How Host-Based Intrusion Detection Systems can Improve the Security of CI/CD Pipelines
DD2482 Essay

Tao Xiong taox@kth.se

Nils Löfberg nilslof@kth.se

April 2023

We certify that generative AI, incl. ChatGPT has not been used to write this essay. Using generative AI without permission is considered academic misconduct.

Contents

1 Introduction

Agile workflows are becoming increasingly prevalent in software development, and most significant companies use DevOps and Continuous Integration/Continuous Deployment (CI/CD) pipelines in their development. CI/CD pipelines, however, are not invulnerable to breaches or attacks. Different Host-Based Intrusion Detection Systems (HIDS) have been developed to improve the pipelines' security. In this essay, we will explore how HIDS works, its benefits to the pipeline, and how it can protect the integrity and reliability of the code in the pipeline.

2 CI/CD Pipelines

2.1 Brief introduction to CI/CD pipelines

The continuous integration/continuous delivery (CI/CD) pipeline is an agile DevOps workflow based on the idea of a frequent and reliable software delivery process. In a CI/CD pipeline, there are four phases: Build, Test, Deliver, and Deploy, which are all deployed continuously in an endless loop. Different automated workflows can be integrated into different parts of the process, and different parts of the pipeline, which helps DevOps teams to quickly, continuously, and reliably deliver the software.

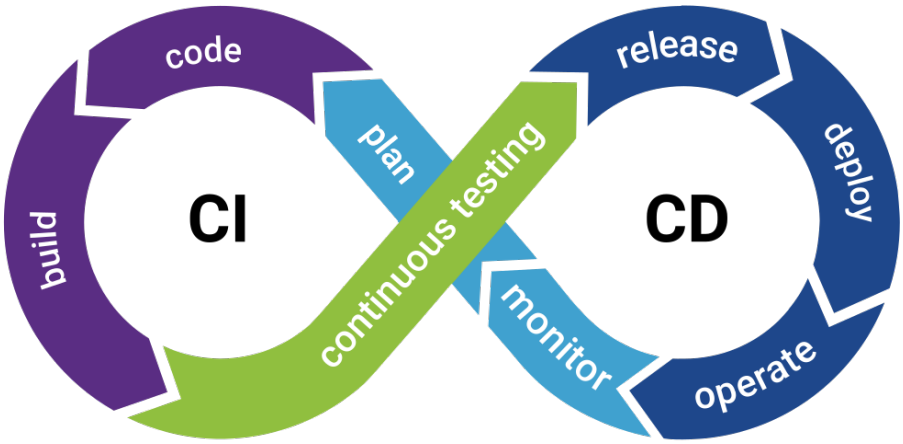

Figure 1: CI/CD workflow

The software integrated into the pipeline ranges from automated testing frameworks to automated code cleanup or access management. Whatever software is chosen to be integrated into the pipeline is dependent on the needs of the

project itself, as some software comes with heavy drawbacks.

As a result of this, The CI/CD pipeline accomplishes the automation of software releases, from initial testing to the final deployment. It also comes with other benefits for software development and deployment, like reducing time to deployment and ease of release.

2.2 Security challenges in CI/CD pipelines

The benefits of working with a CI/CD pipeline are numerous, but it does have its drawbacks. In traditional development, which has big and scarce updates, the code can be more rigorously tested, as it is not released immediately. In a CI/CD pipeline, code can be deployed on the same day that it is written, and even though there are tests, the possibility of errors is still high. This issue has also become more prevalent recently as we are currently experiencing a global uptick in cyber attacks in general (see figure 2)

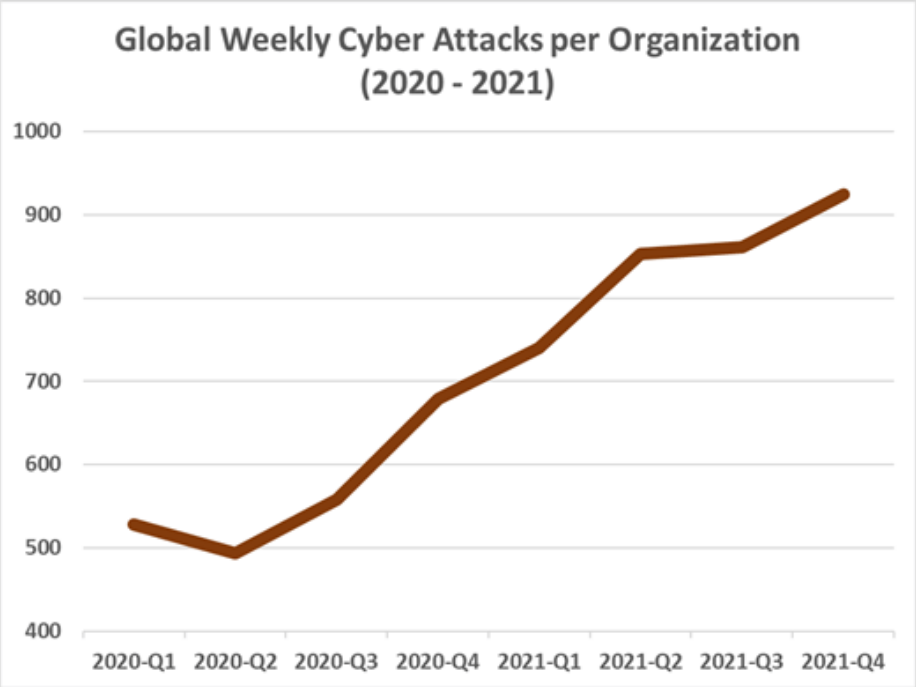

Figure 2: Global weekly cyber attacks per organization [1]

It is impossible to list all the available security issues with any workflow, but the most regarding CI/CD pipelines, the following are the most prevalent [2].

2.2.1 Insecure Frameworks

Implementing frameworks into the pipeline is necessary for the pipeline to live up to its potential. In most cases, these frameworks are written by a third party, not associated with the pipeline, although it is possible to write frameworks yourself. This introduces a security risk, as the framework needs access to the code base to work correctly and can contain both insecure codes as well as malware.

2.2.2 Supply-chain Attacks

Any software project, be it big or small, will almost certainly make use of dependencies and libraries written by third parties. This is common practice in software development and almost impossible to develop without. The problem is that these libraries are often targets for attacks, and weak spots can be found years after release, making them a constant risk.

2.2.3 Insufficient Access Controls

For a pipeline to live up to its potential, it must implement some sort of access control. Not all actors should be able to access all code or use all build tools, as this introduces security risks. The build tools themselves require access to some, but not all, of the pipeline and should therefore be restricted due to the principle of least privilege. If these access controls are improperly managed, the security of the pipeline is reduced.

2.2.4 Untested Code

CI/CD pipelines are designed to accelerate the delivery of code from production. As a result, newly developed code might not be tested properly due to a lack of time. This can result in a weak spot with insecure code that can both break and be susceptible to malware breaches.

3 Overview of Host-Based Intrusion Detection Systems

3.1 Brief introduction to HIDS

Host-based intrusion detection systems, also called HIDS, are applications that monitor a computer or agent for suspicious activities. The monitored activities can include intrusions created by external actors and a misuse of resources or data internally.

There are many different types of HIDS, but generally, they monitor and analyze data for suspicious activities. Then when the HIDS agent detects a potential security threat, it either raises an alert to the DevOps team or quarantines the

problem. The alert provides details about the activity and the potential threat level, enabling the team to take appropriate action. Finally, depending on the severity of the alert, the DevOps team can respond in a variety of ways, including isolating the affected host, disabling user accounts, or rolling back changes. [3]

3.2 Capabilities of HIDS

3.2.1 Host Intrusion Detections Systems Based on Signatures

This type of intrusion detection system focuses on searching for a previously known pattern, identity, or a specific intrusion event. Most IDSs are coming from a definitions database that needs regular updates to keep up with regular and known cyber threats. As long as the database is up to date, this type of IDS will do a good job.

3.2.2 Host Intrusion Detections Systems Based on Anomalies

As opposed to signature-based HIDS, anomaly-based ones rely more on analyzing "trustworthy behavior" and use machine learning techniques to flag malicious behavior. This will translate into a higher false-positive rate, as the system will also flag legitimate behavior as well. Anomaly-based IDS is a good option for determining when someone is probing your network prior to a real attack taking place. The success of this type of IDS also depends on the degree of distribution across the network and the level of training provided by the IT admins.

4 Benefits of Using Host-Based Intrusion Detection Systems in CI/CD Pipelines

4.1 General Benefits

Depending on the type of HIDS used, implementation into the pipeline can range from very simple to relatively difficult. The choice to do so should be based on the needs and regulations of the pipeline and software. Some HIDS can require continuous maintenance to work properly, but it is intended to save time and money in the long run.

Despite any difficulties in implementation or maintenance, the benefits of implementing HIDS into the pipeline of a CI/CD system can be extensive. Basing arguments on the security challenges outlined in subsection 2.2, HIDS can help improve the security of the pipeline on several fronts.

The first argument outlined in subsection 2.2.1 relates to the introduction of third-party software into the pipeline. As the software is introduced into the

pipeline, it will be checked by whatever HIDS is there. If the software contains known malware, a HIDS with Indicator of Compromise (IOC) signature checking will be able to detect it and contain it before it can be run. This, in turn, will stop the entire pipeline from being infected. [4]. Other HIDS could also be able to detect suspicious access requests from the framework and alert the DevOps team that way instead.

The second argument, outlined in subsection 2.2.2 outlines the problem of supply-chain attacks. In these cases, there is little anyone can do to prevent these. What can be done is to monitor the web for known dependency weaknesses and isolate it as soon as possible. Some HIDS can do this in a similar way in that they search for IOC signatures of malware. Apart from that, they can also monitor for suspicious behavior in the system itself and, hopefully, at least disclose that there is a problem, even if the said problem cannot be located right away. Unfortunately, this is not always possible, however. [5]

The fourth argument, outlined in subsection 2.2.4 entailed the fast-paced environment in which CI/CD pipelines live. Code is constantly written, built, tested, and shipped out on a continuous basis, and it can be hard to test the code for breaches or security flaws properly. If any individual device connected to the pipeline has been compromised, the entire system is in danger. By implementing HIDS, organizations can monitor and analyze the behavior of individual devices and systems in real-time, thus being able to identify potential breaches before the damage to the pipeline becomes too severe [6]. Furthermore, these tests can be run on the pipeline as a whole, creating another layer of protection.

On the whole, HIDS can give numerous different levels of security to the pipeline, with a focus on monitoring individual devices and systems as well as the actions they make to the pipeline. This monitoring can result in the early detection of security threats and, thus, a quicker response to eventual incidents. It also gives improved visibility into the pipeline which can mitigate and reduce the number of manual security efforts required. [7]

4.2 OSSEC and FIM

Many different companies use many different HIDS in their pipelines. It is, therefore, impossible to say precisely how HIDS is used in practice in general. We will therefore present a HIDS called OSSEC and a method within it called FIM and explain the connection between it and CI/CD pipelines.

OSSEC (Open Source HIDS SECurity) is a free and open-source host-based intrusion detection system (HIDS) used for security auditing, threat detection, and compliance monitoring [8]. FIM (File Integrity Monitoring)[9] is a feature included in OSSEC that allows OSSEC to monitor changes to files and directories and alert administrators when unauthorized modifications occur. FIM can also be used to detect malware and other malicious activity that modifies

system files. With these capabilities, FIM can be integrated into the DevOps pipeline to avoid some security challenges.

4.3 Qualys

Qualys is an American technology firm that developed its own FIM [10]. Generally speaking, Qualys File Integrity Monitoring (FIM) is a highly scalable app that enables a simple way to monitor critical files, directories, and registry paths for changes in real-time. When it comes to CI/CD pipeline part, Qualys FIM's real-time monitoring helps to detect and report malicious, unauthorized, anomalous activities in CI/CD pipelines. It provides visibility into who made the changes (user and process), full file paths and registry paths, the exact time of the change, and the actual change. Qualys FIM's automated event correlation and alerts help the companies' DevOps system with a smoother incident management process.

By using Qualys FIM, Security and compliance teams can make sure critical files and registries are set to be monitored for production images before they go into production. In CI/CD pipelines, we should integrate Qualys FIM during the Continuous Deployment (CD) phase. Once the instances are in production, Qualys FIM provides comprehensive assurance that critical directories, files, and Windows registries are monitored for changes, which can ensure our CI/CD pipelines are in a safe stage. In that case, we can avoid some security challenges in CI/CD pipelines, like insufficient Access Controls and Untested Code, as we mentioned in subsection 2.2.

As a result of these capabilities, Qualys FIM is used by a wide range of organizations, including Fortune 500 companies, government agencies, and small and medium-sized corporations.

5 Limitations and Challenges

One of the main limitations of HIDS is that they are bound to their host environment. This means that they are unable to detect, and therefore not able to prevent or contain, breaches coming directly from the network, such as distributed denial of service attacks, network scanning, or traffic redirection attacks. These attacks, however, can be solved by a network-based intrusion detection system (NIDS), while HIDS focus on attack coming from within an already connected host.

Another limitation of HIDS is that they rely on the system logs to identify signatures of known security threats and breaches. Since these IOC signatures are compared to knows signatures in a database, HIDS are less equipped to find pre-

viously unknown attacks, commonly referred to as zero-day attacks. Zero-day attacks can still be found by monitoring the system logs for suspicious behavior, but this can easily cause many false positives as well and is not always a recommended practice.

Furthermore, based on the same principles of monitoring system logs, HIDS can not detect a breach that has not already begun. This is because suspicious behaviors or knowing IOC must first appear in the host environment for the HIDS to react, and even then, it can take some time to contain the problem, depending on the scale of the problem and the setting of the HIDS. This is a flaw rooted in the basis of HIDS and cannot be fixed without completely overhauling how HIDS works. This is not to say that there does not exist HIDS with real-time analysis tools, as mentioned previously in section 4.1. [6]

Finally, as mentioned previously, both NIDS and HIDS are prone to give false positives. In other words, they flag suspicious activity when there is none. Besides wasting company time and resources, a likely result of this is that the DevOps team will find it to be tedious to deal with the alerts and flags raised. This, in turn, can lead to the team ignoring important alerts, which reduces the effectiveness of the HIDS as a security measure.

In conclusion, while HIDS provides a useful layer of protection and security for the CI/CD pipeline, it is not an all-encompassing solution. It is important to be aware of what it cannot do as much as it is to be aware of what it can do. HIDS should be used in combination with other security measures, such as NIDS, access management, and vulnerability scanning, Thus providing multiple layers of security from many angles of attack.

6 Conclusion

CI/CD pipelines and general DevOps practices are becoming more and more prevalent in software development. These pipelines improve the efficacy of development but come with some drawbacks in terms of security. HIDS can be used to negate some of the drawbacks when implemented into the pipeline. They do this by analyzing system logs and the activity of individual hosts within the pipeline to detect breaches when they happen and hopefully contain them before too much damage is done. However, they are not a fix-all solution and should be used in conjunction with other security measures to create a multi-layered security to the CI/CD pipeline.

References

[1] Checkpoint. *Check Point Research*. Accessed: 2023-04-21. URL: https://blog.checkpoint.com/security/check-point-research-cyber-attacks-increased-50-year-over-year/.

[2] Checkpoint. *Why CI/CD Security is Critical*. Accessed: 2023-04-19. URL: https://www.checkpoint.com/cyber-hub/cloud-security/what-is-ci-cd-security/.

[3] Heimdalsecurity. *What Is a Host Intrusion Detection System (HIDS) and How It Works*. Accessed: 2023-04-16. URL: https://heimdalsecurity.com/blog/host-intrusion-detection-system-hids/.

[4] Chiadighikaobi Ikenna Rene and Johari Abdullah. "Malicious code intrusion detection using machine learning and indicators of compromise". In: *International Journal of Computer Science and Information Security (IJCSIS)* 15.9 (2017). DOI: 10.1145/2976749.2978315.

[5] Xinyuan Wang. "On the feasibility of detecting software supply chain attacks". In: *MILCOM 2021-2021 IEEE Military Communications Conference (MILCOM)*. IEEE. 2021, pp. 458–463. DOI: 10.1109/MILCOM52596.2021.9652901.

[6] Kaining Lu et al. "An adaptive real-time intrusion detection system using sequences of system call". In: *CCECE 2003-Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No. 03CH37436)*. Vol. 2. IEEE. 2003, pp. 789–792. DOI: 10.1109/CCECE.2003.1226013.

[7] Thorsten Rangnau et al. "Continuous Security Testing: A Case Study on Integrating Dynamic Security Testing Tools in CI/CD Pipelines". In: *2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)*. 2020, pp. 145–154. DOI: 10.1109/EDOC49727.2020.00026.

[8] Opstree. *Host-Based Intrusion Detection Using Ossec*. Accessed: 2023-04-19. URL: https://blog.opstree.com/2022/01/18/host-based-intrusion-detection-using-ossec/.

[9] Kanstantsin Halavan and Evgenij Suharev. "Information integrity monitoring as a part of host intrusion detection system". In: (2017). URL: https://elib.psu.by/handle/123456789/37029.

[10] Qualys. *File Integrity Monitoring*. Accessed: 2023-04-22. URL: https://www.qualys.com/docs/fim-datasheet.pdf.