

Prepared for



Secrets Management and DevOps – A Risked-Based Approach to Eliminating Hard-Coded Secrets

January 2022 EMA White Paper

By Chris Steffen, CISSP, CISA

Research Director, Information Security, Risk and Compliance Management

Hardly a month goes by without a news cycle of the latest cyber-attacks: from a software supply chain attack, to ransomware, to some kind of personal data exposure. They are so numerous that most people only pay attention when it is a “major” breach or happens to a company they have a relationship with. But more often than not, people are never told what really happened for a number of reasons: the companies involved do not want the negative publicity (they want the whole thing to “just go away”), there is an active criminal investigation, or the problem wasn’t really resolved completely. It is the last reason that may be the biggest source of concern: why would you not try to eliminate a known—and previously exploited—vulnerability from your environment?

In some of the more recent high-profile attacks, the attackers were very methodically looking for specific vulnerabilities: a kind found hidden deep within code, otherwise known as hard-coded secrets. In the software development space, hard-coded secrets involve the practice of including or embedding information used for authentication (passwords, tokens, keys, etc.) to connect to services or systems directly into applications. This practice is used by many developers for the ease of authentication, instead of retrieving credentials and keys from an external source or at application’s runtime.

Why Hard-Coded Secrets are so Concerning

Most security professionals immediately recognize the concerns that hard-coded secrets create, but there are four concerns that should be mentioned specifically.

Hard-coded secrets allow for lateral movement. Consider a plausible breach scenario: a company’s code repository is compromised and contained within the code are the hard-coded secrets that provide the attacker the keys to the kingdom—literally all of the information that the hacker needs to traverse the company’s systems and laterally move within the enterprise. The hacker uses the secrets to easily penetrate the shared systems and methodically steals the company’s information. Also, the hard-coded secrets can be found at any point during the software development lifecycle (SDLC). Because the hard-coded secrets facilitate lateral movement, and they can be found in containers, build logs, and K8s, it means that a compromise at any phase of the SDLC potentially grants access to every phase.

The number of hard-coded secret incidents continues to rise. The news is full of accounts of attacks involving the loss of public-facing account passwords and/or credentials, but how the password may have been lost is often not released. From GitHub scans to codebase attacks, attackers know that the information they are looking for is in the millions of lines of unsecured, unencrypted code that is usually far easier to obtain than breaking through the multi-layered firewalls of an enterprise's network edge. For example, the SolarWinds software supply chain attack in 2021 was discovered to be a search by the attackers to find secrets.¹ Those hard-coded secrets are found nearly everywhere: again, through logs, documentation, unencrypted network connections and setup scripts, to name a few obvious locations.

Developers are the new target. Developers are often considered some of the most difficult and demanding technology resources to corral because they have little desire to conform or comply with security protocols and controls. Instead, they are concentrated on delivering applications through a rapid development pace. The attackers know all about the processes and procedures that developers are likely (or NOT likely) to follow and exploit these weaknesses to achieve their ends. They also know that developers tend to be over-credentialed and generally have access to more valuable systems and deeper levels of authorization than a typical end-user, which makes them a high-value target.

Code (and the secrets contained within the code) continues to leak. There are two methods of leaking data: externally and internally. Security teams spend lots of time trying to protect their intellectual property and critical data from external leaks but the internal leak/attacker can often be just as (if not even more) dangerous than the external one, with malicious insiders making up a large percentage of data breaches. Many enterprises use services like GitHub to upload and share their code because it can improve collaboration and save companies from technology disasters. A recent study found hard-coded secrets in over 100,000 GitHub repositories.² Incidents involving the leaking of GitHub repositories seemingly happen every day. Even if security professionals were able to prevent public repository leaks, the root of the problem—that the code still contains the hard-coded secret—still exists.

¹ <https://msrc-blog.microsoft.com/2021/02/18/microsoft-internal-solorigate-investigation-final-update/>

² https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_04B-3_Meli_paper.pdf

Tackling Hard-Coded Secrets Head-On

Dealing with hard-coded secrets is a challenge that security professionals have faced for years, and if the problems outlined prior are any indication, with very little success of remediation. Still, there are steps that organizations can take to mitigate hard-coded secrets from being embedded in their code.

Train Developers

The first approach (and the one that has been ineffective so far) is education. Security teams need to educate developers and company leaders about the dangers of hard-coded secrets. This can be part of a targeted campaign for development teams, but should include a continuing security awareness effort for company leaders. The leaders of the enterprise set the tone of the organization from the top, and many developers have said that the push from business leaders to get applications and new versions out the door is part of the reason they use “shortcuts” in their programming methods. Those shortcut processes nearly always involve less regression and security testing, but also include connectivity shortcuts, such as hard-coded secrets. Providing a clear and concise message from business leaders that these kinds of processes and procedures are not acceptable should be the starting point for every security organization.

After laying the groundwork for an attitudinal process for hard-coded secret mitigation, security teams need to look for technological solutions for remediation, focusing on secrets scanning and secrets management.

Scan for Secrets Within Developer Workflow Integration

Comprehensive secrets scanning is needed in every environment. While EMA research has concentrated primarily on the developers, scanning for secrets in systems, network appliances, and IoT/OT devices is also extremely important and should be part of any secrets-scanning program. On the developer side, it is imperative to understand where the various code for the enterprise is being stored. The security team must know all the various code repositories used by the dev team to have a complete scan. However, scanning code repositories alone is insufficient. Attackers are looking for secrets in any and all phases of the SDLC they breach. Therefore secrets scanning must also include dev tools, build logs, containers, registries, infrastructure as code configurations, and any previous version of source code.

Once all the various repositories have been identified (this alone is a HUGE task for most organizations, since there can be dozens, if not hundreds, of locations where code snippets could potentially reside, and developers are not going to be enthusiastic about the repository discovery process), the security team can work with the dev leaders to establish a process/procedure for scanning for different kinds of secrets.

The security team and development leader must work together to develop a process that includes hard-coded secrets scanning as part of the normal development workflow. By integrating the process as part of normal DevOps, the enterprise accomplishes four things:

- A streamlined process that doesn't reduce developer productivity with additional or manual code reviews
- Buy-in from leaders (and the dev team, if possible) on a smoother process for scanning code for security considerations with agreed upon behavioral expectations and SLAs
- Eliminating hard-coded secrets before they reach the main branch, which helps keep the main branch ready to be built and deployed at all times
- Eliminates the insertion of hard-coded secrets from the very beginning of the SDLC

Developers are not deliberately trying to develop insecure code, but they are often at odds with security leaders about the intrusive nature of security requirements because it slows production and release time. Developing a standard and process that addresses these concerns from the very beginning, then scanning within developer workflows to reinforce the process, potentially solves this problem while creating a dialogue bridge between these silos.

Provide an Alternative to Hard-Coding Secrets

Establishing a communications bridge between security and dev teams also provides the means to collaborate, including potential solutions to replace hard-coded secrets with a secrets/authentication management solution. Technology has significantly advanced since the first developers inserted hard-coded secrets into their codebase. For example, identity and access management (IAM) solutions are commonplace and integrate with simple calls through a rich and well-documented interface. Nearly every secrets management tool/solution available today has some sort of call or method to retrieve tokens or keys on request while ensuring the integrity of the process remains with regular rotation of tokens, keys and passwords.

Reducing Exposure Risk via Defense in Depth

Even the most comprehensive code-scanning process or procedure cannot eliminate all of the risk associated with security breaches. Breaches will continue to happen from compromised accounts, malicious insiders, or code that is leaked from public code storage. It is critical that the organization's security strategy addresses a "defense in depth" approach to secrets management. While most secrets management tools include a scanning component, they should also use solutions for other avenues of secrets exposure, specifically by addressing code leakage and governance.

Code Leakage

Code leakage will likely always be a concern when evaluating risk, at least to some degree. Programming is often collaborative, and every time code is shared, it increases the risk of potential leakage. Code is also stored to make it easily retrievable by development teams, often in public code stores. For example, one of the most popular code repository tools, GitHub, has a unique identity model that makes inadvertent leakage much easier, mainly because developers typically bring their GitHub account with them to each company they work for, in addition to the corporate repositories they can access. Because of this, it is relatively easy to save proprietary code to the wrong repository and expose proprietary code in a publicly accessible personal account. Even with the most stringent precautions, it is still possible for code to be exposed through storage compromise, thus exposing any shared secrets that may have been included in the code. Ideally, the code would not contain shared secrets, but it is also critical to have tools in place to identify and remediate source code leakages as soon as they occur while monitoring and mitigating leakage sources to prevent future exposure.

Code Governance: Authentication & Privilege Management

A secrets management solution should also consider security governance, wherever secrets exist, which is increasingly in multiple phases of the SDLC. For example, attackers are now targeting developers' accounts. Hence, all DevOps tools and infrastructure that contain secrets (repositories, build systems, registries, etc.) should be protected by requiring multifactor authentication (MFA) to reduce the risk of exposing secrets via a compromised account. The process of least privilege access should be similarly enforced wherever secrets reside to reduce risk from unnecessary exposure to users who didn't require access in the first place. Code scanning rules should only be modified by dev leaders or those outside of the development process, while branch protection and security build rules should also be governed with least privilege policies so that they cannot be turned off or modified for malicious purposes, providing additional security and integrity to the code-scanning procedure.

When combined and deployed everywhere secrets exist in one's SDLC, these best practices contribute to significantly decreasing the risk to the business and improving security in the development process. These steps shift the conversation from eliminating hard-coded secrets from the codebase and larger development pipeline to an overall reduction of exposure and risk to the organization.

Conclusion

Taking a risk-based approach to hard-coded secrets ensures that one prevalent attack vector that often leads to major software supply chain breaches is taken off the table for your organization. A complete solution to hard-coded secrets seeks to find and fix any hard-coded secrets, implement a culture of change around the subject, and lowers the risk of exposure of those secrets, all of which will reduce the likelihood of secrets leading to a breach of your software delivery pipeline being compromised.

As the traditional corporate network bastions continue to evolve to a public cloud, the work from anywhere reality, and infrastructure as code scenarios, potential code exposure is at an all-time high. Contained in that code are the digital keys to the enterprise in the form of hard-coded secrets. Mitigating these exposures and risks with a complete secrets management solution is the best practice for any development team.



About Enterprise Management Associates, Inc.

Founded in 1996, Enterprise Management Associates (EMA) is a leading industry analyst firm that provides deep insight across the full spectrum of IT and data management technologies. EMA analysts leverage a unique combination of practical experience, insight into industry best practices, and in-depth knowledge of current and planned vendor solutions to help EMA's clients achieve their goals. Learn more about EMA research, analysis, and consulting services for enterprise line of business users, IT professionals, and IT vendors at www.enterprisemanagement.com. You can also follow EMA on [Twitter](#) or [LinkedIn](#).

This report, in whole or in part, may not be duplicated, reproduced, stored in a retrieval system or retransmitted without prior written permission of Enterprise Management Associates, Inc. All opinions and estimates herein constitute our judgement as of this date and are subject to change without notice. Product names mentioned herein may be trademarks and/or registered trademarks of their respective companies. "EMA" and "Enterprise Management Associates" are trademarks of Enterprise Management Associates, Inc. in the United States and other countries.

©2021 Enterprise Management Associates, Inc. All Rights Reserved. EMA™, ENTERPRISE MANAGEMENT ASSOCIATES®, and the mobius symbol are registered trademarks or common law trademarks of Enterprise Management Associates, Inc.