Here are the Splunk-related tasks that a Site Reliability Engineer (SRE) typically handles on a daily basis:

1. Monitoring and Alerting

Monitor logs from applications, servers, and infrastructure to ensure systems are healthy.

Investigate and resolve alerts triggered by Splunk monitoring.

Create alerts for critical events such as high CPU usage, low disk space, failed deployments, or service outages.

2. Log Analysis and Troubleshooting

Analyze application logs to identify the root cause of incidents or failures.

Correlate logs from multiple systems to detect patterns or issues.

Support development and operations teams in troubleshooting production issues.

3. Dashboard Creation and Maintenance

Create and maintain Splunk dashboards to provide real-time visibility of infrastructure, applications, and service health.

Develop custom dashboards to monitor specific applications or processes.

Continuously optimize dashboards for performance and relevance.

4. Search Query and Report Generation

Write and execute Splunk Search Processing Language (SPL) queries to extract insights from logs.

Generate custom reports for incident analysis, system performance, and capacity planning.

Schedule automated reports to be shared with stakeholders.

5. Incident Management Support

Assist the incident management team by providing log insights during major incidents.

Quickly search logs to find the root cause of high-priority issues.

Correlate data across logs to detect broader infrastructure or application issues.

6. Log Ingestion and Parsing

Ensure proper ingestion of logs from various data sources (servers, containers, cloud services, etc.).

Create and maintain log parsing rules to structure incoming data for easy analysis.

Troubleshoot any issues related to missing or incomplete logs.

7. Health Check of Splunk Infrastructure

Monitor Splunk infrastructure (indexers, search heads, forwarders) to ensure they are running efficiently.

Identify and resolve bottlenecks in Splunk performance.

Upgrade and patch Splunk components as needed.

8. Data Onboarding and Configuration

Onboard new log sources from different applications, databases, cloud services, or infrastructure.

Configure Splunk forwarders and data inputs for new services or infrastructure components.

Validate and test log data to ensure accuracy and completeness.

9. Security and Compliance Monitoring

Monitor security-related events such as unauthorized access attempts, firewall breaches, or data leaks.

Create alerts for potential security incidents.

Generate compliance reports for audits or regulatory requirements.

10. Automation and Integration

Automate log collection, parsing, and alerting using Splunk scripts or APIs.

Integrate Splunk with other tools like ServiceNow, Dynatrace, PagerDuty, etc. for incident and event management.

Automate daily tasks like log archiving or data retention management.

11. Capacity Planning and Optimization

Monitor Splunk license usage to ensure it does not exceed the allocated limit.

Optimize indexing and storage policies to manage Splunk's data retention effectively.

Plan for Splunk infrastructure scaling as log volume increases.

12. Documentation and Knowledge Sharing

Maintain documentation for Splunk dashboards, alerts, and configurations.

Share insights and trends with development, operations, and security teams.

Provide training to team members on using Splunk effectively.

Here are some commonly used Splunk search queries and alert configurations for SRE tasks:

✅ **1. Application Error Monitoring**

Objective: Detect application errors in logs.

Search Query:

```
index=application_logs source="/var/log/app.log"
| where error_code != 200
| stats count by error_code, error_message, host
```

| sort -count

Explanation:

Monitors application logs for errors.

Groups errors by code, message, and host.

Sorts the result to show the most frequent errors first.

Alert Configuration:

Trigger alert if count > 10 within 15 minutes.

Send email notification to the SRE team.

✅ 2. High CPU Usage Alert

Objective: Monitor high CPU usage on servers.

Search Query:

index=system_metrics source="cpu_usage.log"

| stats avg(cpu_usage) as AvgCPU by host

| where AvgCPU > 90

Explanation:

Captures CPU usage logs.

Alerts when CPU usage exceeds 90% on any host.

Alert Configuration:

Trigger if AvgCPU > 90 for 10 minutes.

Send alert to PagerDuty or email.

✅ 3. Failed Deployment Detection

Objective: Identify deployment failures.

Search Query:

index=deployment_logs source="/var/log/deploy.log"

| search "deployment failed" OR "error during deployment"

| stats count by application_name, environment, host

Explanation:

Searches for deployment failure keywords.

Groups failures by application, environment, and host.

Alert Configuration:

Trigger if count > 0 within the last 30 minutes.

Notify DevOps and Application Support teams.

✅ 4. Unauthorized Access Attempt Alert

Objective: Monitor unauthorized login attempts.

Search Query:

```
index=security_logs source="auth.log"
| search "failed password" OR "invalid user"
| stats count by user, host
| where count > 5
```

Explanation:

Tracks login failures.

Flags repeated failed attempts from the same user/host.

Alert Configuration:

Trigger if count > 5 in 30 minutes.

Send alert to the security team.

✅ 5. Disk Space Monitoring

Objective: Alert when disk space usage exceeds 90%.

Search Query:

```
index=system_metrics source="disk_usage.log"
| stats latest(disk_usage) as Usage by host, mount_point
| where Usage > 90
```

Explanation:

Monitors disk usage by host and mount point.

Alerts when disk usage crosses 90%.

Alert Configuration:

Trigger when Usage > 90 for more than 15 minutes.

Send alert via email or webhook.

✅ 6. Application Latency Monitoring

Objective: Detect high application response time.

Search Query:

index=application_logs source="/var/log/access.log"

| timechart avg(response_time) by application_name

| where avg(response_time) > 5000

Explanation:

Tracks average response time.

Flags applications with response times exceeding 5 seconds.

Alert Configuration:

Trigger if latency exceeds 5000ms for 15 minutes.

Send alert to DevOps and SRE teams.

✅ 7. Log Volume Spike Alert

Objective: Detect abnormal increase in log volume.

Search Query:

index=application_logs

| timechart count by host

| where count > 10000

Explanation:

Monitors log volume from each host.

Flags unusually high log volume, indicating potential issues.

Alert Configuration:

Trigger if count > 10000 within an hour.

Notify DevOps team.

✅ 8. Application Crash Alert

Objective: Identify when an application crashes.

Search Query:

index=application_logs

| search "application crashed" OR "service stopped"

| stats count by application_name, host

Explanation:

Searches for crash events.

Groups crashes by application and host.

Alert Configuration:

Trigger if count > 0 within 15 minutes.

Notify the Application Support team.

✅ 9. Splunk License Usage Monitoring

Objective: Monitor Splunk license consumption.

Search Query:

index=_internal source=*license_usage.log

| timechart sum(b) as volume by host

| eval volume_GB=round(volume/1024/1024/1024,2)

| where volume_GB > 10

Explanation:

Tracks Splunk license usage.

Converts bytes to GB and alerts if usage exceeds a threshold.

Alert Configuration:

Trigger if daily usage > 10GB.

Send notification to Splunk administrator.

✅ 10. Service Restart Monitoring

Objective: Track unexpected service restarts.

Search Query:

index=system_logs source="service.log"

| search "service restarted"

| stats count by service_name, host

Explanation:

Tracks service restarts.

Helps detect unstable applications.