Security Document

Group 4

Evgeni,Stefan-Nikola,Vlad,Naadir,Thomas,Nikolay,Renis

Introduction

This report outlines the security measures for a distributed hash cracking project using an Azure Kubernetes Service (AKS) cluster. This university group project involves cracking hashes by distributing the task across multiple pods, each handling a segment of a large wordlist. Ensuring the security of this project is paramount, given the sensitive nature of the data and computations involved.

1.Azure Kubernetes Service (AKS) Security

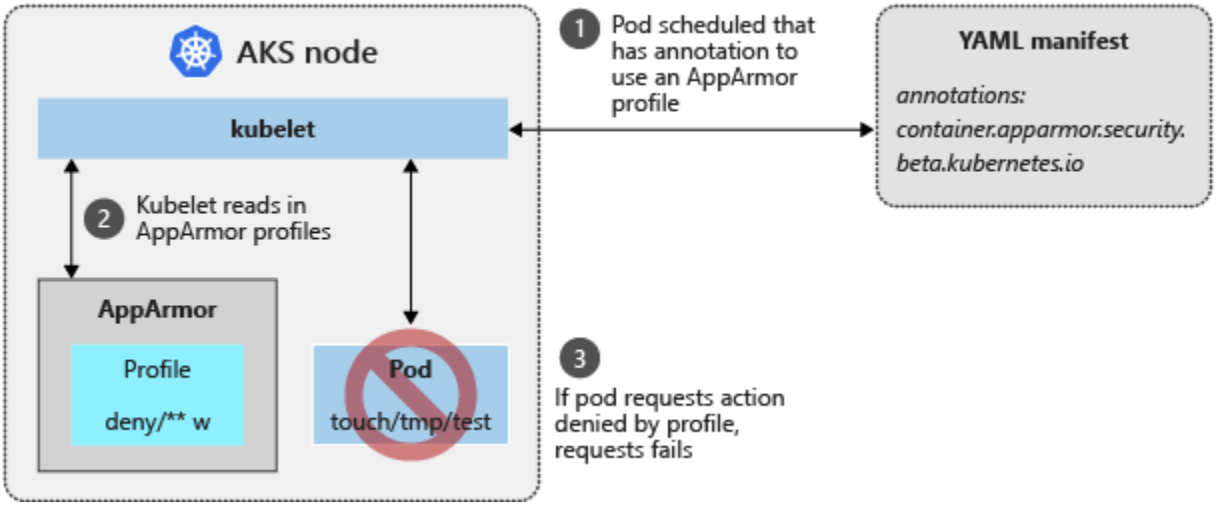
1.1.Network Security

Network policies are implemented to control the communication between pods, ensuring that only authorized pods can communicate, which limits the potential damage from a compromised pod. The AKS cluster is deployed within an Azure Virtual Network (VNet) to isolate it from other resources, and private IP addresses are used to secure traffic within the VNet. Network Security Groups (NSGs) and Azure Firewall are utilized to control and filter traffic at the network boundary, with rules defined to allow only necessary traffic to and from the AKS cluster. Additionally, Azure Private Link is used to access Azure services like Azure Key Vault and Azure Storage over a private endpoint, ensuring that the traffic remains within the Azure network and does not traverse the public internet.

1.2.Identity and Access Management

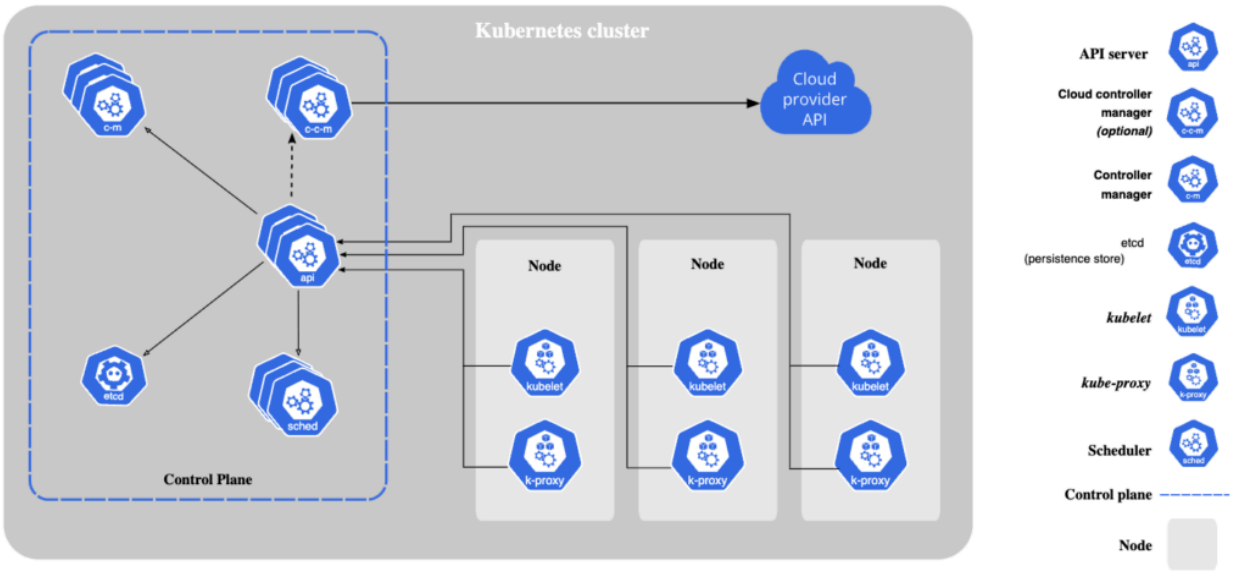
The AKS cluster is integrated with Azure Active Directory (AAD) for managing user access to the Kubernetes API server. Azure Role-Based Access Control (RBAC) is used to grant users and applications only the necessary permissions. Azure Managed Identities are employed to grant the AKS cluster access to other Azure resources without the need to manage credentials, enhancing both security and ease of management. Kubernetes RBAC is

implemented to control access to Kubernetes resources within the cluster, with roles and role bindings defined to enforce the principle of least privilege.



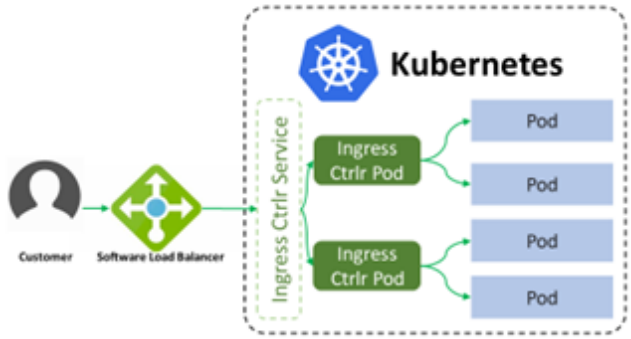
1.3.Cluster and Node Security

Node pools are used to separate workloads with different security requirements, allowing for the isolation of critical tasks from less secure ones and reducing the attack surface. Pod Security Policies (PSPs) are enforced to control the security properties of pods, restricting the use of privileged containers and controlling access to the host network and filesystem. Sensitive data such as passwords and API keys are stored in Kubernetes Secrets, and these secrets are encrypted at rest using Azure Key Vault for enhanced security.

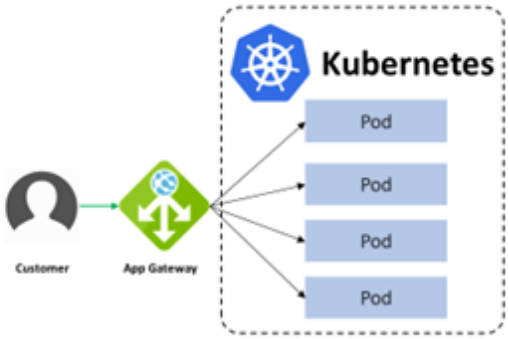


1.4.Container Security

Container images are regularly scanned for vulnerabilities before deployment to the cluster, with Azure Security Center used for continuous image scanning and vulnerability assessment. Docker Content Trust is used to sign container images, ensuring that only trusted images are deployed in the AKS cluster. Runtime security is monitored using tools like Azure Security Center for Kubernetes, which helps detect anomalies and respond to threats in real-time.



In-Cluster Ingress Controller



App Gateway Ingress Controller

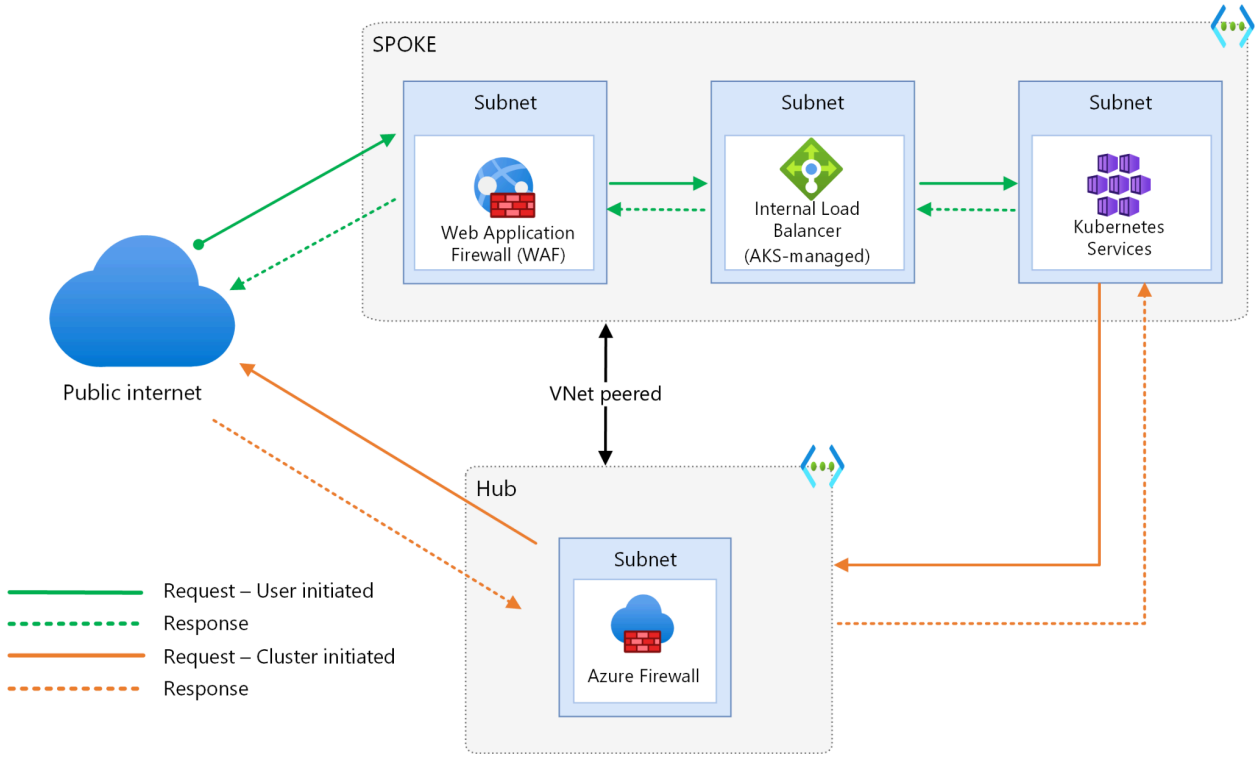
2.Kubernetes Security

2.1.Kubernetes API Security

The Kubernetes API server is protected by using SSL/TLS for secure communication, and IP whitelisting is implemented to limit access to the API server. Audit logging is enabled to keep track of all requests to the Kubernetes API server, aiding in forensic analysis and compliance.

2.2.Network Security

Traffic between pods is encrypted using mutual TLS (mTLS), ensuring that communication between pods is secure and authenticated. Consideration is given to implementing a service mesh like Istio or Linkerd for advanced traffic management, security features such as mTLS, and observability.



2.3.Workload Security

Resource quotas and limits are implemented to prevent resource exhaustion attacks by limiting the CPU and memory usage of pods. Pod Disruption Budgets are used to maintain high availability during voluntary disruptions such as maintenance or upgrades.

2.4.Logging and Monitoring

Centralized logging solutions such as Azure Monitor or the ELK stack are used to collect and analyze logs from the cluster, aiding in security monitoring and incident response. Monitoring and alerts for critical security events are set up using tools like Prometheus, Grafana, and Azure Monitor to detect anomalies and respond to potential security incidents promptly.

3.Risk analysis

Name	Damage	Probability	Countermeasure
DDOS	Medium	low	Load balancer in the cluster that will limit the amount of requests
Connection	High	low	Strong network rules

interference			+ constant data checkups(on input and output)
XSS	High	low	Request filtering + data validation and filter
Brute-force access	High	Medium	We are using RBAC authentication in AKS so that only privileged Azure users can access the cluster
Node fails (calculation output is wrong or pod starts a wrong calculation.	Low	low	K8s manages the health of pods and nodes for us and in moments of issues we get informed.
Hacked pod	low	low	Docker containers are secured using the best principles of all upper level technologies that we are going to use.

4.Conclusion

Implementing robust security measures at multiple layers within the AKS cluster and Kubernetes environment is crucial for protecting the distributed hash cracking project. By following best practices for network security, identity and access management, cluster and node security, container security, and logging and monitoring, the risk of security breaches can be significantly reduced, ensuring the integrity and confidentiality of the data and computations involved.