

Laboratuvar: İzleme için Kubernetes Yapılandırması

Gerekli Tahmini Süre: **45 dk**

Başlarken:

Kubernetes'te İzleme Anlayışı: İzleme, bireysel isteklerin sistemde nasıl hareket ettiğine dair detaylı bilgileri yakalar ve depolar. Bu, isteklerin işlenmesi için geçen süre, isteği işleyen bileşenler ve yol boyunca meydana gelen hatalar hakkında bilgileri içerir. Bu bilgiler daha sonra toplanır ve analiz edilir, böylece geliştiricilerin ve DevOps ekiplerinin uygulama performansını optimize etmelerine yardımcı olabilecek desenler ve eğilimler belirlenir.

Öğrenme Hedefleri:

Bu laboratuvar çalışmasında, Kubernetes'te izleme hakkında bilgi edineceksiniz. Geliştiricilere ve DevOps ekiplerine uygulama performansını optimize etmelerine yardımcı olmak için ayrıntılı bilgileri yakalamayı, depolamayı ve desenler ile eğilimleri tanımlamayı öğreneceksiniz.

Bu laboratuvarı tamamladıktan sonra şunları yapabileceksiniz:

- Kubernetes'te OpenTelemetry izlemeyi uygulamak ve Kubernetes'in verimliliğini maksimize etmek.
- Kubernetes ile OpenTelemetry kullanımı için en iyi uygulamaları uygulamak.
- İzleme için OpenTelemetry'nin faydalarını analiz etmek.

OpenTelemetry izleme sistemini Kubernetes ortamında şu adımları takip ederek uygulayın:

Adım 1. OpenTelemetry Collector'ı Kurun:

Kubernetes kümeniz içinde OpenTelemetry Collector'ı kurarak başlayın. Bu, Helm, Kubernetes manifestleri veya OpenTelemetry Operator kullanılarak yapılabilir.

Adım 2. Collector'ı Yapılandırın:

Collector kurulduktan sonra, uygulamalarınızdan izleri toplamak ve bunları tercih ettiğiniz izleme arka ucuna göndermek için yapılandırın. Bu, istenen ihracatçılar, alıcılar ve işlemcileri belirten bir yapılandırma dosyası oluşturmayı içerir.

İzleme için bir OpenTelemetry Collector yapılandırma dosyasına örnek aşağıda gösterilmiştir.

```
receivers:
  otlp:
    protocols:
      grpc:
exporters:
  jaeger:
    endpoint: <jaeger-endpoint>
    insecure: true
processors:
  batch:
extensions:
  health_check:
service:
  pipelines:
    traces:
      receivers: [otlp]
      processors: [batch]
      exporters: [jaeger]
```

Bu kodda, yapılandırma dosyası, OpenTelemetry Protokolü (OTLP) kullanarak Google Uzak Prosedür Çağrıları (gRPC) aracılığıyla izleri izleyen bir alıcıyı belirtir. Dosya ayrıca izleri bir Jaeger arka uca ileten bir Jaeger dışı aktarıcısını da tanımlar. İzleri arka uca göndermeden önce birleştirerek, toplu işlemci toplayıcının hızını artırmaya yardımcı olur.

Adım 3. Uygulamalarınızı Aletlendirin:

Uygulamalarınızı izleri oluşturmak için OpenTelemetry SDK'sı veya uyumlu bir izleme kütüphanesi ile aletlendirin. Bu, uygulamalarınıza span'ler oluşturmak ve bunları iz bağlamına eklemek için kod eklemeyi içerir.

İşte OpenTelemetry SDK'sı ile basit bir Go uygulamasını aletlendirmek için bir örnek.

```
package main
import (
    "context"
    "go.opentelemetry.io/otel"
    "go.opentelemetry.io/otel/trace"
)
func main() {
    // Create a tracer provider with the default configuration
    provider := otel.GetTracerProvider()
    // Get a tracer instance from the provider
    tracer := provider.Tracer("my-app")
    // Create a span
    ctx, span := tracer.Start(context.Background(), "my-span")
    defer span.End()
    // Do some work
}
```

Bu kodda, OpenTelemetry SDK kullanılarak bir izleyici sağlayıcı ve bir izleyici örneği oluşturulmaktadır. İş, yeni oluşturulan span çerçevesinde yapılmaktadır.

İzlerin arka uca iletildiğinden emin olun. Arka ucunuz tarafından sunulan izleme arayüzü, uygulamalarınızın izleri ürettiğini ve bunları oraya gönderdiğini doğrulamak için de kullanılabilir. Bu, izleri görmenizi ve performans ile diğer sorunları tespit etmenizi sağlar.

Adım 4. İzleri Doğrulayın:

Uygulamalarınızın izler ürettiğinden ve bunları arka uca gönderdiğinden emin olun. Seçtiğiniz arka ucun sağladığı izleme arayüzünü kullanarak izleri görselleştirebilir ve performans darboğazları ile diğer sorunları belirleyebilirsiniz.

Kubernetes ile OpenTelemetry Kullanımı için En İyi Uygulamalar:

1. Uygulama Mimarisi Hakkında Bilgi Edinin:

OpenTelemetry izlemeyi uygulamadan önce, mikro hizmetler, API'ler ve bağımlılıklar dahil olmak üzere uygulamanızın mimarisi hakkında net bir anlayışa sahip olun. Bu, izleme açısından önemli olan ana alanları belirlemeye yardımcı olur.

2. Hedefleri Belirleyin:

İzleme çabalarınızın hedeflerini net bir şekilde tanımlayın, örneğin darboğazları belirlemek veya hizmet sağlığını izlemek gibi. Bu, izleme enstrümantasyon stratejinizi ve arka uç seçimlerinizi şekillendirmeye yardımcı olur.

3. OpenTelemetry En İyi Uygulamalarını Takip Edin:

OpenTelemetry tarafından önerilen en iyi uygulamalara uyun, örneğin izleme öznetelikleri için anlamsal standartlar kullanmak ve aşırı enstrümantasyondan kaçınmak.

4. Dağıtık İzleme Uygulayın:

Mikro hizmetler, API'ler ve bağımlılıklar dahil olmak üzere tüm uygulama yığınızda dağıtık izlemeyi kullanarak performansın kapsamlı bir görünümünü elde edin.

5. Doğru İzleme Arka Uçunu Seçin:

İhtiyaçlarınıza uygun ve izleme hedefleriniz için gerekli işlevselliği sağlayan bir izleme arka ucu seçin. Seçenekler arasında Jaeger, Zipkin ve AWS X-Ray bulunmaktadır.

6. İzleme Verilerini Düzenli Olarak İzleyin ve Analiz Edin:

OpenTelemetry tarafından toplanan izleme verilerini kullanarak uygulamanızın performansını düzenli olarak izleyin ve iyileştirme alanlarını belirleyin.

Kubernetes'te OpenTelemetry izlemeyi uygulayarak, istek yürütmesi hakkında ayrıntılı veriler toplayabilir, sistemin görünürlüğünü artırabilir ve Kubernetes ortamınızın verimliliğini optimize edebilirsiniz. Bu, nihayetinde kullanıcı deneyimini geliştirmeye ve uygulama arzısı risklerini azaltmaya yol açar.

Kubernetes Kullanmanın ve OpenTelemetry (OTEL) ile Gözlemlenebilirliğin Faydaları:

1. Geliştirilmiş Performans:

OpenTelemetry izleme, performans sorunlarını erken aşamada tanımlayıp çözmenizi sağlar. İstek yürütmesi hakkında ayrıntılı veriler toplayarak ve tüm sisteme görünürlük sağlayarak, Kubernetes ortamınızın performansını optimize edebilirsiniz. Bu, uygulama yanıt sürelerinin ve genel sistem verimliliğinin artmasına yol açar.

2. Geliştirilmiş Kullanıcı Deneyimi:

Performansı optimize ederek ve darboğazları azaltarak, OpenTelemetry izleme kullanıcı deneyimini iyileştirir. Kullanıcılar, Kubernetes üzerinde çalışan uygulamalarınızla daha hızlı yanıt süreleri ve daha akıcı etkileşimler deneyimler.

3. Azaltılmış Uygulama Hataları:

OpenTelemetry tarafından sağlanan gözlemlenebilirlik ile potansiyel sorunları uygulama hatalarına neden olmadan proaktif bir şekilde tespit edip çözebilirsiniz. İzleme ve analiz verilerini inceleyerek, potansiyel hataları veya performans düşüşlerini gösteren desenleri ve eğilimleri belirleyebilirsiniz. Bu, uygulama kesintilerini önlemeye ve Kubernetes dağıtımlarınızın istikrarını sağlamaya yardımcı olur.

4. Standartlaştırılmış Telemetri Toplama:

OpenTelemetry, izler, metrikler ve günlükler dahil olmak üzere telemetri verilerini toplamak ve dış aktarmak için standart bir yol sağlar. Bu, diğer gözlemlenebilirlik araçları ve platformları ile kolay entegrasyon sağlamakta ve sisteminizin performansına kapsamlı bir bakış sunmaktadır.

5. Tedarikçi Esnekliği:

OpenTelemetry, tedarikçi bağımsızdır ve çeşitli izleme ve izleme platformları ile entegre edilebilir. Bu esneklik, ihtiyaçlarınıza en uygun izleme arka ucunu seçmenizi sağlar, tedarikçi kilitlenmesini önler ve mevcut araç setinizle uyumluluğu garanti eder.

6. Kubernetes Ekosistemi ile Sorunsuz Entegrasyon:

OpenTelemetry, Prometheus, Jaeger ve Grafana gibi popüler Kubernetes araçları ve platformları ile kolayca entegre edilebilir. Bu entegrasyon, OpenTelemetry tarafından toplanan iz verilerinin görselleştirilmesi, izlenmesi ve analizi için bu araçların yeteneklerinden yararlanmanızı sağlar.

7. Ölçeklenebilirlik ve Dayanıklılık:

Kubernetes, konteynerleştirilmiş uygulamaları çalıştırmak için ölçeklenebilir ve dayanıklı bir altyapı sağlar. OpenTelemetry ile gözlemlenebilirlik birleştirildiğinde, uygulamalarınızın performansına ölçekle ilgili içgörüler elde edebilirsiniz. Bu, potansiyel darboğazları tanımlamaya ve dinamik ve dağıtık bir Kubernetes ortamında kaynak kullanımını optimize etmeye yardımcı olur.

Özetle, OpenTelemetry izleme ile Kubernetes kullanmanın birçok faydası vardır; bunlar arasında geliştirilmiş performans, geliştirilmiş kullanıcı deneyimi, azaltılmış uygulama hataları, standartlaştırılmış telemetri toplama, tedarikçi esnekliği, Kubernetes ekosistemi ile sorunsuz entegrasyon ve uygulamalarınız için ölçeklenebilirlik ve dayanıklılık bulunmaktadır.

Kubernetes ve OpenTelemetry (OTEL) ile Gözlemlenebilirlik Uygulamaları:

Kubernetes ve OpenTelemetry (OTEL) ile gözlemlenebilirlik uygulamaları, özellikle modern yazılım geliştirme ve dağıtım bağlamında çeşitli senaryolar için geçerlidir. İşte Kubernetes ve OpenTelemetry ile gözlemlenebilirliğin bazı özel uygulamaları:

1. Mikroservis Mimarisi:

Kubernetes, mikroservisleri dağıtmak için ölçeklenebilir ve konteynerleştirilmiş bir ortam sağlar. OpenTelemetry ile gözlemlenebilirliği entegre ederek, bireysel mikroservislerin performansı ve etkileşimleri hakkında içgörüler elde edebilir, darboğazları tanımlayabilir, kaynak tahsisini optimize edebilir ve mikroservis tabanlı uygulamanızın sorunsuz çalışmasını sağlayabilirsiniz.

2. Dağıtık Sistemler:

OpenTelemetry izleme, Kubernetes tarafından düzenlenen dağıtık sistemlerde özellikle faydalıdır. Farklı bileşenler ve hizmetler arasında akış halindeki istekleri izlemeyi sağlar; bu, fiziksel konumları veya altyapıları ne olursa olsun. Bu, dağıtık uygulamaların uçtan uca performansını anlamaya, sorunları gidermeye ve genel sistemi optimize etmeye yardımcı olur.

3. Performans Optimizasyonu:

OpenTelemetry ile gözlemlenebilirlik, Kubernetes üzerinde çalışan uygulamalarınızın performansını izleyip analiz etmenizi sağlar. Telemetri verilerini toplayıp analiz ederek, performans darboğazlarını belirleyebilir, kaynak kullanımını optimize edebilir ve uygulamalarınızı en iyi performansı sağlamak için ince ayar yapabilirsiniz.

4. Uygulama İzleme ve Sorun Giderme:

Kubernetes ve OpenTelemetry, uygulamaları gerçek zamanlı olarak izlemek ve sorun gidermek için gerekli araçları sağlar. Metrikleri, izleri ve günlükleri toplayarak, uygulamalarınızın davranışını görselleştirebilir, anormallikleri tespit edebilir ve sorunları teşhis edebilirsiniz. Bu, proaktif izleme ve hızlı sorun giderme imkanı sunarak uygulama güvenilirliğini artırır.

5. Otomatik Ölçeklendirme ve Kaynak Tahsisi:

Kubernetes, talebe dayalı otomatik ölçeklendirme ve kaynak tahsisi için yerleşik özellikler sunar. OpenTelemetry ile gözlemlenebilirlik birleştirildiğinde, uygulama altyapınızı ölçeklendirmek için veri odaklı kararlar alabilirsiniz. Telemetri verilerini analiz ederek, optimal kopya sayısını belirleyebilir ve değişen iş yüklerini karşılamak için kaynakları verimli bir şekilde tahsis edebilirsiniz.

6. Sürekli Entegrasyon ve Dağıtım (CI/CD):

Kubernetes ve OpenTelemetry, CI/CD süreçlerinde kritik bir rol oynar. Konteyner düzenlemesi için Kubernetes'i ve gözlemlenebilirlik için OpenTelemetry'yi kullanarak, uygulamaların CI/CD süreci boyunca kapsamlı bir şekilde test edilmesini, izlenmesini ve optimize edilmesini sağlayabilirsiniz. Bu, uygulama kalitesini korumaya, sorunları erken tespit etmeye ve verimli dağıtım ve geri alma işlemlerine olanak tanır.

7. Bulut Yerel Uygulama Geliştirme:

Kubernetes ve OpenTelemetry, bulut yerel uygulamaların inşasında ana bileşenlerdir. Ölçeklenebilir, dayanıklı ve kolayca yönetilebilen konteynerleştirilmiş uygulamaların geliştirilmesini ve dağıtımını sağlarlar. Gözlemlenebilirlikle, bulut yerel uygulamalarınızın dinamik, dağıtık ortamlarda optimal performans göstermesini sağlayabilirsiniz.

Bunlar, Kubernetes ve OpenTelemetry ile gözlemlenebilirliğin çeşitli uygulama senaryolarında nasıl uygulanabileceğine dair sadece birkaç örnektir. Kubernetes'in konteyner düzenleme yetenekleri ile OpenTelemetry'nin gözlemlenebilirlik çerçevesinin birleşimi, geliştiricilere ve operatörlere modern uygulamaları etkili bir şekilde inşa etme, izleme ve optimize etme yetkisi verir.

Özet:

Tebrikler! İzleme için Kubernetes Yapılandırmasını öğrendiniz.

Bu laboratuvar çalışmasında, Kubernetes'te izleme hakkında bilgi edindiniz. Ayrıca, ayrıntılı bilgileri yakalamayı ve depolamayı, geliştiricilere ve DevOps ekiplerine uygulama performansını optimize etmelerine yardımcı olabilecek desenleri ve eğilimleri tanımlamayı öğrendiniz.

Yazar(lar)
Pallavi Rai

Katkıda Bulunanlar
Anamika Agarwal
Shivam kumar



Skills Network