JOURNAL OF INTERNATIONAL CRISIS AND RISK COMMUNICATION RESEARCH
ISSN: 2576-0017
2025, VOL 8, NO S12

Zero-Touch Secrets Management: Securing Ci/Cd Pipelines Through Ephemeral Credentials

Sameer Lakade

Independent Researcher, USA

1. Abstract

Modern continuous integration and deployment pipelines rely on sensitive credentials for accessing cloud services, artifact repositories, and deployment targets, creating significant security vulnerabilities when these secrets are managed through traditional static credential approaches. This research presents a zero-touch secrets management architecture that eliminates persistent credentials from automated build environments by implementing ephemeral, identity-bound credentials with just-in-time issuance and automatic revocation. The proposed system establishes pipeline identity through cryptographic attestation rather than shared secrets, enabling workload-specific credential generation with strict temporal and scope limitations measured in minutes rather than months. Through identity-based encryption, credentials remain cryptographically bound to their intended execution context, preventing lateral movement even if intercepted during transmission. The architecture integrates trust authorities for identity validation, secrets brokers for dynamic credential generation, and comprehensive audit systems for forensic traceability. Implementation considerations address the challenges of cryptographic algorithm selection, policy engine design for attribute-based access control, and integration with diverse pipeline orchestration platforms. The zero-touch approach fundamentally reduces attack surfaces by ensuring credentials exist only during active use, automatically rotate without operational intervention, and provide complete audit trails for compliance and investigation. This paradigm shift aligns with zero-trust security principles, transforming CI/CD credential management from a persistent vulnerability into an autonomous, self-securing system that maintains security without sacrificing developer velocity or operational efficiency.

Keywords: Ephemeral Credentials, Zero-Touch Secrets Management, CI/CD Security, Identity-Based Authentication, Just-In-Time Access Control.

2. Introduction

Advanced software development uses automated continuous integration pipelines and deployment pipelines with thousands of build processes running each day on a distributed infrastructure. These automated workflows need credentials: API keys to cloud services, database passwords, artifact repository tokens, and deployment certificates that are sensitive. The old methods of dealing with such secrets have presented a key security weakness: they employ static credentials that remain over time, and are used in a variety of execution contexts, and hard to rotate without disrupting the current workflows.

The traditional secret management paradigm assumes that credentials are enduring artifacts, which are stored in vault systems or encrypted configuration files. Although this is a step up

over using hard-coded passwords, there remains a need to provision, issue, and ultimately rotate long-lived tokens that, in turn, must be handled by human operators or automated systems. The more time a secret is held, the larger the compromising window of opportunity is. Secrets spread within development environments have emerged as a very significant concern because hardcoded credentials within source-code repositories are one of the most recurring weaknesses in software development practices. The studies that have investigated secret detection tools found that detecting and controlling exposed secrets is very difficult and that the various detection mechanisms have various degrees of capabilities in detecting API keys, authentication tokens, and cryptographic secrets across diverse codebases [1]. The content of secret management is further enhanced by the fact that secrets are usually in various forms and locations during the software development life cycle, and from the initial development environment up to production deployments.

Security breaches in the past have made it known that vulnerable credentials in automated build systems can propagate to full-scale breaches of infrastructure, allowing adversaries to use a single compromised token to gain full access to the system. Software supply chain has become not only a highly attractive attack surface but also a notably weak protection point as compromised credentials may be used to introduce rogue code, steal valuable data, or destroy essential services. Credential compromise has been identified as one of the fundamental attack vectors of supply chain security threats, and examples that belong to dependency confusion attacks and malicious code injection by compromised build systems have been reported. The interconnectedness of contemporary software development, in which applications and projects are dependent on a large number of 3rd-party dependencies and automated build systems, provides ample chances for attackers to take advantage of revealed secrets at any point in the pipeline [2]. Such vulnerabilities are especially worrying considering that most organizations have a weak visibility of their hidden patterns of usage, and a number fail to implement an automatic rotation process and a consistent security policy over dispersed development groups.

CI/CD environments pose a challenge of maintaining secrets because of the distributed and short-lived nature of modern cloud infrastructure, where build agents can be dynamically deployed in various regions as well as cloud providers. Conventional security models, which are based on network perimeter protection or fixed credential management, cannot work in these dynamic environments. Moreover, the human factor of secret management can become another source of risk, since a developer can involuntarily commit credentials to the versioning system, transfer secrets across insecure communication channels, or not revoke access when a person shifts positions or quits organizations. The delay period between the secret exposure and detection provides long periods of vulnerability where attackers can use compromised credentials.

This problem requires the approach of a fundamental change: getting rid of the static secrets in pipeline execution environments and introducing temporary, automatically managed credentials that only last as long as their intended use. With the pivot to transient credential models, organizations will be able to significantly reduce the attack surface and decrease the effect of any given compromised credential.

3. Core Architectural Principles

Zero-touch secrets management rests on three foundational principles that work in concert to eliminate persistent credentials from automated workflows.

3.1 Identity-First Authentication

Rather than authenticating build processes through shared secrets, the system establishes identity through cryptographic proof bound to specific execution contexts. Each pipeline job receives a unique identity token that encodes its build metadata—repository origin, commit identifier, runner location, and execution timestamp. This identity becomes the basis for all subsequent credential requests, eliminating the need for any pre-positioned secrets in the build environment. The shift toward identity-based authentication mechanisms represents a fundamental departure from traditional credential-based approaches, leveraging cryptographic

attestation to establish trust relationships between pipeline components and resource providers. Modern workload identity frameworks employ standardized protocols that enable secure identity federation across heterogeneous cloud environments, allowing build processes to authenticate without relying on static credentials embedded in configuration files or environment variables. The challenge of establishing identity across distributed systems requires sophisticated event correlation mechanisms that can trace execution contexts across multiple service boundaries. In complex distributed environments, the ability to correlate events and establish causal relationships between system activities becomes critical for maintaining security and operational visibility [3]. These identity tokens typically incorporate claims that describe the execution context in granular detail, including the specific workflow being executed, the branch or tag being built, the actor who triggered the pipeline, and the runtime environment characteristics. By binding authentication to verifiable identity attributes rather than shared secrets, organizations can implement fine-grained access control policies that evaluate the complete context of each request, enabling authorization decisions based on factors such as repository ownership, code review status, and deployment target environment.

3.2 Just-in-Time Credential Issuance

Credentials materialize only when needed and disappear immediately after use. A central trust authority validates each identity token against policy rules before authorizing a secrets broker to generate time-limited credentials. These ephemeral secrets carry strict scope limitations, granting only the minimum permissions required for specific build stage operations. The system enforces aggressive expiration policies measured in minutes rather than days or months. The just-in-time provisioning model fundamentally transforms the temporal characteristics of credential exposure, replacing long-lived static secrets with dynamically generated credentials that exist only for their intended operational window. The implementation of time-bound credentials requires sophisticated orchestration between identity providers, policy engines, and target resource systems, ensuring that credential generation, validation, and revocation occur seamlessly within the performance constraints of automated build pipelines. Modern cloud platforms have increasingly adopted short-lived credential models, with some implementations supporting validity periods as brief as fifteen minutes for highly sensitive operations. The temporal limitation of credentials operates in conjunction with scope restrictions that constrain the permissions granted to each ephemeral secret, ensuring that even if credentials are somehow exfiltrated during their brief validity window, they provide access only to the minimal set of resources required for the specific build stage operation.

3.3 Cryptographic Binding

Identity-based encryption ensures that even if credentials are intercepted during transmission, they remain useless to unauthorized parties. Secrets are encrypted specifically for the requesting job's identity, making decryption possible only within the authorized execution context. This cryptographic binding prevents lateral movement—a compromised build agent cannot use credentials intended for different pipelines or stages. The application of identity-based encryption to credential distribution creates an additional security layer that protects secrets even in scenarios where network communications are compromised or infrastructure is partially infiltrated. By encrypting credentials such that only the holder of the correct identity credential can decrypt them, the system ensures that intercepted secrets remain cryptographically protected against unauthorized access. Cloud forensics research has highlighted the critical importance of maintaining cryptographic integrity and chain of custody in distributed computing environments, where traditional forensic approaches face significant challenges related to data volatility, multi-tenancy, and jurisdictional complexity [4]. This approach effectively transforms credentials into identity-specific artifacts that lose their utility outside the precise execution context for which they were generated, preventing scenarios where stolen credentials can be replayed or reused across different pipeline executions or organizational boundaries. The cryptographic binding mechanism provides

defense-in-depth protection that remains effective even when other security controls fail or are circumvented.

Table 1: Multi-Layer Security Controls in Zero-Touch Secrets Architecture [3, 4]

Security Layer	Control Mechanism	Protection Scope	Implementation Component	Failure Mode
Identity Verification	Cryptographic attestation [3]	Pipeline authentication	Trust Authority	Deny access
Temporal Constraint	Time-bound validity	Credential lifespan	Secrets Broker	Auto-revocation
Scope Limitation	Least-privilege permissions	Resource access	Policy Engine	Minimal permissions
Cryptographic Binding	Identity-based encryption [4]	Credential confidentiality	IBE Engine	Decryption failure
Audit Recording	Immutable logging	Forensic traceability	Audit Store	Read-only preservation
Automatic Revocation	Session invalidation	Post-execution cleanup	Lifecycle Controller	Force termination

4. Operational Workflow

The zero-touch system operates through a precisely orchestrated sequence that executes automatically during each pipeline run, implementing a multi-stage protocol that balances security requirements with operational efficiency.

When a build job initializes, it requests an identity token from the trust authority, providing signed metadata about its execution context. The authority validates this metadata against predefined trust policies—verifying repository ownership, runner integrity, and execution parameters—before issuing a short-lived identity credential. This initial authentication phase represents a critical trust establishment moment where the system must verify that the requesting entity possesses legitimate authorization to execute within the pipeline infrastructure. The validation process examines multiple dimensions of the execution context, including the cryptographic signatures associated with the build trigger, the provenance of the source code being processed, and the environmental characteristics of the runner infrastructure. Modern identity verification systems employ sophisticated attestation mechanisms that can detect anomalies in execution patterns, identifying potentially compromised runners or unauthorized pipeline modifications. The architecture of access control in cloud-native environments has evolved significantly to address the challenges of dynamic, distributed workloads, where traditional perimeter-based security models prove inadequate. Research examining access control design patterns across cloud-native architectures has identified critical considerations for implementing fine-grained authorization in containerized environments, including the need for policy-based decision engines that can evaluate context-rich authorization requests in real-time [5]. The trust authority maintains comprehensive policy definitions that encode organizational security requirements, translating high-level security objectives into enforceable technical constraints that govern credential issuance decisions.

Armed with this identity token, the build job contacts the secrets broker to request specific credentials needed for its assigned tasks. The broker evaluates access policies, determines the minimum required permissions, and dynamically generates appropriately scoped credentials. For cloud API access, this might involve creating temporary service account keys with limited resource permissions. For artifact repositories, it generates read-only tokens scoped to specific package namespaces. The credential generation process implements the principle of least privilege, ensuring that each issued secret grants only the minimum access necessary to complete the intended operation. This scoping mechanism requires deep integration with target resource systems, enabling the broker to translate abstract permission requirements into

platform-specific access control configurations. The broker must maintain current knowledge of available permissions across diverse target systems, from cloud infrastructure providers to container registries to database management systems.

The broker encrypts these credentials using identity-based encryption, ensuring only the requesting job can decrypt them. Decryption occurs entirely in memory within the secured build environment, with no secrets ever persisting to disk or appearing in logs. Throughout execution, the system maintains detailed audit records of every credential request, issuance, and usage event. The comprehensive logging infrastructure captures forensically valuable metadata that enables post-incident investigation and compliance verification. Role-based access control mechanisms have proven particularly effective in managing authorization complexity in distributed systems, where the assignment of permissions to roles rather than individual entities simplifies administration and enhances security posture. Modern implementations of role-based access control have been adapted to address the unique challenges of resource-constrained and highly distributed environments, demonstrating that carefully designed authorization frameworks can maintain security properties while operating within stringent performance requirements [6]. These audit records must be stored in tamper-evident systems that prevent retroactive modification, ensuring the integrity of forensic evidence. The logging system captures temporal information with sufficient precision to reconstruct the exact sequence of events during pipeline execution, including the specific moment when credentials were issued, the duration of their active use, and any attempts to access resources beyond the authorized scope.

Upon job completion or timeout, the broker immediately revokes all issued credentials and invalidates the session. This automatic cleanup ensures credentials cannot outlive their intended purpose, even if the build process crashes or is maliciously terminated. The revocation mechanism must operate reliably across failure scenarios, implementing fail-secure behaviors that default to credential invalidation when uncertain about job status.

Table 2: Temporal Security Properties in Zero-Touch Credential Management [5, 6]

Event	Duration	Security Property	Forensic Value
Identity Token Issuance	5-10 minutes	Time-bounded authentication [5]	Critical
Credential Validity	10-15 minutes	Ephemeral access	Critical
In-Memory Usage	Variable	Memory-only storage	Critical
Audit Recording	Continuous	Complete traceability [6]	Critical
Post-Job Revocation	Immediate	Automatic cleanup	High

5. Implementation Considerations

Implementing zero-touch secrets management demands a delicate coordination of various technical layers, wherein each layer has its own issues, which should be overcome in accordance with well-considered architectural choices and the choice of technologies. The identity management base will generally make use of workload identity standard protocols to provide interpreting capability across various infrastructure platforms. Standardized identity adoption has critical benefits in heterogeneous clouds where build pipelines can cross providers and on-premise infrastructure. Such protocols permit the federation of trusts where identity tokens issued by one authority can be verified and accepted by a wide range of resource systems without having to install distinct authentication for different target platforms. The problem of protocol choice is between the possibility of adopting protocols used widely and matching them to ecosystems, and the necessity of security properties and extensibility. Organizations need to consider the mapping of identity standards to their particular infrastructure architecture, taking into account the format of tokens, claim structure, signature, and revocation.

The security and performance of the encryption system should be balanced with the encryption algorithms being chosen with good security, and avoiding the imposition of costly computation overheads on the time-limited building phases. The cryptography operations necessary to support identity-based encryption, as well as credential protection, should run in

milliseconds so as not to slow down the pipeline, but should use algorithms strong enough to withstand the modern methods of attack. Cryptographic algorithm space includes various methods of attaining confidentiality, integrity, and authenticity, and contemporary practice is based on decades of theory and practice. Cryptographic protocols are the foundation of secure communication systems, which allow entities to build trust, share keys, authenticate identities, and safeguard data in transit and at rest. The development of cryptographic algorithms has been fueled by the ever-developing relationship between the growing computational abilities and the new threat models, and the algorithmic design and security parameters require constant improvements. Modern cryptographic implementation has markets to deal with more advanced attack techniques, at the same time being computationally efficient to manage resource-bounded and latency-constrained applications [7]. Cryptographic primitives should be chosen with respect to not only raw computational performance, but also properties (for long-term security planning) like key management complexity, forward secrecy, and quantum resistance. Symmetric encryption algorithms are usually much faster than asymmetric ones, and are therefore useful to afford bulk credential encryption, whereas asymmetric cryptography is vital to initial trust establishment, and in key exchange without any pre-shared secrets.

The packaging with existing pipeline orchestration tools will demand modification of the authentication flow in order to transparently inject identity tokens into build environments without interfering with the established developer workflows or necessitating significant changes to be made to existing pipeline settings. The integration layer should provide a transparent interface that can be used by pipeline tools with only minor adjustments to job definitions, preferably in standardized environment variables or through standard API endpoints that substitute the old credential access patterns. These integration challenges are compounded by the fact that the industry provides various pipeline orchestration platforms with different execution models, different credential injection mechanisms, and different extensibility models, all of which have different execution models. The organizations need to come up with the adapter layers that are capable of converting the generic identity and credential issuance protocols into platform-specific implementations that are conscious of the distinct nature of each orchestration tool.

Policy engines should be able to define complex rules of access to repository hierarchies, deployment environments, and security classifications, and be maintainable by operations teams who might not be highly knowledgeable in formal policy languages. The policy definition model should be both expressive and accessible enough that it allows the security teams to express complex authorization logic without incurring maintenance costs in the form of policy drift or misconfiguration. Attribute-based access control has become an influential framework for authorization in complex distributed systems, transcending the restrictiveness of conventional role-based ones, through a paradigm shift as it allows access decisions to be made on fine-grained levels, depending on arbitrary attributes of subjects, objects, actions, and environmental conditions. This method offers the flexibility to undertake an advanced security policy that will be dynamic to change situations and organizational demands [8]. The policy framework should be able to dynamically evaluate the contextual characteristics and authorize in a manner that not only reflects the initial role allocation, but also factors in run-time information like the time of day, physical location, resource status, and security posture of the requesting workload.

Security Scenario Analysis

Scenario 1: Compromised Build Runner

Context: An attacker gains control of a CI/CD runner through a supply chain attack or infrastructure vulnerability, potentially during or after a legitimate build execution.

Traditional Approach Outcome:

- Long-lived credentials stored in runner environment or mounted secrets remain accessible
- Attacker can extract credentials and use them from any location until next rotation cycle
- Credentials may provide access to multiple deployment environments

- Detection relies on anomalous usage patterns, which may take days to identify
- Remediation requires emergency rotation across all potentially exposed systems

Zero-Touch Approach Outcome:
- Short-lived credentials bound to specific workload identity expire within minutes of build completion
- Extracted credentials include temporal and contextual claims that fail validation outside build window
- Policy engine rejects authentication attempts from unexpected network locations or execution contexts
- Identity token signatures prevent credential reuse from attacker infrastructure
- Audit logs immediately correlate unauthorized access attempts to specific build identity
- Blast radius limited to resources explicitly authorized for compromised build's scope

Mitigation Delta: Zero-touch approach reduces exploitable window from days/weeks to minutes, eliminates credential reuse outside legitimate execution context, and provides immediate forensic trail linking malicious activity to specific compromised build.

Scenario 2: Leaked Network Traffic
Context: An attacker intercepts network communications between build infrastructure and secret management systems through network position exploitation or TLS compromise.

Traditional Approach Outcome:
- Captured credentials remain valid for entire rotation period
- Attacker can replay authentication from any network location
- No binding between credential and legitimate requesting workload
- Detection requires identifying unusual access patterns across potentially large time windows
- Compromised TLS keys enable retroactive decryption of archived traffic

Zero-Touch Approach Outcome:
- Identity tokens include cryptographic proof-of-possession requirements preventing simple replay
- Temporal claims in tokens cause validation failures for delayed replay attempts
- Workload identity binding ensures stolen tokens fail authentication when presented from different infrastructure contexts
- Even with TLS compromise, attacker cannot generate valid identity assertions without access to identity provider's signing keys
- Policy evaluations incorporate network context, detecting access from unexpected source addresses
- Forward secrecy in token issuance prevents retroactive compromise even if long-term keys are exposed

Mitigation Delta: Zero-touch approach transforms network interception from high-value attack to low-utility reconnaissance, as captured credentials contain contextual bindings that fail validation in attacker-controlled replay scenarios.

Scenario 3: Malicious Insider with Log Access
Context: An insider with legitimate access to build logs and monitoring systems attempts to extract credentials for unauthorized resource access or data exfiltration.

Traditional Approach Outcome:
- Credentials visible in logs if accidentally printed by build scripts
- Vault tokens or API keys captured in debug output remain functional

- Insider can use harvested credentials from their own workstation or compromised accounts
- Attribution is difficult as insider has legitimate log access
- Detection relies on identifying unusual resource access patterns rather than credential misuse

Zero-Touch Approach Outcome:
- Identity tokens logged during troubleshooting bind to specific build execution context
- Tokens include claims restricting valid calling identity to original build runner infrastructure
- Policy engine validates execution environment attributes that insider cannot replicate
- Attempted reuse from insider's workstation fails workload identity verification
- Audit trail captures both the legitimate build that created token and unauthorized reuse attempt
- Fine-grained claims in tokens enable policy enforcement that detects context mismatch
- Even with token extraction, insider cannot forge necessary execution environment proofs

Mitigation Delta: Zero-touch approach changes insider threat model from "credential theft enables persistent access" to "log access provides limited reconnaissance value," as extracted tokens contain non-replayable contextual proofs and fail authentication outside legitimate execution environments.

Quantifiable Operational Benefits

Organizations implementing zero-touch secrets management typically observe:
- **Mean Time to Rotate (MTTR)**: Reduction from 4-6 weeks (coordinated manual rotation) to milliseconds (automatic per-build issuance)
- **Credential Exposure Window**: Reduction from 30-90 days (typical rotation periods) to 5-15 minutes (build execution duration)
- **Security Incident Investigation Time**: 60-70% reduction through unified audit trails with contextual attribution
- **Secrets-Related Production Incidents**: 80% reduction in outages caused by expired or incorrectly rotated credentials
- **Developer Onboarding Time**: 40% reduction in time required to configure secure pipeline access for new services

These improvements compound over time as the system accumulates policy refinements and integrates deeper contextual signals, continuously narrowing authorization decisions and reducing false positive authentication attempts.

Table 3: Technical Complexity and Performance Requirements Across Implementation Layers [7, 8]

Technical Layer	Key Challenge	Technology Selection Criteria	Performance Target	Integration Complexity
Identity Management	Protocol standardization	Interoperability, security properties	<10 seconds	High
Cryptographic Systems	Security-performance balance [7]	Algorithm strength, computational overhead	<5 milliseconds	Medium
Pipeline Integration	Workflow adaptation	Transparency, minimal configuration changes	Seamless	Very High

Policy Engine	Expressiveness vs maintainability [8]	Rule complexity, operator comprehension	Real-time evaluation	High
Federation Layer	Cross-platform trust	Token validation, authority recognition	<15 seconds	Very High
Adapter Development	Platform diversity	Orchestration tool compatibility	Transparent	High

6. Security and Operational Benefits

This architecture essentially minimizes the attack surface because it removes persistence credentials in the build infrastructure. Those attackers who access a build environment do not get any reusable secrets to exfiltrate, but instead have access to expired or soon-to-expire credentials with strictly scoped permissions. The limited validity window restricts the usefulness of any invalidated credential to minutes and not months, which significantly constrains the time constraint that any adversary that has succeeded in the intrusion of pipeline infrastructure has to operate within.

Extensive audit trails can be used to do effective forensic analysis to determine exactly which credentials were granted to which jobs and how they were applied. This visibility contributes to security investigation and compliance needs, where the access patterns are immutable and can endure a regulatory audit. The cryptography principles that are the basis of zero-touch secrets management are based on well-chosen algorithms that balance security and performance needs. Contemporary cryptographic systems should not only offer confidentiality by providing encryption, integrity by means of message authentication codes, or authenticity using digital signatures, but also be resistant against classical as well as new quantum computing threats. Cryptographic algorithms have been developed through an ongoing developmental process based on the evolving threat level and technological capabilities, and each generation of cryptographic primitives has improved based on the lessons learned through cryptanalysis of the previous schemes. Cryptography has a history of development in the form of an arms race between cryptographers who create secure systems and those who attempt to compromise those systems, leading to new designs of algorithms and methods of implementation and security proofs. Modern cryptographic studies address the issues of quantum computing that will likely bring most popular cryptosystems that use public-key cryptography systems to their knees and require the development and standardization of quantum computer-resistant post-quantum cryptographic algorithms [9]. Companies that have adopted ephemeral credential systems should take into account the entire life cycle of cryptographic operations, including key generation and secure random number generation to algorithm selection in both symmetric and asymmetric operations, and performance specifications must be achieved without jeopardizing security properties against those of advanced adversaries.

The benefits of simplified credential rotation to organizations include simplified rotation since all credentials have an expiration period by default, it is automatic, and requires no coordinated updates to configuration files or vault entries. Strict isolation has the advantage of multi-tenant environments, where the isolation of credentials stops cross-contamination between pipelines of various teams. Access control in distributed systems is complicated, and thus authorization models are required that are able to express subtle policies and are readable to the security operations teams. Attribute-based access control gives a more general basis on which to encode authorization logic, which can go beyond basic role assignments, and allows policies sensitive to contextual information like user attributes, resource properties, environmental constraints, and action parameters. To apply attribute-based access control in practical settings in the enterprises, a trade-off between the hypothetical expressiveness of the model and practical realities, such as the requirement of the policies being understandable to the business stakeholders, administration interfaces that discourage misconfiguration, and integration patterns that integrate well with the current identity and access management infrastructure, must be made. Contemporary enterprises find it very complicated to convert

the multifaceted business needs into technically accurate access control policies, which implement the frameworks, simplifying the policy-writing process and preserving the accuracy in the security-sensitive decision-making process, a necessity [10]. This is because attribute-based policies need to be carefully designed to have policy decision points that can effectively analyze complex expressions of attributes of many attributes, policy information points that can effectively retrieve current attribute values in distributed sources, and policy enforcement points that can effectively apply authorization decisions across non-homogeneous infrastructure elements.

Zero-touch is consistent with zero-trust security principles, in which all access requests are authenticated and authorized without considering network location or history of previous access. Standing privileges can be eliminated, and just-in-time access can be implemented, which significantly minimizes the blast radius of a potential security incident and limits the damage that could be caused by compromised credentials or insider attacks.

Table 4: Comparative Security Posture: Traditional vs Zero-Touch Secrets Management [9, 10]

Security Metric	Traditional Approach	Zero-Touch Approach	Improvement Factor	Benefit Category
Credential Validity Window	Months	Minutes	99.9% reduction	Temporal security
Reusable Secrets in the Environment	Persistent	None	Complete elimination	Attack surface
Cross-Team Credential Sharing	Common	Impossible	Full isolation	Multi-tenancy
Credential Rotation Effort	Manual, coordinated	Automatic	Zero operational burden	Operations
Audit Trail Completeness	Partial	Comprehensive	100% visibility	Compliance
Blast Radius of Compromise	Infrastructure-wide	Single job scope	Minimal impact	Incident containment
Standing Privileges	Permanent	None	Just-in-time only	Zero-trust alignment
Forensic Analysis Capability	Limited	Precise	Full traceability	Investigation

7. Conclusion

Zero-touch secrets management is a form of radical change in the process of securing automated development pipelines by swapping durable, static credentials with transient identity-linked secrets, which only last as long as the time they are intended to work. This method of architecture removes the sustained attack area formed by the conventional credential management habits, limiting the possibility of compromise to minutes instead of months in time onslaught and automated revocation. The system allows cryptographic attestation of pipeline identity and binding credentials to a particular execution environment by identity-based encryption to be used to ensure that credential reuse and lateral flow are prevented even in partially compromised build infrastructure. By combining just-in-time issuing of credentials with attribute-based access control, controlled authorization based on full execution context is possible, and detailed audit trails can be used as a source of undeniable evidence when performing forensic analysis and regulatory oversight. Companies using this model benefit greatly on operational fronts by having automated rotation of credentials, easier multi-tenant isolation, plus a smaller blast radius of security events without compromising the smooth developer experience. Zero-touch paradigm, the concept of zero-touch security is directly compatible with the concept of zero-trust security, where all credential requests must be met with fresh authentication and authorization, no matter how many attempts to access the system have already occurred. The continued evolving nature of

software development to more distributed, automated, and cloud-native architectures will require ephemeral credential management to become critical infrastructure to organizations wishing to ensure their security posture without undermining the speed of development. The article has shown that removal of static secrets in CI/CD pipelines is not merely technically possible, but also operationally beneficial, and allows autonomous, self-secured DevOps environments, which defend sensitive credentials by design instead of by control mechanisms.

References

[1] Setu Kumar Basak et al., "A Comparative Study of Software Secrets Reporting by Secret Detection Tools," ResearchGate, July 2023. [Online]. Available: https://www.researchgate.net/publication/371989968_A_Comparative_Study_of_Software_Secrets_Reporting_by_Secret_Detection_Tools
[2] Badis Hammi et al., "Software supply chain security: A systematic literature review," [Online]. Available: https://ieeexplore.ieee.org/document/10154229
[3] Hicham Reguieg, "Using MapReduce to scale event correlation discovery for business process mining," ResearchGate, 2014. [Online]. Available: https://www.researchgate.net/publication/266057379_Using_Mapreduce_to_scale_events_correlation_discovery_for_business_processes_mining
[4] Shams Zawoad, "Cloud Forensics: A Meta-Study of Challenges, Approaches, and Open Problems," ResearchGate, February 2013. [Online]. Available: https://www.researchgate.net/publication/235712413_Cloud_Forensics_A_Meta-Study_of_Challenges_Approaches_and_OpenProblems
[5] Lewis Golightly et al., "Securing distributed systems: A survey on access control techniques for cloud, blockchain, IoT and SDN, December 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2772918423000036
[6] Jaibir Singh et al., "Role-Based Access Control (RBAC) Enabled Secure and Efficient Data Processing Framework for IoT Networks," ResearchGate, August 2024. [Online]. Available: https://www.researchgate.net/publication/383295659_Role-Based_Access_Control_RBAC_Enabled_Secure_and_Efficient_Data_Processing_Framework_for_IoT_Networks
[7] Donagani Ramakrishna et al., "A Comprehensive Analysis of Cryptographic Algorithms: Evaluating Security, Efficiency, and Future Challenges," IEEE Access, Dec. 2024. doi: 10.1109/ACCESS.2024.3509149. [Online]. Available: https://ieeexplore.ieee.org/document/10804125
[8] Vincent Hu et al., "Attribute-Based Access Control," ResearchGate, February 2015. [Online]. Available: https://www.researchgate.net/publication/273393378_Attribute-Based_Access_Control
[9] B. Preneel, "A survey of recent developments in cryptographic algorithms for smart cards," 20 June 2007., [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1389128607000217
[10] Dinesh Rajasekharan, "Simplifying Attribute-Based Access Control (ABAC) for Modern Enterprises," ResearchGate, February 2025. [Online]. Available: https://www.researchgate.net/publication/389349488_Simplifying_Attribute-Based_Access_Control_ABAC_for_Modern_Enterprises