**10**

**Operational Excellence Considerations**

Application maintainability is one of the main aspects that a solution architect should consider during architectural design. Every new project starts with lots of planning and resources at the beginning. Teams spend the initial months creating and launching your application. After the production launch, the application needs several things to be taken care of to keep operating. You need to continually monitor your application to find and resolve any issues on a day-to-day basis.

The operations team needs to handle application infrastructure, security, and any software issues to make sure your application is running reliably without any problems or issues. Often, an enterprise application is complex in nature, with a defined **Service-Level Agreement** (**SLA**) regarding application availability. Your operations team needs to understand business requirements and prepare themselves accordingly to respond to any event.

Operation excellence should be implemented across various components and layers of architecture. In modern microservice applications, there are so many moving parts involved that make system operations and maintenance a complicated task. Your operations team needs to put proper monitoring and alert mechanisms in place to tackle any issues that can hamper the business flow. Operational issues involve coordination from several teams for *preparation* and *resolution*. Operation expenditures are one of the significant costs that an organization puts aside to run a business.

In this chapter, you will learn various design principles applicable to achieving operational excellence for your solution.

The operational aspect needs to consider every component of the architecture. You will get an understanding of the right selection of technologies to ensure operational maintainability at every layer of software application. You will learn the following best practices of operational excellence in this chapter:

* Designing principles for operational excellence
* Selecting technologies for operational excellence
* Achieving operational excellence in the public cloud

By the end of this chapter, you will know various processes and methods to achieve operational excellence. You will learn about best practices that you can apply throughout application design, implementation, and post-production to improve application operability.

**Designing principles for operational excellence**

Operational excellence is about running your application with the minimal possible interruption to gain maximum business value. It is about applying continuous improvements to make the system efficient.

The following sections talk about the standard design principles that can help you strengthen your system's maintainability. You will find that all operational excellence design principles are closely related to and complement each other.

**Automating manual tasks**

Technology has been moving fast in recent times, and so IT operations need to keep up with that, where hardware and software inventories are procured from multiple vendors. Enterprises are building hybrid cloud and multi-cloud systems, so you need to handle both on-premises and cloud operations. All modern systems have a decidedly more extensive user base, with various microservices working together and millions of devices connected in a network. There are many moving parts in an IT operation, so this makes it difficult to run things manually.

Organizations maintain agility, and operations have to be fast to make use of the required infrastructure for new service development and deployment. The operations team has a more significant responsibility to keep services up and running and recover quickly in case of an event. Now, it is required to take a proactive approach in IT operations, rather than waiting for an incident to happen and then reacting.

Your operations team can work very efficiently by applying automation. Manual jobs need to be automated so that the team can focus on more strategic initiatives rather than getting overworked with tactical work. Spinning up a new server or starting and stopping services should be automated by taking an **Infrastructure as Code** (**IaC**) approach. Automating active discovery and response for any security threat is most important, to free up the operations team. Automation allows the team to devote more time to innovation.

For your web-facing application, you can detect anomalies in advance before they impact your system, using machine learning prediction. You can raise an automated security ticket if someone exposes your server to the world with HTTP port 80. You can pretty much automate the entire infrastructure and redeploy it multiple times as a *one-click solution*. Automation also helps to prevent human error, which can occur even if a person is doing the same job repetitively. Automation is now a must-have for IT operations.

**Making incremental and reversible changes**

Operational optimization is an ongoing process, whereby continuous effort is required to identify the gap and improve upon it. Achieving operational excellence is a journey. There are always changes required in all parts of your workload to maintain it—for example, often, operating systems of your server need to be updated with a security patch provided by your vendor. Various software that your application is using need a version upgrade. You need to make changes in the system to adhere to new compliance requirements.

You should design your workload in such a way that it allows all system components to get updated regularly, so the system will benefit from the latest and most significant updates available. Automate your flow so that you can apply small changes to avoid any significant impact. Any changes should be reversible, to restore system working conditions in the case of any issues. Incremental changes help to do thorough testing and improve overall system reliability. Automate any change management to avoid human error and achieve efficiency.

**Predicting failures and responding**

Preventing failures is vital to achieving operational excellence. Failures are bound to happen, and it's critical to identify them as far in advance as possible. During architecture design, anticipate failure to make sure you design for failure so that nothing will fail. Assume that everything will fail all the time and have a backup plan ready. Perform regular exercises to identify any potential source of failure. Try to remove or mitigate any resource that could cause a failure during system operation.

Create a test scenario based on your SLA that potentially includes a system **Recovery Time Objective** (**RTO**) and **Recovery Point Objective** (**RPO**). Test your scenario, and make sure you understand their impact. Make your team ready to respond to any incident by simulating in a production-like scenario. Test your response procedure to make sure it is resolving issues effectively and create a confident team that is familiar with response execution.

**Learning from mistakes and refining**

As operational failures occur in your system, you should learn from the mistake and identify the gap. Make sure those same events do not occur again, and you should have a solution ready in case a failure gets repeated. One way to improve is by running **root cause analysis**, also called **RCA**.

During RCA, you need to gather the team and ask five *whys*. With each *why*, you peel off one layer of the problem, and, after asking the last *why*, you get to the bottom of the issue. After identifying the actual cause, you can prepare a solution by removing or mitigating the resources and update the operational runbook with the ready-to-use solution.

As your workload evolves with time, you need to make sure the operation procedure gets updated accordingly. Make sure to validate and test all methods regularly, and that the team is familiar with the latest updates in order to execute them.

**Keeping the operational runbook updated**

Often, a team overlooks documentation, which results in an outdated runbook. A runbook provides a guide to executing a set of actions in order to resolve issues arising due to external or internal events. A lack of documentation can make your operation people-dependent, which can be risky due to team attrition. Always establish processes to keep your system operations people-independent, and document all the aspects.

In the runbook, you want to keep track of all previous events and actions taken by team members to resolve them, so that any new team members can provide a quick resolution of similar incidents during operation support. Automate your runbook through the script so that it can get updated automatically as new changes roll out to the system.

Your runbook should include the defined SLA in relation to RTO/RPO, latency, scalability performance, and so on. The system admin should maintain a runbook with steps to start, stop, patch, and update the system. The operations team should include the system testing and validation result, along with the procedure to respond to the event.

Automate processes to annotate documents as a team applies changes to the system, and also after every build. You can use annotation to automate your operation, and it is easily readable by code. Business priorities and customer needs continue to change, and it's essential to design operations to support evolution over time.

**Selecting technologies for operational excellence**

The operations team needs to create procedures and steps to handle any operational incidents and validate the effectiveness of their actions. They need to understand the business need to provide efficient support. The operations team needs to collect systems and business metrics to measure the achievement of business outcomes.

The operational procedure can be categorized into three phases—planning, functioning, and improving. Let's explore technologies that can help in each phase.

**Planning for operational excellence**

The first step in the operational excellence process is to define operational priorities to focus on the high business impact areas. Those areas could be applying automation, streamlining monitoring, developing team skills as the workload evolves, and focusing on improving overall workload performance. There are tools and services available that crawl through your system by scanning logs and system activity. These tools provide a core set of checks that recommend optimizations for the system environment and help to shape priorities.

After understanding the priorities, you need to design the operation, which includes the workloads to design and building the procedures to support them. The design of a workload should consist of how it will be implemented, deployed, updated, and operated. An entire workload can be viewed as various application components, infrastructure components, security, data governance, and operations automation.

While designing for an operation, consider the following best practices:

* Automate your runbook with scripting to reduce human error, which creates an operating workload.
* Use resource identification mechanisms to execute operations based on defined criteria such as environment, various versions, application owner, and roles.
* Make incident responses automated so that, in the case of an event, the system should start self-healing without much human intervention.
* Use various tools and capabilities to automate the management of server instances and overall systems.
* Create script procedures on your instances to automate the installation of required software and security patches when the server gets started. These scripts are also known as bootstrap scripts.

After the operation design, create a checklist for operational readiness. These checklists should be comprehensive to make sure the system is ready for operation support when going live in production. This includes logging and monitoring, a communication plan, an alert mechanism, a team skillset, a team support charter, a vendor support mechanism, and so on. For operational excellence planning, the following are the areas where you need appropriate tools for preparation:

* IT asset management
* Configuration management

Let's explore each area in more detail, to understand the available tools and processes.

**IT asset management**

Operational excellence planning requires a list of IT inventories and tracks their use. These inventories include infrastructure hardware such as physical servers, network devices, storage, end-user devices, and so on. You also need to keep track of software licenses, operational data, legal contracts, compliance, and so on. IT assets include any system, hardware, or information that a company is using to perform a business activity.

Keeping track of IT assets helps an organization to make strategic and tactical decisions regarding operational support and planning. However, managing IT assets in a large organization could be a very daunting task. Various **IT Asset Management** (**ITAM**) tools are available for the operations team to help in the asset management process. Some of the most popular ITAM tools are **SolarWinds**, **Freshservice**, **ServiceDesk Plus**, **Asset Panda**, **PagerDuty**, **Jira Service Desk**, and so on.

IT management is more than tracking IT assets. It also involves monitoring and collecting asset data continuously, to optimize usage and operation costs. ITAM makes the organization more agile by providing end-to-end visibility and the ability to apply patches and upgrades quickly. The following diagram illustrates **ITAM**:

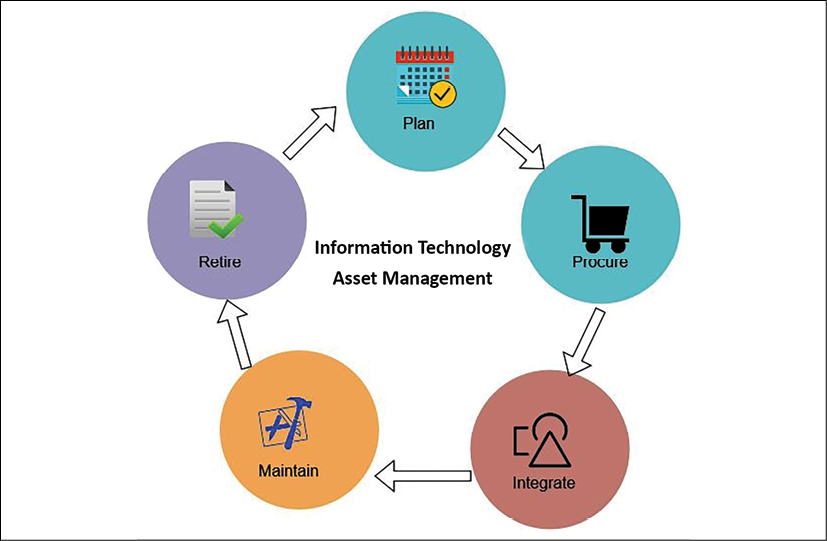


Figure 10.1: ITAM process

As shown in the preceding diagram, the ITAM process includes the following phases:

* **Plan**: An asset life cycle starts with planning, which is a more strategic focus to determine the need for overall IT assets and procurement methods. It includes cost-benefit analysis and total cost of ownership.
* **Procure**: In the procurement phase, organizations acquire the asset based on the outcome of planning. They may also decide to develop some holdings as required—for example, in-house software for logging and monitoring.
* **Integrate**: In this phase, an asset is installed in the IT ecosystem. It includes operation and support of the asset, and defines user access—for example, installing a log agent to collect logs from all the servers in a centralized dashboard, and restricting monitoring dashboard metrics to the IT operations team.
* **Maintain**: In the maintenance phase, the IT operations team keeps track of assets, and acts to upgrade or migrate based on the asset life cycle—for example, applying a security patch provided by the software vendor. The other example is keeping track of the end of life for licensed software, such as a plan to migrate from Windows Server 2008 to Windows 2022, as the old operating system is getting to the end of its life.
* **Retire**: In the retirement phase, the operations team disposes of the end-of-life asset. For example, if an old database server is getting to the end of its life, then the team takes action to upgrade it and migrates the required users and support to the new server.

ITAM helps organizations adhere to **ISO 19770** compliance requirements. It includes software procurement, deployment, upgrade, and support. ITAM provides better data security and helps to improve software compliance. It provides better communication between business units such as operation, finance, marketing teams, and frontline staff. Configuration management is another aspect that helps to maintain IT inventory data along with details such as owner and their current state. Let's learn more about it.

**Configuration management**

Configuration management maintains **Configuration Items** (**CIs**) to manage and deliver an IT service. CIs are tracked in the **Configuration Management Database** (**CMDB**). The CMDB stores and manages system component records with their attributes such as their type, owner, version, and dependency with other components. The CMDB keeps track of whether the server is physical or virtual, the operating system and its version (that is, Windows 2022 or **Red Hat Enterprise Linux** (**RHEL**) 8.0), the owner of the server (that is, support, marketing, or HR), and whether it has a dependency on other servers such as order management, and so on.

Configuration management is different from asset management. Asset management handles the entire life cycle of an asset, from planning to retirement, while CMDB is a component of asset management that stores configuration records of an individual asset. As shown in the following diagram, configuration management implements the integration and maintenance part of asset management:

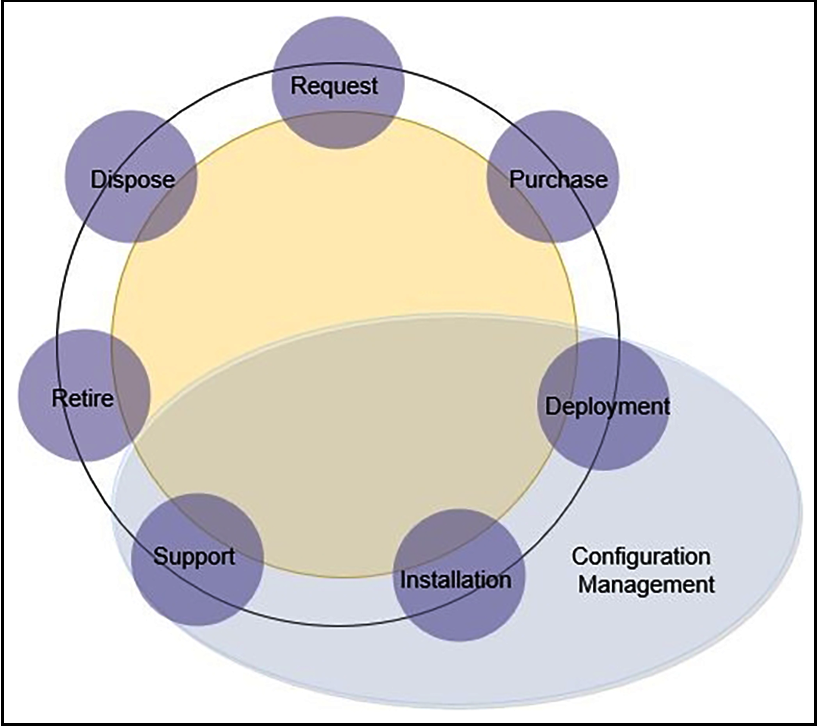


Figure 10.2: IT asset life cycle versus configuration management

Configuration management, as shown in the preceding diagram, implements the deployment, installation, and support part of asset management. The configuration management tool can help the operations team to reduce downtime by providing readily available information on asset configuration.

Implementing effective change management helps us to understand the impact of any changes in the environment. The most popular configuration management tools are Chef, Puppet, Ansible, and Bamboo. You will learn more details about them in *Chapter 12*, *DevOps and Solution Architecture Framework*.

IT management becomes easier if your workload is in a public cloud such as **Amazon Web Services** (**AWS**), Microsoft Azure, or **Google Cloud Platform** (**GCP**). Cloud vendors provide inbuilt tools to track and manage IT inventories and configuration in one place. For example, AWS provides services such as AWS Config, which tracks all IT inventories that spin up as a part of your AWS cloud workload, and services such as **AWS Trusted Advisor**, which recommends cost, performance, and security improvements, which you can use to decide how to manage your workload. You can see an example in the following screenshot:

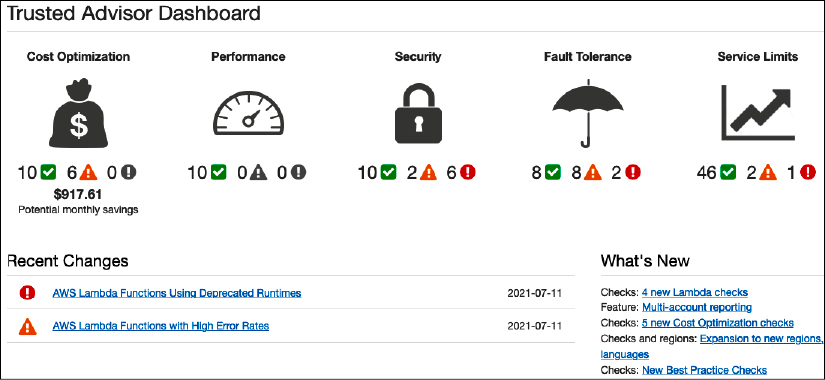


Figure 10.3: AWS Trusted Advisor dashboard

As shown in the preceding screenshot, the AWS **Trusted Advisor Dashboard** is showing 6 security issues that you can further explore to find out more details.

Configuration management helps to continuously monitor and record your IT resource configurations and allows you to automate the evaluation of recorded configurations against desired configurations. Configuration management offers the following benefits:

* **Continuous monitoring**: Continuously monitor and record configuration changes of your IT resources.
* **Change management**: Track the relationships among resources and review resource dependencies prior to making changes.
* **Continuous assessment**: Continuously audit and assess the overall compliance of your IT resource configurations with your organization's policies and guidelines.
* **Enterprise-wide compliance monitoring**: View the compliance status across your enterprise and identify non-compliant accounts. You can dive deeper to view the status for a specific region account.
* **Manage third-party resources**: Publish the configuration of third-party resources such as GitHub repositories, Microsoft Active Directory resources, or any on-premises and on-cloud server.
* **Operational troubleshooting**: Capture a comprehensive history of your AWS resource configuration changes to simplify troubleshooting of your operational issues.

Configuration management helps you to perform security analysis, continuously monitor the configurations of your resources, and evaluate their configurations for potential security weaknesses. It helps you to assess compliance with your internal policies and regulatory standards by providing you visibility into the configuration of your IT resources as well as third-party resources, and evaluating resource configuration changes against your desired configurations on a continuous basis.

The enterprise creates a framework such as an **Information Technology Infrastructure Library** (**ITIL**), which implements an **Information Technology Service Management** (**ITSM**) best practice. An ITIL provides a view on how to implement ITSM.

In this section, you learned about asset management and configuration management, which is part of the ITIL framework and more relevant to operational excellence. ITSM helps organizations to run their IT operations daily. You can learn more about ITIL from its governing body AXELOS by visiting their website (<https://www.axelos.com/best-practice-solutions/itil>). AXELOS offers ITIL certification to develop skills in the IT service management process. As you have learned about planning, let's explore the functioning of IT operations in the next section.

**The functioning of operational excellence**

Operational excellence is determined by proactive monitoring and quickly responding to recover in the case of an event. By understanding the operational health of a workload, it is possible to identify when events and responses impact it. Use tools that help understand the operational health of the system using **metrics** and **dashboards**. You should send log data to centralized storage and define metrics to establish a benchmark.

By defining and knowing what a workload is, it is possible to respond quickly and accurately to operational issues. Use tools to automate responses to operational events supporting various aspects of your workload. These tools allow you to automate responses for operational events and initiate their execution in response to alerts.

Make your workload components replaceable, so that rather than fixing the issue you can improve recovery time by replacing failed components with known good versions. Then, analyze the failed resources without impacting a production environment. For the functioning of operational excellence, the following are the areas where appropriate tools are needed:

* Monitoring system health
* Handling alerts and incident response

Let's understand each area in more detail with information on the available tools and processes.

**Monitoring system health**

Keeping track of system health is essential to understanding *workload behavior*. The operations team uses system health monitoring to record any anomalies in the system component, and acts accordingly. Traditionally, monitoring is limited to the infrastructure layer keeping track of the server's CPU and memory utilization. However, monitoring needs to be applied to every layer of the architecture. The following are the significant components where monitoring is applied.

**Infrastructure monitoring**

Infrastructure monitoring is essential and is the most popular form of monitoring. Infrastructure includes components required for hosting applications. These are the core services such as storage, servers, network traffic, load balancer, and so on. Infrastructure monitoring may consist of metrics, such as the following:

* **CPU usage**: Percentage of CPU utilized by the server in a given period
* **Memory usage**: Percentage of **Random-Access Memory** (**RAM**) utilized by the server in a given period
* **Network utilization**: Network packets *in and out* over the given period
* **Disk utilization**: Disk read/write throughput and **Input/Output Operations per Second** (**IOPS**)
* **Load balancer**: Number of request counts in a given period

There are many more metrics available, and organizations need to customize those monitoring metrics as per their application monitoring requirement. The following screenshot shows a sample monitoring dashboard for network traffic:

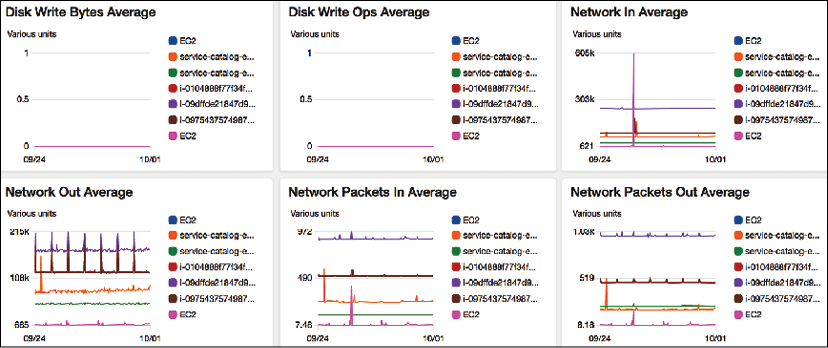


Figure 10.4: Infrastructure monitoring dashboard

You can see the preceding system dashboard showing a spike in one day in the **Network In Average** pane, with color-coding applied for different servers. The operations team can dive deep into each graph and resources to get a more granular view to determine overall infrastructure health.

**Application monitoring**

Sometimes, your infrastructure is all healthy except applications having an issue due to some bug in your code or any third-party software issues. You may have applied some vendor-provided operating system security patch that messed up your application. Application monitoring may include metrics, such as the following:

* **Endpoint invocation**: Number of requests in a given period
* **Response time**: Average response time to fulfill the request
* **Throttle**: Number of valid requests spilled out as the system runs out of capacity to handle the additional requests
* **Error**: Application throws an error while responding to a request

The following screenshot shows a sample application endpoint-monitoring dashboard:

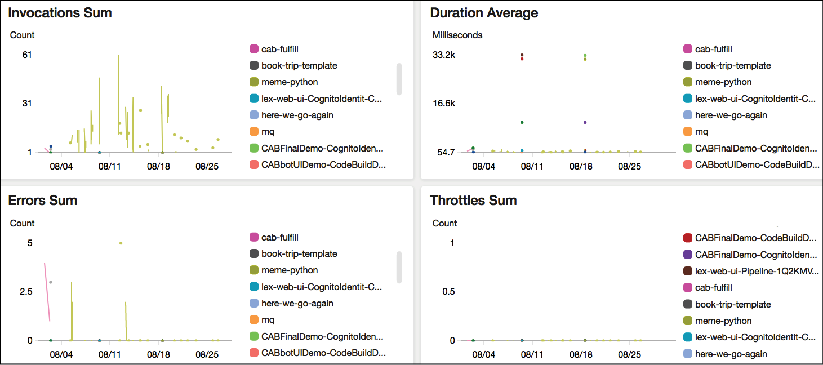


Figure 10.5: Application monitoring dashboard

There could be many more metrics based on application and technology—for example, a memory garbage collection amount for a Java application, a number of HTTP POST and GET requests for a RESTful service, a count of 4XX client errors, a count of 5XX server errors for a web application, and what they might be looking for that would indicate poor application health.

**Platform monitoring**

Your application may be utilizing several third-party platforms and tools that need to be monitored. These may include the following:

* **Memory caching**: Redis and Memcached
* **Relational database**: Oracle Database, Microsoft SQL Server, Amazon **Relational Database Service** (**RDS**), PostgreSQL
* **NoSQL database**: Amazon DynamoDB, Apache Cassandra, MongoDB
* **Big data platform**: Apache Hadoop, Apache Spark, Apache Hive, Apache Impala, Amazon **Elastic MapReduce** (**EMR**)
* **Containers**: Docker, Kubernetes, OpenShift
* **Business intelligence tool**: Tableau, MicroStrategy, Kibana, Amazon QuickSight
* **Messaging system**: MQSeries, **Java Message Service** (**JMS**), RabbitMQ, **Simple Queue Service** (**SQS**)
* **Search**: Elasticsearch, Solr search-based application

Each of the aforementioned tools has its own set of metrics that you need to monitor to make sure your application is healthy as a whole. The following screenshot shows the monitoring dashboard of a relational database platform:

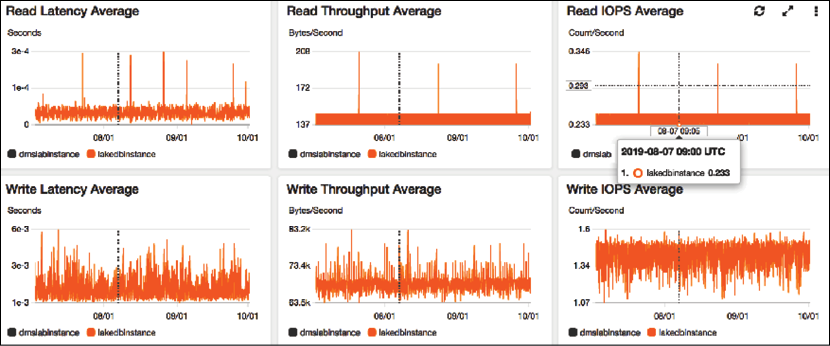


Figure 10.6: Platform monitoring dashboard for a Relational Database Management System (RDBMS)

In the preceding dashboard, you can see the database has lots of write activity, which is showing that the application is continuously writing data. On the other hand, read events are relatively consistent except for some spikes.

**Log monitoring**

Traditionally, log monitoring was a manual process, and organizations took a reactive approach to analyze logs when issues were encountered. However, with more competition and increasing expectations from users, it has become essential to take quick action before the user notices the issue. For a proactive approach, you should have the ability to stream logs in a centralized place and run queries to monitor and identify the issue.

For example, if some product page is throwing an error, you need to know the error immediately and fix the problem before the user complains, else you will suffer a revenue loss. In the case of any network attack, you need to analyze your network log and block suspicious IP addresses. Those IPs may be sending an erroneous number of data packets to bring down your application. Monitoring systems such as AWS CloudWatch, Logstash, Splunk, Google Stackdriver, and so on provide an agent to install in your application server. The agent will stream logs to a centralized storage location. You can directly query to central log storage and set up alerts for any anomalies.

The following screenshot shows a sample network log collected in a centralized place:

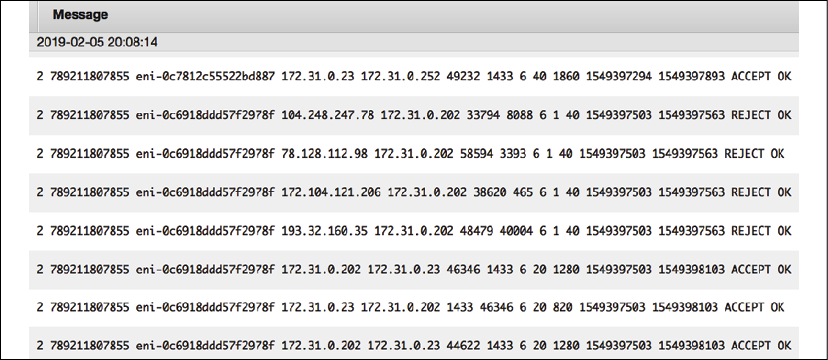


Figure 10.7: Raw network log streamed in a centralized datastore

You can run a query in these logs and find out the top 10 source IP addresses with the highest number of reject requests, as shown in the following screenshot:

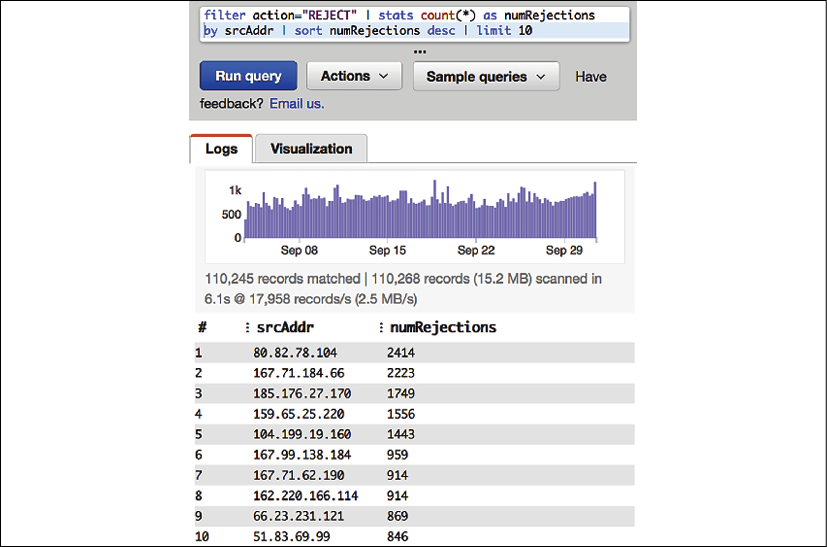


Figure 10.8: Insight from raw network log by running query

As shown in the preceding query editor, you can create a graph and put an alarm in for if the number of rejections detected crosses a certain threshold, such as more than 5,000.

**Security monitoring**

Security is a critical aspect of any application. Security monitoring should be considered during solution design. As you learned when we looked at security in the various architectural components in *Chapter 8*, *Security Considerations*, security needs to be applied at all layers. You need to implement security monitoring to act and respond to any event. The following significant components show where monitoring needs to be applied:

* **Network security**: Monitor any unauthorized port opening, suspicious IP address, and activity
* **User access**: Monitor any unauthorized user access and suspicious user activity
* **Application security**: Monitor any malware or virus attack
* **Web security**: Monitor a **Distributed Denial of Service** (**DDoS**) attack, SQL injection, or **Cross-Site Scripting** (**XSS**)
* **Server security**: Monitor any gap in security patches
* **Compliance**: Monitor any compliance lapses such as violations of **Payment Card Industry** (**PCI**) compliance checks for payment applications or the **Health Insurance Portability and Accountability Act** (**HIPAA**) for healthcare applications
* **Data security**: Monitor unauthorized data access, data masking, and data encryption at rest and in transit

For monitoring, you can use various third-party tools such as Imperva, McAfee, Qualys, Palo Alto Networks, Sophos, Splunk, Sumo Logic, Symantec, Turbot, and so on.

One example of security monitoring using **Amazon Detective** for the AWS cloud is shown below:

Graphical user interface, application

Description automatically generated

Figure 10.9: Security monitoring using Amazon GuardDuty

While you are putting application monitoring tools in place to monitor all components of your system, it is essential to monitor the monitoring system. Make sure to monitor the host of your monitoring system. For example, if you're hosting your monitoring tool in Amazon **Elastic Compute Cloud** (**EC2**), then AWS CloudWatch can monitor the health of EC2.

**Handling alerts and incident response**

Monitoring is one part of operational excellence functioning; the other part involves handing alerts and acting upon them. Using alerts, you can define the system threshold and when you want to work. For example, if the server CPU utilization reaches 70% for 5 minutes, then the monitoring tool records high server utilization and sends an alert to the operations team to take action to bring down CPU utilization before a system crash. Responding to this incident, the operations team can add the server manually. When automation is in place, autoscaling triggers the alert to add more servers as per demand. It also sends a notification to the operations team, which can be addressed later.

Often, you need to define the alert category, and the operations team prepares for the response as per the alert severity.

The following levels of severity provide an example of how to categorize alert priority:

* **Severity 1**: Sev1 is a critical priority issue. A Sev1 issue should only be raised when there is a significant customer impact, for which immediate human intervention is needed. A Sev1 alert could be that the entire application is down. The typical team needs to respond to these kinds of alerts within 15 minutes and requires 24/7 support to fix the issue.
* **Severity 2**: Sev2 is a high-priority alert that should be addressed in business hours. For example, the application is up, but the rating and review system is not working for a specific product category. The typical team needs to respond to these kinds of alerts within 24 hours and requires regular office hours' support to fix the issue.
* **Severity 3**: Sev3 is a medium-priority alert that can be addressed during business hours over days—for example, the server disk is going to fill up in 2 days. The typical team needs to respond to these kinds of alerts within 72 hours and requires regular office hours' support to fix the issue.
* **Severity 4**: Sev4 is a low-priority alert that can be addressed during business hours over the week—for example, **Secure Sockets Layer** (**SSL**) certification is going to expire in 2 weeks. The typical team needs to respond to these kinds of alerts within the week and requires regular office hours' support to fix the issue.
* **Severity 5**: Sev5 falls into the notification category, where no escalation is needed, and it can be simple information—for example, sending a notification that deployment is complete. Here, no response is required in return since it is only for information purposes.

Each organization can have different alert severity levels as per their application needs. Some organizations may want to set four levels for severity, and others may go for six. Also, alert response times may differ. Maybe some organization wants to address Sev2 alerts within 6 hours on a 24/7 basis, rather than waiting for them to be addressed during office hours.

While setting up an alert, make sure the title and summary are *descriptive* and *concise*. Often, an alert is sent to a mobile (as an SMS) or a pager (as a message) and needs to be short and informative enough to take immediate action. Make sure to include proper metrics data in the message body.

In the message body, include information such as *The disk is 90% full in production-web-1 server* rather than just saying *The disk is full*. The following screenshot shows an example alarm dashboard:

Graphical user interface, text, application, email

Description automatically generated

Figure 10.10: Alarm dashboard

As shown in the preceding alarm dashboard, there is one alarm in progress when a NoSQL Amazon DynamoDB database table called **testretail** is using a low write capacity unit and causing unnecessary additional cost. The bottom and top two alarms have an **OK** status as data is collected during monitoring that is well within the threshold. There may be other alarms showing **Insufficient data**, which means there are not enough data points to determine the state of resources you are monitoring. You should only consider this alarm valid if it can collect data and move into the **OK** state.

Testing of incident response in the case of critical alerts is important to make sure you are ready to respond as per the defined SLA. Make sure your threshold is set up correctly so that you have enough room to address the issue, and, also, don't send too many alerts. Make sure that as soon as the issue is resolved, your alert gets reset to the original setting and is ready to capture event data again.

An incident is any unplanned disruption that impacts the system and customer negatively. The first response during an incident is to recover the system and restore the customer experience. Fixing the issue can be addressed later as the system gets restored and starts functioning. The automated alert helps to discover the incident actively and minimizes user impact. This can act as a failover to a disaster recovery site if the entire system is down, and the primary system can be fixed and restored later.

For example, Netflix uses the **Simian Army** (<https://netflixtechblog.com/the-netflix-simian-army-16e57fbab116>), which has **Chaos Monkey** to test system reliability. Chaos Monkey orchestrates the random termination of a production server to test if the system can respond to disaster events without any impact on end users. Similarly, Netflix has other monkeys to test various dimensions of system architecture, such as **Security Monkey**, **Latency Monkey**, and even **Chaos Gorilla**, which can simulate outage of the entire availability zone.

Monitoring and alerts are critical components to achieving operational excellence. All monitoring systems typically have an alert feature integrated with them. A fully automated alert and monitoring system improves the operations team's ability to maintain the health of the system, provide expertise to take quick action, and excel in the user experience.

As you monitor your application environment, it's important to apply continuous improvement and achieve excellence. Let's learn in more detail about improving operational excellence.

**Improving operational excellence**

Continuous improvement is required for any process, product, or application to excel. Operational excellence needs continuous improvement to attain maturity over time. You should keep implementing small incremental changes as you perform **RCA** (**Root Cause Analysis**) and learn lessons from various operations' activities.

Learning from failure will help you to anticipate any operational event that may be planned (such as deployments) or unplanned (such as utilization surge). You should record all lessons learned and update remedies in your operation runbook. For operational improvement, the following are the areas where you need appropriate tools:

* IT operation analytics
* Root cause analysis
* Auditing and reporting

**IT operations analytics**

**IT Operations Analytics** (**ITOA**) is the practice of gathering data from various resources to make a decision and predict any potential issue that you may encounter. It's essential to analyze all events and operational activities in order to improve. Analyzing failures will help to predict any future event and keep the team ready to provide the appropriate response.

Implement a mechanism to collect the logs of operations events, various activities across workloads, and infrastructure changes. You should create a detailed activity trail and maintain an activity history for audit purposes.

A large organization could have hundreds of systems generating a massive amount of data. You need a mechanism to ingest and store all logs and event data for a length of time, such as 90 or 180 days, to get insight. ITOA uses big data architecture to store and analyze multiple terabytes of data from all over the place. ITOA helps to discover any issue that you could not find by looking at individual tools and helps to determine dependencies between various systems, providing a holistic view.

As shown in the following diagram, each system has its own monitoring tool that helps to get insights and maintains individual system components. For operation analytics, you need to ingest this data in a centralized place. All operation data collection in one place gives a single source of truth, where you can query required data and run analytics to get a meaningful insight:

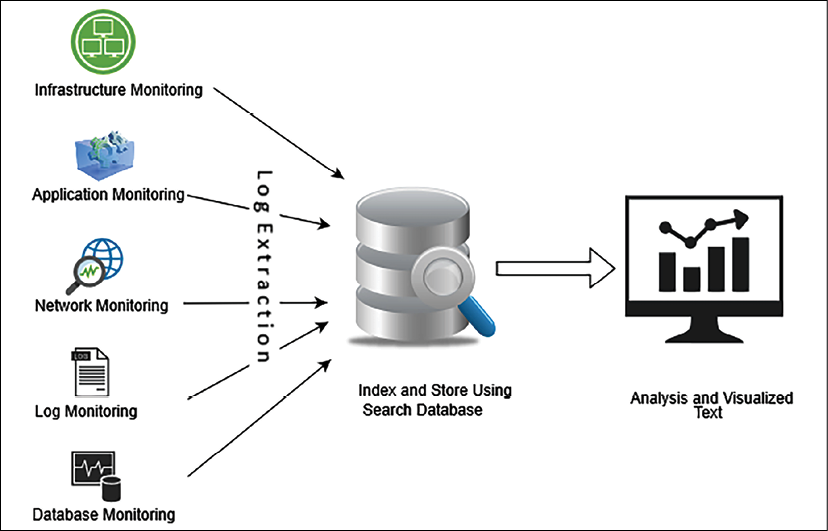


Figure 10.11: Big data approach for ITOA

To create an operation analytics system, you can use scalable big data storage such as Amazon**Simple Storage Service** (**S3**). You can also store data in an on-premises Hadoop cluster. For data extraction, the agent can be installed in each server, which can send all monitoring data to a centralized storage system. You can use the Amazon CloudWatch agent to collect data from each server and store it in S3.

Third-party tools such as **ExtraHop** and **Splunk** can help to extract data from various systems.

Once data is collected in centralized storage, you can perform a transformation to make data ready for search and analysis. Data transformation and cleaning can be achieved using a big data application such as Spark, MapReduce, AWS Glue, and so on. To visualize the data, you can use any business intelligence tool such as Tableau, MicroStrategy, Amazon QuickSight, and so on. Here, we are talking about building an **Extract, Transform, and Load** (**ETL**) pipeline. You will learn more details in*Chapter 13*, *Data Engineering for Solution Architecture*. You can further perform machine learning to do predictive analysis on a future event.

You will learn more about machine learning in *Chapter 14*, *Machine Learning Architecture*.

**Root cause analysis**

For continuous improvement, it is essential to prevent any errors from happening again. If you can identify problems correctly, then an efficient solution can be developed and applied. It's important to get to the root cause of the problem to fix the problem. **Five whys** is a simple, yet most effective, technique to identify the root cause of a problem.

In the *five whys technique*, you gather the team for a retrospective look at an event and ask five consecutive questions to identify actual issues. Take an example where data is not showing up in your application monitoring dashboard. You will ask five whys to get to the root cause.

**Problem**: The application dashboard is not showing any data.

1. Why: Because the application is unable to connect with the database
2. Why: Because the application is getting a database connectivity error
3. Why: Because the network firewall is not configured to the database port
4. Why: Because the configuring port is a manual check and the infrastructure team missed it
5. Why: Because the team doesn't have the tools for automation

**Root Cause**: Manual configuration error during infrastructure creation.

**Solution**: Implement a tool for automated infrastructure creation.

In the preceding example, at first glance the issue looks like it is related to the application. After the *five whys* analysis, it turns out to be a bigger problem and there is a need to introduce automation to prevent similar incidents.

RCA helps the team to document lessons learned and continuously build upon it for operational excellence. Make sure to update and maintain your runbook-like code and share best practices across the team.

**Auditing and reporting**

Auditing is one of the essential activities to create recommendations and identify any malicious activity in the system by internal or external interference. An audit becomes especially important if your application needs to be compliant as per regulatory body requirements—for example, PCI, HIPAA, **Federal Risk and Authorization Management Program** (**FedRAMP**), **International Organization for Standardization** (**ISO**), and so on. Most of the compliant regulatory bodies need to conduct regular audits and verify each activity going on in the system to prepare a compliance report and grant a certificate.

An audit is essential to prevent and detect security events. A hacker may silently get into your system and systematically steal information without anyone noticing. Regular security audits can uncover a hidden threat. You may want to conduct a regular audit for cost optimization to identify if resources are running idle when not required. Also, determine resource demand and available capacity so that you can plan.

In addition to alert and monitoring, the operations team is also responsible for saving the system from any threat by enabling and conducting the audit. An IT audit makes sure you safeguard IT assets and license protection and that you ensure data integrity and operations adequately to achieve your organizational goal. The following screenshot shows a data audit stored in an Amazon S3 bucket using Amazon Macie, which is a data security and data privacy service that uses machine learning and pattern matching to discover and protect your sensitive data in AWS.

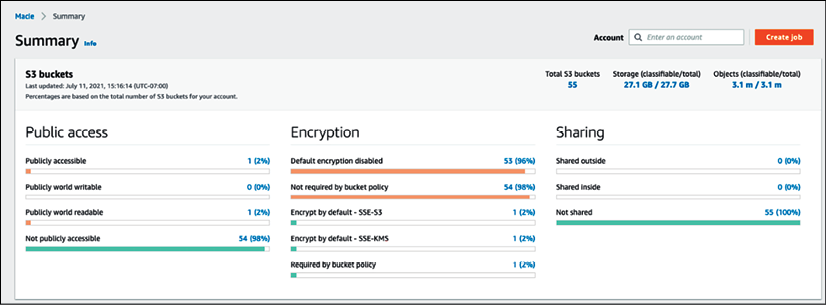


Figure 10.12: Data audit report summary from Amazon Macie

The data audit report in the preceding screenshot shows data accessibility, encryption, and data sharing reports along with data storage and size details.

Auditing steps include planning, preparing, evaluation, and reporting. Any risk item needs to be highlighted in the report, and follow-ups will be conducted to address open issues.

For operational excellence, the team can perform internal audit checks to make sure all systems are healthy and that proper alerts are in place to detect any incidents.

**Achieving operational excellence in the public cloud**

A public cloud provider such as AWS, GCP, or Azure provides many inbuilt capabilities and guidance to achieve operational excellence in the cloud. Cloud providers advocate automation, which is one of the most essential factors for operational excellence. Taking the example of the AWS cloud, the following services can help to achieve operational excellence:

* **Planning**: Operational excellence planning includes the identification of gaps and recommendations, automating via scripting, and managing your fleet of servers for patching and updates. The following AWS services help you in the planning phase:
  + **AWS Trusted Advisor**: AWS Trusted Advisor checks your workload based on prebuilt best practices and provides recommendations to implement them
  + **AWS CloudFormation**: With AWS CloudFormation, the entire workload can be viewed as code, including applications, infrastructure, policy, governance, and operations
  + **AWS Systems Manager**: AWS Systems Manager provides the ability to manage cloud servers in bulk for patching, updates, and overall maintenance
* **Functioning**: Once you have created operational excellence best practices and applied automation, you need continuous monitoring of your system to be able to respond to an event. The following AWS services help you in system monitoring, alerts, and automated responses:
  + **Amazon CloudWatch**: CloudWatch provides hundreds of inbuilt metrics to monitor workload operation and trigger alerts as per the defined threshold. It provides a central log management system and triggers an automated incident response.
  + **AWS Lambda**: The AWS service used to automate responses to operational events is AWS Lambda.
* **Improving**: As incidents come into your system, you need to identify their pattern and root cause for continuous improvement. You should apply the best practice to maintain the version of your scripts. The following AWS services will help you to identify and apply system improvements:
  + **Amazon OpenSearch**: OpenSearch helps to learn from experience. Use OpenSearch to analyze log data to gain insight and use analytics to learn from experience.
  + **AWS CodeCommit**: Share learning with libraries, scripts, and documentation by maintaining them in the central repository as code.

AWS provides various capabilities to run your workload and operations as code. These capabilities help you to automate operations and incident response. With AWS, you can easily replace failed components with a good version and analyze the failed resources without impacting the production environment.

On AWS, aggregate the logs of all system operation and workload activities, and infrastructure, to create an activity history, such as with AWS CloudTrail. You can use AWS tools to query and analyze operations over time and identify a gap for improvement. In the cloud, resource discovery is easy, as all assets are located under the API- and web-based interfaces within the same hierarchy. You can also monitor your on-premises workload from the cloud. For security auditing in the AWS cloud Amazon GuardDuty and Amazon Detective provide great insight and details across multiple accounts.

Operational excellence is a continuous effort. Every operational failure should be analyzed to improve the operations of your application. By understanding the needs of your application's load, documenting regular activities as a runbook, following steps to guide issue handling, using automation, and creating awareness, your operations will be ready to deal with any failure event.

**Summary**

Operational excellence can be achieved by working on continuous improvement as per operational needs, and lessons learned from past events using RCA. You can achieve business success by increasing the excellence of your operations. Build and operate applications that increase efficiency while building highly responsive deployments. Use best practices to make your workloads operationally excellent.

In this chapter, you learned about the design principles to achieve operational excellence. These principles advocate operation automation, continuous improvement, taking an incremental approach, predicting failure, and being ready to respond.

You learned about various phases of operational excellence and the corresponding technology choices. In the planning phase, you learned about ITAM to track the inventory of IT resources and identify dependencies between them using configuration management.

You learned about alerts and monitoring in the functioning phase of operational excellence. You considered various kinds of monitoring, with examples such as infrastructure, application, log, security, and platform monitoring. You learned about the importance of alerts, and how to define alert severity and respond to it.

During the improvement phase of operational excellence, you learned about analytics in IT operations by building a big data pipeline, methods to perform RCA using the *five whys*, and the importance of auditing to save the system from any malicious behaviors and unnoticed threats. You learned about operational excellence in the cloud and different inbuilt tools that can be utilized for operational excellence in the AWS cloud.

As of now, you have learned best practices in the areas of performance, security, reliability, and operational excellence. In the next chapter, you will learn about best practices for cost optimization. You will also learn about various tools and techniques to optimize overall system costs and how to leverage multiple tools in the cloud to manage IT expenditure.