



Law Enforcement Achieves High Security & Software Development Speed with JFrog

Law Enforcement has special security requirements, such as internal networks with no internet access, that make it challenging to access open-source packages while ensuring integrity and increasing the efficiency of software development operations.



Successful deployment
in a fully isolated
Air Gap environment



Leveraging of Open
Source without direct
internet access



Automated threat detection
to speed up development
operations

0

Developers with
Direct Internet Access

1,000+

Open Source
Packages / Month

80%

Reduction in Curation
Time per Package



This leading law enforcement agency, leverages open source package curation to deliver quality software and speed up release cycles. Due to strict security constraints, there are no direct Internet connections or access to open-source repositories. After the successful adoption of the JFrog Platform on their internal network, they engaged JFrog Professional Services to help them detect potential vulnerabilities before they enter the network, gain access to multiple curated OSS repositories, and increase development efficiency despite having to implement a solution in a DMZ using Air Gap technology.

Company Overview

Software for law enforcement was valued at [\\$11.9 B last year](#) and is expected to grow by an impressive 12% per year through 2033. To meet this incredible demand for software solutions, law enforcement agencies are looking for ways to speed up development and deliver more software in less time. Security requirements and limited access to open source software (OSS), however, make this a very challenging goal.

Like so many government and military software development operations, this law enforcement agency does not permit direct internet access from their internal network — especially their software development environment. This policy is likely to continue due to the growing number of cyber threats and the introduction of potential vulnerabilities into the software supply chain.

With hundreds of developers turning out hundreds of releases per month, and an ever-growing list of requests for additional applications, the law enforcement agency's DevOps team was looking for a way to better leverage the benefits of OSS without compromising their high-security architecture.

Challenges

High-security internet access environments often employ stringent measures to ensure the integrity and security of their systems. Overcoming architectural challenges and improving operational efficiencies were key to the success of this project.

It is important to emphasize that prior to implementation of the current solution, it took a developer an average of 30 min to select, request and receive an open source package which required custom programming and human intervention to support each repository and did not include scanning for vulnerabilities prior to uploading to the internal network.

Likewise, achieving improved overall control, enhanced security and better compliance were also major challenges that had to be met. Besides meeting these objectives, the solution also had to result in faster release cycles, transparency for developers, increased automation for DevOps and advanced vulnerability detection for Security.

Overcoming Architectural Restrictions

These components, typically used in highly secure environments, become a real challenge when providing a solution for accessing open-source packages hosted in public repositories on the Internet:

- ✔ **Air Gap and the DMZ:** In this setup, there is a physical separation between internal and external networks, which also includes a DMZ that enables an indirect exchange of data between the networks.
- ✔ **No open ports:** In this environment, all ports that could facilitate direct communication between the internal network and the Internet are closed, forcing all communications to be routed via the DMZ.
- ✔ **Vault file transfers:** A secure vault is used to transfer files between the internal network and the DMZ. In this context, the vault acts as a centralized location where files are temporarily stored and securely managed during the transfer process.

Improving Operational Performance

In addition to the restrictions of the physical architecture, another important aspect of the challenge was how to introduce more efficient procedures for developers requesting open-source packages. Before deployment of the current solution, the main issues that had to be addressed included:

Time-Consuming Download Process — Before the current solution, the process for downloading OSS packages and their dependencies from the Internet was complicated and time-consuming requiring developers to:

- ✔ Check if the requested package has been approved and stored in a local registry
- ✔ Open a ticket with IT to request new or updated packages
- ✔ Wait for approval and downloading of new packages to the DMZ
- ✔ Regularly check the local registry to see if the new packages are available

Custom Repository-Specific Solutions — The mechanism for requesting and receiving an open source package was based on custom programming which included human intervention. The code for each repository had to be written and maintained independently which made it very challenging to add more repositories making operations extremely difficult to scale. It also prevented developers from accessing additional repositories which might contain more suitable and secure versions of the functionality they require.

Limited Vulnerability Detection — Due to the complexities of transferring files from the Internet via the DMZ, it was only possible to scan open-source packages for potential vulnerabilities after they had been introduced into the development environment.

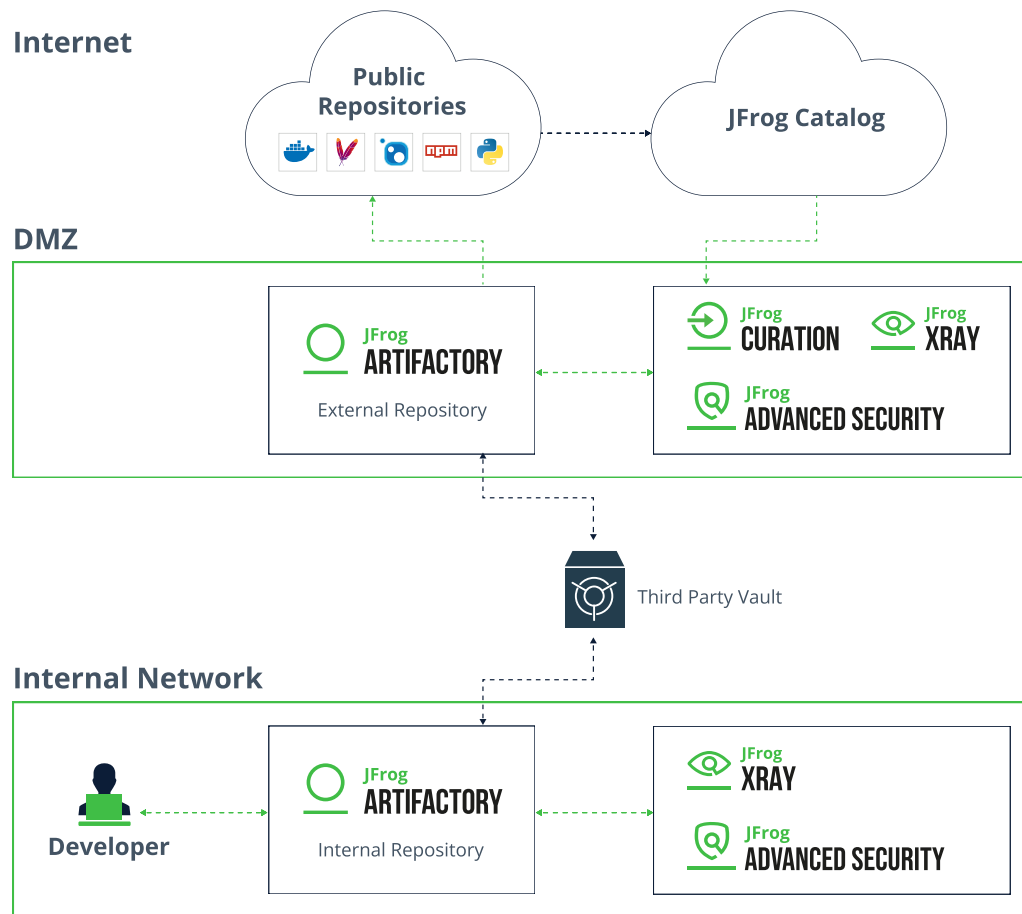
While this procedure may have succeeded in protecting the integrity of the internal network while allowing for the controlled use of external resources, it was a liability from both a developer experience and operational perspective, negatively impacting production efficiency and limiting developers from accessing the OSS resources they require.

In addition to the restrictions experienced by individual developers, from a DevOps perspective, not only was it slowing down the entire software development process, but also delayed releases due to vulnerabilities that were only discovered during the testing phase of the SDLC.

Solution

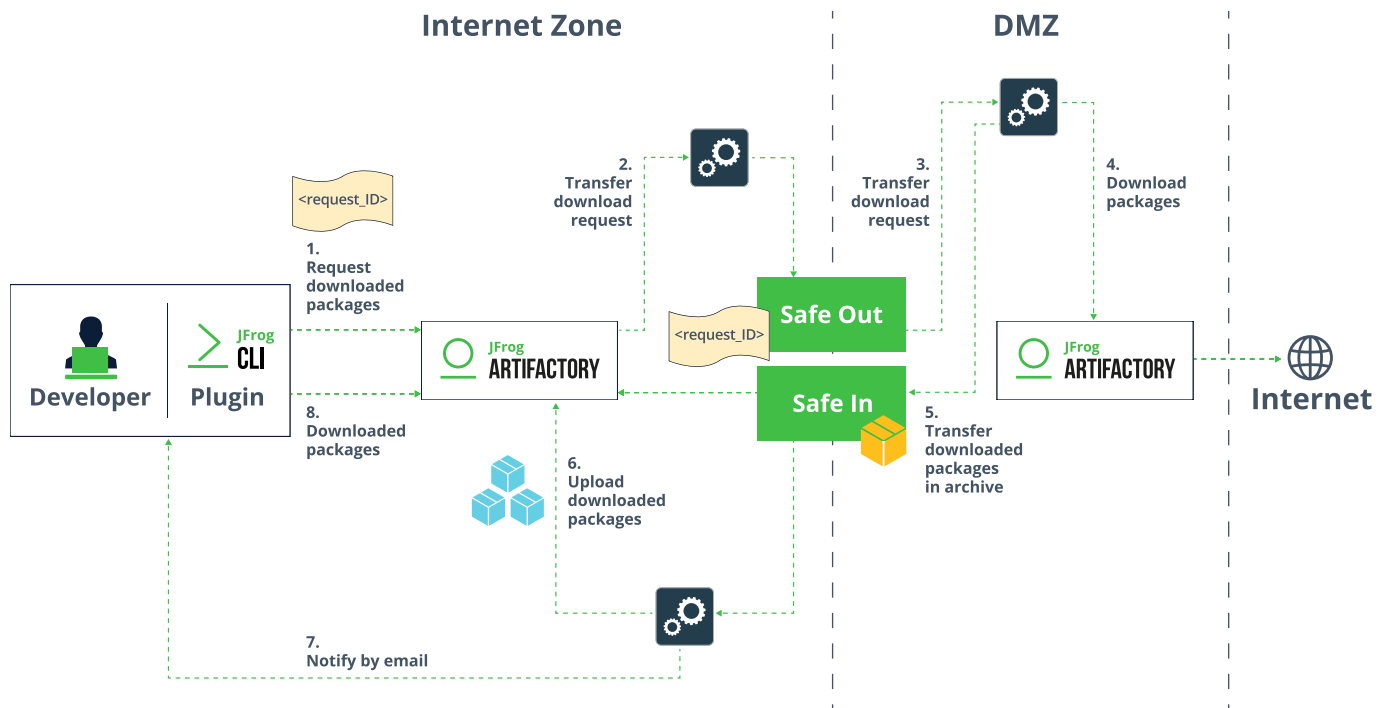
After careful analysis of the security and operational requirements, [the JFrog Professional Services](#) team suggested several options, based on different architectures and security approaches. After a number of in-depth discussions, they tailored the best solution based on the agency's needs, preferences, regulations and deployment considerations. In the end, the law enforcement agency opted for an architecture that separated the internal and external networks, while expanding secure access to multiple repositories and scanning for vulnerabilities before files are permitted to enter the DMZ.

The solution required multiple instances of JFrog Artifactory, JFrog Xray, JFrog Curation, and JFrog Advanced Security to manage and secure access to open-source software packages within a high-security Air Gap environment. Here is how the solution was implemented:



Multiple instances of JFrog Artifactory, Xray and Advanced Security were deployed on both the internal and external networks.

1. **JFrog Artifactory:** The first instance of Artifactory had already been deployed on the internal network. The second instance, Artifactory (EXT) was deployed inside the DMZ on the external network. On the internal network, Artifactory (INT) handled the download requests and sent them to the second instance. The approved requests were then scanned for vulnerabilities and downloaded from the Internet to the external instance of Artifactory (EXT). Approved packages are then sent to Artifactory (INT) for secure upload to the development environment. The entire process is automated and takes a developer 3-5 minutes per package request including direct and transitive dependencies.



The scalable JFrog Platform provides a seamless developer experience, handling multiple package types and easily integrating with leading tools in the development ecosystem

- 2. JFrog Catalog:** Provides a complete list of OSS packages and CVEs that help security professionals identify vulnerabilities, understand the operational risks and map connections between vulnerabilities, packages and dependencies. The data is also leveraged for contextual analysis and serves as a single source of truth for all OSS packages associated with a particular release.
- 3. JFrog Curation:** Open-source packages are scanned for vulnerabilities before being permitted to enter the DMZ. This enables early blocking of malicious or risky packages before they can become an issue. All external packages are curated automatically before they can be uploaded to the external Artifactory instance, eliminating potential threats before they can reach the development environment.
- 4. JFrog Xray & Advanced Security:** These modules continuously scan OSS packages for vulnerabilities and detect any packages that are either vulnerable or do not comply with licensing policies. They can also be fully integrated with CI/CD processes to automatically fail builds when potentially threatening vulnerabilities are detected. For purposes of ultimate traceability and adherence to emerging standards, they also support generation and export of a Software Bill of Materials (SBOM) in industry standard formats like SPDX and CycloneDX with support for VEX. Likewise, they also include advanced security features such as contextual analysis (for CVE applicability) and secrets detection.

Overall, this solution leverages the JFrog platform to ensure secure and efficient management of OSS packages in high security environments, while improving access to the most appropriate open source repositories, minimizing developer involvement and increasing development efficiency. That's a pretty significant accomplishment considering the architectural and operational constraints.

Results

JFrog Professional Services was instrumental in analyzing and understanding the agency's security needs and Air Gap architecture. After presenting a number of options, the agency deployed an automated solution where the [JFrog Platform](#) was deployed both on the internal network and inside the DMZ. After implementing the proposed solution, the following objectives were achieved:

- ✔ A seamless efficient software developer experience
- ✔ A single scalable solution for all open source package types
- ✔ A flexible platform with integration to complementary tools in the ecosystem

The agency's data security team were thrilled with JFrog Curation's ability to detect vulnerabilities prior to entering the DMZ and prevent them from ever reaching their internal software development environment. Likewise, the DevOps team were excited about having a scalable unified interface that is compatible with over [30 of the leading OSS package types](#), as opposed to the previous custom solution that could handle only NPM package types and could not be leveraged for other repositories.

By deploying multiple instances of JFrog Artifactory, Curation, Xray and Advanced Security both inside and outside the internal network, this law enforcement agency was able to cut the developer time for requesting and receiving open source packages down to 5 min per package, improving development efficiency by over 80% and providing a single unified and scalable interface for uploading multiple OSS package types from leading repositories such as NPM, Docker, Nuget and Pypi.

From a security perspective, automated curation of open source packages ensured that potential vulnerabilities were flagged in the DMZ and were never permitted to enter the development environment. These improvements enabled this leading law enforcement agency to maintain strict security requirements while improving the quality and efficiency of its software development operations.

If you are interested in how JFrog can help your government, military, law enforcement or other high security agency achieve greater access to open source software while maintaining high security and improving development efficiency, then [let's set up a demo](#) to see how the JFrog Platform can bring improved speed and security to your software development operations.

v1.0240129

ABOUT JFROG

JFrog empowers thousands of DevOps organizations globally to build, secure, distribute, and connect any software artifact to any environment using the universal, hybrid, multi-cloud JFrog Platform.



Start for free at <https://jfrog.com/start-free>

LEGAL STATEMENT

Copyright © 2024 JFrog LTD. JFrog, the JFrog logo, and JFrog Artifactory are trademarks or registered trademarks of JFrog LTD or its subsidiaries in the United States and other countries. All other marks and names mentioned herein may be trademarks of their respective companies.



www.jfrog.com



www.twitter.com/jfrog



www.facebook.com/artifrog/



www.linkedin.com/company/jfrog-ltd