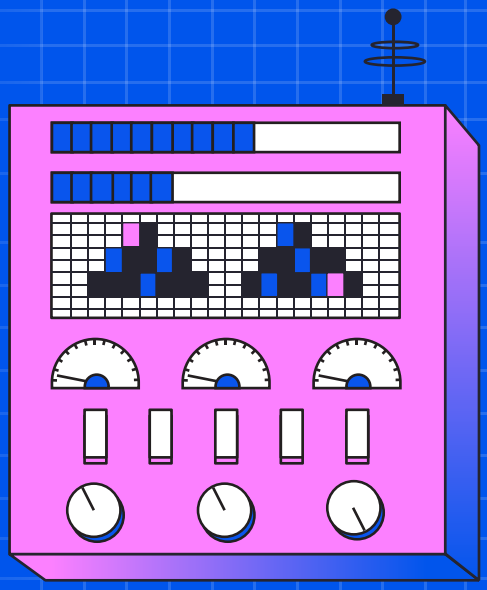
WIZ

Cheat Sheet

# Advanced Cloud Security Best Practices



In today's digital age, cloud infrastructures have become the backbone of countless businesses and services, so ensuring their security is more critical than ever. Yet, with myriad threats, configurations, and solutions out there, it's easy to feel overwhelmed.

Enter Wiz's cheat sheet: more than just a guide, it's a beacon in the often foggy landscape of cloud security. Crafted for both novices and seasoned professionals, this cheat sheet exceeds traditional advice. It combines theory with real-world impact analysis, provides actionable steps, and even delves into hands-on code snippets for those who prefer a direct approach. This toolkit is tailored to empower every cloud professional to navigate the complexities of cloud security with confidence and precision.

## Infrastructure security

### 1 Harden virtual machines

A VM that isn't hardened against threats can become the Achilles' heel of your entire infrastructure. Once compromised, it can lead to a cascading number of issues, including data breaches, unauthorized data manipulation, and lateral movement within the network that will allow attackers to explore and exploit other vulnerabilities.

#### Actionable items

- **Regularly patch and update VM software**

  Keeping your VM software up to date ensures that known vulnerabilities are addressed, reducing potential attack vectors.

- **Monitor VM access and activity logs**

  By monitoring who accesses the VM and what activities are carried out, you can quickly detect and respond to any unauthorized or suspicious actions.

- **Establish multi-factor authentication for VM access**

  By requiring multiple forms of verification before granting access to a VM, you're adding an additional layer of security that can deter unauthorized users and reducethe risk of breaches.

### Code suggestion

```
# Disable unnecessary services on a Linux VM to reduce potential
entry points for attackers:

$ sudo systemctl disable [service-name]
```

## 2 Secure storage buckets

A single misconfigured storage bucket on a cloud-based object storage service can cause disaster. Over the years, numerous high-profile data leaks have been attributed to improperly secured buckets. These leaks expose sensitive data and degrade an organization's reputation, leading to regulatory fines and loss of customer trust.

### Actionable items

- **Ensure default settings are set to private**

  Always start with a "deny all" stance and then explicitly allow access as needed. This ensures that you don't accidentally expose data.

- **Use bucket policies to fine-tune access controls**

  Rather than broad permissions, use granular policies to give specific access to users, roles, or services. This minimizes the risk of overexposure.

- **Regularly audit bucket permissions**

  Periodically review and validate bucket permissions to ensure they align with current operational needs and security best practices.

### Code suggestion

```
// Here is a sample S3 bucket policy to restrict access to a
specific IP address, ensuring that only trusted sources can access
the data:

{

    "Version": "2012-10-17",

    "Id": "IPAllowPolicy",

    "Statement": [{

        "Effect": "Allow",

        "Principal": "*",

        "Action": "s3:*",

        "Resource": "arn:aws:s3:::your-bucket-name/*",
```

```
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "your-ip-address"
                }
            }
        }]
    }
```

## 3 Isolate network resources

Without proper network segmentation, an attacker with access to one part of the network might easily traverse and gain access to other sensitive parts. By isolating network resources, you create barriers that can contain and limit the damage from such intrusions.

### Actionable items

- **Use VPCs and subnets to segment resources**

  Virtual private clouds (VPCs) and subnets allow you to create isolated areas within your cloud environment, ensuring that resources within one segment are shielded from potential threats in another.

- **Implement Network Access Control Lists (NACLs) and security groups**

  While both NACLs and security groups control inbound and outbound traffic, they can each protect you in different ways. NACLs operate at the subnet level, and security groups operate at the instance level. Using them in tandem provides layered security.

- **Regularly review and update network configurations**

  With growing cloud usage, it's crucial to regularly review and update network configurations. By acting proactively in this way, you're making sure your network segmentation strategies remain robust and effective against threats such as SQL injections or cross-site scripting attacks

### Code suggestion

```
# Create a new VPC in AWS to start the process of network
segmentation:

aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

# Application security

## 1 Secure coding practices

Insecure coding practices can open the door to cyber threats. Beyond the immediate threat to data and services, such vulnerabilities can deteriorate trust, damage brand reputation, and lead to regulatory penalties.

### Actionable items

- **Regularly review and update coding practices**

  As new threats emerge and the technology landscape evolves, coding practices must be revisited and refined to remain effective.

- **Train developers on secure coding principles**

  A well-informed development team is the first line of defense against vulnerabilities. Requiring regular certification updates, training workshops, and group sessions can empower developers with up-to-date knowledge and the best methodologies for secure coding.

- **Implement code review processes**

  Peer reviews and automated code scanning tools can help identify and rectify potential vulnerabilities before they make it to production.

### Code suggestion

```
# The following code snippet demonstrates a secure approach in
Python to prevent SQL injection attacks using parameterized
queries:

import sqlite3


def get_user_data(user_id):

    conn = sqlite3.connect('example.db')

    cursor = conn.cursor()

    cursor.execute("SELECT * FROM users WHERE id=?", (user_id,))

    data = cursor.fetchall()

    return data
```

## 2 Update third-party libraries

An outdated or compromised third-party library can be a ticking time bomb. Even if the application's proprietary code is a fortress of security, a single vulnerable library can provide an entry point for attackers.

The ripple effect of a compromised library can be vast, affecting all applications and systems that depend on it.

### Actionable items

- **Regularly check for updates and patches for third-party libraries**

  Developers should make it a routine to check for and apply updates to the libraries they use. This introduces new functionalities while also addressing possible security risks.

- **Use tools to scan for vulnerable libraries automatically**

  There are several tools available that can scan your project's dependencies and alert you to known vulnerabilities. Integrating such tools into your development process can provide an added layer of security.

### Code suggestion

```
# This command demonstrates how to use npm (Node.js) to identify
and get insights on outdated packages, which can be potential
security risks:

npm outdated
```

## 3 Monitor application logs

The real-time nature of log monitoring offers a proactive approach to security. Instead of reacting to a breach after it has occurred, monitoring allows for immediate detection and response.

For instance, a sudden surge in failed login attempts might be a sign that you're in the middle of a brute-force attack. By catching and addressing such anomalies early, organizations can thwart potential threats before they escalate, minimizing damage and reducing recovery efforts.

### Actionable items

- **Set up real-time log monitoring**

  Use tools and platforms that continuously monitor application logs, ensuring you're always aware of what's happening within your application.

- **Create alerts for suspicious activities**

  Rather than sifting through logs manually, set up alerts for specific events or patterns that might indicate malicious activity. This will save time in situations where threats require immediate attention.

- **Regularly review and analyze logs**

  Automated alerts are invaluable, but periodic manual reviews can offer deeper insights, helping to identify subtle patterns or long-term trends that automated systems might overlook.

### Code suggestion

```
# Setting up Filebeat to forward logs to Logstash (or directly to
Elasticsearch):

# Install Filebeat

sudo apt-get install filebeat


# Configure Filebeat:

sudo nano /etc/filebeat/filebeat.yml


# Sample configuration to forward system logs:

filebeat.inputs:

- type: log

  enabled: true

  paths:

    - /var/log/*.log


# Specify Logstash (or Elasticsearch) as the output:

output.logstash:

  hosts: ["your-logstash-host:5044"]


# Start Filebeat:

sudo service filebeat start
```

# Identity and access management (IAM)

## 1 Rotate access keys

A compromised access key can be a silent disaster. Unlike a brute-force attack or a system glitch, unauthorized actions using a valid access key might not raise immediate alarms. Significant damage could have been done by the time the breach was detected.

Regularly rotating these keys ensures that even if an old key is leaked, it becomes obsolete after a short period, thereby limiting potential misuse.

### Actionable items

- **Set automated rotations for access keys**

  Implement a routine where access keys are automatically rotated at regular intervals. This ensures that keys are always fresh and that compromised keys have limited lifespans.

- **Use IAM roles for applications instead of long-term access keys**

  Assign IAM roles to cloud-based compute instances or other services instead of hardcoding access keys in applications. This way, they get temporary credentials to access other cloud services, enhancing security.

- **Monitor access key usage**

  Regularly review Amazon CloudTrail or equivalent log services on other clouds to monitor how and when access keys are used. Unexpected patterns can indicate misuse.

### Code suggestion

```
# Use the AWS CLI to rotate access keys: First, create a new access
key, then deactivate or delete the old one, as shown here:

aws iam create-access-key --user-name UserName

aws iam delete-access-key --access-key-id OldAccessKeyId --user-
name UserName
```

## 2 Use service-linked roles

By leveraging service-linked roles, organizations can balance functionality and security. These roles ensure that cloud services can seamlessly interact with resources without the risk of over-permission. In the event of a potential service vulnerability, the blast radius is contained, as the service can only perform actions within the confines of its service-linked role.

### Actionable items

- **Review and limit permissions for service-linked roles**

  Even though these roles come with predefined permissions, reviewing them periodically to ensure they align with the principle of least privilege is essential.

- **Regularly audit roles to ensure they align with current needs**

  Some service-linked roles might become redundant or require adjustments as your cloud environment and usage patterns evolve. Regular audits can help identify and rectify such discrepancies.

- **Implement strict access controls for VMs**

  Limiting who can access your virtual machines and setting specific permissions ensures that only authorized personnel can make changes, limiting threats.

### Code suggestion

```
# Using AWS CLI to create a service-linked role tailored for a
specific AWS service:

aws iam create-service-linked-role --aws-service-name
servicename.amazonaws.com
```

## 3 Monitor IAM access patterns

Unusual IAM access patterns are primarily indicators of misuse or misconfiguration. For instance, an unexpected spike in login attempts could indicate a brute force attack, while access requests from unfamiliar IP addresses might suggest unauthorized access attempts.

By detecting and addressing these anomalies early, organizations can nip potential security threats in the bud, safeguarding their data and resources.

### Actionable items

- **Set up alerts for unusual IAM access patterns**

  Implement real-time monitoring tools to detect and notify you of unusual IAM activities to ensure fast action before any damage is done.

- **Regularly review IAM access logs**

  Periodic manual reviews of IAM access logs can offer deeper insights, helpin to identify subtle patterns or trends that automated systems might overlook.

- **Implement multi-factor authentication (MFA)**

  Enhance security by requiring a second form of authentication for IAM users, so that even if credentials are compromised, unauthorized access is disabled.

### Code suggestion

```
# Use AWS CloudWatch to set up an alarm for any activity from the
root account:

aws cloudwatch put-metric-alarm --alarm-name RootAccountUsage --
metric-name RootAccountUsage --namespace AWS/Usage --statistic
Maximum --period 300 --threshold 1 --comparison-operator
GreaterThanThreshold --evaluation-periods 1 --alarm-actions
[ARN_of_SNS_Topic]
```

# Incident response

## 1 Predefined incident response plan

Without a predefined plan, organizations risk making ad-hoc decisions that could worsen the situation. On the other hand, a well-structured and rehearsed incident response plan can provide clarity and direction.

Following such plans helps make sure incidents are handled methodically, evidence is preserved, stakeholders are informed, and recovery is swift. That will minimize operational disruptions, financial implications, and help maintain trust and reputation in the eyes of customers and partners.

### Actionable items

- **Document the incident response process**

  This should be a comprehensive document detailing every step from detection to recovery. It should also include post-incident analysis to help improve responses after incidents take place.

- **Assign roles and responsibilities for different stages of the response**

  Clearly define who does what. For instance, while technical teams might handle operations and recovery, communication teams might handle external communication and PR.

- **Conduct regular drills**

  Just having a plan isn't enough. Regularly simulate incidents to test the plan's effectiveness and the team's preparedness.

### Code suggestion

```
# The following is a sample AWS CLI command to isolate a
potentially compromised EC2 instance by disabling its source/
destination check, preventing it from sending or receiving
unintended traffic:

aws ec2 modify-instance-attribute --instance-id instance_id --no-
source-dest-check
```

## 2 Conduct security drills

The actual value of security drills lies in their ability to transform theoretical plans into practical action. By regularly testing the incident response plan, teams can ensure that they're not just on paper but in practice.

Drills can uncover logistical challenges, communication breakdowns, or technical shortcomings that might be overlooked in planning sessions.

## Actionable items

- **Schedule regular security drills**

  Like software needs regular updates, so does your incident response strategy. Regular drills ensure the plan evolves with the changing threat landscape and organizational changes.

- **Involve all relevant stakeholders**

  While the IT and security teams play a central role, it's essential to involve other departments, such as PR, legal, and customer support to ensure a holistic response.

- **Document the outcomes**

  After each drill, conduct a debriefing session. Document what went well, what didn't, and the lessons learned.

## Code suggestion

```
# The following demonstrates a command to simulate a DDoS attack
using hping3. This can help test systems' resilience and mitigation
strategies' effectiveness:

hping3 -c 10000 -d 120 -S -w 64 -p 80 --flood --rand-source
target_ip
```

# 3 Automated incident detection

The speed and efficiency of automated detection tools can be game-changing. By identifying and flagging incidents in real time, these tools drastically reduce the time between the onset of an incident and its detection.

This rapid response ability can mean the difference between a slight setback and a major security catastrophe. By catching threats early, organizations can act swiftly to contain and mitigate them, minimizing damage, downtime, and potential financial losses.

## Actionable items

- **Integrate automated detection tools into your infrastructure**

  Whether it's intrusion detection systems (IDS), security information and event management (SIEM) solutions, or AI-driven threat detection platforms, ensure your infrastructure is equipped with the latest automated detection technology.

- **Set up alerts for detected incidents**

  While automated tools can detect incidents, human judgment is crucial in determining the appropriate response. Ensure that any detected incidents trigger alerts, notifying the relevant teams or personnel to take action.

- **Regularly update and fine-tune detection algorithms**

  The threat landscape is ever-evolving. Periodically update the detection algorithms for new threat patterns and false positives.

### Code suggestion

```
# Use the security monitoring service AWS GuardDuty or similar
service on other clouds, which analyze and process VPC traffic,
event logs, and DNS logs. This command enables GuardDuty for threat
detection:

aws guardduty create-detector --enable
```

# Data protection and privacy

## 1 Automated compliance checks

Automated compliance checks act as a proactive shield, identifying potential deviations from required standards before they escalate into violations.

By ensuring continuous compliance checks, you can operate with more confidence, knowing that your organization  is aligned with industry standards and free from the looming threat of non-compliance penalties. If a check finds an issue, you can quickly respond and right the ship.

### Actionable items

- **Integrate compliance check tools into your infrastructure**

  Utilize tools that continuously monitor and evaluate your infrastructure against established benchmarks, ensuring that any deviations are promptly identified.

- **Schedule regular compliance reports**

  While real-time monitoring is essential, periodic comprehensive reports provide a holistic view of the organization's compliance posture.

- **Stay on top of regulations**

  Compliance requirements aren't static. They evolve in response to industry developments, technological advancements, and societal changes. Ensure that your automated checks are updated to reflect the latest requirements.

### Code suggestion

```
# Use of AWS Config, a service that allows you to review, audit,
and analyze the setups of your AWS resources. This command starts
recording the configuration changes, aiding in compliance checks:

aws configservice start-configuration-recorder --configuration-
recorder-name default
```

## 2 Threat intelligence integration

In the world of cybersecurity, being reactive often means being too late. Adopting real-time threat intelligence transitions the approach from reactive to proactive security. By staying updated with the newest threat patterns, vulnerabilities, and attack vectors, you can preemptively strengthen your organization's defenses, patch vulnerabilities, and prevent attacks before they materialize.

### Actionable items

- **Subscribe to threat intelligence feeds**

  Partner with reputable threat intelligence providers or platforms to receive real-time updates on emerging threats. This could range from open-source platforms to specialized commercial services.

- **Integrate feeds into security tools and processes**

  Ensure that the threat intelligence is not just consumed but also acted upon. Incorporate your security information into your security information and event management (SIEM) systems, intrusion detection systems (IDS), and additional security instruments that make threat identification and reaction fully automated.

- **Regularly review and validate intelligence**

  While threat-intelligence feeds provide valuable insights, it's essential to validate and contextualize this information to your specific environment. Not all threats are relevant to all organizations, so prioritize based on your infrastructure, industry, and threat landscape.

### Code suggestion

```
# The following command shows how to fetch threat intelligence data
from a platform like MISP (Malware Information Sharing Platform &
Threat Sharing). This command retrieves the latest threat data in
JSON format:

curl -H "Authorization: api_key" -o threats.json https://
misp_instance/events/restSearch/json
```

# Conclusion

Cloud security isn't something you can sleep on. By regularly revisiting and refining your practices according to these guidelines, you're not just ensuring a resilient cloud environment, but also staying ahead in the cybersecurity game.

Wiz can help you do that, providing a comprehensive analysis engine that can automatically review and update your:

- Cloud security posture management (CSPM)
- Kubernetes security posture management (KSPM)
- Cloud workload protection (CWPP) + vulnerability management

- Infrastructure-as-code (IaC) scanning
- Cloud Infrastructure entitlement management (CIEM)
- Data security posture management (DSPM)