## 1. VM Security Best Practices

Virtual Machines are often prime targets for attackers, so Azure recommends minimizing their exposure to public networks. One of the key best practices is to **restrict access as much as possible**, especially avoiding public IP addresses for VMs unless absolutely necessary. Instead, administrators should use **Azure Bastion**, a secure and fully managed service that allows RDP and SSH access through the Azure Portal without exposing the VM to the internet.

**Limiting access** through **Network Security Groups (NSGs)** is another foundational step. NSGs allow administrators to define inbound and outbound security rules at the subnet or VM level, controlling which IPs or subnets can interact with the VM. These should be configured according to the principle of least privilege—only allowing traffic from known, necessary sources. Additionally, enabling just-in-time VM access can reduce exposure windows by opening ports only when needed and only for a limited time.

Other essential VM security measures include enabling **disk encryption** (using Azure Disk Encryption or server-side encryption), turning on **Azure Defender for Servers**, and using **update management** to ensure that the operating system and installed software are patched regularly. Finally, all VM access should be audited, and logging should be turned on via Azure Monitor and Log Analytics to track any unusual or unauthorized activity.

---

## 2. Networking Security Best Practices

Networking in Azure should be designed to prevent unintended exposure. One of the most important practices is to **not expose VNets to the public internet** unless absolutely required. By default, VNets are private, but services within them can gain internet access or become publicly accessible if misconfigured. Thus, all public access must be scrutinized or avoided altogether.

To tightly control traffic flow, **Network Security Groups (NSGs)** should be used at the subnet or NIC level to permit or deny traffic based on IP, port, and protocol. **Service Endpoints** and **Private Endpoints** further enhance network security by enabling access to Azure services over the Azure backbone network rather than through the internet. This is critical for services like Azure SQL, Storage, or Key Vault, which support these access methods to help isolate resources from external networks.

For more complex environments, a **hub-and-spoke topology** is often recommended. In this model, a central hub VNet acts as a control and monitoring point with shared services like firewalls or VPN gateways, while individual workloads are isolated in spoke VNets. This structure provides a scalable and secure design that allows for centralized security enforcement while maintaining network segmentation between teams or applications.

---

## 3. Database Security Best Practices

Databases often store the most sensitive information in an application stack, so their security is paramount. One of the first best practices is to **enable encryption both at rest and in transit**. Azure services like Azure SQL Database and Cosmos DB provide built-in support for Transparent Data Encryption (TDE) and TLS-based encryption for in-transit data, which should be enforced for all connections.

Access control is equally critical. It's best to **connect the database to a VNet** using **private endpoints** to ensure that all traffic flows over the internal Azure network. This prevents exposure to the public internet. Azure allows you to configure **firewall rules** to restrict external access to specific IPs or ranges, ensuring that only trusted sources can initiate connections.

To secure authentication and service-to-service interactions, **Managed Identities** are preferred over embedding credentials in code or config files. This integrates with Azure AD, allowing services like App Service or Function Apps to connect securely to the database without hardcoded secrets. Additionally, auditing and threat detection should be enabled to monitor access patterns, alert on suspicious activity, and support compliance requirements.

---

## 4. App Service Security Best Practices

Azure App Services host web applications and APIs and can be targeted by malicious users if not properly secured. A key recommendation is to **deploy an Application Gateway in front of the App Service**, especially one with a Web Application Firewall (WAF) enabled. This provides centralized SSL termination, layer 7 filtering, and protection from OWASP top 10 vulnerabilities.

App Services should be integrated with **private endpoints or VNet service endpoints** to isolate them from public exposure. When internal-only access is needed, this setup ensures traffic never leaves the Azure backbone. Additionally, developers should **leverage Azure AD authentication** with **Multi-Factor Authentication (MFA)** to protect user logins, especially for admin or back-office interfaces. Azure AD can be directly integrated into App Service for identity management.

For app-to-resource authentication (e.g., connecting to storage or databases), **Managed Identities** are the best practice. They allow your app to securely access other Azure services without storing credentials in the codebase or configuration files. Logging, diagnostics, and automatic scaling features of App Service should also be configured to monitor behavior and enforce reliability and security policies.

---

## 5. Use Key Vault

**Azure Key Vault** is a centralized service to **secure secrets, keys, and certificates** used by applications and services. It supports hardware security module (HSM)-backed keys for added protection and adheres to strict compliance standards. Key Vault is essential for securing secrets like database connection strings, API keys, tokens, and even entire TLS/SSL certificates.

Access to Key Vault is **highly restricted** and can be controlled using **Azure RBAC** or **Access Policies**. You can grant granular access (e.g., allow reading a secret but not modifying it) and combine this with

**Managed Identities** so services can programmatically retrieve secrets without the need for embedded credentials. All requests to the Key Vault are logged and can be monitored through Azure Monitor.

Applications and automation tools can access Key Vault through the **REST API**, making it easy to integrate into CI/CD pipelines, runtime environments, and scripts. Importantly, Key Vault is **cost-effective**, offering a pay-per-use model without requiring dedicated infrastructure. For most applications in Azure, using Key Vault is a foundational security measure that drastically reduces the risk of accidental leaks or misuse of sensitive information.

---

## 6. Defender for Cloud (formerly Security Center)

**Microsoft Defender for Cloud** is a unified security management platform that provides advanced threat protection across Azure, hybrid, and multi-cloud environments. It offers **security recommendations** tailored to your environment, such as enabling MFA, disabling unused ports, or enabling Just-in-Time access for VMs. These recommendations help organizations prioritize and implement best practices to improve their security posture.

The service also monitors resources in real time and generates **security alerts** when suspicious activities are detected. These alerts can originate from anomalous VM behaviors, unusual authentication attempts, or insecure configurations. Alerts are classified by severity and can be integrated with Azure Sentinel, SIEMs, or ticketing systems for response automation and incident tracking.

Lastly, Defender for Cloud maintains an **inventory of all monitored assets**, giving you visibility into the security configuration and compliance state of each one. You can also use it to assess regulatory compliance through built-in standards (e.g., ISO 27001, CIS benchmarks). Whether you’re a small business or a large enterprise, Defender for Cloud helps enforce governance, reduce attack surfaces, and react quickly to potential threats.