Wie gut verstehen Sie die Kosten Ihrer Kubernetes-Workloads?

Überflüssiges aufspüren. Kosten optimieren.
Performance sicherstellen.

splunk>
a CISCO company

Vorwort

Kubernetes bietet Flexibilität und Geschwindigkeit, die Teams können schnell iterieren und auf Bedarfe reagieren. Die Vorteile bringen jedoch neue Komplexität mit sich. Da Kubernetes-Umgebungen laufend skalieren und sich verändern, ist nur schwer zu überschauen, was läuft, was wichtig ist und was das alles kostet. Tatsächlich schätzt Gartner die weltweiten Public-Cloud-Ausgaben der End-User bereits auf $ 723 Milliarden.

Dem Splunk-Bericht Die neuen Regeln des Datenmanagements zufolge sind bei 91 % der Unternehmen die Datenmanagement-Kosten gestiegen. Kein Wunder, denn die Teams sammeln Unmengen von Telemetriedaten von allen Enden ihrer Umgebungen. Doch ohne Kontext sind diese Daten nur teures Rauschen. Und der Kontext fehlt häufig – gerade in Kubernetes-Umgebungen, wo die Ressourcen kurzlebig sind und die Skalierung automatisch erfolgt. Die Teams können dann nur Mutmaßungen über Ressourcennutzung, Leistungsengpässe und Kostentreiber anstellen, anstatt gezielt Maßnahmen zu ergreifen.

Die Teams haben alles auf ihren Dashboards, wissen aber nicht, was sie verbessern sollten und wo sie ansetzen sollen. Observability schließt diese Lücke, korreliert Telemetrie-Signale mit Performance-Daten und Kosten und zeigt auf, was funktioniert und was zu teuer ist, sodass die Teams sich auf das konzentrieren können, was zählt.

Die Entscheidungen, wie Kubernetes-Workloads bereitgestellt und skaliert werden sollen, sind stets Kompromisse. Überprovisionierung sichert die Stabilität der Services, ist aber Verschwendung. Unterprovisionierung senkt zwar die Kosten, ist aber riskant. Die Teams müssen klar erkennen können, wie sich ihre Workload-Konfigurationen und Skalierungsentscheidungen auswirken.

Dieses E-Book unterteilt das Kubernetes-Kostenmanagement in drei Phasen: **Transparenz,** sodass sichtbar wird, was ausgeführt wird; **Optimierung** wichtiger Workloads, wo Leistung, Kosten und Zuverlässigkeit am meisten ins Gewicht fallen; und die **Operations,** wo es gilt, Veränderungen einen Schritt voraus bleiben. Observability ermöglicht all dies, indem sie die Ressourcennutzung mit den Kosten verknüpft, Ineffizienzen aufdeckt und nachvollziehbar macht, wie sich Änderungen mit der Zeit auf die Performance auswirken.

Mehr zu Troubleshooting und Fehlerbehebung in Kubernetes finden Sie in unserem Leitfaden Troubleshooting in Kubernetes-Umgebungen.

Observability spielt eine entscheidende Rolle beim Management der Kubernetes-Performance und bei Entscheidungen über Kostenfragen. Mit Observability können die Teams:

- Einblick in den Zustand von Anwendungen und Services gewinnen
- Kostentreiber und Ineffizienzen identifizieren
- Informierte Entscheidungen zum Ausgleich von Performance und Kosten treffen

Die Illusion von Kontrolle

Viele Teams glauben, dass sie ihre Kubernetes-Umgebungen im Griff haben. Sie können ja überschauen, was ihre Services tun, haben die Uptime im Monitoring und können den Ressourcenverbrauch innerhalb der Silos sehen. Doch unter der Oberfläche hat die operative Sicht blinde Flecken ohne Ende.

In vielen Unternehmen arbeiten die Teams unabhängig voneinander, mit eigenen Tools, nach eigenen Standards und eigenen Observability-Verfahren. Das führt aber oft zu Ressourcenüberschneidungen, übersehenen Signalen und instabiler Performance. Und selbst wenn die Teams ihre jeweilige Datenansicht verlässlich finden, so hat doch niemand eine klare Übersicht über das Gesamtsystem.

Diese fragmentierte Transparenz ergibt so etwas wie eine **unsichtbare Infrastruktur:** Workloads, Services oder Cluster, die still und leise Ressourcen verbrauchen, ohne dass ihre Effizienz oder ihr Nutzen hinterfragt wird. Solche Lücken bleiben oft unbemerkt – bis die Kosten explodieren, die Performance einbricht oder ein überraschender Ausfall eine genauere Untersuchung erzwingt.

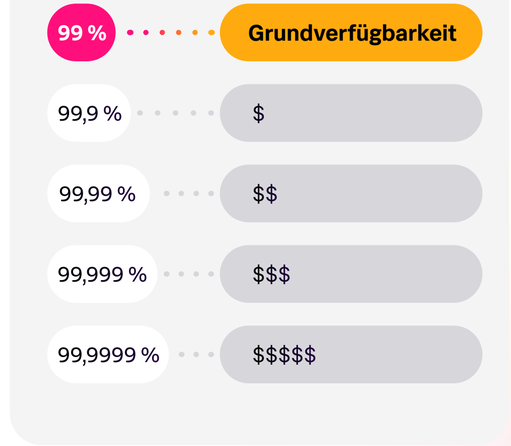
Als Cloud Computing noch am Anfang stand, entstand rasch eine Unmenge von **Schatten-IT.** Man zückte die Kreditkarte und beschaffte sich, was gerade gebraucht wurde. Diese Freiheit machte zwar Tempo, warf aber ernsthafte Probleme auf: Die Teams hatten am Ende Dubletten-Umgebungen, es häuften sich Kosten außerhalb des offiziellen Budgets, und es gab Sicherheitslücken, die niemand auf dem Radar hatte. Ohne zentrale Übersicht war kaum zu sagen, was gerade lief, wo es lief und warum. Das hat eine Zeit lang funktioniert. Irgendwann funktionierte es nicht mehr.

Bei Kubernetes ist die Situation ganz ähnlich. Kubernetes macht Bereitstellungen einfach und skaliert enorm schnell – doch was konkret läuft und warum es läuft, gerät mehr und mehr aus dem Blick.

Ohne gemeinsame Standards und Transparenz können die Teams den Verbrauch nicht überschauen und werden auf Ineffizienzen erst dann aufmerksam, wenn es zu spät ist.

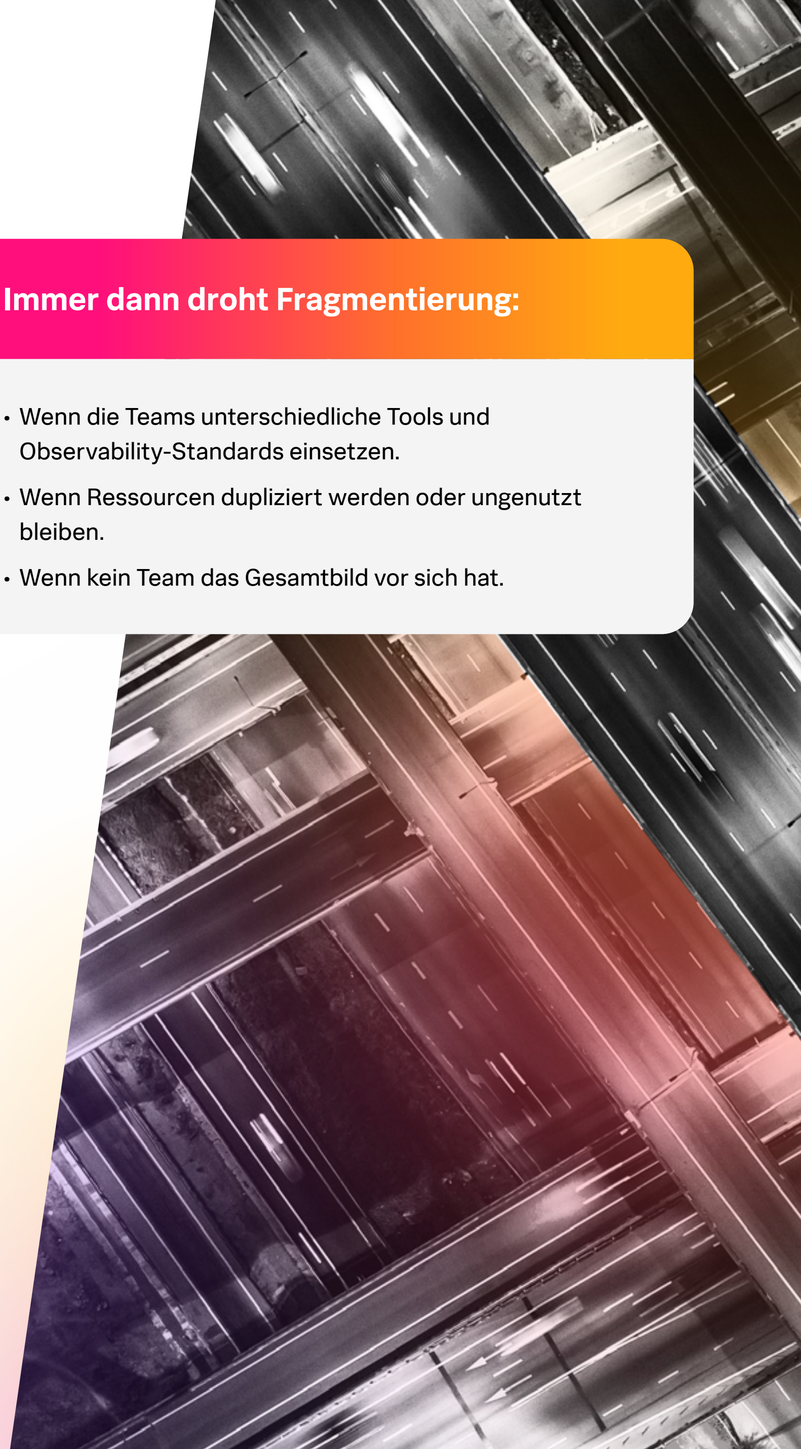
Darum gibt es Platform Engineering. Damit lassen sich Tools vereinheitlichen und Standards durchsetzen, die Experience der Dev-Teams wird optimiert und die teamübergreifende Zusammenarbeit gefördert. Das Plattform-Team kann die Fragmentierung reduzieren und der Engineering-Leitung einen klaren Überblick darüber verschaffen, was läuft, wo es läuft und welche Auswirkungen es hat.

Und jetzt die Gretchenfrage: Wissen Sie, was Ihre Kubernetes-Workloads wirklich kosten? Manchen fehlen vielleicht Ressourcen, während andere unbemerkt Budget verbrennen. **Und in Ihrem gesamten Stack klebt auf jedem zusätzlichen Neuner der Verfügbarkeit ein Preisschild.** Das ist die eigentliche Herausforderung: zu wissen, wann diese Ausgaben Mehrwert bringen und wann nicht.



Immer dann droht Fragmentierung:

- Wenn die Teams unterschiedliche Tools und Observability-Standards einsetzen.
- Wenn Ressourcen dupliziert werden oder ungenutzt bleiben.
- Wenn kein Team das Gesamtbild vor sich hat.



Phase 1

Transparenz: Das Gesamtbild im Blick

Damit Sie Ihre Kubernetes-Umgebung verstehen, benötigen Sie zunächst klare, verlässliche Sichtbarkeit. Das bedeutet mehr als Systemstatusprüfungen und Verfügbarkeitsmetriken. Sie müssen auch wissen, wie die Ressourcen genutzt werden, wer dafür verantwortlich ist und wie sie sich auf der Kostenseite darstellen. Das ist die Voraussetzung aller Optimierung.

Die erste Phase besteht darin, Klarheit zu schaffen, wohin die Ausgaben gehen. Sie müssen identifizieren, welche Workloads welche Kosten aufwerfen, wie sich die Kosten auf die Teams bzw. die Umgebungen verteilen und ob die Ressourcenanforderungen der tatsächlichen Nutzung entsprechen. So lassen sich überprovisionierte Cluster entdecken oder dauerhafte Überprovisionierungen, die auf einmalige Lastspitzen zurückzuführen sind. Solche Spitzen sind typische Kostentreiber, weil die Teams dann zusätzliche Kapazitäten vorhalten oder weil eine automatische Skalierung ausgelöst wird.

Durch Kostenzuordnungen und Prognosen wird diese Transparenz praktisch nutzbar. Kostenzuordnungen schreiben den Verbrauch den verantwortlichen Teams bzw. Services zu. Prognosen machen auf der Grundlage der Verbrauchsmuster Vorhersagen über die künftigen Ausgaben. Beides hilft den Teams, von der reinen Kostenreaktion zu einer belastbaren Bedarfsplanung zu gelangen. Sie können präziser budgetieren und ihre Skalierungen am realen Bedarf ausrichten.

Ein Beispiel: Das DevOps-Team entdeckt einen unerwarteten Anstieg der Cloud-Ausgaben bei einem bestimmten Namespace. Mithilfe eines Kubernetes-fähigen Observability-Tools kann es diesen Anstieg auf einen längst vergessenen Testservice zurückführen, der immer weiter skaliert wurde. Die Workload wurde einfach nie beendet. Durch die Kostenzuordnung und mit Effizienzmetriken kann das Team das Problem lokalisieren, die Workload stilllegen und solche Mehrverbräuche in Zukunft unterbinden.

Die **Kostenzuordnung** stellt dar, was läuft, wo die Verantwortung dafür liegt und was es kostet. Wenn die Teams bestimmte Services auf deren tatsächliche Eigentümer zurückverfolgen können, dann wird es einfacher, die Umgebung um nicht mehr benötigte Services zu bereinigen. Sie müssen dann nicht mehr rätselhafte Workloads haschen, sondern können der Spur folgen und in Ordnung bringen, was Ärger macht – etwa indem sie den Verbrauch anpassen, den Budgetrahmen reduzieren oder manches komplett einstellen.

Derartige Probleme treten meist in Umgebungen auf, in denen Entwicklungs- und Engineering-Teams autonom arbeiten und ihre Services selbstständig skalieren können. Ohne zentrales Reporting und ohne Budgetkontrolle laufen unnötige Ausgaben dann wochen- oder gar monatelang vor sich hin. Um dies zu verhindern, braucht es Transparenz auf Workload-Ebene mit korrelierten Verantwortlichkeiten von Teams und Anwendungen.

Effektive Transparenz bedeutet, dass Systeme und ihre Daten drei Merkmale erfüllen. Sie sind:

- **Zugänglich** – alle können die Daten einsehen, die sie für ihre Arbeit brauchen.
- **Kontextualisiert** – Erkenntnisse sind auf den tatsächlichen Verbrauch bzw. die jeweiligen Kosten bezogen.
- **Umsetzbar** – den Teams ist klar, was als Nächstes zu tun ist.

Prognosen erleichtern den Teams vorbeugende Maßnahmen. Durch Analysen des bisherigen Verbrauchs können Engineering- und Operations-Teams saisonale Spitzen, Trends und Ausreißer erkennen, sodass sie einfacher Grenzwerte festlegen, Kapazitäten planen und Probleme erkennen können, ehe unnötige Kosten auflaufen.

Signale wie das Risiko von Ressourcenmangel, die Effizienzrate und Indikatoren ungenutzter Ressourcen können helfen, versteckte Ineffizienzen aufzudecken. Diese Signale sind besonders nützlich, wenn sie aggregiert und gewichtet werden. Manche Plattformen fassen dies sogar auf Cluster- oder Namespace-Ebene zusammen und heben so hervor, was zuerst Aufmerksamkeit erfordert.

Reines Monitoring ist allerdings nicht genug. **Transparenz** sollte Informationen auch zugänglich machen, kontextualisieren und umsetzbar machen. Ein komplexes Dashboard mit Hunderten von Diagrammen ist ohne klare Eigentümer und ohne Priorisierung nur ein weiteres Bild vom Rauschen.

Transparenz fördert auch die Eigenverantwortlichkeit. Wenn die Teams auf Kostenaufstellungen, Effizienztrends und Ressourcenverbrauch zugreifen können, können sie besser begründet abwägen. Dann stellt ein Team vielleicht fest, dass seine aggressive Autoskalierungsrichtlinie zwar für hohe Zuverlässigkeit sorgt, aber zu einem Preis, der untragbar ist. Mit den richtigen Daten kann das Team beurteilen, ob Kosten und Performance ausbalanciert sind.

Kosteneffektive Kubernetes-Observability bedeutet letztlich immer gemeinsames Verständnis: Transparenz-Tools sollten den Teams von Engineering, Operations und Betriebswirtschaft helfen, den Verbrauch der Infrastruktur an den Geschäftszielen und den Budget-Gegebenheiten auszurichten. Das heißt: Es muss offen über Erwartungen und über die Workload-Performance gesprochen werden – und darüber, in welchen Bereichen eine Optimierung am meisten bewirkt. Wenn die Teams dieselben Signale und Erkenntnisse vor sich haben, fällt es leichter, Prioritäten abzustimmen, Verschwendung einzudämmen und besser zusammenzuarbeiten.

Risiko von Ressourcenmangel (Starvation Risk): Dieses Signal zeigt auf Grundlage historischer Verbrauchsmuster an, wie wahrscheinlich es ist, dass einer Workload die Rechen- oder Speicherressourcen ausgehen, was Performance-Einbußen, Fehler oder Instabilität zur Folge haben kann.

Effizienzrate (Efficiency Rate): Diese Metrik gibt an, welchen Anteil der angeforderten bzw. zugewiesenen Ressourcen eine Workload tatsächlich nutzt. Eine niedrige Effizienzrate ist ein Marker von Verschwendung und bedeutet, dass Sie Ressourcen bezahlen, die die Workload gar nicht verbraucht.

Indikatoren ungenutzter Ressourcen (Idle Resource Indicators): Diese Anzeichen deuten auf Ressourcen hin (Knoten, Pods oder bestimmte angeforderte Ressourcen), die zwar bereitgestellt werden und Kosten verursachen, aber nicht aktiv genutzt werden oder längere Zeit deutlich unterfordert sind.

Phase 2

Optimierung: Auf der Suche nach dem Gleichgewicht

Wenn die Teams einen klaren Überblick über ihre Kubernetes-Umgebungen haben, besteht die nächste Herausforderung darin, Erkenntnisse in intelligente, messbare Verbesserungen umzusetzen. Anstatt nur auf Kostenspitzen oder Leistungsprobleme zu reagieren, können die Teams wiederholbare Prozesse einrichten und ihre Workloads gezielt optimieren.

Den Balanceakt dürfen Sie sich wie eine Wippe vorstellen: Am einen Ende sitzt die **Performance.** Dazu gehören Reaktionsfähigkeit, Verfügbarkeit und alle Komponenten der Ausfallsicherheit, was meist nach Uptime oder anhand der Service-Erwartungen bemessen wird. Am anderen Ende nimmt die **Kostenkontrolle** Platz. Überprovisionierung sorgt zwar für stabile Services, vergeudet aber Rechenleistung und Speicherplatz. Unterprovisionierung spart Kosten, birgt aber das Risiko von Latenzen, Performance-Einbußen und Ausfällen.

Das richtige Gleichgewicht lässt sich nur im Kontext finden. Wie wichtig ist eine bestimmte Workload? Wer ist darauf angewiesen? Welches Leistungsniveau ist akzeptabel, und was sind Sie bereit, dafür auszugeben?

In Unternehmen ohne klare Optimierungssignale werden Entscheidungen über Skalierung und Feinabstimmung meist von Gewohnheiten bestimmt, von Befürchtungen oder vom Druck, Ausfälle „um jeden Preis“ zu vermeiden. Die Teams neigen dann dazu, die CPU- und Speicherzuweisungen von Haus aus aufzudoppeln, um nur ja nicht das Risiko einer Unterprovisionierung einzugehen. Dieser Ansatz ist jedoch kaum skalierbar, insbesondere in großen Umgebungen, wo sich kleine Ineffizienzen rasch aufschaukeln.

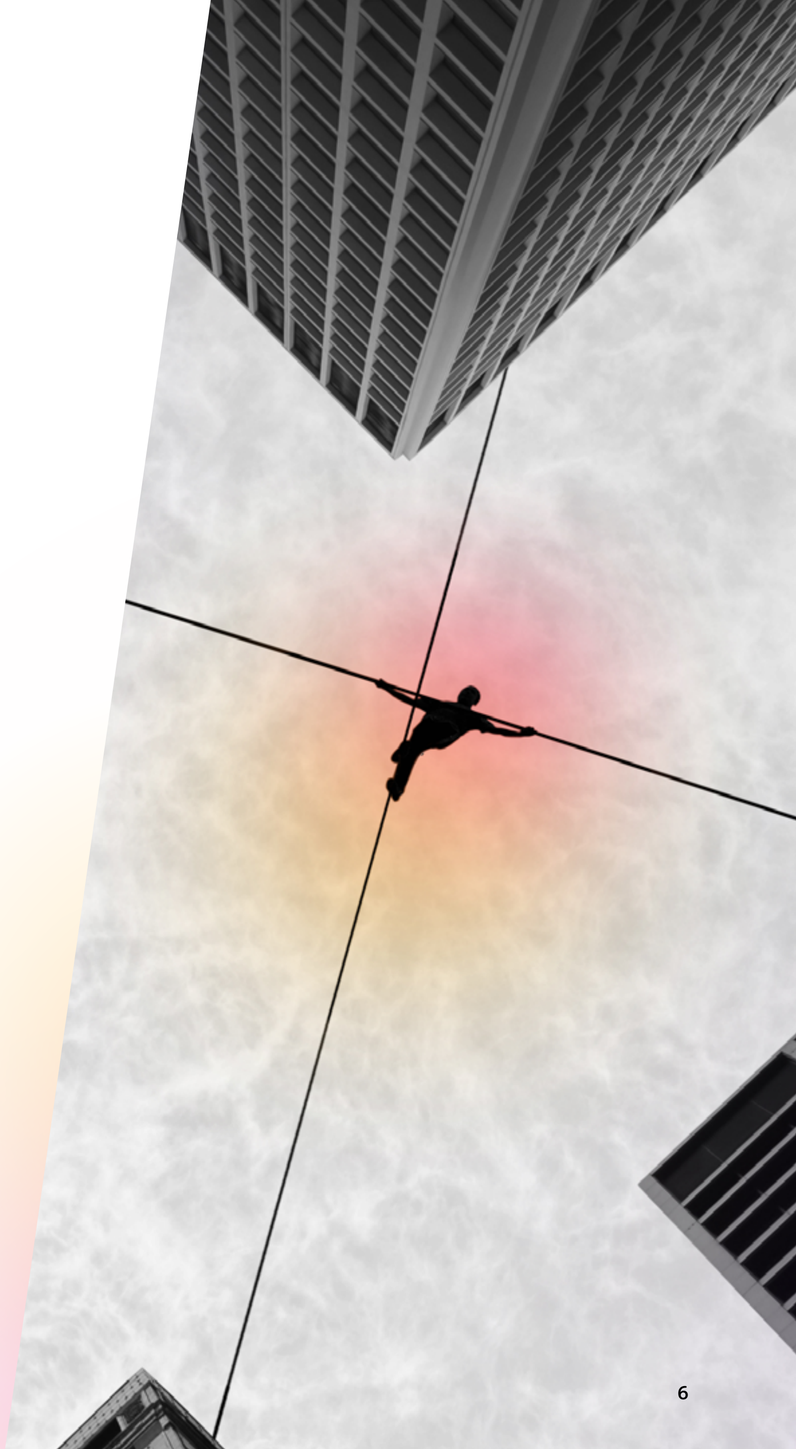
Das **historische Workload-Verhalten** liefert den entscheidenden Kontext für eine bessere Entscheidungsfindung. Die Teams können Verbrauchsmuster, Ladespitzen und vergangene Incidents analysieren und daraus besser durchdachte Skalierungsrichtlinien entwickeln. Plattformen, die solche Daten integrieren, können auch Empfehlungen geben oder Risikoindikatoren liefern, die bei der Gewichtung der Optimierungsmaßnahmen nach Dringlichkeit helfen.

Ein Team, das eine Workload mit Stapelverarbeitung betreut, könnte so z. B. feststellen, dass zwischen der täglichen Ausführung des Jobs erhebliche Leerlaufzeiten liegen. Dann ist es sinnvoller, nicht rund um die Uhr fixe Ressourcen vorzuhalten, sondern den Service so zu konfigurieren, dass er nach Ausführung herunterskaliert und erst bei Bedarf wieder neu provisioniert wird. Solche Entscheidungen erwachsen aus dem Zusammenspiel von gründlicher Observability, historischen Trends und verlässlicher Automatisierung.

Manche Plattformen bieten heutzutage auch **Optimierungsempfehlungen** auf der Basis des tatsächlichen Verbrauchs. Derartige Empfehlungen schlagen eine angemessene Dimensionierung der Workloads vor, um die angeforderten CPU- und Speicherressourcen dem tatsächlichen Bedarf anzupassen. Sie markieren auch Workloads, bei denen akuter Ressourcenmangel droht, sowie Workloads, die schlicht überdimensioniert sind.

Fragen Sie sich:

- Ist die Workload geschäftskritisch?
- Bezahlen wir für Leistung, die eigentlich unnötig ist?
- Können wir wenig genutzte Services konsolidieren oder ganz einstellen?



Erfolgsentscheidend ist die **Priorisierung.** In großen Kubernetes-Umgebungen können die Teams nicht alles auf einmal optimieren. Mit Observability-Tools, die aufzeigen, wo die größten Möglichkeiten liegen, können die Teams ihre Engineering-Zeit auf das konzentrieren, was am meisten bringt. Das können z. B. Workloads sein, die offensichtlich nicht ausgelastet sind oder unverhältnismäßig hohe Kosten verursachen. In überschaubaren Umgebungen können die Teams die Optimierung manchmal direkter angehen, aber auch hier gilt: Starke Signale beschleunigen die Arbeit und sorgen für bessere Entscheidungen.

Optimierung bedeutet Anpassung auf Grundlage der tatsächlichen Nutzung der Infrastruktur. Es geht darum, sicherzustellen, dass jede Workload die Kosten, die sie aufwirft, auch wert ist. Die Teams sollten sich z. B. fragen: Bezahlen wir für eine Performance, die die User auch wahrnehmen? Können wir die Zuverlässigkeit auch mit weniger Ressourcen aufrechterhalten? Gibt es Services, die wir konsolidieren oder ganz einstellen sollten?

Zuverlässigkeit und Effizienz ins Gleichgewicht zu bringen, ist eine Aufgabe, die Zusammenarbeit erfordert. Die Leute aus SRE, Entwicklung und Platform Engineering brauchen eine gemeinsame Sicht auf das Workload-Verhalten und auf den Ausgleich von Lasten und Kosten. Ohne gemeinsamen Kontext besteht das Risiko, dass die Optimierungsbemühungen in die Irre gehen oder ganz zum Erliegen kommen.

Ein Schichtenmodell für Anwendungen (Application Tiering) kann sich ebenfalls als hilfreich erweisen. Denn nicht jeder Service darf als geschäftskritisch gelten. Die Teams können ihre Workloads nach Business-Relevanz kategorisieren: von unverzichtbaren Systemen, die hohe Verfügbarkeit erfordern, bis hin zu Best-effort-Services, die gelegentliche Unterbrechungen vertragen können.

Das ermöglicht eine klügere Balance unter Einbeziehung der Kritikalität: Bei Anwendungen mit hoher Priorität sind vermutlich höhere Ausgaben für Performance und Redundanz gerechtfertigt, während sich die Workloads der unteren Schicht strengere Kontrollen gefallen lassen müssen.

Ebenso wichtig ist, dass die Teams erkennen, wann keine Optimierung erforderlich ist. Die verfügbare Zeit ist begrenzt, und es ist sinnvoll, sich auf die Bereiche zu konzentrieren, die besonders kostspielig oder unbedingt erfolgskritisch sind oder sich unmittelbar auf die User auswirken. Ein Reporting-Job, der im Hintergrund läuft, dessen Ausführung kaum Kosten verursacht und von dem die User nichts mitbekommen, wird vermutlich kein direktes Eingreifen erforderlich machen. Stattdessen sollten die Teams ihre Aufmerksamkeit den Services widmen, die umfangreich sind, sich auf die Kundschaft auswirken oder im Betrieb unverhältnismäßig teuer kommen.

Diese Balance entsteht nicht von allein. Der Ausgleich muss erst geschaffen werden – zwischen dem, was Sie brauchen, und dem, was Sie finanzieren können. Dazu braucht es menschliches Urteilsvermögen, klare Signale aus Ihren Umgebungen und den Mut, sich mit Dingen zu befassen, von denen andere lieber die Finger lassen.

Phase 3

Operations: Kubernetes auf Effizienzkurs halten

Mit einmaliger Optimierung ist es freilich nicht getan. Wenn die Workloads angepasst und die Ressourcen richtig dimensioniert sind, müssen die Teams dafür sorgen, dass die Verbesserungen auch auf lange Sicht Bestand haben. Kubernetes-Umgebungen verändern sich laufend. Ständige Updates, Verbrauchsmuster, die sich immer wieder ändern, und Team-Prioritäten, die sich weiterentwickeln, sind typische Merkmale. Die Änderungen scheinen zunächst geringfügig, doch wenn sie nicht mitverfolgt und gemanagt werden, können sie irgendwann die Vorteile früherer Optimierungen zunichte machen.

Beim dritten Schritt zu einem hohen operativen Reifegrad geht es also darum, die **Effizienz aufrechtzuerhalten.** Selbst bestens eingestellte Workloads gleiten nach und nach in die Ineffizienz ab, wenn sie nicht aufmerksam betreut werden. Das kann daran liegen, dass sich die Verbrauchsmuster ändern, das kann an zusätzlichen Funktionen liegen oder an Änderungen an ganz anderer Stelle der Infrastruktur. Um die einmal erzielten Vorteile zu sichern, braucht es dreierlei: intelligente Warnmeldungen, historischen Kontext und die Fähigkeit, Probleme rechtzeitig zu erkennen.

Die **Anomalieerkennung** spielt hier eine entscheidende Rolle. Wenn die Kosten sprunghaft steigen oder der Ressourcenverbrauch sich unerwartet ändert, brauchen die Teams Frühsignale. Das können z. B. Warnmeldungen sein, wenn ein Service plötzlich mehr Speicher belegt als die Normallinie verzeichnet oder wenn die Kosten eines Namespaces über die Schwellenwerte hinausgehen. Anhand solcher Warnmeldungen können die Teams den Sachverhalt überprüfen, bevor sich Abweichungen zu größeren Problemen auswachsen.

Damit dieser Prozess handhabbar bleibt, benötigen Teams eine Möglichkeit, das Rauschen auszufiltern, sodass sie sich auf die relevanten Trends konzentrieren können. Alarmmüdigkeit ist ein echtes Problem, vor allem bei komplexen Systemen. Am leichtesten haben es Teams mit Plattformen, die Signale mit hoher Relevanz hervorheben, z B. dass die Workload-Effizienz absinkt oder das Risiko von Ressourcenmangel steigt. Dann können sich die Fachleute direkt auf das konzentrieren, was zuerst Aufmerksamkeit erfordert.

Budgets und Kostenschwellen können die **Eigenverantwortung** stärken. Unternehmen, die ihren Teams und Projekten eigene Budgets zuweisen, schaffen quasi natürliche Kontrollpunkte. Wenn ein Team sein Budget überschreitet, muss das besprochen werden – nicht als Maßregelung, sondern als ein Anlass, Bereitstellungsentscheidungen oder Verbrauchsrichtlinien neu zu bewerten.

Es gibt Plattformen, die Workloads, Namespaces oder Cluster nach Ressourcenverbrauch, Risikosignalen und Effizienz **einstufen und bewerten.** So können die Teams leichter erfassen, worauf sie sich als Nächstes konzentrieren sollten, insbesondere in großen Umgebungen mit begrenzter Bandbreite. Ein Tracking nach Label oder Namespace macht es außerdem leichter, Verantwortlichkeiten zuzuweisen, Trends zu erkennen und sicherzustellen, dass der Verbrauch den Prioritäten des Geschäfts entspricht. Diese Struktur ist auch sinnvoll für strategische Diskussionen über Richtliniendurchsetzung, Automatisierung und Ressourcensteuerung.

Nachhaltige Optimierung bedeutet:

- Kontinuierliches Monitoring mit Warnmeldungen
- Regelmäßige Neubewertung der Workloads
- Teamübergreifende Zusammenarbeit und Verantwortung

Mit Frameworks, die auf Richtlinien aufsetzen, lässt sich leichter für Konsistenz sorgen. Wenn die Plattform z. B. feststellt, dass eine neue Workload mehr CPU-Leistung anfordert als standardmäßig vorgesehen ist, kann sie die Bereitstellung zur Überprüfung markieren oder – in strengeren Umgebungen – den Start unterbinden. Solche Mechanismen tragen dazu bei, dass alle Teams sich an die gemeinsamen Richtlinien halten und dass potenzielle Probleme frühzeitig erkannt werden. Dann bleibt die Effizienz gewahrt und die Entwicklungsgeschwindigkeit bleibt hoch.

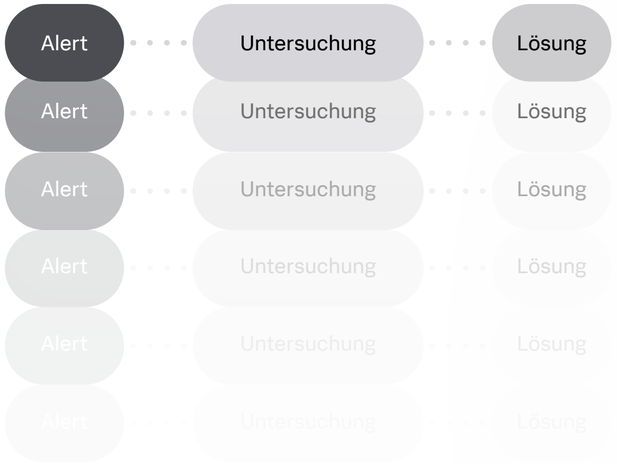
Auf lange Sicht hängt Effizienz von **kontinuierlichem Feedback** ab. Das Feedback sollte zeitnah, konkret und an die relevanten Metriken gebunden sein. Die Teams benötigen klare, konsistente Indikatoren, die ihnen zeigen, wie gut es gelingt, die Effizienz aufrechterhalten. Das System sollte dieses Bewusstsein durch Zusammenfassungen, Vergleiche und geeignete Erkenntnisse wach halten.

Betrachten wir einmal folgendes Szenario: Ein Team hat erst im letzten Quartal seinen teuersten Service optimiert. Ohne kontinuierliche Transparenz droht dieser Fortschritt aber wieder zu verfallen. Vielleicht fügt ein Team eine neue Funktion hinzu – und damit neue Last. Und ein anderes Team aktualisiert die Abhängigkeiten oder entfernt das Autoscaling. Die einzelnen Änderungen erscheinen gering, aber die Auswirkungen summieren sich. Um die Effizienz aufrechtzuerhalten, müssen diese Veränderungen frühzeitig erkannt werden und die Teams müssen mit derselben Umsicht darauf reagieren wie bei der ursprünglichen Optimierung.

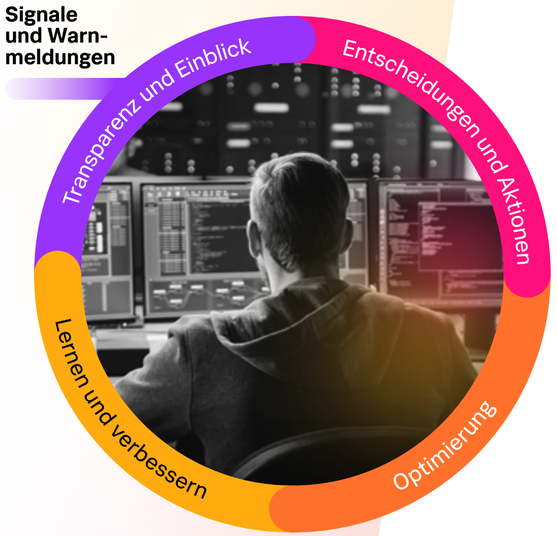
Es lohnt sich auch, frühere **Annahmen regelmäßig zu überprüfen.** Es kann z. B. sein, dass Workloads, die einmal für Spitzenverbrauch optimiert wurden, nun eine andere Funktion erfüllen oder mit anderen Lademustern arbeiten müssen. Eine viertel- oder halbjährliche Revision solcher Entscheidungen kann den Teams helfen, Diskrepanzen frühzeitig zu erkennen und kostspielige Versehen zu vermeiden.

Auch in dieser Phase ist **Zusammenarbeit** wichtig. Die Teams arbeiten zwar oft für sich, doch die operative Effizienz liegt in geteilter Verantwortung. Das Plattform-Team kann z.B. allgemeine, clusterübergreifende Trends aufzeigen, während die für die einzelnen Services Zuständigen den Kontext liefern, der für die Interpretation und für Maßnahmen notwendig ist. Die Tools und Prozesse zur Effizienzsteigerung sollten also nicht nur die Transparenz fördern, sondern auch die Koordination.

Klassisches Monitoring



Moderne Observability



Diese Phase führt uns natürlich wieder zum Anfang des Zyklus zurück. Jede Erkenntnis, jede Warnmeldung und jede Entscheidung geht in die nächste Transparenzrunde ein. Observability ist der rote Faden, der sich durch alle Phasen dieser Reise zieht, von der ersten Erkenntnis bis zur kontinuierlichen Optimierung. Langfristig erfolgreich sind diejenigen Teams, die Optimierung nicht als Ziel, sondern als Teil ihres Tagesrhythmus betrachten.

Fazit

Wer Kubernetes-Umgebungen effizient, zuverlässig und kostenbewusst betreiben will, braucht nicht nur Transparenz, sondern Kontext, operative Disziplin und den Willen zu kontinuierlicher Verbesserung. Dieses E-Book hat uns durch einen Zyklus geführt, der drei wesentliche Phasen durchläuft: **Transparenz, Optimierung** und **Operations.** Diese drei bilden zusammen die Grundlage eines verantwortungsvollen Kubernetes-Betriebs.

Transparenz ist die erste Phase und wohl auch die wichtigste. Transparenz ist die Voraussetzung, dass die Teams überhaupt sehen können, wie Ressourcen genutzt werden, wo Kosten entstehen und wer verantwortlich ist. Transparenz geht Hand in Hand mit Observability, die Daten, Kontext und Signale liefert, sodass die Teams ihre komplexen Kubernetes-Umgebungen verstehen können.

Es folgt die zielgerichtete **Optimierung.** Die Teams tunen ihre Workloads, passen Konfigurationen an und richten die Infrastruktur am Bedarf aus. In dieser Phase bewerten die Teams die Performance in Relation zu den Kosten und zum Nutzen, den die Workloads bieten.

Bei den laufenden **Operations** geht es schließlich darum, Resilienz zu schaffen und dauerhaft für Effizienz zu sorgen. Mit durchgängigen Richtlinien, Anomalieerkennung und gemeinsamer Verantwortung können die Teams ihre Optimierungsgewinne sichern und schleichenden Kursabweichungen vorbeugen.

Wo Observability sorgfältig implementiert ist, schafft sie die Voraussetzungen dafür, dass die Teams auf Basis realer Informationen ihre Cluster, Pods und Nodes überprüfen, evaluieren und optimieren können. Nicht zuletzt ist Observability die Grundlage, auf der das Unternehmen weiter gefasste Herausforderungen bewältigt: die Stack-übergreifende Budgetierung, die Reduzierung von Workload-Silos und die Kostenoptimierung als stets gegenwärtige Priorität.

Ein solcher Ansatz ist heute wichtiger denn je. Da die Cloud-Kosten weiter wachsen und die Entwicklungsteams mit immer weniger Ressourcen immer mehr leisten müssen, wird es eine Führungsaufgabe, dass Sie die Performance Ihrer Infrastruktur und deren Kosten kennen. Ganz gleich, ob Sie eine Plattform, ein Budget oder ein Geschäftsergebnis verantworten – Sie müssen sich darauf verlassen können, dass Ihre Systeme zweckmäßig und effizient arbeiten.

Die Unternehmen, die in diesem Bereich führend sind, haben in der Regel einiges gemeinsam: Sie unterziehen die Workload-Effizienz regelmäßig einer Bewertung und konzentrieren sich auf Verbesserungen in Punkten, wo dem Geschäft am meisten gedient ist; die damit verbundenen Kompromisse gehen sie überlegt ein, speziell im Spannungsverhältnis zwischen Performance und Kosten. Diese Balance muss immer wieder neu gefunden werden. Teams mit transparentem Einblick in Verbrauch, Kosten und Performance können das Gleichgewicht effektiver und mit größerer Sicherheit herstellen.

Tools sind nur ein Teil der Lösung. Der wahre Nutzen entsteht erst, wenn Entscheidungen über die Infrastruktur im vollen Bewusstsein der Zugeständnisse bei Performance, Zuverlässigkeit und Kosten getroffen werden. Wenn Teams in die Lage versetzt werden, auf der Grundlage dieses Wissens zu handeln, können sie schneller reagieren, intelligenter skalieren und ihre Systeme effektiv kontrollieren, auch bei zunehmender Komplexität.

Nun liegt es an Ihnen, die Frage zu beantworten. Die Daten liegen vor, aber die eigentliche Herausforderung besteht darin, sie in umsetzbares Wissen zu verwandeln. Also: Verfügen Sie über die Transparenz und die operative Disziplin, sodass Sie wirklich wissen, warum Ihre Kubernetes-Workloads das kosten, was sie kosten? Oder lassen Sie zu, dass sich in der Komplexität Ineffizienzen einnisten, die still und heimlich Ihr Budget aufzehren?

Der Kubernetes-Kreislauf von Kosten und Performance:

- **Transparenz:** Kosten und Verbrauch verstehen.
- **Optimierung:** Nach Performance und Relevanz angemessen dimensionieren.
- **Operations:** Optimierungsvorteile sichern – mit Richtlinien, Warnmeldungen und Verantwortlichkeit.

Setzen Sie Ihre Kubernetes-Reise fort

Sie wissen nun, wie Sie die Balance von Kosten und Performance finden. Dann sehen Sie sich Ihr Kubernetes doch genauer an:

- Laden Sie den Leitfaden **Troubleshooting in Kubernetes-Umgebungen** herunter und erfahren Sie, wie Observability bei der Diagnose, bei Behebung und Vermeidung kritischer Probleme hilft.
- Entdecken Sie **Kubernetes-Monitoring von Splunk**.

Starten Sie Ihre kostenlose Testversion von Splunk Observability Cloud.
Schaffen Sie gründliche Transparenz bei Kubernetes-Kosten und -Performance und finden Sie heraus, wo sich Optimierungen am meisten lohnen.

splunk> a CISCO company

25_CMP_ebook_How-well-do-you-understand-your-kubernetes-workloads_v6_GER