systems the code moves through, and cybersecurity teams get involved at some point (unfortunately, often toward the end) to ensure a secure deployment.

This means there are many different types of people, identities (human and machine), processes, environments, automation tools, complex integrations, and supply chains across each of an organization's various CI/CD pipelines that can potentially introduce a vulnerable point of entry for attackers. It only takes one point of vulnerability—one brief window of opportunity—for a vigilant attacker to devastate an organization.

SECURING SOURCE CODE

As noted in the first chapter, secrets and other credentials (such as application credentials and cloud access keys) should never be hard-coded into source code under any circumstances. When using a popular code repository (such as GIT), it's easy to inadvertently configure it—and the credentials within—to be publicly accessible. It's especially tempting during the earliest stages of development and testing with the first lines of rough code, long before any real release is on the horizon. Meanwhile, attackers constantly use tools to persistently scan repositories for exposed credentials, and the results can be catastrophic.

Remember, developers often leverage shortcuts to maintain velocity. Writing credentials directly into the source code can be relatively fast and simple, but can result in deadly consequences.

Code signing ensures that any third-party code pulled from a repository for use in a build is legitimately developed and validated by the third party and does not include an attacker's injected code. Additional techniques can further prevent malicious code injection. For example, code minification that removes unnecessary characters

from sensitive code and code obfuscation tools rename functions and variables to obscure their purpose and can add redundant code to confuse an intruder.

Another best practice is to segment access to the codebase. For example, only grant development teams access to the specific codebase they need to perform their jobs and restrict which administrators control access.

SHIFTING LEFT

Another essential approach is to involve the cybersecurity team early in the development cycle to help build secure applications—a concept often referred to as "shifting left" (see **FIGURE 6**). This helps prevent encountering intrusive security problems when the code is near deployment and after critical security decisions have already been made.

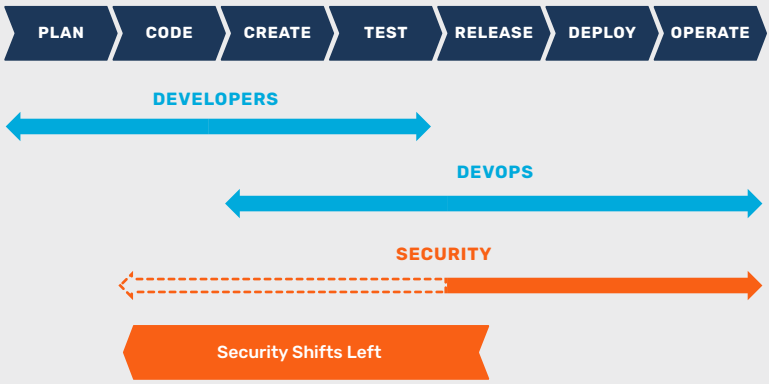


FIGURE 6: Example of a "shift-left" model where the cybersecurity team is involved earlier in the lifecycle

SECURING DEVELOPER INFRASTRUCTURE

Beyond the code itself, it's equally imperative to secure all the development team's application computing resources. Secure databases with strong, parameterized queries and encode the resulting output. Lock down file management by validating file types and restrict types to only those that meet business requirements, then ensure only authenticated users can upload files.

Don't just lock down the tools and platforms that make up the build environment, though; lock down processes as well. Structure all build processes with security in mind from their inception. For example, as mentioned earlier, segment access to the codebase, perform a manual review after pivotal pull requests (e.g., by forcing MFA with a development team's manager), and always discard the build environment after it has served its purpose so that no credentials can possibly be pulled from it. You should also integrate automation tools and Infrastructure-as-Code (IaC) tools (such as Terraform or Ansible) with your identity security management solution.

Securing all development pipelines, processes, and build environments ultimately makes it possible to develop secure applications, many of which are often themselves SaaS applications deployed to

Pipeline Tools

Developers utilize a wide array of tools throughout the pipeline. For example, AWS offers multiple solutions for its environments that each serve different CI/CD functions, such as AWS CodeCommit, AWS CodeBuild, AWS CodeDeploy, and AWS CodePipeline.

cloud environments. Also ensure that development environments are always entirely segmented and separated from production environments to ensure that no identities, credentials, or resources are ever shared between the two. In fact, few—if any—developers should ever have access to production environments.

Supporting Developer Autonomy

We all know you cannot force security on developers. If security measures are onerous, they'll simply find a workaround. Instead, empower cybersecurity teams to *"meet developers where they are"* and enhance their freedom to innovate by implementing automated system-level actions so that the cybersecurity team doesn't need to intervene with hands-on measures that would disrupt a developer's workflow.

INTEGRATE & AUTOMATE

For example, implement a solution that will automatically check all source code for hard-coded secrets and other vulnerabilities during check-in. If the scan catches a secret, it should immediately generate an alert so that a rapid response can prevent what might otherwise result in a disastrous leak.

Leverage code signing certificates in development environments that differ from those used in production environments. Similarly, ensure development keys utilize a different root of trust than production keys. Automatically generate logs for all code commits, code signings, certifications, and certificate authorizations to ensure audit requests can be easily met and don't become burdensome.

Seamlessly integrated and automated solutions satisfy everyone's requirements, ensuring development environments are secured without interrupting the developer's workflow or running them through obtrusive manual security processes.

Beware Weak Links in the Supply Chain

Regardless of all the security measures implemented across the development architecture, at the end of the day, the organization is only as secure as the weakest link in its supply chain.

Every single entity that touches source code or its development process in any way is considered a link in the supply chain. This includes deployment infrastructure, APIs, development tools, and third-party source code (such as proprietary or open source code).

Consider again the infamous SolarWinds attack as a cautionary tale.

The breach was a dramatic wakeup call for many organizations and their executives and served as a powerful call to action for software developers to ensure the security of their digital supply chains.

The SolarWinds Incident

An attacker compromised the SolarWinds CI/CD pipeline in 2020, back when securing the software supply chain was not widely recognized as a concern. The attacker injected their own code into the company's proprietary source code, which was then inadvertently deployed by tens of thousands of SolarWind customers.

By modifying the code, the attacker deployed their malware, "SunSpot," which ran with elevated privileges as it scanned various builds of SolarWinds Orion (the company's IT monitoring product), then inserted its own well-hidden backdoor.

President Biden even signed an Executive Order to introduce new regulations, including federal implementation of Zero Trust cloud architecture, supply chain security improvements, and the implementation of a Cybersecurity Safety Review Board.

95% of organizations have suffered some form of breach along their supply chain, but the story doesn't end there. The good news is that these breaches have raised awareness across industries, leading to the advancement of more robust security tools and methodologies going forward.

CHAPTER 4

How CyberArk and AWS Work Together

CyberArk's solutions address and solve each of the challenges detailed in these chapters with capabilities designed to help protect enterprises' developers, environments, cloud architectures, CI/CD pipelines, human and machine identities, and much more, all without adding obstacles to developers' agility. Most importantly, many of these solutions are designed to enhance and improve existing capabilities available from AWS.

A Holistic Approach

Implementing comprehensive identity security across all varieties of enterprise architecture and across all types of human and machine accounts is no simple task. CyberArk recommends a pragmatic, holistic, phased approach that eases the transition while leaving no stone unturned.

Refer to the CyberArk Blueprint for an example. It provides a prescriptive framework with a risk-based approach to building privilege management programs for both human and machine identities. Protecting human identities across the workforce (including IT identities, developer identities, and so forth) is based on three guiding

principles: prevent credential theft, stop lateral and vertical movement, and limit privilege escalation and abuse across all environments (including hybrid and multi-cloud environments).

The Blueprint's first three stages address Infrastructure-as-a-Service (IaaS) administrators, CI/CD consoles, and environments' various dynamic applications to ensure all privileges are secured across all development and deployment processes.

CyberArk AWS Solutions

AWS already provides solid capabilities to secure developer and machine identities, but as environments become more complex, additional capabilities from security partners can strengthen and streamline the enterprise's security posture. In many cases, CyberArk's Identity Security Platform builds on and strengthens existing AWS capabilities by providing greater granularly of access, ZSP, centralized secrets discovery across AWS and multi-cloud environments, centralized secrets management (regardless of compute environment or application type), and policy-based rotation.

CyberArk is an official Amazon Web Services advanced technology partner and has achieved AWS Security Competency and AWS Digital Workplace Competency. Furthermore, CyberArk is AWS Outposts Service Ready and counts over 100 Certified AWS Solutions Architects among its teams.

For example, CyberArk's cloud security and machine identity solutions go beyond simply securing SaaS apps with SSO; they secure every user's web session and protect every cloud VM with dynamic privilege access, all the while ensuring seamless Just-In-Time (JIT) access to CSPs via a ZSP methodology.

SEAMLESS AWS INTEGRATION

CyberArk offers the broadest set of solutions that seamlessly integrate with AWS Cloud services across multiple levels. In fact, CyberArk's portfolio is available in the AWS Marketplace, where you can find more than 25 out-of-the-box AWS integrations.

Each service integrates with a corresponding AWS service to ensure daily operation remains smooth while enhancing secure access to AWS resources. CyberArk solutions even interact directly with AWS Security Hub to improve threat detection.

SECURING HUMAN AND MACHINE IDENTITIES

The CyberArk approach to multi-layer identity security is specifically purpose-built for AWS. CyberArk services provide JIT access to AWS resources with ZSP. These services secure AWS EC2 access, DevOps tools and processes, console access, and WorkSpaces instances.

Meanwhile, CyberArk safeguards all human and machine identities via a centralized service that protects all AWS Secrets Manager instances.

DEVELOPER VELOCITY & AUTONOMY

CyberArk solutions resolve the challenge of satisfying both cybersecurity and developer teams' requirements by maintaining each developer's velocity and autonomy without sacrificing secure access to developer resources.

CyberArk adheres to a philosophy of "meeting developers where they are" by providing completely transparent integrations with CSP's native vaults, cloud-agnostic APIs, dynamic secrets, and automated secrets rotation. These solutions even conveniently integrate with ChatOps and IT Service Management (ITSM) platforms.

CENTRALLY DISCOVERING, MANAGING, & ROTATING SECRETS IN AWS SECRETS MANAGER

Developers are increasingly securing secrets in secrets stores such as AWS Secrets Manager, which is a great first step to securing these machine identities. Security teams typically face some key challenges, though.

First, the security team likely does not have visibility into all the developers' secrets stores, therefore there will be secrets stored and used that they are unaware of. That poses quite the challenge given that the security team is responsible for, well, security.

Second, there is likely a multitude of vaults, many of which likely store copies of the same secret. This vault sprawl makes tracking, auditing, and secrets rotation challenging, especially given that organizations are often reluctant to rotate a secret if they don't know everywhere it is used. Rotating such a secret could very well cause an outage or application failure.

Third, policies implemented to meet an enterprise's compliance requirements typically require secrets rotation.

Addressing these challenges requires a centralized means of not only managing secrets but automating their rotation as well, ideally without ever impacting how developers use their CSP's native secret stores.

CyberArk Secrets Hub does exactly that for secrets in AWS Secrets Manager and other cloud environments. For example, CyberArk Secrets Hub can scan all secret stores in an AWS Organization to provide security teams with visibility across all their various secrets stores. Security teams can discover previously unknown secret stores and secrets while gaining insights into both managed and unmanaged secrets, such as when a secret was last used or rotated. Once secrets have been discovered, the security team can establish

risks and priorities. Meanwhile, integrations with provisioning tools such as Terraform enable organizations to automate onboarding existing and newly created secrets.

CyberArk earned the highest score for the Secrets Management use cases in Gartner's® 2024 and 2023 Critical Capabilities for Privileged Access Management while CyberArk was named a leader in Gartner's 2024 and 2023 Magic Quadrant™ for Privileged Access Management.

SIMPLIFYING & CENTRALLY SECURING ACM-MANAGED CERTIFICATES

While Amazon Certificate Manager (ACM) provides powerful certificate management capabilities, CyberArk's TLS Protect certificate lifecycle management solution helps simplify the enrollment of certificates from Amazon's and others' CAs into ACM and provision them to support cloud services. CyberArk also provides automation tools to improve the IT team's efficiency and help simplify certificate management and automatic renewal across multi-cloud and hybrid environments with centralized certificate discovery and management.

CyberArk solutions also help simplify certificate provisioning with the existing Amazon Elastic Load Balancer (which automates provisioning from any CA) and Amazon CloudFront (which automates the entire certificate lifecycle by provisioning certificates from any CA). These features enable automatic certificate revocation upon terminating an Amazon EC2 instance.

SECURE CLOUD ACCESS

CyberArk Secure Cloud Access secures multi-cloud environments via one single platform to apply global access policies. CyberArk follows a "no agents, no drama" approach, keeping usage simple and preserving native access to cloud consoles across multiple cloud platforms.

Secure Cloud Access maintains ZSP and adheres to Least Privilege via on-demand JIT elevation mechanisms to ensure users only get the right permissions to the right resources at the right time. This even extends to those critical "what-if" scenarios by ensuring engineers can securely request and quickly receive dynamic break-glass access during an emergency.

ACCELERATED TIME-TO-COMPLIANCE

CyberArk solutions accelerate time-to-compliance with the Systems and Organization Controls (SOC) 2, the National Institutes of Standards and Technology (NIST) agency's standards, and various other industry standards via access control mechanisms that maintain ZSP.

Learn More

You can learn more today about securing all identities across all architectures by checking out CyberArk's services in the AWS Marketplace or by requesting a demo from CyberArk so you can get a first-hand look at the solutions yourself.

ABOUT CYBERARK

CYBERARK®
The Identity Security Company

CyberArk is the global leader in identity security. Centered on intelligent privilege controls, CyberArk provides the most comprehensive security offering for any identity — human or machine — across business applications, distributed workforces, hybrid cloud workloads and throughout the DevOps lifecycle. The world's leading organizations trust CyberArk to help secure their most critical assets. To learn more about CyberArk, visit www.cyberark.com, read the CyberArk blogs or follow us on Twitter via @CyberArk, LinkedIn, or Facebook.

ABOUT ACTUALTECH MEDIA

ActualTech Media, a Future B2B company, is a B2B tech marketing company that connects enterprise IT vendors with IT buyers through innovative lead generation programs and compelling custom content services.

ActualTech Media's team speaks to the enterprise IT audience because we've been the enterprise IT audience.

Our leadership team is stacked with former CIOs, IT managers, architects, subject matter experts and marketing professionals that help our clients spend less time explaining what their technology does and more time creating strategies that drive results.

If you're an IT marketer and you'd like your own custom Gorilla Guide® title for your company, please visit actualtechmedia.com.