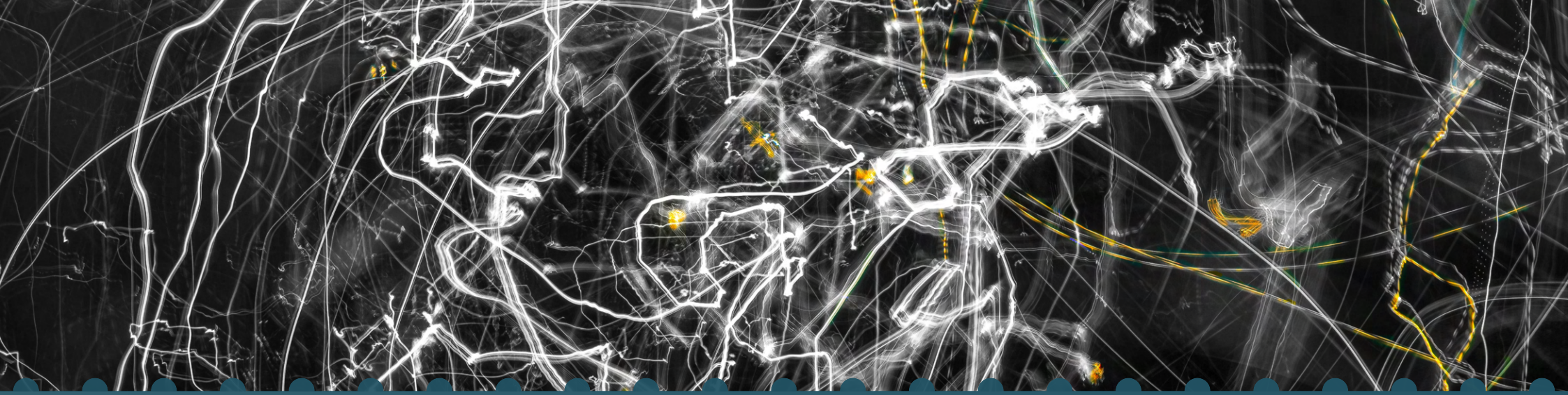


# Exploring Our Automatic hardening using Tailor-made AppArmor profiles

Discover how our cutting-edge technology makes managing AppArmor easier than ever before. We'll walk you through the challenges of Modern application security, AppArmor profile creation, and show you how our solution can provide reliable and effective security for your containerized applications.

# Modern Software is made possible by open-source

Modern cloud applications use open-source libraries and frameworks, which provide developers access to a vast array of pre-built functionality that accelerate the development process.

One of the defining characteristics of modern cloud applications is their reliance on open source. Today, it is common to have hundreds of dependencies in a given codebase. These dependencies are essential building blocks enabling developers to rapidly build powerful and flexible applications. However, the sheer number of dependencies that modern applications rely on makes maintenance and security challenging.

CI/CD help teams set up pipelines to deliver code to the production environment frictionless. The CI/CD pipeline builds, tests, and deploys the solution, automatically. This enable developers to quickly iterate on their code, catch and fix errors early in the development process, and deploy new features and bug fixes quickly and confidently.

# Security implications across the supply chain are real threats to your software

The fast developer velocity made possible by modern toolchains also bring multifaceted security threats, which require careful consideration and mitigation to ensure the production environment remains secure.

1. **Vulnerabilities in the container.** Insecure configurations, outdated dependency versions, or malicious code hidden within the image constitute a severe threat. Attackers can exploit these vulnerabilities to gain unauthorized access to the application or underlying infrastructure.

2. **The container runtime environment.** Containers share the underlying operating system and kernel, meaning a vulnerability in one container can compromise the entire host. Ensuring that containers are properly configured, patched, and monitored is crucial to prevent attacks such as container breakouts, privilege escalations, and unauthorized access to sensitive data.

3. **The container orchestration systems.** Access to the underlying infrastructure requires privileged access for some user accounts, making these accounts a high-value target for attackers. Proper access controls and monitoring are crucial to prevent unauthorized access or tampering.

# Protecting your service

Several security measures are required to make the production environment secure.

1. **Image Security.** Scan for vulnerabilities and malicious code; only trusted images should be used. Secure configurations and software updates should also be applied to mitigate known vulnerabilities.
2. **Network Security.** Access controls, network segmentation, and isolation must be implemented to ensure containers run securely. Intrusion detection systems and continuous monitoring help track events and inform teams.
3. **Orchestration & Infrastructure Security.** Proper access controls and monitoring are in place to ensure that container orchestration systems are not compromised.

**25k**

New Vulnerabilities discovered in 2022 alone.

**2/3**

of breaches uses already known vulnerabilities.

# Containerized software have large attack surfaces that requires more complex security

As the threat level evolves, traditional measures might not be sufficient to address the unique challenges posed by containerized applications.

AppArmor profiles can be used to isolate and secure Containers from other containers on the host system. AppArmor proactively protects the container from external and internal threats by specifying what system resources an indicial container can access and what resources are denied, such as file access and what sockets and ports to use.

However, creating and maintaining AppArmor profiles is complex and time-consuming, significantly when the containerized application constantly changes during development. Each time the application changes, the AppArmor profile may need to be updated to ensure it's still securing the application properly.

AppArmor is a security feature in Linux that allow profiles for individual applications or services. These profiles can restrict an application's access to certain resources, such as files, network sockets, and system calls. Read more at https://apparmor.net

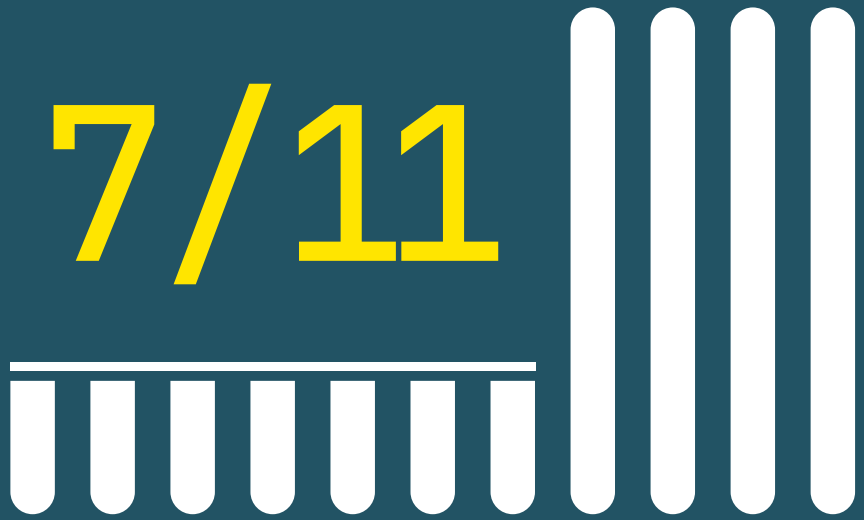
# Introducing bifrost security

bifrost automatically generate AppArmor profiles. Hooking into the development pipeline and testing environments, bifrost uses audit event log data to tailor-made the profiles.

1. **Enhanced security.** The AppArmor profiles help prevent attackers from exploiting vulnerabilities that might exist in the container, known and unknown ones, reducing the risk of security incidents.

2. **Increased compliance.** bifrost continuous auditing of the software behavior provides better insights. bifrost can block, log and alert when irregularities occur, supporting PCI, ISO 27001, and SOC2.

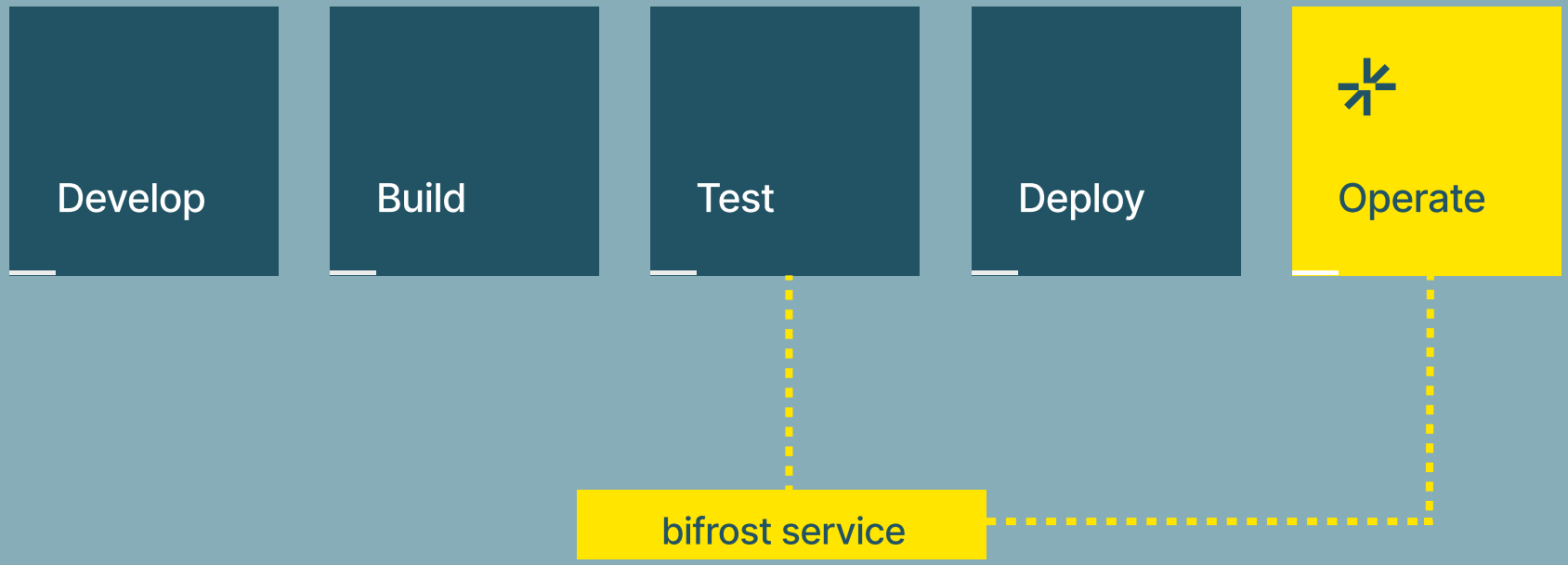
3. **Reduced workload.** By automating the process of AppArmor profile creation, the tool saves time and effort, allowing the organization to focus on developing core features.




Successfully blocked exploits using bifrost profile*

*Research demonstration showing a custom profile blocking 7 of 11 identified vulnerabilities compared to no profile.

# bifrost integrates into the CI/CD pipeline

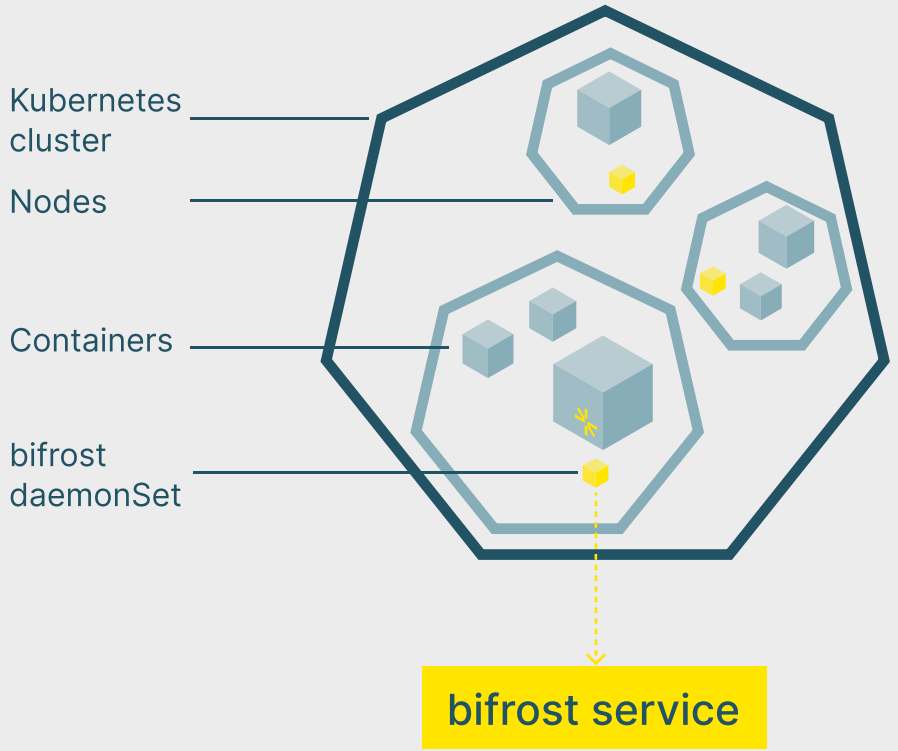



Our lightweight agent stream behavioral non-sensitive signals from a test/staging environment to capture the intended behavior of the containerized software.

At deployment, a tailor-made security profile is generated for the container. At runtime, the profile will block and alert any attempts of unintended behavior from the container, stopping potential attacks.

# Integrated, bifrost can audit and push profiles

Our lightweight and open-sourced agent is easily deployed into the Kubernetes cluster as a daemonSet. With a pre-configured helm-chart, limited permissions, and resources, deployment is straightforward.




1. **Describe the environment.** name clusters and environment, and the services in bifrost portal.

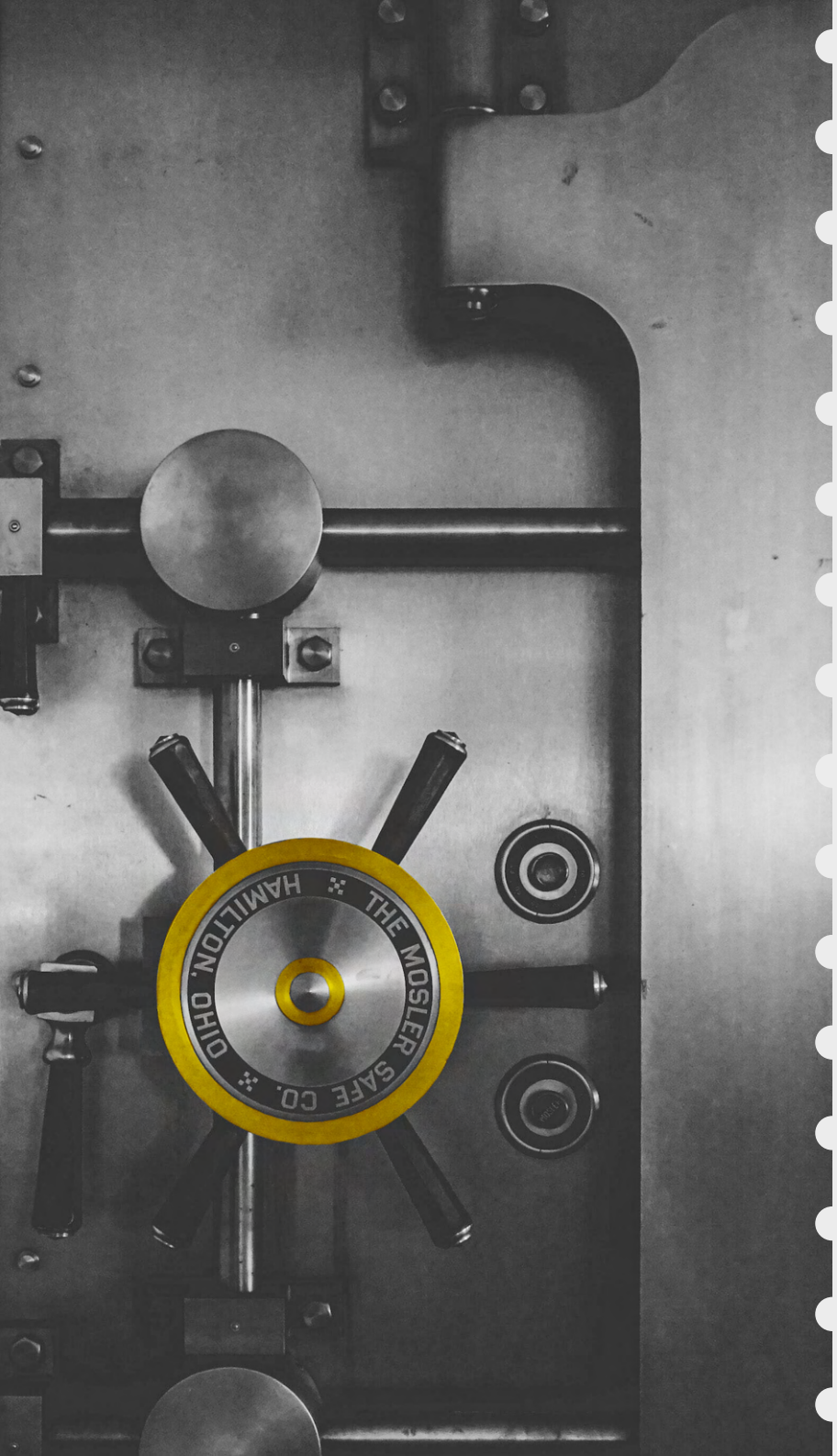
2. **Deploy the daemonSet.** Using the pre-configured helm chart, drop in the bifrost agent into the cluster.

3. **Update container deployment.** Using the bifrost API, adjust the deployment for the given service to a given environment. This will let the agent know what container to audit.

4. **Deploy container.** Next deployment, the bifrost agent will start streaming the audit events to the bifrost service.

5. **Request a tailored profile.** To try out a profile, use bifrost API again at the deployment stage. The agent will distribute an AppArmor to the correct cluster, ready to be used.

6. **Get notified.** Setup notifications for any potential security boundary violation that might occur.




# With bifrost you'll have security that evolves with your software

Automatic, tailor-made AppArmor profiles that continuously capture the moving organism that is modern software help companies raise their runtime security posture while providing better insights, less manual work, and fewer false positive alerts.

## Sounds interesting?

Visit bifrostsec.com or get in touch.

Hannes Ullman
hannes@bifrostsec.com
+46 (0) 733 211 638