White Paper

How to Choose a Secrets Management Solution

cpl.thalesgroup.com

THALES
Building a future we can all trust

Table of Contents

Executive Summary

Modern IT environments are powered by an exponentially growing number of secrets. As organizations adopt cloud computing, containers, microservices and DevOps practices, the number of secrets required for authentication and encryption proliferates rapidly.

Secrets like passwords, API keys, SSH keys and encryption certificates enable services, applications, and systems to communicate securely. However, uncontrolled sprawl also introduces security risks, with many high-profile breaches tied to compromised secrets.

This white paper examines the urgent need for robust secrets management solutions. It provides a detailed overview of the secrets management challenge, from risks to requirements. Guidance is provided on planning, evaluating, and selecting a secrets management system aligned to organization needs.

Why is secrets management needed?

Secrets are credentials, certificates and keys used to authenticate between different machines or from human to machine interaction. Automated processes, virtual machines and any process that runs a service needs to have access to that resource for authentication using credentials, certificates, and keys.

Several trends in the last few years such as containerization, cloud transformation, DevOps, and automation contributed to a massive increase in the number of secrets that are being used everywhere, in every environment including hybrid cloud, on-prem, and multi-cloud environments (Figure 1).

With this proliferation, stronger management is needed, otherwise, it can cause a lot of breaches such as Uber, Scotiabank, and Nvidia. With all kinds of examples in the last few years, attackers gain access to environments using stolen secrets. Your secrets are in danger of compromising the entire network by having them within the attack vector and elevating the privileges during that kind of attack resulting in a data breach.

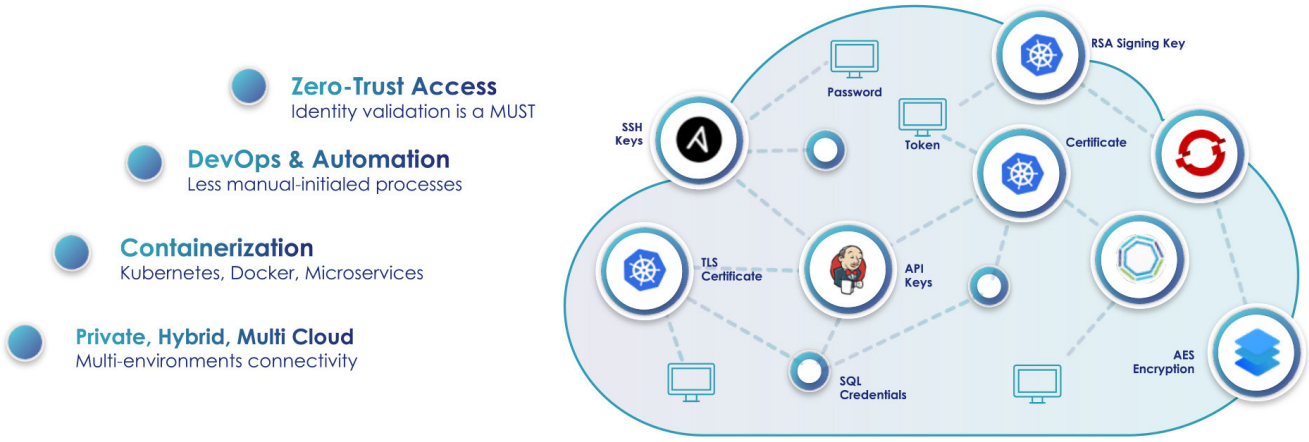

Figure 1: Massive increase in use of secrets

Challenges with secrets management

Sprawled everywhere

Secrets are scattered throughout IT environments, not just in source code but also in configuration files, automation scripts, tools like Ansible/Chef/Puppet, and more. This makes secrets difficult to track and secure.

Needed in ANY environment

Secrets need to be available across multiple environments like development, test, staging, and production. They also need to span on-prem and multi-cloud infrastructures across regions. Replicating and managing secrets is complex.

Mission critical

Applications require secrets to operate and interconnect with other applications. You need highly available systems to operate to have your applications running.

Containers can spin up a high amount of traffic. Within peak time as an example, you may have 3,000 of those containers running in parallel. All of them need to have the secret to authenticate and run.

Siloed solutions

Many teams have also implemented siloed secrets management solutions like cloud provider tools or open source vaults. This leads to fragmentation and inconsistencies, with poor organizational visibility. Migrating and unifying secrets management is difficult. Secrets management is much more than just simply vaulting your secrets.

You may find yourself in a situation when you have siloed solutions in which different secrets and different teams are leveraging diverse types of solutions. You might have within your organization some existing solutions with secrets management at different maturity levels. Part of your team may not even use any type of secrets manager while others might leverage the cloud service provider's secrets manager. Or they may be leveraging some kind of open source project. There is no one unified standard resulting in an organizational challenge that you need to tackle.

The purpose of secrets management

The purpose of secrets management is all about controlling the sprawl and reducing the risk of secrets exposure, while keeping your business running. It is important to control, govern, and reduce the risk of exposure to secrets. Imagine you can eliminate those secrets within the configuration file, source code, or infrastructure that runs the application which provides them in the runtime for that service. Instead, the secrets management service will provide centralized governance, Just-in-Time access, role-based access management and more.

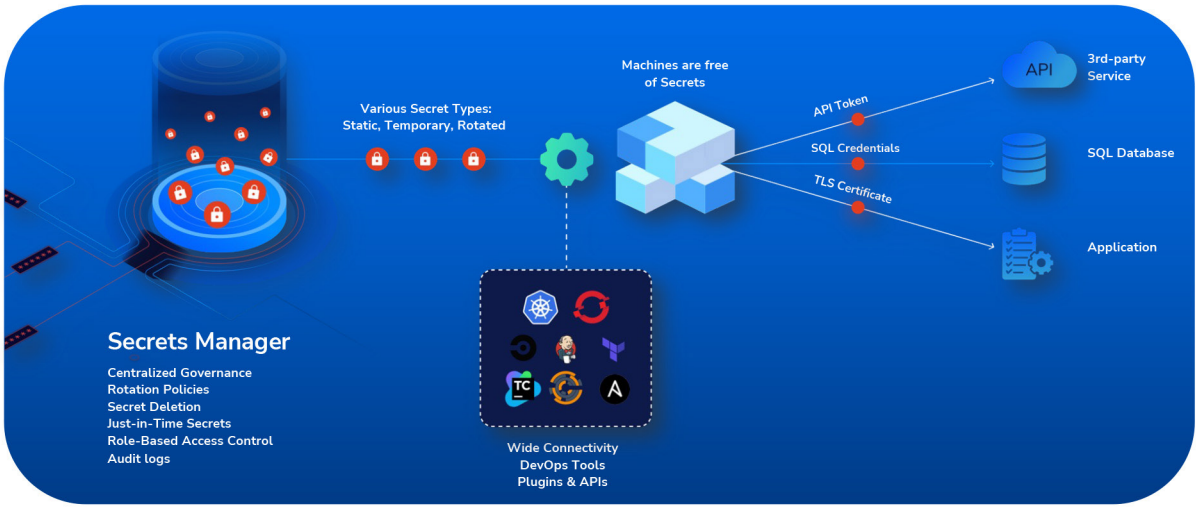

Figure 2: A purposeful secrets management design

To do this, the application would need to leverage a connectivity layer. That could be plugins, APIs, Command Line Interface, or SDK for the application to connect to the secrets management system. It fetches the secrets to use them whenever the application needs to authenticate to a third-party service, database, or other application. And eventually in memory, those secrets will be destroyed and will be eliminated. Whenever the secret is not being used, it will not be stored within the application itself (Figure 2).

"By 2025, 30% of enterprises will have adopted broad-spectrum Data Security Platforms, up from less than 10% in 2021, due to the pent up demand for higher levels of data security and the rapid increase in product capabilities."

– Gartner, 2023 Strategic Roadmap for Data Security Platform Adoption, 22 September 2022, Joerg Fritsch, Brian Lowans

Eight key requirements of a secrets management system

1. Support different types of secrets

A secrets management system should support many diverse types of secrets. Some requirements include:

- A workload requires access to a database, it will need database credentials.
- An automated process will need access to a Linux server, it will require an SSH key in order to authenticate to that server.
- An application requires access to a certain SaaS environment or whatever is provided and it will need an API key.

There are a lot of examples of SSH keys, API keys, certificates for SSH and PKI encryption keys, including AES and RSA needed to either sign certificates or to encrypt certain data. All of these diverse types of secrets need to be supported by the secrets management system.

2. Multi-platform rotation capabilities

Having a rotation capability requirement allows you to reduce the risk of someone compromising that credential. If they obtain that credential, then by automatically and periodically rotating that password, the compromised password would be completely meaningless. The password itself can be rotated by any automated process by your secrets management system. Now understanding the rotation mechanism, it is particularly important for a secrets management system to be able to rotate different types of passwords on diverse types of platforms. There are a lot of technologies out there that need wide support. For example, diverse types of databases need the ability to rotate SSH key, API key or rotate anything that you have within your environment using a custom rotator.

Figure 3: Reduce risk with rotation capabilities

However, supporting complex policies is also required. For example, you may need a rotation to happen at 2 am, once every 30 days or every hour. Policies are required for a secrets management solution to provide and enable setting the time and the frequency for that rotation.

3. Temporary Just-in-Time secrets

Support of static secrets and the ability to rotate them are important, but those kinds of credentials are just there, static and waiting to be compromised. This is the classic or traditional way of creating identities. However, there is a mechanism that helps to elevate to a more advanced security practice when adopting secrets management. Temporary secrets, also known as Just-in-Time access, should be considered as a requirement.

What do temporary secrets look like? It simply means that whenever an identity is being created, then it also must be deleted. It is being destroyed once it is used. Instead of having standing privilege, a workload requiring access to a database would ask the secrets management system to bring the temporary identity that can leverage the secrets management system for that database. It would create the temporary credentials and provide them to the application itself for later use. After it is used and the container spins off or after the application is done with the tasks that it requires, then that credential is deleted. This is what the secrets management manages.

Figure 4: Just-in-Time access

Zero Trust is also a part of Just-in-Time secrets. Consider a compliance and audit process running within an environment. If temporary secrets are leveraged, you will be able to go through the audit process, show the auditors your database as an example that there are no privileges there at all. There are no identities beyond the default identities because the identities that were leveraged were created and deleted, according to the actual usage. In terms of identity and access management, this is also referred to as zero standing privileges.

4. Secure access

Considering who can access your secrets and what they can do once inside your secrets management system is another requirement. This can have several layers.

Authentication

The first layer is who should be able to authenticate your secrets management system. There are many identities such as micro functions, team members, workloads. No matter the type of identity, you need to set the identity to be able to authenticate.

Role-based access control (RBAC)

You need to set the exact identity permissions allowing that identity permission to edit or create a certain secret. Or another identity could have the permission to delete a particular secret. This flexibility is required.

Business unit segregation

With multiple teams, each one of those teams within the organization will need to have their own independence to control their own secrets. Segregation of duties allows the separation to be either logical or physical for certain types of secret management systems.

Zero knowledge

When secrets are managed, you need assurance that no one in the chain of use – other than the endpoint needing the secret – has access to the secrets. This requires a secure mechanism of storing and retrieving secrets.

Government access

You must also consider whether you are concerned about government access. If so, you must ensure that the secrets management solution you choose prevents the government from accessing your secrets.

5. Connectivity

Another fundamental requirement is connectivity. Some repositories may be secured with several types of secrets. How would a certain workload be able to interconnect with the secrets management system? There are several methods for this such as a SDK if it is a source code that requires a particular secret, command line interface if it needs to be injected within scripts, or an API to be etched.

Source Code, Scripts, CI/CD, DevOps, Production

Figure 5: Many required plugins

Even more importantly is connectivity via plugins if leveraging an automation platform like Jenkins, CircleCI, Ansible, Chef, Puppet, or others. They all require a plugin to fetch the secrets when used. It is important that these tools be able to call for secrets from a secrets management system (Figure 5).

Another type of connectivity consideration is support for a rich user interface. A rich UI helps to understand what is happening and provides a great user experience as well.

6. Multi-environment

Where do you need your secrets to be available? Usually, the response is everywhere. Another requirement for a secrets management system is that it must support different regions and different environments – on prem, hybrid or multi-cloud. The system must replicate and sync secrets globally across all environments for development, test, staging, production in multiple regions and clouds. One hurdle is that organizations are dynamic. A secrets management solution needs to seamlessly support the dynamic nature of environments that are spinning up or down.

Figure 6: Supports on prem, hybrid or multi-cloud

7. Mission critical support

The mission criticality requirement of a secrets management system is making sure that all of the secrets that workloads require are available when needed. Otherwise, those workloads cannot operate. Production can be down if the secrets are not available. High availability, redundancy and scale are required.

High availability

The system must be available for all those transactions and fetching requests from the workloads. Highly available 24 by 7 every time the application requires it.

Redundancy

The second consideration is redundancy. What happens when it is not available? For example, you can leverage caching mechanisms. There are many solutions where a secrets management solution needs to make sure that you have the full redundancy whenever needed.

Scale

A third consideration is scale. When all those workloads work together and operate simultaneously, they are requesting for those secrets altogether. The operation of the secrets management system needs to handle all those requests concurrently.

8. Governance

The last fundamental requirement is around governance.

Tracking

When a workload is asking for a secret, it needs to capture comprehensive audit trails and logs of all secret access and administrative actions.

Visibility

Next is visibility. Once all the secrets actions are tracked, those actions must be presented on a reasonable dashboard allowing visibility of what has happened with what kind of secret and who has accessed them as well as isolate any identity that affects those secrets.

Event log and forwarding

In many cases, organizations require the logs to be gathered and forwarded to their internal log management systems, SIEM systems or to the internal security systems for later analysis.

Window into siloed vaults

Siloed solutions for secrets management deployed within the organization, whether from a cloud service provider or open source solution, needs to have the secrets management system coexist with existing solutions allowing full governance and understanding of where secrets are throughout the organization.

> Within a strong and true secrets management solution, you'll have a solution for easing the migration of siloed processes.

A simplified process of migration for a new secrets management enterprise-wide solution is necessary. For those siloed solutions that will not be migrated, visibility and an understanding of where those secrets are is crucial.

Key planning considerations

With thorough planning, organizations can implement modern secrets management to reduce risks, improve operations, simplify management, and accelerate innovation. Key planning considerations include:

- Identifying all secrets requirements across existing infrastructure and applications
- Assessing gaps in current secrets management and risks
- Determining availability and scalability needs
- Evaluating options for consolidation of siloed management solutions
- Developing access control policies and RBAC architecture
- Building a migration plan for progressively transitioning secrets management
- Integrating with existing identity management systems
- Enforcing controls demanded by regulatory compliance

Summary

Today's cloud-native, DevOps environments demand solutions to control secrets sprawl, enforce least privilege access, maintain availability, and reduce attack surfaces. Unmanaged secrets introduce unacceptable cyber risks and hinder operations.

This white paper provided a comprehensive overview of the secrets management challenge, from risks to requirements. Organizations must align their strategies to these realities, evaluating their needs against these criteria when selecting a secrets management platform.

With robust and proactive secrets management, enterprises can securely enable cloud transformation, automation, and innovation through centralized secrets orchestration. Taming secrets sprawl provides multifaceted benefits from reduced breaches to simplified IT architecture.

Resources

At Thales, we provide a unified data security platform that simplifies the management of access, data discovery, data protection, and control of your most critical data and critical information. See how Thales CipherTrust Secrets Management is a state-of-the-art secrets management solution powered by Akeyless. It meets these requirements and overcomes the challenges by protecting and automating access to secrets across DevOps tools and cloud workloads including secrets, credentials, certificates, API keys, and tokens.

For more information:

- **Go to the Thales Partner Solutions Series: CipherTrust Secrets Management for DevSecOps webinar by Oded Hareven, CEO & Co-Founder (Akeyless) and John Wohlfarth, Director, Business Development (Thales)**
- **View demos**
- **Schedule a free demo/POC**
- **Start your 90-day free trial**
- **Go to Thales CipherTrust Secrets Management webpage**

- **Manage your secrets and data in one platform**
- **Quick to deploy and enterprise-ready**
- **Lower cost of ownership**

About Akeyless

Akeyless' unique combination of innovative technology and cloud-native architecture enables enterprises to quickly secure DevOps, cloud workloads, and legacy environments, while meeting compliance and regulations.

About Thales

Today's businesses and governments depend on the cloud, data and software to deliver trusted digital services. That is why the most recognized brands and organizations around the world, rely on Thales to help them protect sensitive information and software wherever it is created, stored or accessed – from the cloud and data centers to devices and across networks. As the global leader in data security, identity & access management, and software licensing, our solutions enable organizations to move to the cloud securely, achieve compliance with confidence, create more value from their software and deliver seamless digital experiences for millions of consumers every day.

THALES

Building a future we can all trust

Contact us

For all office locations and contact information,
please visit cpl.thalesgroup.com/contact-us

cpl.thalesgroup.com