

CHAPTER 4

Infrastructure Security Patterns

Organizations typically focus on building applications and data-driven software to harvest value from their data. To create secure and reliable applications, setting up an infrastructure is a must. Security configurations must be done correctly in order to ensure that the applications function correctly, without any vulnerabilities or security concerns.

The previous chapter explored the infrastructure security patterns and discussed how to set up public cloud resources.

This chapter covers the following topics:

- Physical security
- Built-in Azure security controls
- Azure Tenant Security
- Container security
- Securing Azure resources

Physical Security

When you use public cloud platforms like Azure, the Microsoft Azure team has to secure the data centers and resources used in the background for the IaaS, PaaS, and SaaS services.

These data centers and resources should comply with industry standards to maintain the security and reliability of the application. Managing and maintaining these data centers and resources is handled by the Microsoft operations staff in the background, so customers can continue their work without the headaches of configuration, management, and maintenance.

Microsoft Azure has a globally distributed infrastructure that can support thousands of online services and secure facilities all over the world. There are a total of 58 regions and 140 countries worldwide. Microsoft Azure regions comprise an interconnected set of data centers via a massive network. This network includes distribution, load balancing, and encryption for ongoing traffic between the regions and outside the regions. Microsoft Azure has the largest number of regions across the world compared to other cloud providers. See Figure 4-1.

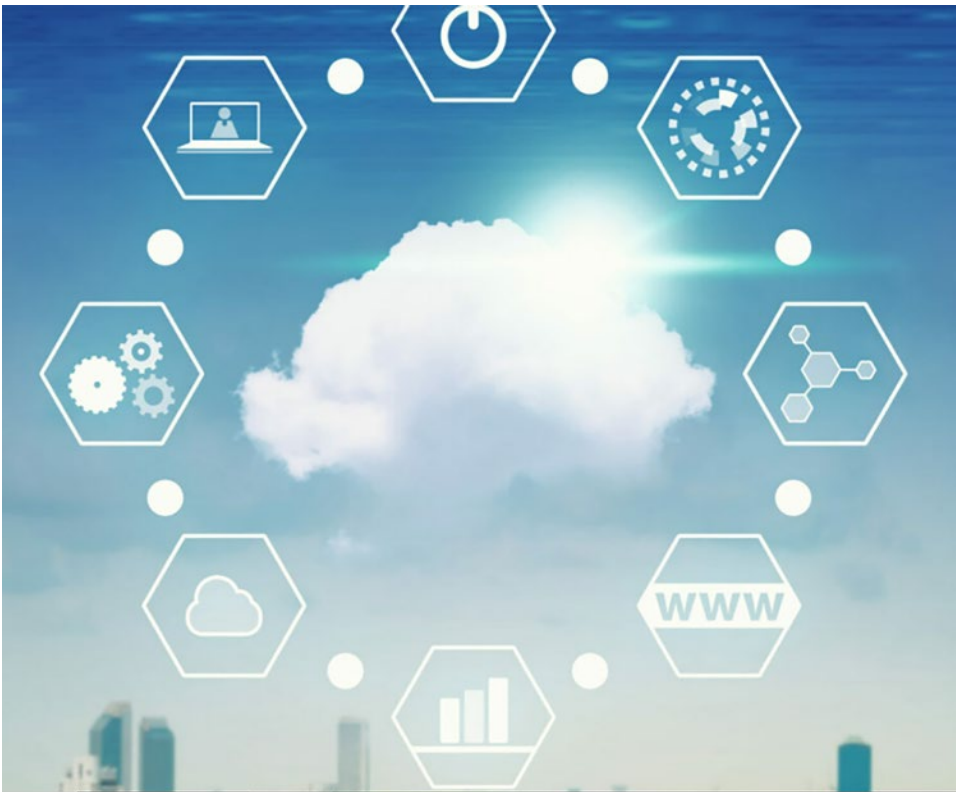


Figure 4-1. Azure regions worldwide

Azure regions are organized into geographies. This ensures that the data residency and the compliance requirements are honored with respect to the geography and the regions. These geographies are also fault tolerant, in case of any failures in the region, with dedicated infrastructure.

Each Azure region consists of availability zones. Availability zones are physical, separate locations within an Azure region. Each availability zone consists of multiple data centers with cooling, networking, and compute capabilities.

Figure 4-2 shows the overall global infrastructure with region pairs and availability zones within the data boundary for back up, disaster recovery, and high availability. This geographically distributed data center enables Microsoft to reduce network latency and backup failover.

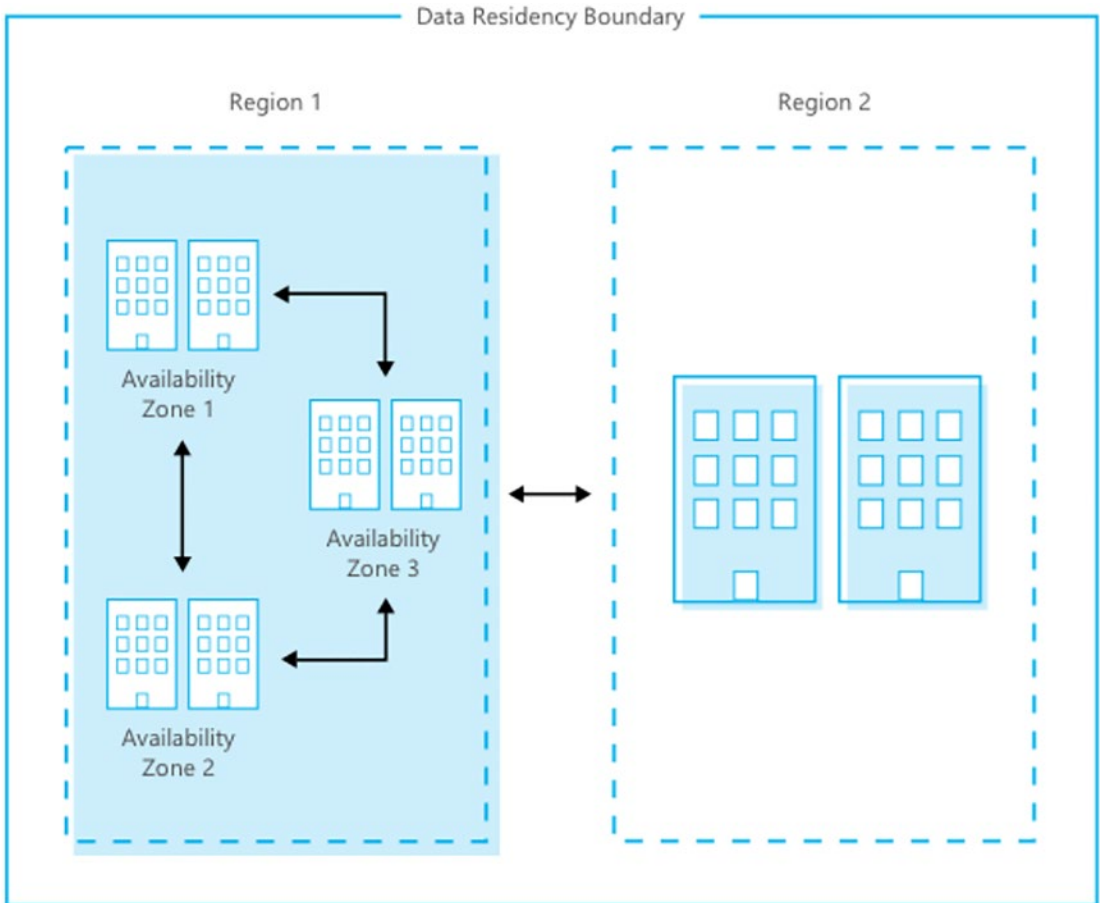


Figure 4-2. Azure regions and availability zone

- Physical security reviews: For a public cloud, you have to periodically perform a physical security review to address the Azure security requirements. End users don't have physical access to the Azure services and data centers where the Azure resources reside.
- Data bearing devices: Microsoft uses best practices and various industry standard procedures to handle the data-bearing devices. There might be various hard drives that can't be wiped.

- **Compliance:** The Microsoft Azure infrastructure also has a set of international compliance standards, such as ISO 27001, HIPAA, SOC 1, and SOC 2.

Let’s explore in detail the built-in Azure security controls.

Built-in Azure Security Controls

While using a public cloud service, it is very important for enterprise organizations to know about the shared responsibility between the cloud customer and the public cloud offerings. Public clouds have three types of services: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Responsibility for these services differs based on the type of service you choose.

Once you move the workload to the cloud, responsibility for the security will vary based on the type of cloud service you use. For any type of cloud deployment, customers have to use their own data and identities. The end customer or the cloud consumer is responsible for protecting the data security, the on-premises resources, and the cloud components. The following features should always be maintained by the cloud consumers:

- Data
- Endpoints
- Account
- Access management

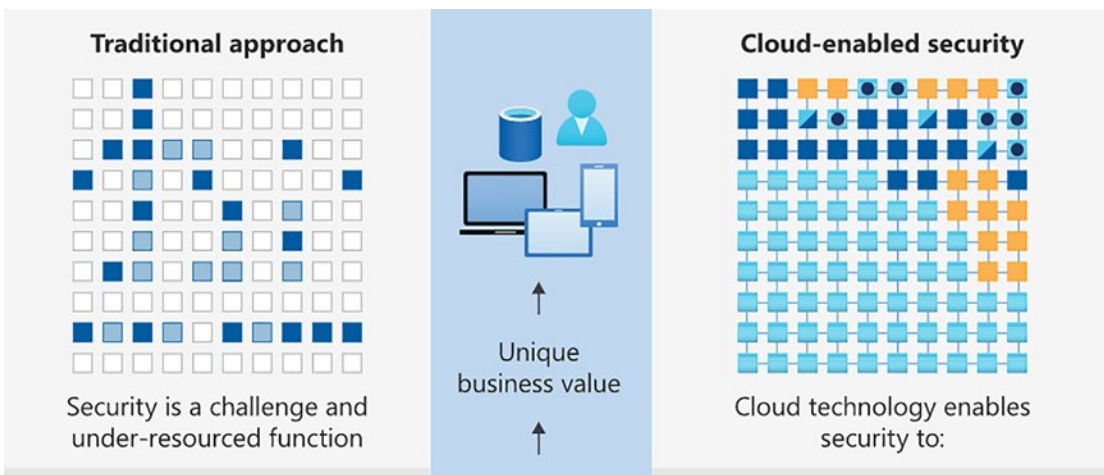


Figure 4-3. Shared responsibility: traditional vs cloud

Figure 4-3 shows the responsibility of the traditional application as well as the cloud application. One of the major reasons for using Azure cloud and its service is that it has a range of built-in security services and features that can be easily integrated to monitor the security features.

- Azure platform: With the Azure cloud platform, you can select from a range of operating systems, programming languages, frameworks, databases, and tools. See Figure 4-4.

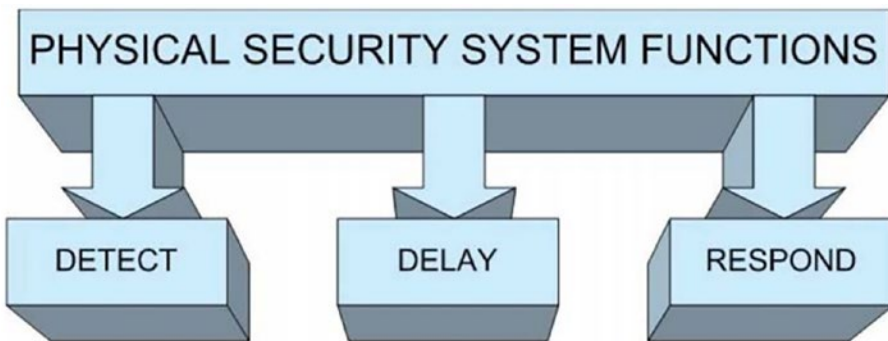


Figure 4-4. Azure platform security

Azure's public cloud infrastructure is designed to provide facility to applications to host millions of customers and provide a foundation when the security requirements are met. In addition to this, Azure also provides various security configurations so you can customize your security based on the needs and requirements.

- Azure Resource Manager: With Azure Resource Manager, you can work with resources to deploy, update, and delete all resources in the application. See Figure 4-5.

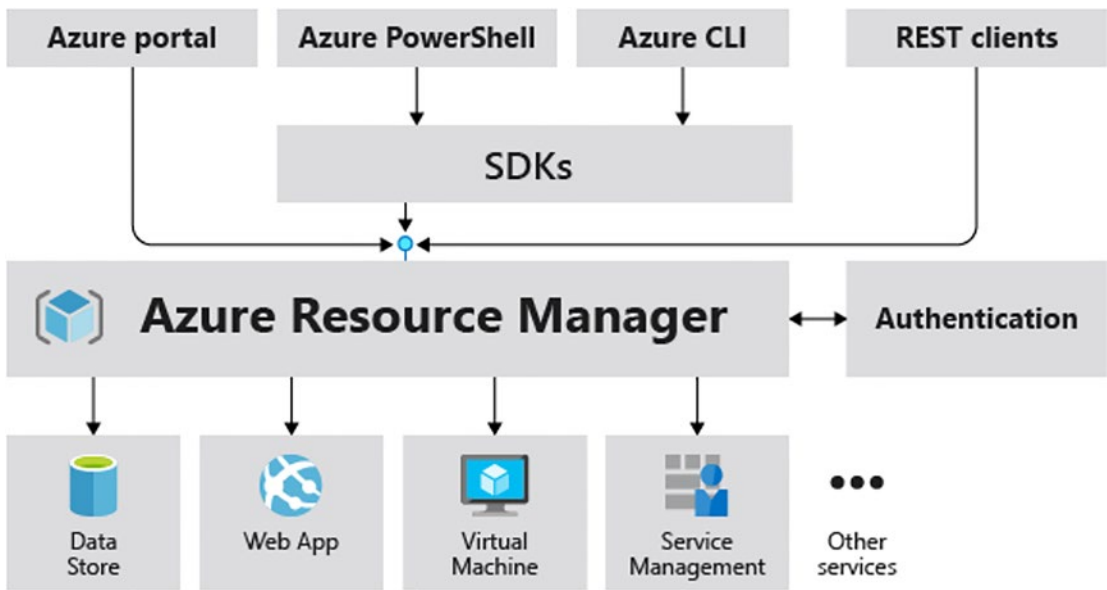


Figure 4-5. *Azure Resource Manager*

ARM Template deployment will work differently with different environments, such as development, testing, acceptance, and production. It provides security and auditing features so that you can manage resources after the deployment.

- **Application Insight:** With Application Insight, you can monitor applications when they are running as well as during testing and deployment. You can create charts and tables to find out more about how responsive the application is or to determine the overall performance of the application. See Figure 4-6.

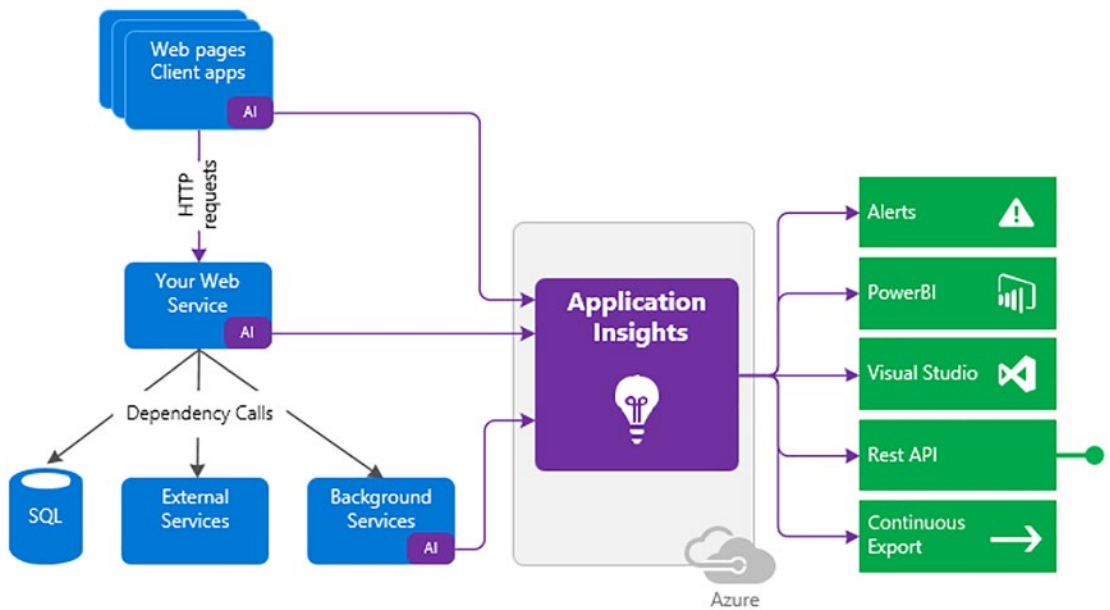


Figure 4-6. Application Insight

In case of any failures or performance issues, you can search through the logging data to determine the root cause.

- **Azure Monitor Logs:** These provide an IT management solution for on-premises as well as other cloud providers like AWS, Azure, and so on. They can be used to perform security as well as forensic logs. See Figure 4-7.

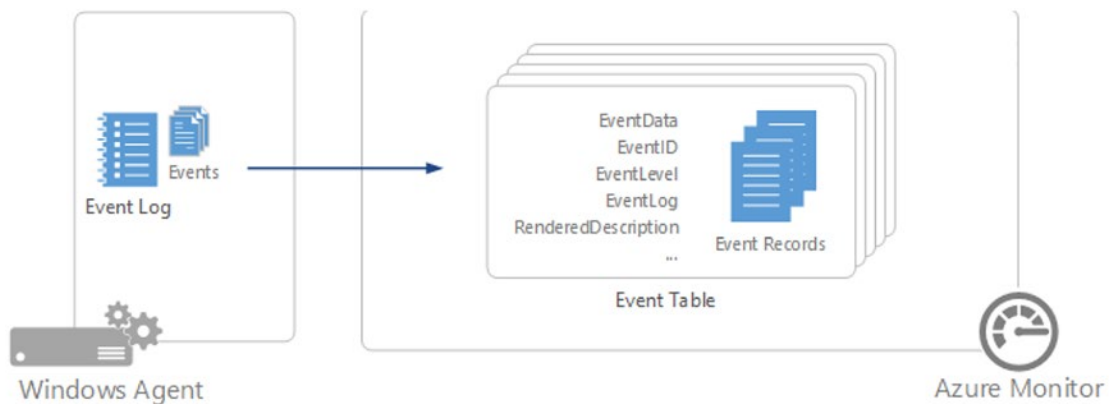


Figure 4-7. Azure Monitor Logs

- Azure Advisor: This is a personalized cloud service within Azure that’s used to determine and optimize the security, cost, and performance of your cloud workload. Azure advisor uses configurations and telemetry to improve overall security, performance, reliability, and cost of applications as well services. See Figure 4-8.

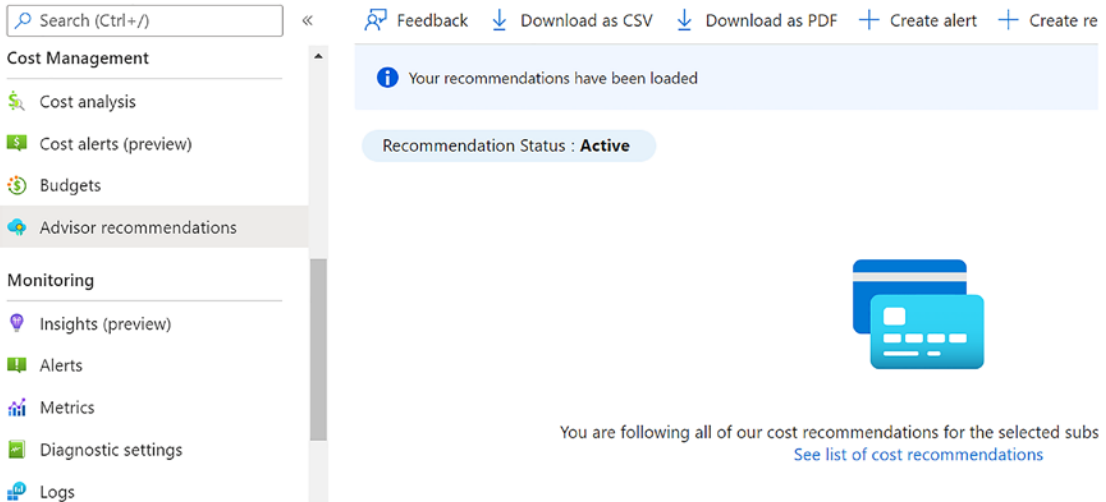


Figure 4-8. Azure Advisor recommendations

- As shown in Figure 4-9, you can look into the cost, security, reliability, and performance as well as operational excellence recommendations from Azure Advisor.

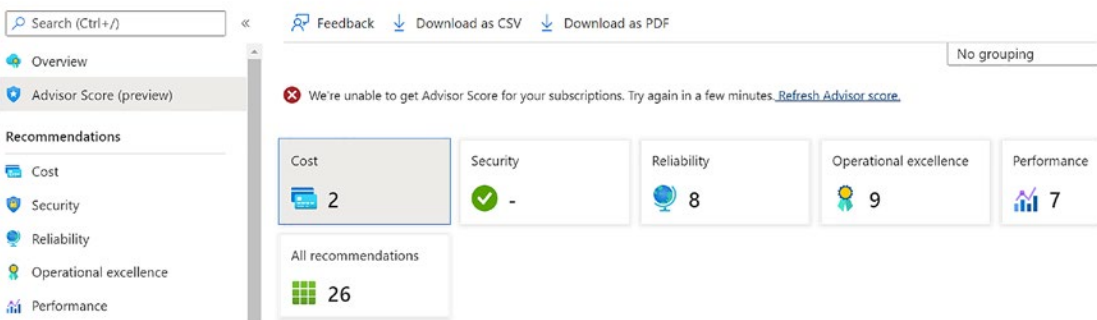


Figure 4-9. Azure Advisor Recommendations categories

From the left pane, you can select Security to learn more about the security recommendations in detail. With this feature, you can significantly improve the overall security of the application in Azure. See Figure 4-10.

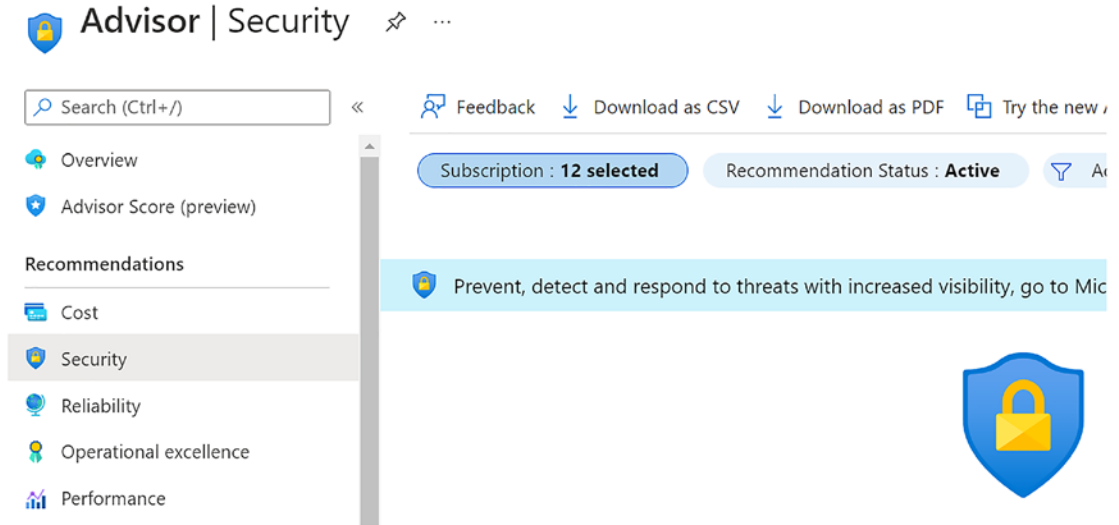


Figure 4-10. Azure Advisor security recommendations

- **Web application firewall (WAF):** With a web application firewall, you can protect your web applications from common attacks such as SQL injection, cross-site scripting (XSS), and session hijacking. It also helps enterprise organizations follow layered security approaches to make the application more secure and safe. Azure web application firewall on Azure application gateway provides protection from the web applications using the common security vulnerabilities. They target the attacks happening on the cloud resources and act based on the core rule set from the open web application security project. It operates as an application delivery control and provides a Transport Layer security layer, which was earlier known as the secure socket layer. This includes policy management and E2E TLS support. See Figure 4-11.

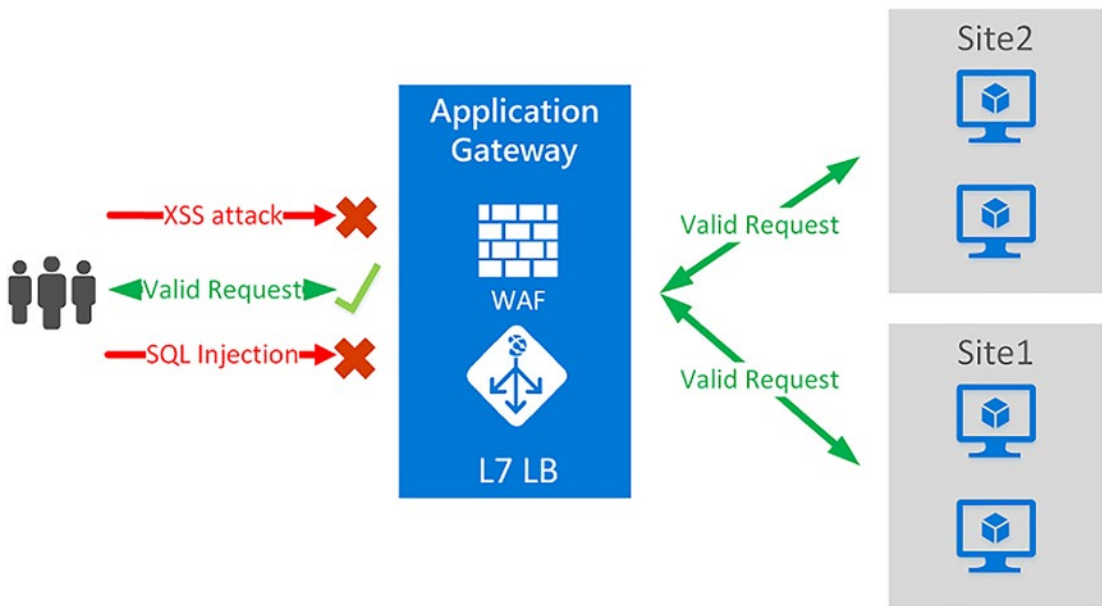


Figure 4-11. Azure Web Application Firewall

A layered security architecture incorporates the pros and cons of various security products. With layered security, you can put multiple security controls into the IT environment. A fundamental component of layered security is the perimeter of defense for traffic from the network. The Azure app service environment provides an isolated runtime environment that’s deployed in the Azure Virtual Network. The major goal is to hide the API backends from Internet access. With Azure Virtual Network, you can also use Network Security Groups (NSGs) to allow or deny inbound or outbound traffic to and from the website and web application. You can also use private endpoints to restrict public access to the same.

Layered security refers to the multiple components to protect operations at multiple levels or layers. This security approach deploys multiple controls to protect sensitive areas of the technology where attackers can get access to the system or application. With a layered approach, you need to define the firewall, patch management, multifactor authentication, endpoint protection, and security awareness training to improve the overall security of the application. See Figure 4-12.



Figure 4-12. Layered security architecture

Let's now look at the Azure Tenant Security concepts.

Azure Tenant Security

Azure Tenant Security (AzTS) is a more scalable and robust solution. The Azure Tenant Security solution is based on the Azure functions and it is a central scan model where you can perform scanning using the managed identity. See [Figure 4-13](#).

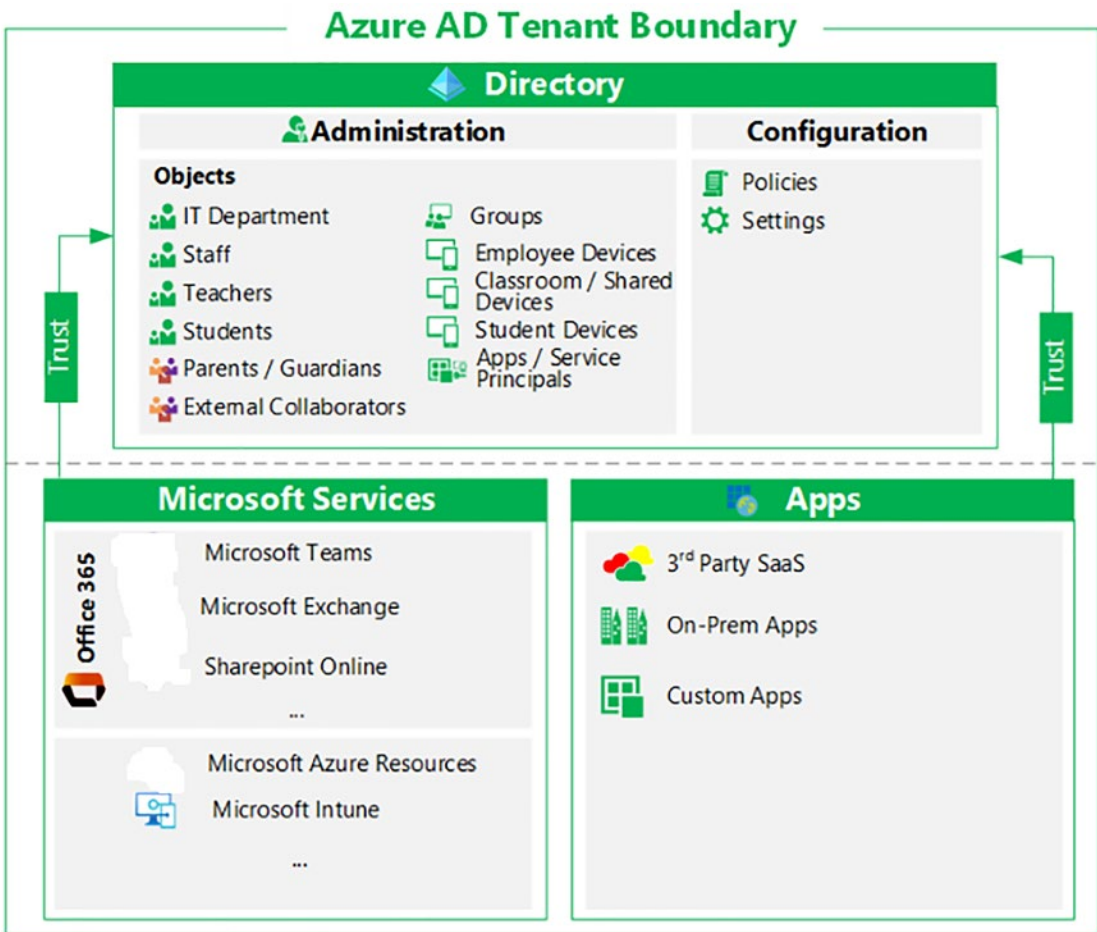


Figure 4-13. Layered security architecture

With the Azure Tenant Security solution, you can scale efficiently and use less overhead to get the same level of visibility. It is also designed to accelerate migration to Azure security offerings such as Policy, Security Center, Management Groups, and Azure Resource graphs. .

Now that you understand the basics of Azure Tenant Security, let’s look at the step-by-step process for using Azure Tenant Security.

1. Download the PowerShell script from GitHub at:

<https://github.com/azsk/AzTS-docs/blob/main/TemplateFiles/DeploymentFiles.zip?raw=1>

2. Open the folder in Windows Explorer and then open the ExecutionScript.ps file using the PowerShell IDE. See Figure 4-14.











 AzTSConsolidatedSetup	Windows PowerShell Script
 AzTSDeploymentTemplate	JSON Source File
 AzTSKeyVaultTemplate	JSON Source File
 AzTSSetup	Windows PowerShell Script
 ConfigureWebUI	Windows PowerShell Script
 ExecutionScript	Windows PowerShell Script
 KeyVaultMonitoringAlertTemplate	JSON Source File
 MonitoringAlertTemplate	JSON Source File
 OnDemandScan	Windows PowerShell Script
 TokenProvider	Windows PowerShell Script

Figure 4-14. Deployment script

3. Now, change the information for \$TenantId. You can get the information from the Azure Active Directory. See Figure 4-15.

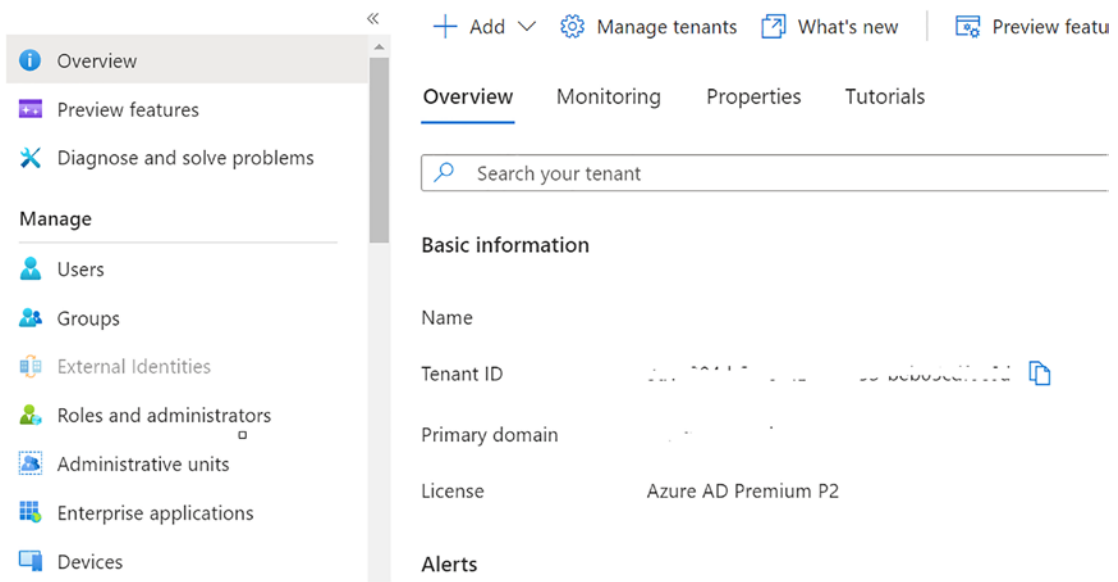


Figure 4-15. Azure Active Directory

4. Once the details are up to date, execute the script that will create the Azure services, as shown in Figure 4-16.

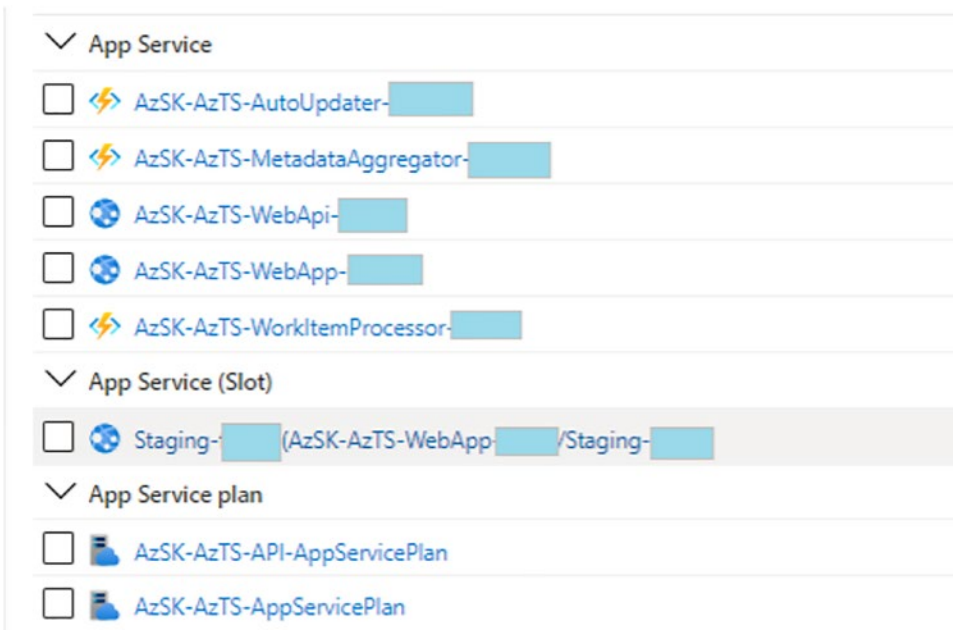


Figure 4-16. Azure services

5. Even though the services have been created, when you open the web app, the data is still blank. Be sure to execute this PowerShell command:

```
Start-AzSKTenantSecuritySolutionOnDemandSca -SubscriptionId
$HostSubscriptionId1 -ScanHostRGName $HostResourceGroupName1
```

6. Once that command is executed, it will open the web page shown in Figure 4-17.

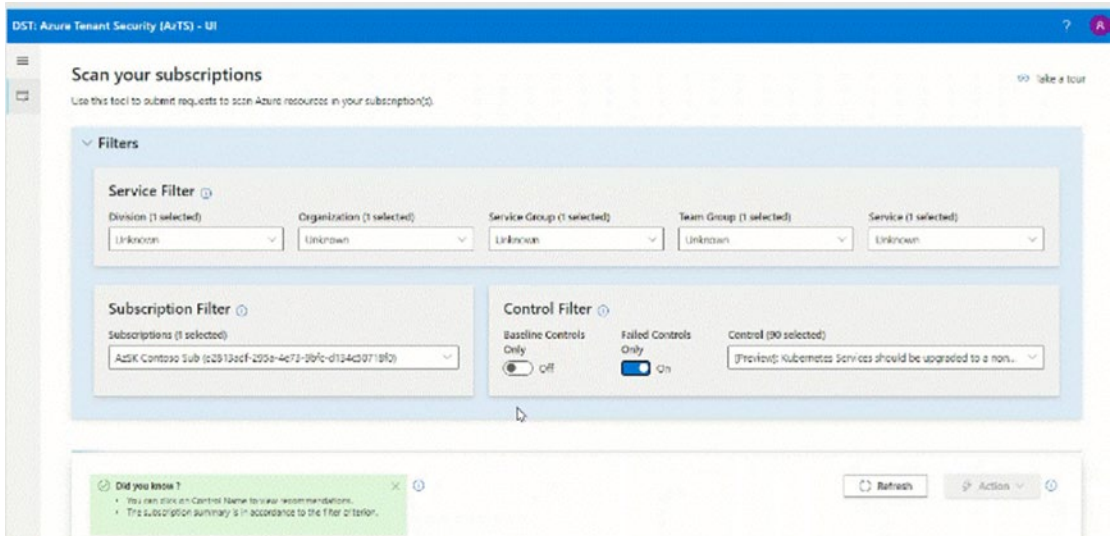


Figure 4-17. Azure Tenant Security

Azure Tenant Security can be used to obtain visibility of cloud subscriptions across multiple subscriptions in the organization. It is a DevOps kit that helps you move closer to the implementation of cloud security compliance on the Azure platform.

With Azure Tenant Security, you can perform the following activities:

- Scan multiple subscriptions with a central scan tool in a cheaper and more timely manner
- Auto-scale without any external constraints
- Speed up the effort of native features
- Enable incremental transition to the controls for custom code

Let's take a quick tour of container security.

Container Security

Considering the popularity and ease of setting up a Kubernetes cluster, many organizations use Kubernetes to orchestrate their containerized applications. Organizations need to consider and adopt Kubernetes security best practices for their containerized workloads. See Figure 4-18.



Figure 4-18. K8 security checklist

- Enable Kubernetes with role-based access control (RBAC): With RBAC, you can define permissions and determine who can access the Kubernetes API and related permissions at a granular level. Kubernetes combines the authorization controllers so it is better to disable the legacy attribute based access control. It is advisable to avoid granting cluster-level permission, even during troubleshooting. See Figure 4-19.

Role-Based Access Control

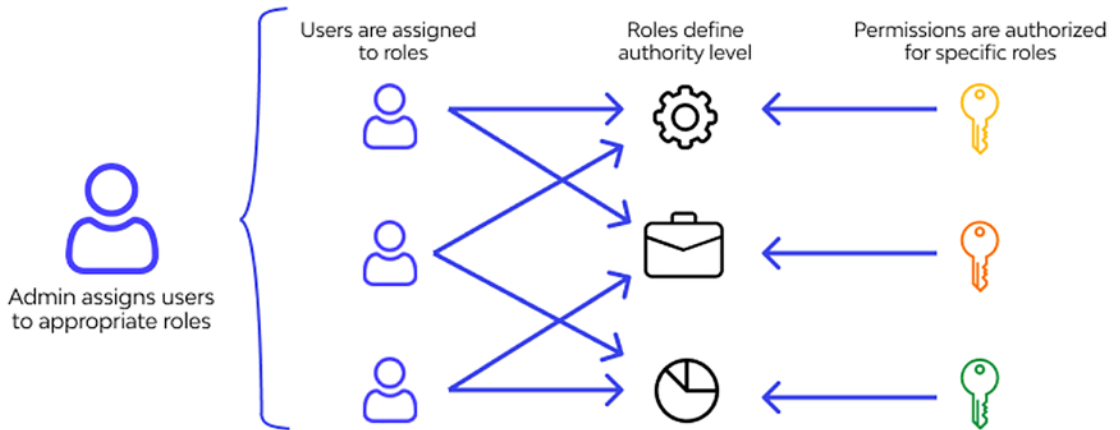


Figure 4-19. Role-based access control

- **Keep Kubernetes up to date:** As it is open source, Kubernetes has many contributors. Thus, it is important to stay up to date with the latest version, considering the security vulnerabilities and related updates.
- **Integrate Kubernetes with third-party authentication:** You can integrate Kubernetes with third-party tools, such as GitHub, for authentication. These tools provide additional security features like multifactor authentication, and so on.
- **Limit direct access to the Kubernetes nodes:** Limit SSH access to the Kubernetes nodes to avoid the risk of unauthorized access. You can also use Kubernetes authorized plugins to control user access to these resources.
- **Set up administrative boundaries:** You can create a Kubernetes namespace to partition the resources into logical groups. Resources created in one namespace won't be accessible to another namespace. You can even create policies to separate access between the namespaces.

For example:

```
{
  "apiVersion": "abac.authorization.kubernetes.io/v1beta1",
  "kind": "Policy",
  "spec": {
    "user": "sagar",
    "namespace": "cloud1",
    "resource": "pods",
    "readonly": true
  }
}
```

- **Enable network isolation:** Running multiple applications on a Kubernetes cluster runs the risk of one application interfering with another. Therefore, network isolation is important to ensure containers can only communicate with appropriate containers. See [Figure 4-20](#).

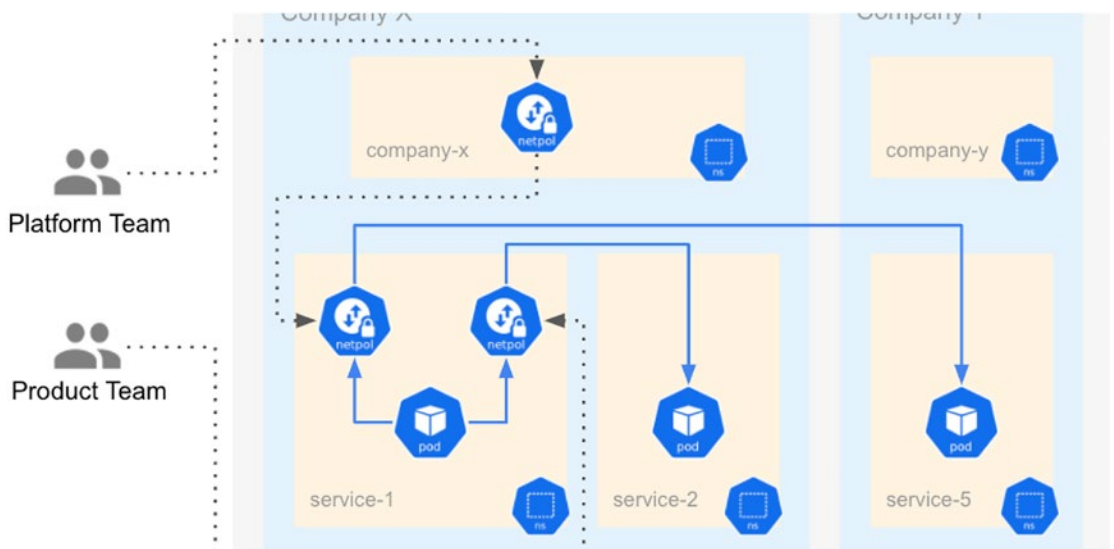


Figure 4-20. K8 network isolation

- Enable security context for pods and containers: While designing containers and pods, it is advised to configure the security context for the pods, containers, and volumes. You can define security context properties in the YAML format. See Figure 4-21.

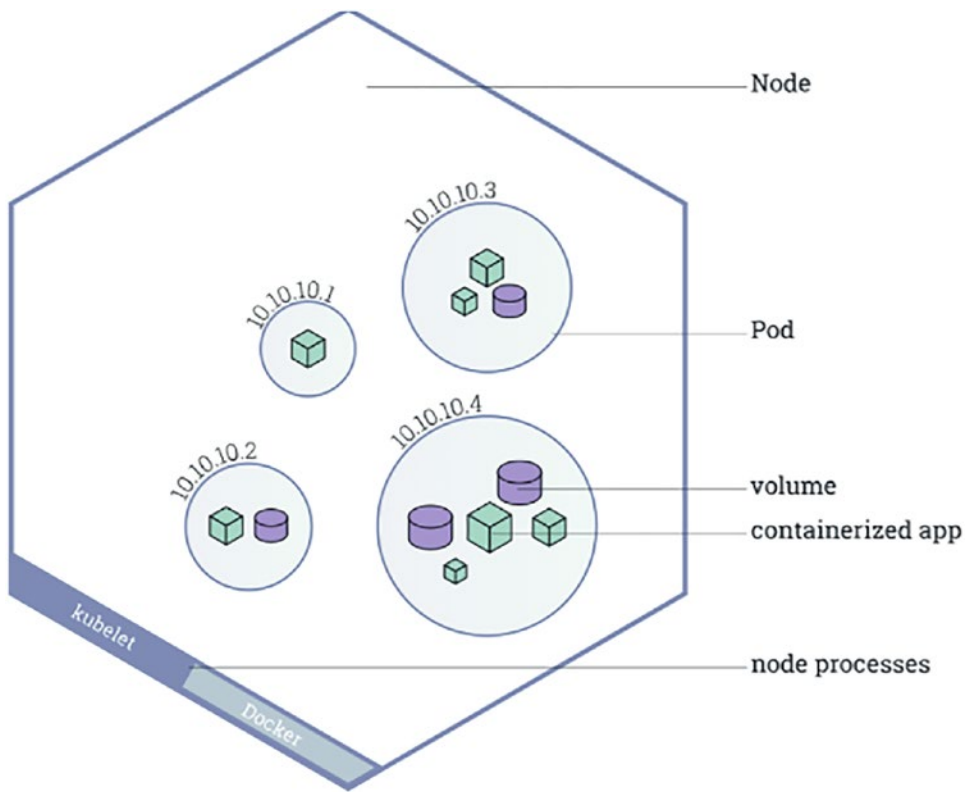


Figure 4-21. Node, pod, volume and node process

For example, pod definition with security context parameters:

```
apiVersion: v1
kind: Pod
metadata:
  name: first-pod-security-context
spec:
  containers:
    # specification of the pod's containers
  securityContext:
    readOnlyRootFilesystem: true
    runAsNonRoot: true
```

- Enable Kubernetes cluster logging: Enable logging for all cluster-related activities. You can also integrate the output of cluster logs with tools like Google Stackdriver logging or ElasticSearch.
- Secure secrets: Kubernetes cluster secrets contain sensitive information such as passwords, tokens, and SSH keys. Kubernetes supports encryption to ensure communication between the API servers is protected with TLS/SSL. It is also recommended to frequently rotate secrets to make it harder for attackers to gain unauthorized access to the cluster.
- Protect etcd with TLS, firewalls, and encryption: etcd stores the state of the cluster and confidential information in the form of secrets. It contains all the confidential information and is the highest-value target for attackers. If unauthorized users get access to etcd, the entire cluster is vulnerable to security attacks. See Figure 4-22.

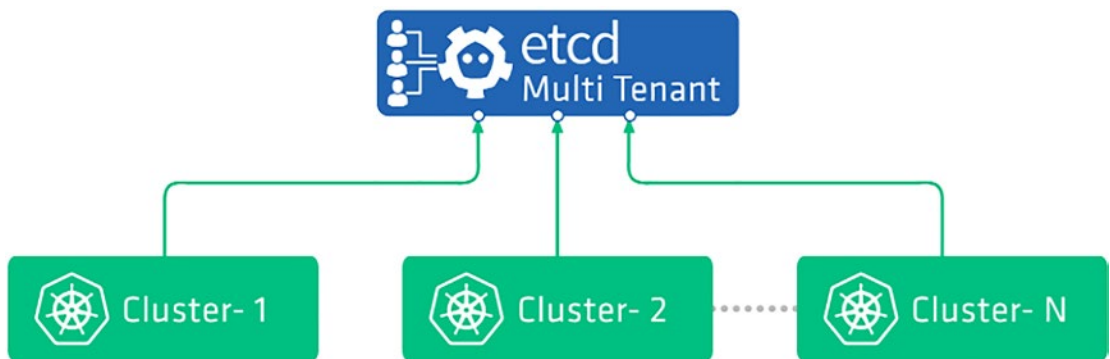


Figure 4-22. *etcd*

- Set up process whitelisting: Whitelisting is an effective way of identifying unexpected running processes. For this, you first need to understand the overall application behavior over a period of time. Then, use this pattern/list to whitelist the workload to the Kubernetes cluster. It is also difficult to do runtime analysis at the process level, but there are various solutions available in the market that minimize the overhead.

- **Lock Kubelet:** Kubelet is basically an agent that runs on each cluster node that interacts with the container runtime to launch the pods. Various configuration options are available to lock Kubelet to improve the overall security of the cluster.
- **Disable anonymous access with `--anonymous-auth=false`:** Unauthenticated requests won't be able to access the cluster and they will get an error response.
- **Set `--authorization mode`:** It is recommended to set the value for this variable to `AlwaysAllow` to verify that requests are authorized.
- **Set `--read-only-port=0`:** This configuration will enable read-only ports to prevent anonymous users from accessing information about running workloads.

Let's now look at how to secure Azure resources.

Securing Azure Resources

When you use a public cloud like Azure, deploying services and components is very easy and flexible. Before deploying the cloud application in the production environment, you have to make sure that the application follows the checklist for all the operational security requirements.

Consider this best practices checklist:

- **Azure Role-Based Access Control (RBAC):** Azure has various built-in roles to provide end users with the required permissions to the users, groups, or service principles. See [Figure 4-23](#).

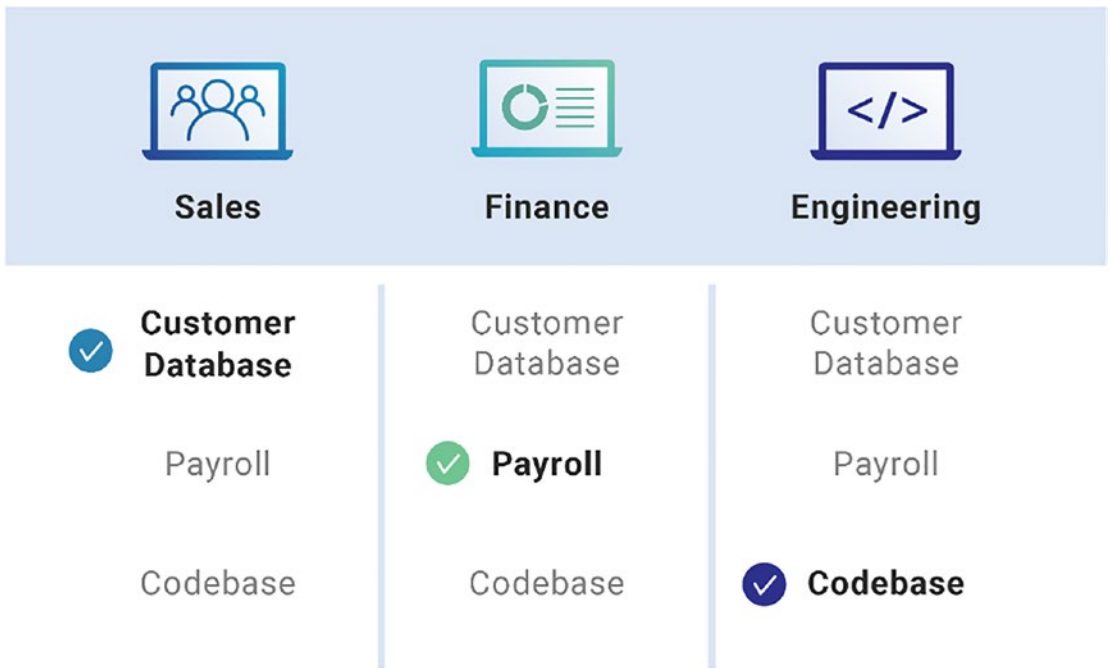


Figure 4-23. Role-based access control

- Data storage: In order to access and authenticate the data stored in the storage account, you can use the shared access signature (SAS). See Figure 4-24.

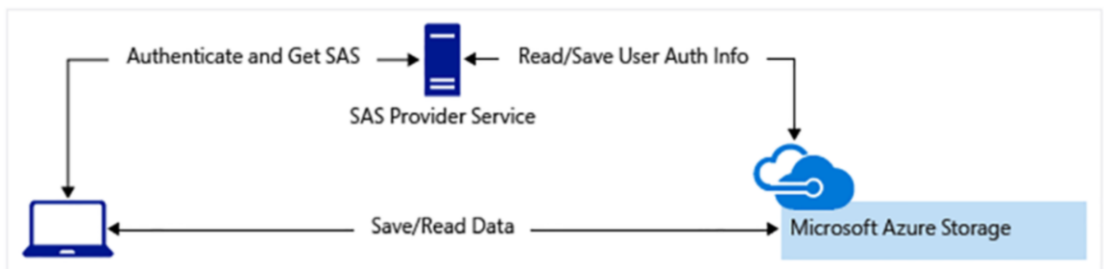


Figure 4-24. Data storage

- You can also use the TLS (Transport Layer Encryption) with HTTPS for Azure File shares. You can also use client-side encryption to secure the data sent to the storage accounts. This can be controlled with encryption keys. For Azure Virtual machines, you use Azure disk encryption to store the data. You can use the Storage Service Encryption to encrypt the data stored on the disk and data files.
- Using the Azure storage analytics, we can monitor the authorization type based on the storage account keys or shared access signature. See Figure 4-25.

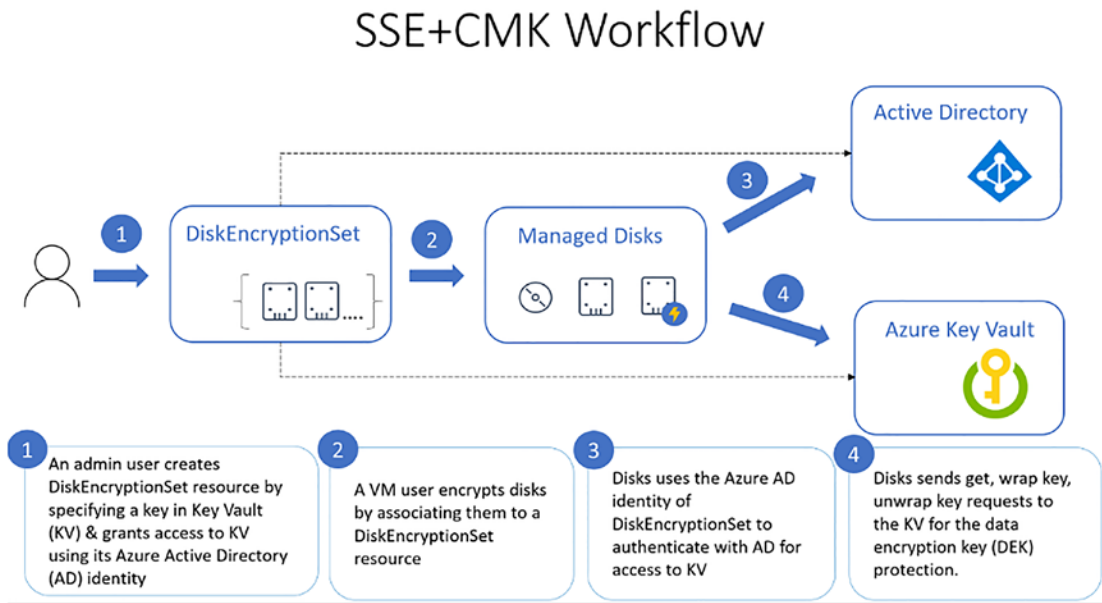


Figure 4-25. Disk Encryption with SSE

- Identity and Access Management: The first step is to sync your on-premises active directory with Azure Active Directory. You can also use single sign-on to enable access to the SaaS applications based on the accounts in Azure. You can also enable multifactor authentication for users to make authentication and authorization more secure. Using Azure AD premium, you can actively monitor suspicious activities. See Figure 4-26.

- Instead of using personal accounts, developers can use applications like Microsoft software development lifecycle.

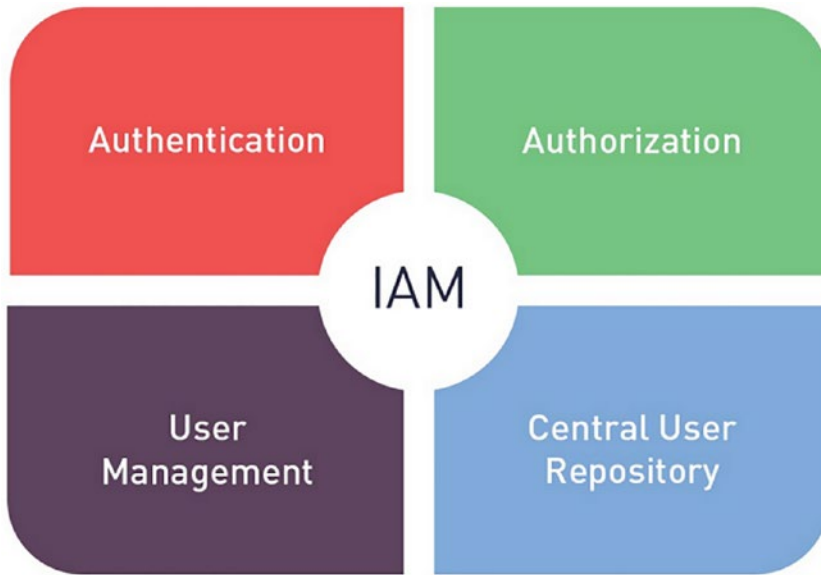


Figure 4-26. Identity and access management

Conclusion

This chapter explored how cloud providers use and manage public security of their infrastructure. In addition to this, you also took an in-depth tour of the built-in security controls available on the Microsoft Azure cloud. You also learned about Azure Tenant Security, with a step-by-step guide, and learned how to follow best practices to improve container security. Finally, you explored the security controls available in Azure's services and resources.