Tableau Server 확장성

Server 관리자를 위한 기술 가이드

Neelesh Kamkolkar
제품 관리자, 데이터 및 성능

목차

개요

많은 조직에서 Tableau는 성과 달성에 있어 매우 중요한 위치를 차지하고 있습니다. 기업의 많은 부서들이 데이터의 인사이트 발견을 통해 중요한 가치를 깨닫게 됨에 따라 IT 팀에서는 비즈니스 부서와 협력하여 기업 전반의 분석 플랫폼으로 Tableau를 제공하고 있습니다. 기업에서 이러한 용도로 Tableau를 배포하기 시작하면서 기업 설계자와 IT 리더는 Tableau Server가 데이터와 콘텐츠, 사용자에 맞춰 확장되는 방식을 이해하고, 다양하고 이질적인 엔터프라이즈 IT 플랫폼에 Tableau Server를 통합 및 배포하는 방법을 파악하여 현재뿐 아니라 미래의 비즈니스 분석 요구를 지원하는 것이 매우 중요합니다.

이 백서는 기업 설계자와 IT 리더를 대상으로 하며, Tableau의 아키텍처와 증가하는 워크로드에 따른 Tableau의 확장 방식을 자세히 살펴봅니다.

Tableau에서는 Tableau Server 10 확장 방식과 그 결과가 이전 제품 버전과 어떻게 다른지 테스트했습니다. 고객의 요청에 따라 확장성 테스트 범위를 확장하여 백그라운드 워크로드와 사용자 확장성까지 포함했습니다.

시스템에는 Tableau Server의 성능과 확장성에 영향을 미칠 수 있는 여러 요소가 있습니다. 몇 가지 중요한 시스템 변수에 통합 문서 디자인, 서버 구성, 인프라 조정, 데이터 환경, 컴퓨팅 용량 및 네트워킹이 있습니다. 이러한 요소는 다양한 사용자 프로필 및 배포에 따라 가변성이 높습니다. 따라서 특정 확장성 테스트 결과는 이러한 변수의 조정 또는 변경에 따라 다를 수 있습니다. 변수를 줄이고 차단하기 위해 폐쇄된 네트워크 실험실의 실제 시스템에서 테스트를 수행함으로써 외부 시스템의 영향으로 인한 측정값의 가변성을 최소화하도록 노력하였습니다. 그런 다음 Tableau Server의 실제 사용을 모델링하여 확장성 메트릭을 측정하였습니다.

이를 위해 피크 사용량에서 Tableau Server의 실제 운용 환경 배포 분석으로 시작한 다음 자동화된 테스트로 이 사용량을 모델링했습니다. 이 2단계 접근법은 매우 현실적인 워크로드를 모방하며, 시스템에서 사용자 및 백그라운더 워크로드 (주로 데이터 추출 및 사용자 알림)가 적용될 때 실제적인 차이를 시뮬레이션합니다. 이러한 환경에서 비주얼라이제이션과 상호 작용할 때 사용자의 대기 시간 또는 일정에 따라 실행 중인 백그라운더 작업 수 등이 변수로 작용합니다.

실제 운용 조건을 모방하기 위한 테스트의 일환으로 차이를 모델링했고, Tableau Server 클러스터에 작업자 노드를 더 추가했을 때 시스템이 선형적으로 확장되는 것을 확인했습니다. 이 실험에서는 실제 운용 환경에서 확인된 조건을 상회하는 피크 수준으로 서버 로드를 증가시켰습니다. 실제 운용 환경의 사용량은 업무 시간 동안 빈번하게 짧은 주기로 피크에 이르는 경향이 있습니다. 실험 동안 처리량에 나타난 대로 로드를 측정했습니다. 처리량은 지정된 시간 동안 서버가 처리하는 작업량, 즉 초당 트랜잭션 수입니다. 아래 표시된 대로 실험에서 초당 4~18개의 트랜잭션까지 처리량이 확대된 것을 알 수 있습니다. 각 막대는 열 머리글에 설명된 토폴로지에서 실행된 실험을 나타냅니다(독립 실행형 서버, 주 서버 + 1 작업자 노드 등).

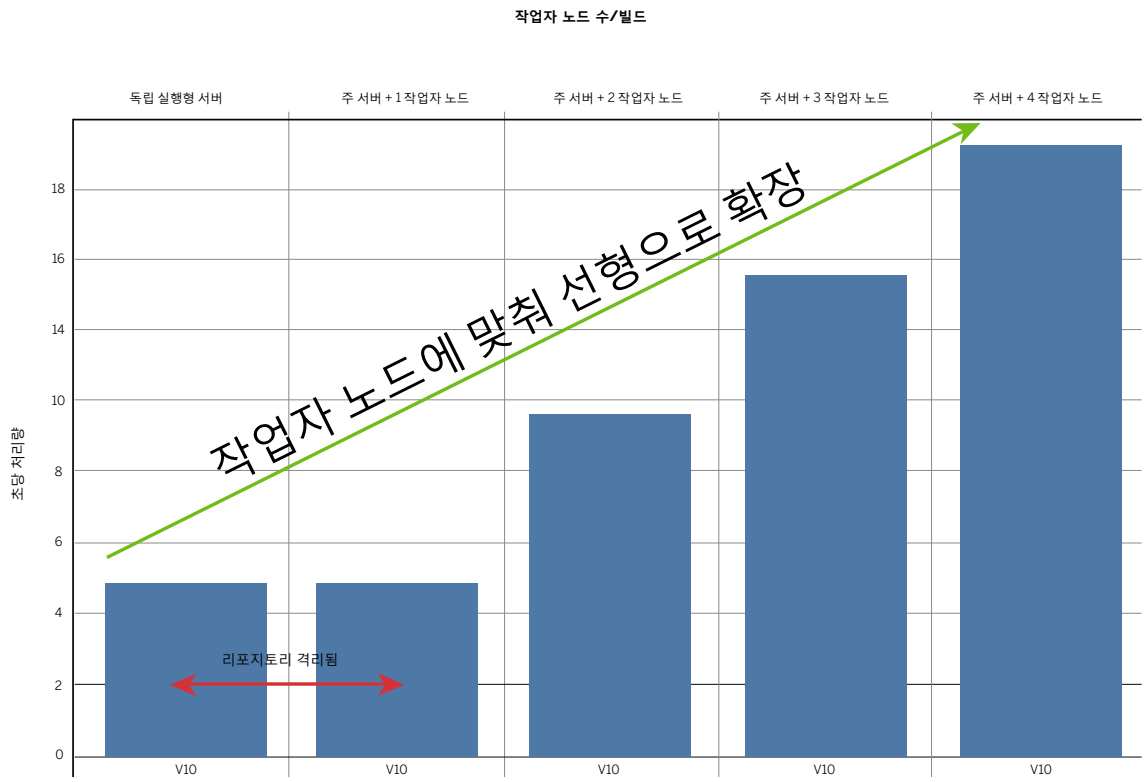

그림 1: 초당 시나리오 처리량

8코어의 단일 서버를 통해 처리할 수 있는 최대 활성 사용자 로드 수를 확인했습니다. 활성 사용자 로드는 Tableau Server에서 작동하는 자동화된 작업 집합을 말합니다. 활성 사용자 로드에 포함되는 작업에 대해서는 이 백서 후반에 자세히 설명되어 있습니다. 8코어의 Tableau Server에서 지원할 수 있는 최대 활성 사용자 수를 측정한 다음 이 활성 사용자 수를 확장 단위로 사용해서 동일한 8코어 작업자 노드를 클러스터에 추가하여 선형으로 로드를 증가시킬 수 있는지 테스트했습니다.

그 결과 단일 8코어 서버는 로드가 지속적으로 피크 수준을 유지하는 현실적인 모델링 상태에서 최대 112명의 활성 사용자를 지원할 수 있음을 알 수 있었습니다. 이 기준을 선형으로 확장하여, 즉 각각 8코어가 있는 4개의 동일 작업자 노드와 리포지토리 및 기본 설치만 된 주 컨트롤러 노드를 추가하여 448명의 활성 사용자를 지원할 수 있었습니다. 이 수는 시스템에서 동시에 작업 중인 사용자를 나타내므로 모든 사용자가 항상 시스템을 사용하는 것은 아니라고 추정하면 어디에서든 8~32코어가 장착된 클러스터 구성을 통해 500~10,000명의 사용자를 지원할 것으로 예상할 수 있습니다. 나중에 확인하겠지만 이러한 추정은 다양한 매개 변수에 따라 다릅니다. 다른 모든 변수가 동일한 상태에서 사용자 수에 따른 서버 크기 조정은 분석 사용과 데이터 조건에 따라 다릅니다. 테스트를 통해 나타난 결과는 Tableau Server 클러스터에 작업자 노드를 더 추가하는 방식의 Tableau 아키텍처를 사용하여 사용자를 선형으로 계속 확장할 수 있다는 것입니다.

이 테스트에서 오류를 반환하거나 시간 초과가 시작되는 지점을 찾기 위해 서버 로드를 계속 증가시켰습니다. 이러한 스트레스 테스트를 통해 서버의 확장성과 성능 한계를 정의하는 상한선을 알아냈습니다. 이 한계를 수량화할 때 운용 환경의 급격한 증가 로드 특성 분석을 넘어 지속적인 최대 로드까지 계속 테스트를 진행했습니다. 따라서 실제로 좀 더 안정적인 일반적인 상황에서는 이 백서에 제안된 사용량보다 더 많은 양을 플랫폼에서 지원합니다. Tableau Server 아키텍처는 선형으로 확장되므로 현재의 요구와 향후 성장을 충족하도록 확장 가능함을 확신할 수 있을 것입니다.

각 토폴로지와 사용 사례는 아래 테이블에 설명되어 있습니다. 위험 프로필 열에는 자동 페일오버, 하드웨어 장애에 대한 노출, 피크 로드 시 사용 가능한 여유 공간 등 다양한 요소를 토대로 한 토폴로지의 위험 노출 정도를 나타내는 변수가 설명되어 있습니다.

배포구성	사용 사례	위험 프로필
독립 실행형 서버	단일 서버 배포	가용성 위험 높음, 리소스 경합 높음, 최대 로드로 성능 문제가 있을 때 여유 공간 없음
1주 서버 + 1작업자 노드	2노드 배포	가용성 위험 높음, 리포지토리 사용으로 리소스 경합 낮음, 사용량 최고 시 여유 공간 없음
1주 서버 + 2작업자 노드	3노드 배포	가용성 위험 중간, 리소스 경합 낮음, 수평 확장 향상, 사용량 최고 시 여유 공간 거의 없음
1주 서버 + 3작업자 노드	4노드 배포	가용성 위험 중간, 리소스 경합 낮음, 확장 향상, 사용량 최고 시 여유 공간 적당
1주 서버 + 4작업자 노드	5노드 배포	가용성 위험 낮음, 확장 향상, 사용량 최고 시 여유 공간 충분[1]

테이블 1: 배포 토폴로지 및 사용 사례

[1]사용 가능한 여유 공간은 다양한 요소에 따라 다르며, 보장되지 않습니다.

이 수치는 방법론 및 테스트 맥락에서 이해하는 것이 중요합니다. 이를 위해 백서의 나머지 부분에서는 기존 BI 기술과 비교하여 Tableau 아키텍처의 차별성, 방법론 및 확장성 결과 분석에 대해 중점적으로 다룹니다.

오래된 패러다임을 깨는 VizQL

기존 비즈니스 인텔리전스(BI) 솔루션에 익숙하거나 Tableau를 처음 사용하는 경우 기존 BI와 비교하여 Tableau의

핵심적인 몇 가지 차이점을 이해하는 것이 좋습니다. 우리는 기존 BI의 "쿼리를 먼저하고, 시각화를 나중에"라는 방식에 근본적으로 이의를 제기해왔습니다. 대신 실시간으로 데이터를 탐색하고 질문하면서 인사이트를 발견한다고 믿습니다. Tableau에는 10년 넘게 쿼리와 시각화 작업을 단일 플랫폼에 결합하는 VizQL™이라는 특허 기술을 제공해왔습니다. 이렇게 강력한 기능의 결합으로 최종 사용자는 데이터를 시각화하면서 동시에 데이터에 대해 무제한의 질문, 즉 쿼리, 필터링 및 분석을 수행할 수 있습니다. VizQL™은 사용자 질문과 동작을 표현하는 Tableau의 기본 언어로서, 사용자 질문과 동작을 엔터프라이즈 환경이나 클라우드의 모든 데이터 집합에 대해 실행될 수 있는 쿼리로 변환합니다. 이 기술은 10년 이상의 엔지니어링 투자를 통해 발전해왔으며 차세대 데이터 원본과 분석 요구 사항을 지원하도록 계속 진화하고 있습니다.

사전 정의된 정적 요구 사항에 따라 설계 및 개발된 기존 BI 보고서와는 달리, Tableau 비주얼라이제이션은 상호 작용 및 협업할 수 있도록 만들어졌습니다. 사용자는 복잡한 SQL 쿼리나 조인을 작성하지 않아도 데이터에 대해 질문할 수 있습니다. 사용자는 질문에 대한 답을 얻기 위해 기존 소프트웨어의 또 다른 개발 단계를 기다릴 필요가 없습니다. 대신 기존 비주얼라이제이션을 반복하고 계속 분석을 수행하여 비즈니스나 프로젝트에 대한 질문의 답을 구할 수 있습니다.

기존 BI를 사용하는 경우 최적화가 미리 구성된 대상 시스템에서 실행되도록 쿼리가 설계된 특정 SLA(서비스 수준 계약)를 충족하는 로드 테스트 정적 보고서에 익숙할 수 있습니다. 정적 보고서는 범위와 쿼리 세트가 고정되어 있고 대부분 개발자에게 최적화되어 있으며 여러 주에 한 번 정도 생성됩니다. 정적 보고서를 위해 SLA를 정의하고 설정하는 일은 상대적으로 쉽지만 보고서를 변경하면 전체 개발 단계를 재설정해야 하는 애로사항은 SLA에 제대로 고려되지 않는 실정입니다.

반면에 Tableau 비주얼라이제이션은 사용자의 탐색 동작에 맞추어 새로운 쿼리를 다시 생성하거나 제출합니다. 데이터를 신속하게 검색할 수 있도록 하는 VizQL의 로컬 브라우저 캐싱 및 최적화 기능 덕분에 사용자는 쿼리 결과를 기다리는 대신 분석 흐름을 유지할 수 있습니다.

VizQL은 언어로서 VizQL 서버 프로세스에 포함되며 다른 다양한 분산 서버 프로세스와 조화를 이루어 확장 및 사용 가능하고 안전하며 안정적인 비즈니스 분석 플랫폼을 제공합니다.

Tableau 아키텍처

Tableau의 유연한 아키텍처는 확장이 용이하도록 설계되었습니다. 이 아키텍처를 통해 Tableau는 엔터프라이즈 표준 플랫폼 또는 클라우드 분석을 강화하는 플랫폼으로 실행될 수 있습니다. Tableau Server는 가장 복잡한 엔터프라이즈 운용 환경 인프라 요구 사항뿐 아니라 간단한 부서 또는 작업 그룹 수준 배포도 지원합니다.

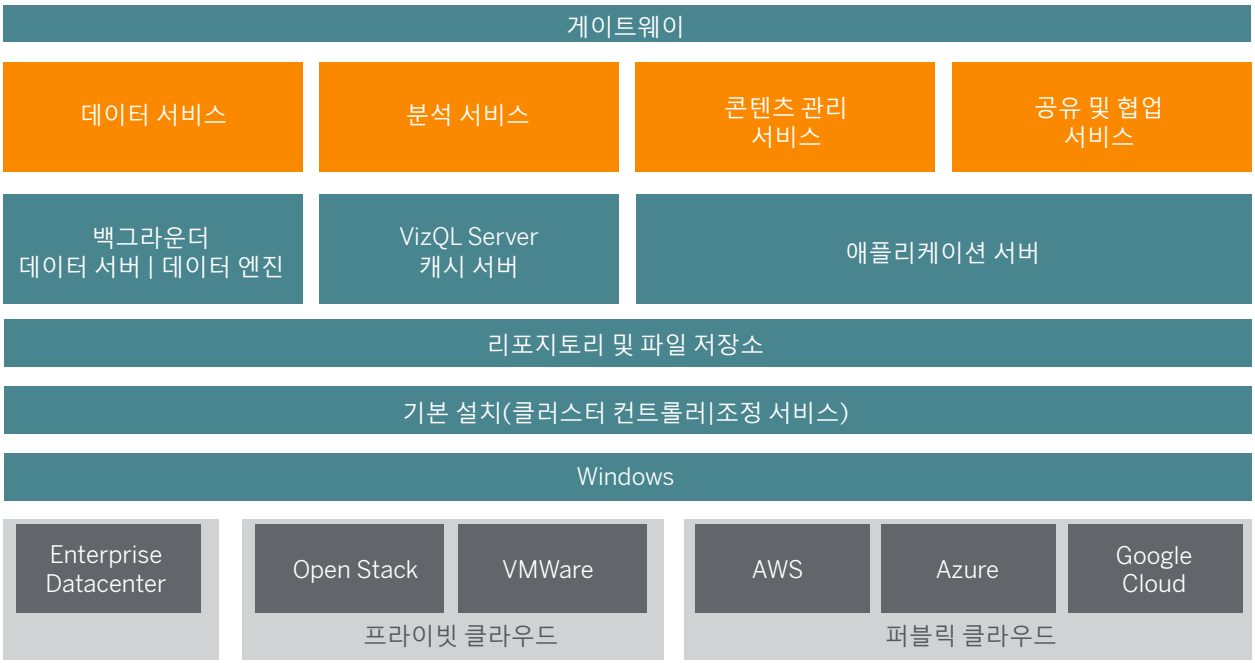

그림 2: Tableau Server 아키텍처

Tableau Server는 간단하게 설치 및 구성할 수 있습니다. 설치한 뒤에는 여러 서버 프로세스(그림 2에 파란색으로 표시)가 함께 작동하여 다양한 계층의 서비스를 제공합니다.

게이트웨이 프로세스는 모든 Tableau 클라이언트에서 발생하는 트래픽을 클러스터의 사용 가능한 서버 노드로 리디렉션하는 구성 요소입니다.

데이터 서비스는 데이터 최신성, 공유 메타데이터 관리, 관리되는 데이터 원본 및 인메모리 데이터를 제공하는 논리적으로 그룹화된 서비스입니다. 데이터 서비스를 운영하는 기본 프로세스는 백그라운더, 데이터 서버 및 데이터 엔진 프로세스입니다.

VizQL과 캐시 서버 프로세스로 구성된 분석 서비스는 사용자 관련 비주얼라이제이션 및 분석 서비스와 캐싱 서비스를 제공합니다.

콘텐츠 관리 서비스와 공유 및 협업 서비스는 애플리케이션 서버 프로세스로 운영됩니다. 사용자 로그인과 콘텐츠 관리(프로젝트, 사이트, 보안 권한 등) 및 관리 활동과 같은 핵심 Tableau Server 기능은 애플리케이션 서버 프로세스에서 제공합니다.

위의 모든 서비스는 리포지토리 프로세스를 기반으로 하여 사용됩니다. 이 프로세스에는 사용 권한, 통합 문서, 데이터 추출, 사용자 정보 및 메타데이터 등 구조화된 관계형 데이터가 포함됩니다. 파일 저장소 프로세스는 클러스터 전체에서 데이터 추출 파일의 중복성을 보장하며 모든 클러스터 노드에서 로컬로 추출을 사용할 수 있게 합니다. 로드가 많아지면 클러스터 전체에서 추출 파일을 로컬로 사용하여 처리와 렌더링 속도를 높일 수 있습니다.

Tableau의 아키텍처는 유연하여 어디에서나 플랫폼을 실행할 수 있습니다. Tableau Server를 온프레미스, 프라이빗 클라우드 또는 데이터 센터의 Amazon EC2™, Google Cloud Platform™ 또는 Microsoft Azure™에 설치할 수 있습니다. 또한 Tableau 분석 플랫폼은 VMware ESXi™ 또는 Microsoft Hyper-V™ 등 가상화 플랫폼에서 실행될 수 있습니다. Tableau Serve에서 최상의 성능을 보장하려면 각 가상화 플랫폼별 모범 사례를 따르는 것이 좋습니다.

개별 서버 프로세스에 대한 자세한 내용은 Tableau Server 관리 가이드를 참조하십시오.

인메모리 및 라이브 - 통합 아키텍처

Tableau는 이기종 플랫폼을 지원하여 수십 개 공급업체에서 가장 많이 사용되는 50개 이상의 데이터 원본에 연결됩니다. 아키텍처의 유연성을 통해 신속한 인메모리 분석을 원하는지 또는 라이브 데이터 저장소에 직접 연결하여 분석을 유도할지 선택할 수 있습니다. 어느 쪽을 선택하든 다양한 비즈니스 요구나 데이터 원본에 대해 라이브 연결과 인메모리 분석 간을 손쉽게 전환할 수 있습니다.

Tableau는 메모리에 매핑된 파일로 로드되는 TDE(Tableau 데이터 추출)라는 전용 열 저장소로 데이터를 추출하여 빠르게 액세스할 수 있는 인메모리 분석을 지원합니다. 추출은 Tableau Desktop 사용자가 생성하거나 Tableau API를 사용하여 외부 비즈니스 프로세스에서 생성할 수 있습니다. Tableau Server 아키텍처는 사용자가 생성하는 추출이 최적화된 관리 일정에 따라 업데이트되고 최신 상태를 유지할 수 있는 지원 기능을 기본 제공합니다.

데이터에 연결하고 이를 Tableau Server에 가져오는 방법과 상관없이 데이터 분석과 캐싱, 추출 새로 고침 및 기타 관련 작업을 위한 메모리 리소스가 충분해야 합니다.

순수한 인메모리 도구와 달리 Tableau의 메모리가 차지하는 공간은 전체 클러스터에 걸쳐 있으며 단일 서버에 축적되지 않습니다. 오히려 메모리 사용이 외부에서 공유되는 쿼리 캐시 및 세션 수준 캐시를 저장하는 기타 프로세스용 캐시 서버를 통해 클러스터 전체에 분산됩니다. 따라서 로드 증가로 인해 메모리에 미치는 영향이 워크로드에 따라 클러스터 전체에 퍼집니다.

테스트 접근법 및 방법론

대상 애플리케이션을 블랙박스로 취급하는 대부분의 기존 로드 테스트 프로젝트와 달리 Tableau Server의 경우 아키텍처에 대해 충분히 이해한 상태에서 로드 테스트를 수행해야 합니다. Tableau Server는 온프레미스, 클라우드 등 어디에서나 실행되도록 만들어졌습니다. 또한 모든 규모의 팀과 조직에서 사용할 수 있도록 설계되었습니다. 따라서 Tableau Server의 기본 설치는 대부분의 표준 배포에 잘 맞습니다. 기업 전체에서 Tableau Server를 확장하고 배포하려면 다양한 워크로드를 처리하는 방법과 몇 가지 간단한 구성 조정으로 배포 시나리오의 결과를 향상하는 방법에 대해 알아야 합니다.

버전 10 확장성 목표에 대해 다음 질문에 답하는 것으로 시작해보겠습니다.

1. 다음 2가지 일반 시나리오에 하드웨어를 추가하면 Tableau Server 10이 선형적으로 확장됩니까?
   - 최종 사용자 워크로드 확장
   - 백그라운더 워크로드 확장

2. 백그라운더 프로세스를 전용 작업자 노드로 이동하기에 적합한 시기는 언제입니까?

3. 이전 버전과 비교해서 Tableau Server 10 성능이 어떻게 다릅니까?

이 백서의 이전 버전에서는 특별히 최종 사용자 확장성에 초점을 맞추었지만 많은 고객들이 백그라운더 워크로드의 확장 방법에 대해 문의했습니다. 백그라운더 워크로드는 데이터의 최신성(추출 새로 고침) 및 분석이 광범위하게 사용되는 방식(구독 알림)을 제어합니다.

방법론

성능 및 확장성 개선은 Tableau Server 버전 10의 기본 목표였습니다. 따라서 실제 운용 준비가 된 엔터프라이즈급 테스트 방법론의 개발이 핵심 요구 사항이기도 했습니다. Tableau에서는 이 섹션에 설명된 방법론을 v10의 출시 전 버전에서부터 실행하기 시작했습니다. 신속한 개발을 반복하여 수행하는 동안 현재 방법론을 통해 약 20개의 확장 및 성능 관련 버그를 발견하여 수정했습니다. v10 출시 후에도 계속 테스트를 진행하여 10.0.1 MR(유지 관리 버전)에는 많은 버그가 수정되었습니다. 이 백서에서는 일반적으로 Tableau Server v10에 대해 언급하지만 테스트 결과는 10.0.1 MR을 기반으로 합니다. v10에서 최상의 확장성 및 성능 이점을 실현하려면 조직에서 10.0.1 MR 이상을 실행해야 합니다.

Tableau에서는 현실적인 고객 시나리오를 대변하는 워크로드를 수집하고 테스트하기 위해 확장성 사례를 계속 개발해 왔습니다. 배포의 확장성에 영향을 주는 변수는 많지만 배포 계획 때 고려해야 하는 중요한 요소는 다음과 같습니다.

- 사용자 영향 – 셀프 서비스 사용 및 사용자 채택: 얼마나 많은 사용자가 분석을 사용합니까? 정보가 충분히 제공된 의사 결정을 내리기 위해 사용자는 얼마나 자주 분석을 활용합니까? 사용자가 작성하는 비주얼라이제이션은 얼마나 복잡합니까?

- 데이터 영향 – 최신성, 크기 및 위치 데이터가 얼마나 큽니까? 데이터가 어디에 있습니까? 정확한 비즈니스 의사 결정을 위한 정보를 제공받으려면 데이터가 얼마나 최신 상태여야 합니까?

효과적인 비즈니스 의사 결정을 위한 분석 사용				
높음 (매초)		7. 예: WW 데이터 탐색 Tableau Public (미국 대통령 선거) 30K 뷰/시간	8. 예: 판매 할당량 대시보드, TV의 Tableau	9. 예: 항공 교통 컨트롤러, 금융 모니터링, 거래 실행
중간 (매시간)		4. 예: 일일 상점 재고 보험 고객 분석 마케팅 (타게팅)	5. 예: 환자 수용 능력 대리점 관리	6. 예: 지원 에스컬레이션 대시보드, 금융 포트폴리오 대시보드 사기 조사
낮음 (매일)		1. 예: 엔지니어링 - 선박 공간 저당 재고 기존 BI	2. 예: 인기품목 판매 리드 추적	3. 예: 고속도로 웹 교통정보 대시보드
		낮음 (매일)	중간 (매시간)	높음 (매초)
		효과적인 비즈니스 의사 결정을 위한 데이터 새로 고침 빈도		

그림 3: 분석 사용 및 데이터 새로 고침 빈도 매트릭스

이 고려 사항을 모두 테스트하기 위해 테스트를 통합하여 서버에 백그라운더 워크로드를 늘리면서 동시에 점차적으로 최종 사용자 로드를 증가시켜야 했습니다. 이 방법론으로 최종 사용자의 서비스 품질에 대한 백그라운더 워크로드의 영향을 연구할 수 있었습니다. 분석 사용의 낮음, 중간, 높음 및 백그라운더 워크로드 격리의 효과를 모델링하기 위해 실험실에서 다양한 워크로드로 400회 이상 반복하여 테스트를 실행했습니다. 시스템 확장성을 연구했으며 로드가 많을 때에만 드러나는 버그 또한 수정했습니다.

실제 환경과의 테스트 연관성을 높이기 위해 먼저 이 테스트를 실행하기에 적합한 워크로드 집합을 수집해야 했습니다.

실제 워크로드 특성 분석

v10의 경우 사용량이 피크인 기간 동안 실제 운용 환경의 Tableau Server가 어떻게 사용되는지 관찰하고 특성을 분석했습니다.

모델링할 통합 문서와 테스트의 워크로드 특성을 확인하기 위해 사용자가 3,000명인 운용 환경에서 Tableau Server 로그 파일을 분석했습니다. 또한 많이 사용된 비주얼라이제이션과 통합 문서를 확인하고 이러한 통합 문서 전체에서 사용량 분산을 계산한 다음 사용 특성을 분석했습니다. 예를 들어 요청 간 시간 간격(씽크 타임이라고도 함)을 조사했습니다. 다음은 운용 서버의 워크로드를 모델링하기 위해 수행한 단계입니다.

테스트 모델링 단계

1. 사용량이 피크인 기간의 운용 서버 로그를 가져옵니다.

2. 서버에서 걸린 가중 평균 시간별 상위 N 통합 문서-뷰를 확인합니다.

3. 가중 평균 = 평균 응답 시간 * 요청 수

4. 상위 N 통합 문서-뷰 중에서 상대적인 가중치를 계산합니다.

5. 선택된 각 통합 문서-뷰:
   a. 새로 고침=y를 사용하여 비주얼라이제이션 로드의 비율을 찾습니다. 이는 최신 데이터를 얻기 위해 적극적으로 데이터를 새로 고친 사용자 수를 알아내는 방법입니다.
   b. 가중치별 상위 N 상호 작용을 찾습니다.
   c. 상호 작용 간 평균 씽크 타임을 계산합니다.

6. 모델 확인:
   a. 통합 문서-뷰의 부트스트랩 TBB(테스트 간 시간) 사이 평균 시간을 찾습니다.
   b. 서버에서 걸린 가중 평균 시간별 상위 N 백그라운더 작업을 찾습니다.

7. 다양한 작업(예: 구독 및 추출 새로 고침)별로 백그라운더 트래픽을 분류합니다.
   a. 상위 백그라운더 작업 간 평균 시간을 찾아서 모델에 포함합니다.
   b. 다양한 일정의 구독 수를 찾습니다.
   c. 추출 새로 고침(데이터 서버에 게시된 추출, 통합 문서 추출)의 크기와 유형을 찾습니다.

위의 데이터를 모두 사용하여 사용량이 피크인 동안 실제 사용자가 운용 서버를 사용하는 방식을 나타낸 현실적인 워크로드 조합을 모델링했습니다. 마지막으로 운용 로그 분석에서 백그라운더 추출 새로 고침 및 구독에 대한 워크로드 기반 모델을 생성했습니다.

워크로드 조합은 다음 테이블에 요약되어 있습니다.

워크로드 이름	설명	이전 버전과의 비교 방법
실제 운용 서버 워크로드	이 워크로드는 IT에서 관리하는 중요 애플리케이션으로서 조직의 3000명 사용자를 지원하는 운용 Tableau Server의 사용량 분석과 특성 분석을 기반으로 했습니다.	여기에 지정된 대로 이 백서의 결과 데이터만 9.3과 10.0을 비교할 수 있습니다. 테스트 방법론이 크게 달랐으므로 9.0 백서를 비롯한 이전 백서와 결과를 비교할 수 없습니다.
실제 운용 서버 워크로드 + 새 기능	이 워크로드는 v10의 새 기능을 수행한 통합 문서와 위의 워크로드를 결합했습니다.	9.3에는 수행할 v10 기능이 없으므로 9.3과 10.0을 비교할 수 없습니다. 이 백서 결과의 최고 활용 방법은 v10 확장 특성을 알리는 것입니다.
백그라운더 조합	이 워크로드는 운용 워크로드 분석 기반으로 실제 통합 문서를 모델링하며 운용 환경 미러링을 예약합니다.	10.0 확장성 테스트 백서에 소개된 새로운 조합으로, 이전 백서 결과와 비교할 수 없습니다.

테이블 2: 워크로드 조합 설명

그런 다음 이 조합을 각각 가져와서 격리된 확장성 실험실의 물리적인 시스템에서 최종 사용자와 백그라운더 로드를 점차적으로 증가시키며 독립적으로 실행했습니다. 클러스터가 최대 용량이 되면 작업자 노드를 한 번에 하나씩 추가하여 로드를 계속 확장했습니다. 각 실험을 수행하면서 시스템 작동 방식을 확인했습니다. 실험이 진행되는 동안 응답 시간, 처리량, 오류율 등 핵심 성능 지표를 기록했습니다. 또한 JMX를 사용하여 시스템 메트릭과 애플리케이션 서버 메트릭을 기록했습니다. 각 테스트를 수행할 때마다 데이터 연관성을 살피고 워크로드가 증가하면 시스템이 어떻게 작동하는지 분석했습니다. 또한 이 과정에서 민첩한 개발 프로세스를 통해 확장성 버그를 찾아 수정했습니다.

백그라운더 방법론

백그라운더 서버 프로세스는 시스템 수준 및 사용자 수준 백그라운드 작업을 처리합니다. 일상적인 리포지토리 유지 관리 작업과 같은 시스템 수준 작업은 백그라운더에서 수행합니다. 사용자 수준 작업은 사용자가 시스템에서 사용자를 대신하여 실행하도록 제출했을 수 있는 작업입니다. 예를 들어 사용자는 서버에 추출을 게시한 다음 일정에 따라 추출에 대해 반복 데이터 새로 고침을 구성할 수 있습니다. 이러한 작업 집합을 통해 새로 고침 작업이 생성됩니다. 백그라운더는 사용자를 대신하여 작업 목록을 검토하고 작업을 실행하는 프로세스입니다. 관리 부서에서 데이터를 새로 고칠 때까지 사용자가 기다릴 필요가 없으므로 이 시나리오는 효과적인 셀프 서비스에 매우 중요합니다. 하지만 Tableau Server를 관리하는 관리 팀에서 이러한 유형의 로드를 위한 용량을 계획하지 않는 경우 최종 사용자에게 서비스 품질 문제가 발생할 수 있습니다. 백그라운더 서비스의 최적화 정도는 서버 크기 조정과 계획에 매우 중요한 구성 요소입니다. 다른 컴퓨터에 백그라운드 서비스를 분리하여 워크로드를 격리할지 고려해야 합니다.

워크로드를 시간별로 분리하여 처리할 수 있는 간단한 여러 모범 사례가 백서 끝 부분에 설명되어 있습니다. 하지만 백그라운더를 동일한 컴퓨터에서 분석 서비스로 실행하는 경우 서버의 리소스 공유 제한으로 인해 로드가 많을 때 최종 사용자 서비스 품질에 영향을 줄 수도 있습니다.

이런 이유로 분석 서비스와 동일한 컴퓨터에 함께 배치되었을 때 백그라운더가 최종 사용자 확장에 미치는 영향을 알아보고자 했습니다. 또한 자체 하드웨어에서 격리되었을 때 로드를 증가시키면 백그라운더가 어떻게 확장되는지 수량화하고자 했습니다.

이를 알아보기 위해 백그라운더를 별도로 실행할 4개의 물리적 코어가 있는 컴퓨터를 사용했습니다. 같은 컴퓨터에서 다른 Tableau Server 프로세스는 실행하지 않았습니다. Tableau Server 버전 9.3 및 10.0에서 동일한 운용 모델링 워크로드를 실행했습니다. 사용자 수준 작업에 집중할 수 있도록 워크로드에 추출 새로 고침과 구독이 포함되었습니다. 8개 일정의 400개 구독이 워크로드에 포함되었습니다. 구독 알림의 성공 및 Tableau가 모든 구독을 완료하기까지 걸린 시간도 알아보았습니다.

격리 환경 표준화

성능 실험실에서 다음과 같은 사양의 동일한 물리적 시스템에서 확장성 테스트를 실행했습니다.

서버 유형	Dell PowerEdge R620
운영 체제	Microsoft Windows Server 2012 R2 Standard 64비트
CPU	2.6GHz 1x8 물리적 코어, 하이퍼 스레딩 사용(16개의 논리적 코어)
메모리	64GB

테이블 3: 테스트 환경의 각 노드에 대한 하드웨어 사양

이 테이블에 물리적 코어가 표시되어 있지만 하이퍼 스레딩 사용을 중지하지 않는 것이 좋습니다. 여기에 참조된 경우를 제외하고, 일관성을 위해 이 백서의 물리적 코어 수를 언급할 때는 물리적 시스템에 항상 하이퍼 스레딩이 활성화되어 있다고 가정합니다.

배포 토폴로지

클러스터는 하나 이상의 주 (컨트롤러) 노드와 하나 이상의 작업자 노드로 구성되어 있습니다. 이 테스트에서는 작업자 노드가 다음과 동일한 프로세스 구성 프로필을 공유했습니다.

프로세스	주 서버 tsperf-212.perf.dev.tsi.lan	작업자 노드 1 tsperf-213.perf.dev.tsi.lan	작업자 노드 2 tsperf-215.perf.dev.tsi.lan	작업자 노드 3 tsperf-216.perf.dev.tsi.lan	작업자 노드 4 tsperf-217.perf.dev.tsi.lan
클러스터 컨트롤러	✓	✓	✓	✓	✓
게이트웨이	✓	✓	✓	✓	✓
애플리케이션 서버		✓	✓	✓	✓
VizQL Server		✓ ✓	✓ ✓	✓ ✓	✓ ✓
캐시 서버		✓ ✓	✓ ✓	✓ ✓	✓ ✓
검색 및 찾아보기		✓	✓	✓	✓
백그라운더		✓	✓	✓	✓
데이터 서버		✓ ✓	✓ ✓	✓ ✓	✓ ✓
데이터 엔진		✓	✓	✓	✓
파일 저장소		동기화 중	동기화 중	동기화 중	동기화 중
리포지토리	✓				

새로 고침 상태 — ✓ 액티브 · 사용 중 · ✓ 패시브 · ⚠ 라이선스 없음 · ✕ 중단 · 상태 알 수 없음

그림 4: 프로세스 구성

여기에 표시된 작업자 프로세스 구성은 기본 구성입니다. 환경 및 사용 시나리오에 대해 구성하는 프로세스의 수와 유형에 따라 어느 정도의 확장성을 얻을 수 있습니다.

주 노드는 클러스터 컨트롤러, 게이트웨이 및 리포지토리 프로세스를 포함하는 기본 설치로 구성됩니다. 코어 라이선스 스키마로 배포되면 이러한 주 노드 구성 유형은 코어 개수에 포함되지 않습니다.

앞서 설명된 사용자 워크로드를 시뮬레이션하기 위해 로드 생성기(TabJolt)를 사용하여 워크로드를 확장했습니다. TabJolt는 Tableau Server 9.0 이상에서 사용하도록 특별히 설계된 '포인트 앤 런(point-and-run)' 로드 및 성능 테스트 도구입니다. 아래 그림은 테스트 실행의 논리적 뷰를 보여줍니다.

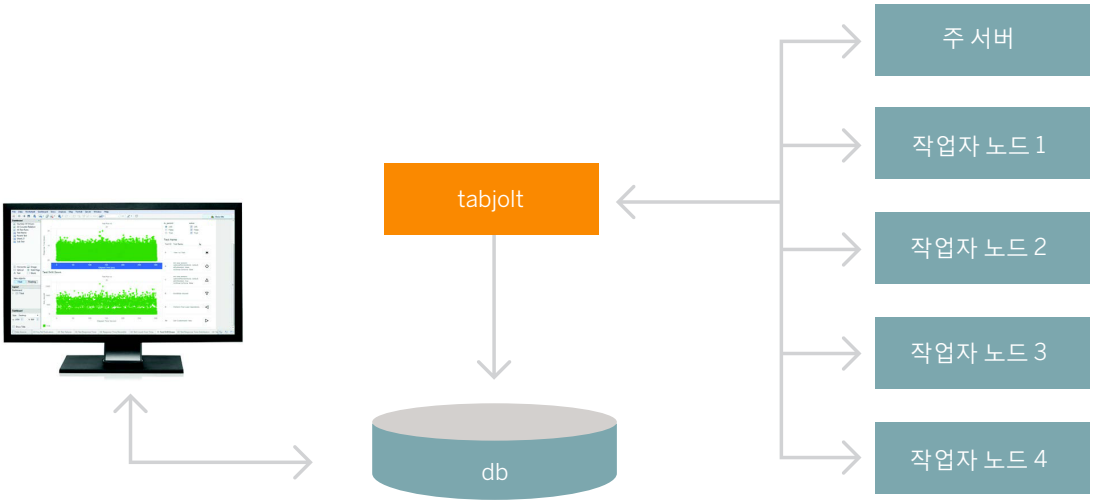

그림 5: 테스트 환경에 대한 논리적 뷰

각 테스트를 반복 실행하여 분석할 데이터를 수집했지만 먼저 몇 가지 메트릭과 정의를 살펴보고 결과를 알아보겠습니다.

측정 및 보고

CPU, 메모리, 디스크에 대한 시스템 메트릭을 비롯하여 하드웨어 성능과 확장성을 알아보기 위한 다양한 메트릭을 측정했습니다. 또한 응답 시간, 처리량, 오류율, 실행 기간 등 성능 및 확장성 메트릭을 측정했습니다.

이 백서에 설명된 데이터를 이해하기 위해 몇 가지 정의를 간단히 검토해 보겠습니다.

시나리오

시나리오는 서버에서 일어나는 최상위 사용자 활동입니다. 이전 버전의 백서에서는 서버가 게스트 액세스를 활성화했을 때 비주얼라이제이션 로드와 상호 작용 시간에 초점을 맞추었습니다. 이번 백서에서는 로그인과 같은 애플리케이션 서버 서비스 및 기타 서비스를 포함하도록 워크로드를 확대했습니다. 최종 사용자는 TabJolt의 사용자 지정 버전을 사용하여 시뮬레이션한 운용 환경 워크로드 모델링을 기반으로 일련의 단계를 진행합니다.

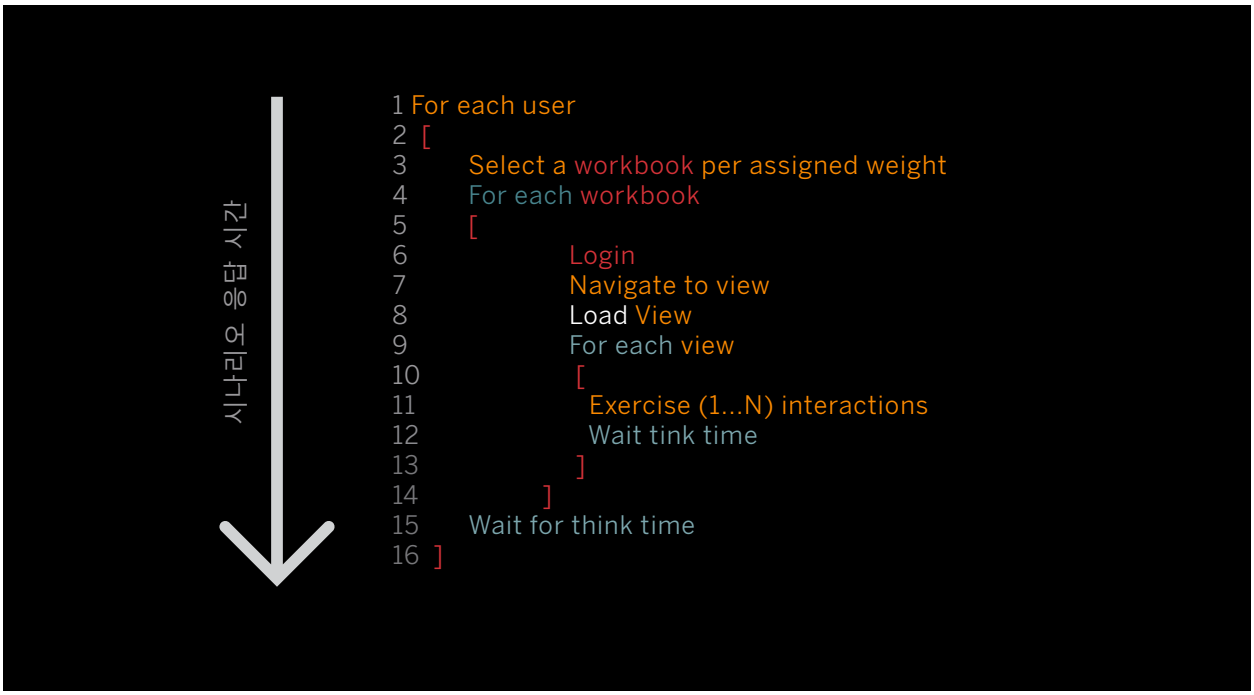

그림 6: 시나리오 테스트 알고리즘

테스트 모델에는 시나리오를 완료하는 데 필요한 시간이 포함되어 있으므로 이 백서에 보고된 응답 시간은 간단한 로드 및 상호 작용 모델보다 깁니다. 응답 시간은 고객 관점에서 트랜잭션에 대해 측정 및 보고됩니다. 즉, 시나리오는 지연 시간과 같은 환경적 네트워크 변수를 포함하여 실제 최종 사용자의 관점을 설명합니다.

응답 시간

응답 시간은 최종 사용자의 요청에 서버가 응답하는 데 걸리는 시간(초)으로 측정됩니다. 사용자가 서버에 로그인하고, 비주얼라이제이션으로 이동하여 이 비주얼라이제이션에서 필터를 변경하고, 비주얼라이제이션이 업데이트되고 렌더링되기를 기다린 다음 이를 분석(씽크 타임)하는 예를 생각해 보겠습니다. 사용자가 서버에 로그인한 뒤 사용자 씽크 타임이 끝날 때까지 총 '벽시계' 시간은 시나리오의 반복에 대한 응답 시간으로 보고됩니다.

시나리오 처리량

시나리오 처리량은 초당 완료된 성공적인 시나리오 수(TPS)입니다. 시나리오 TPS는 시스템의 컴퓨팅 한도에서 특정 토폴로지에 대해 한 시간 동안 시나리오를 실행하여 계산했습니다. 총 시나리오 수를 한 시간의 초 수(3600)로 나눈 값이 TPS입니다.

예를 들어 테스트 실행 기간 동안 8코어 Tableau Server 10에서 시나리오를 실행하여 16,372개의 시나리오를 완료했습니다. 이는 약 4.5(16,372/3600) TPS로 해석됩니다. 이 실험에서는 TPS가 1 미만이었던 운용 환경에서 확인된 양보다 훨씬 더 많이 서버 로드를 증가시켰습니다. 실험 데이터가 상당히 증가하는데 이는 선택한 하드웨어 및 확장 대상에 대한 최고 성능 용량을 파악하기 위해 서버 로드를 한도까지 늘렸기 때문입니다.

하지만 운용 환경 배포에서는 사용량이 피크일 때 로드가 갑자기 증가하며, 사용자 환경은 대개 자동 테스트 실행 프레임워크보다 훨씬 느려집니다. 이 차이를 기억하시기 바랍니다. 예를 들어 Tableau Public은 초당 약 12개까지의 대규모 비주얼라이제이션 로드(노출수라고도 함)를 지원하는데 이는 Tableau 비주얼라이제이션에 대해 주당 7백만 뷰 이상을 지원함을의미합니다.

활성 사용자

활성 사용자는 사용량이 피크인 1시간 동안 Tableau Server를 동시에 사용하는 사용자 수를 측정하는 메트릭입니다. 이제 시나리오에 로그인, 비주얼라이제이션 로드, 사용자 상호 작용, 검색 및 기타 동작이 포함됩니다. 활성 사용자는 사용량이 피크인 1시간 동안 이러한 유형의 동작을 수행하는 사용자로 정의됩니다.

서버 설치를 통해 지원할 수 있는 활성 사용자 수를 알아내기 위해 단일 서버에서 응답 시간 저하, 2% 이상의 오류율(Tableau Server HTTP 오류), 80% 이상의 CPU 사용률 등이 없이 유지될 수 있는 사용자 수를 확인하는 것으로 시작했습니다.

운용 환경 워크로드 특성 분석을 기반으로 특정 통합 문서에 가중치를 할당했습니다. 그런 다음 할당된 가중치를 기반으로 임의의 통합 문서를 선택한 다음 앞서 설명한 전체 시나리오를 완료할 스레드(가상 사용자)를 만들었습니다. 시나리오가 끝날 때 스레드는 지정된 씽크 타임 동안 기다립니다. 이 기간이 끝나면 스레드는 반복을 완료하고 종료됩니다. 테스트가 진행되는 동안 다른 메트릭 중에서 시나리오 처리량, 응답 시간, 오류율, CPU 및 메모리 사용량을 측정했습니다. CPU 사용률이 80% 아래로 유지되고 오류율이 2% 미만인 상태로 응답 시간이 저하되지 않는 한 계속해서 활성 스레드 수를 늘렸습니다. 이 임계값 중 하나가 깨지면 이 값을 해당 토폴로지에서 합리적으로 지원할 수 있는 활성 사용자 수에 대한 적정값으로 간주했습니다.

서버가 선형으로 확장된다는 사실을 입증하기 위해 단일 시스템의 적정값을 찾은 다음 가상 사용자 수를 다음 증분까지 선형으로 증가시켰고 새로운 로드 상태에서도 미리 설정된 조건을 여전히 충족하는 것을 확인했습니다.

결과

지금까지 테스트 실행 방식, 사용한 배포 및 메트릭을 살펴 보았습니다. 이제 결과를 알아보겠습니다.

Tableau Server 10의 선형 확장

첫 번째 질문은 'Tableau Server 10이 어떻게 확장되는가'였습니다. 사용자 로드가 증가하는 경우, 클러스터에 작업자 노드를 추가하면 Tableau Server 10이 로드에 맞춰 선형적으로 확장되는 것을 확인했습니다. 아래 그림에서는 작업자 노드 증가에 따라 초당 완료된 사용자 시나리오 수를 보여줍니다.

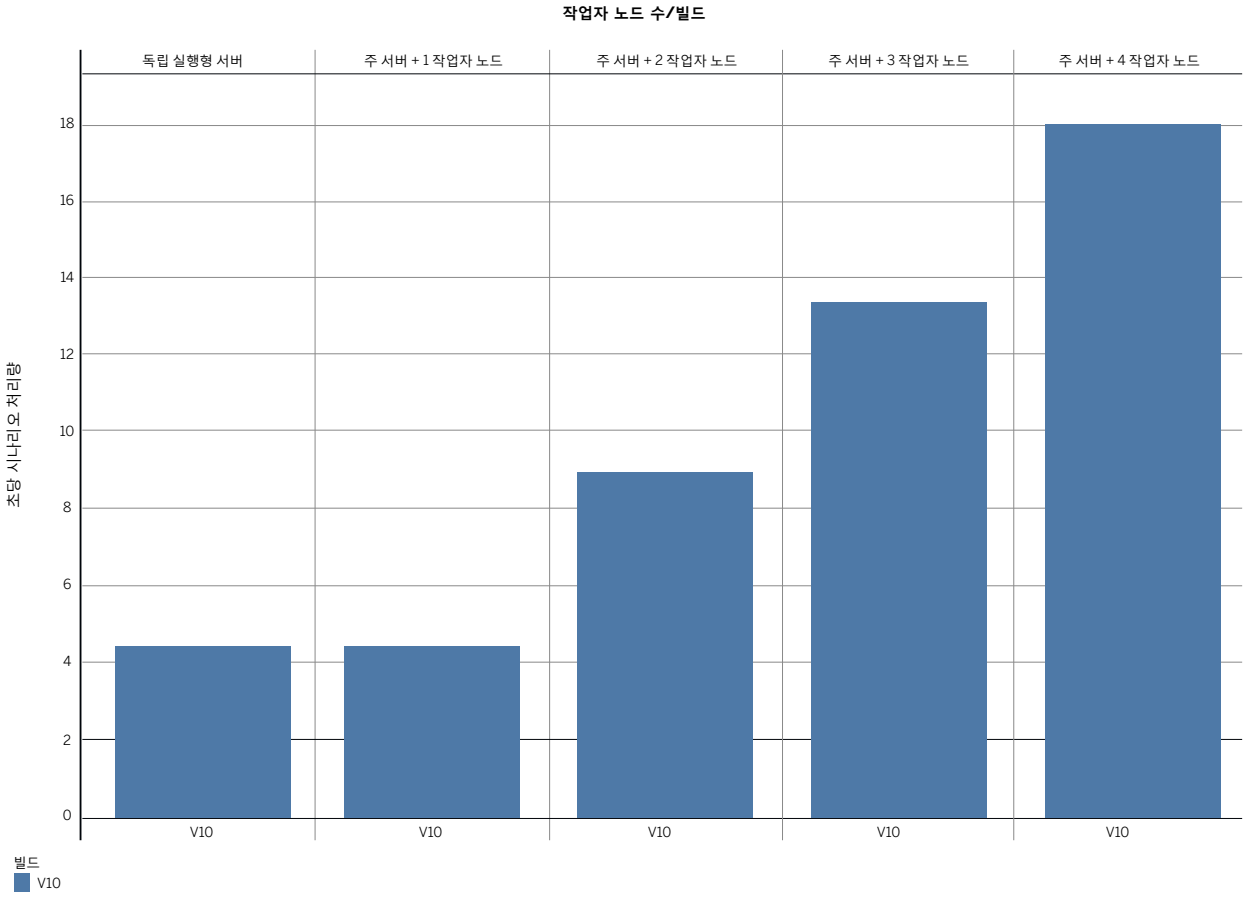

그림 7: 초당 시나리오 처리량

각 열은 클러스터 토폴로지를 보여줍니다. 맨 왼쪽 열은 단일 서버 설정을 보여줍니다. 두 번째 열은 하나의 작업자 노드가 있는 주/작업자 노드 클러스터 구성을 보여줍니다. 나머지 각 열은 앞서 설명된 대로 작업자 노드가 하나씩 더 추가되었습니다. 막대의 높이는 초당 평균 시나리오 처리량(TPS)을 표시합니다. TPS는 서버에서 처리하는 작업의 양을 나타냅니다. 보다시피 작업자 노드가 추가될 때마다 TPS가 선형적으로 증가했습니다. 클러스터에 로드를 증가시키면 클러스터 전체의 CPU 활용률이 평균 약 80%가 됨을 볼 수 있습니다. 아래 그림은 로드가 증가되고 신뢰 구간이 95%인 클러스터 전체의 CPU 활용률을 보여줍니다.

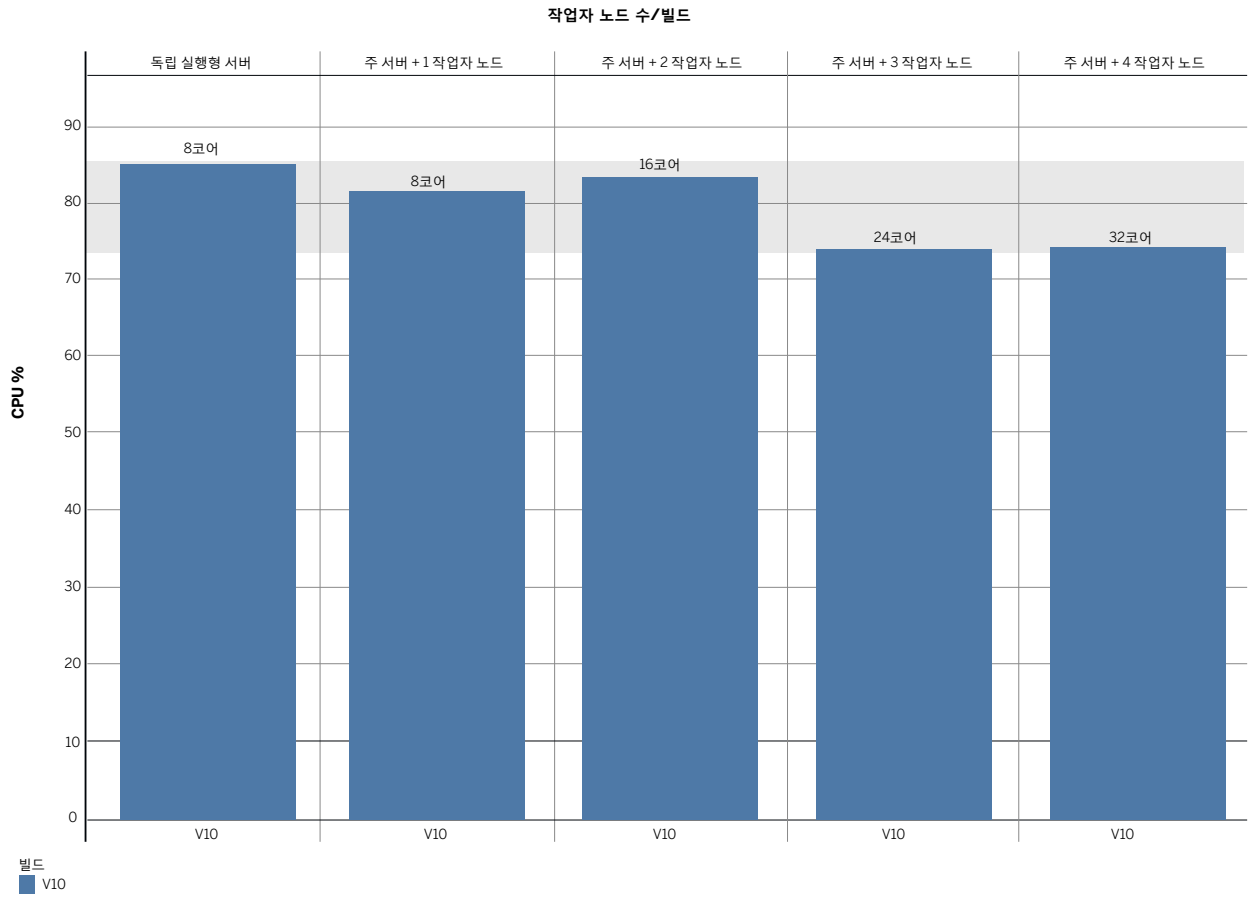

그림 8: 로드 증가에 따른 클러스터 전체의 CPU 활용률

그림 8에 나타난 대로 클러스터 전체에 CPU 로드를 분산하도록 작업자 노드를 더 추가하면 급격한 로드 증가에 대한 여유 공간이 어느 정도 제공되어 시스템이 최적화됩니다. 해당 클러스터에 작업자 노드를 더 적게 구성하면 작업자 노드가 더 많은 클러스터에 비해 CPU 활용률이 비교적 더 높습니다. 작업자 노드가 더 적은 경우 컴퓨팅 기반 프로세스가 한정된 리소스를 놓고 경쟁합니다. 테스트가 진행되는 동안 서버의 오류율을 관찰하고 이 오류율이 방법론의 일환으로 설정했던 2% 목표 내로 유지된 것을 확인했습니다. 워크로드에 따라 클러스터가 더 적은 시스템으로 제한되거나 용량이 한정된 경우 오류율이 더 높아지고 서비스 품질이 저하될 수 있습니다.

다음 질문은 '동일한 테스트 방법론을 사용했을 때 Tableau Server 10과 Tableau Server 9.3은 어떻게 다르냐' 였습니다.

결과는 각 버전에서 동일한 방법론을 실행했을 때 Tableau Server 10 처리량이 Tableau Server 9.3과 비교하여 향상되었다는 것입니다.

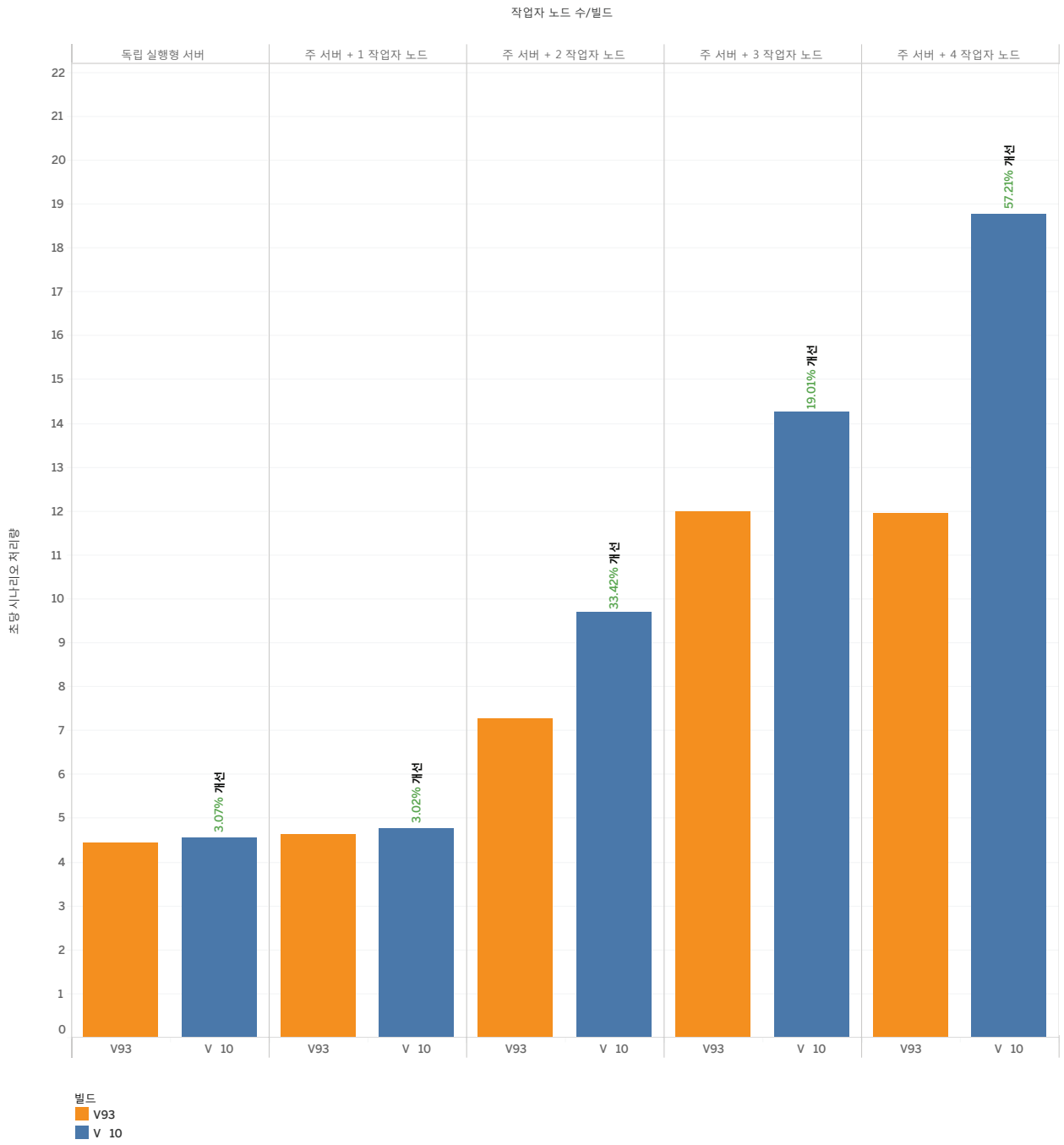

그림 9: 9.3과 10.0.1의 초당 시나리오 처리량 비교

그림 9는 Tableau Server 9.3의 시나리오 처리량(주황색 막대)을 Tableau Server 10의 시나리오 처리량(파란색 막대)과 비교한 것입니다. 각 패널은 열 머리글에 설명된 대로 클러스터 토폴로지의 구성을 보여줍니다. 앞에서 명시된 대로 클러스터에 노드를 추가하면 Tableau Server 10이 선형적으로 확장되었을 뿐만 아니라 Tableau Server 9.3과 비교하여 시나리오 처리량도 향상되었습니다.

버전을 비교할 때 가장 적절한 방법은 실제 환경의 사용자 시나리오를 반영하는 향상된 테스트 방법론으로 두 버전을 비교하는 것입니다. 따라서 Tableau Server 10 결과를 이전 백서의 결과와 비교하는 것은 테스트 방법론이 상당히 다르기 때문에 정확하지 않습니다.

처리량을 관찰하는 것도 중요하지만 전체 테스트 시나리오에서 최종 사용자 응답 시간이 성능 기준에 충분히 부합하며 로드가 증가해도 장애가 발생하지 않는 것을 목표로 했습니다.

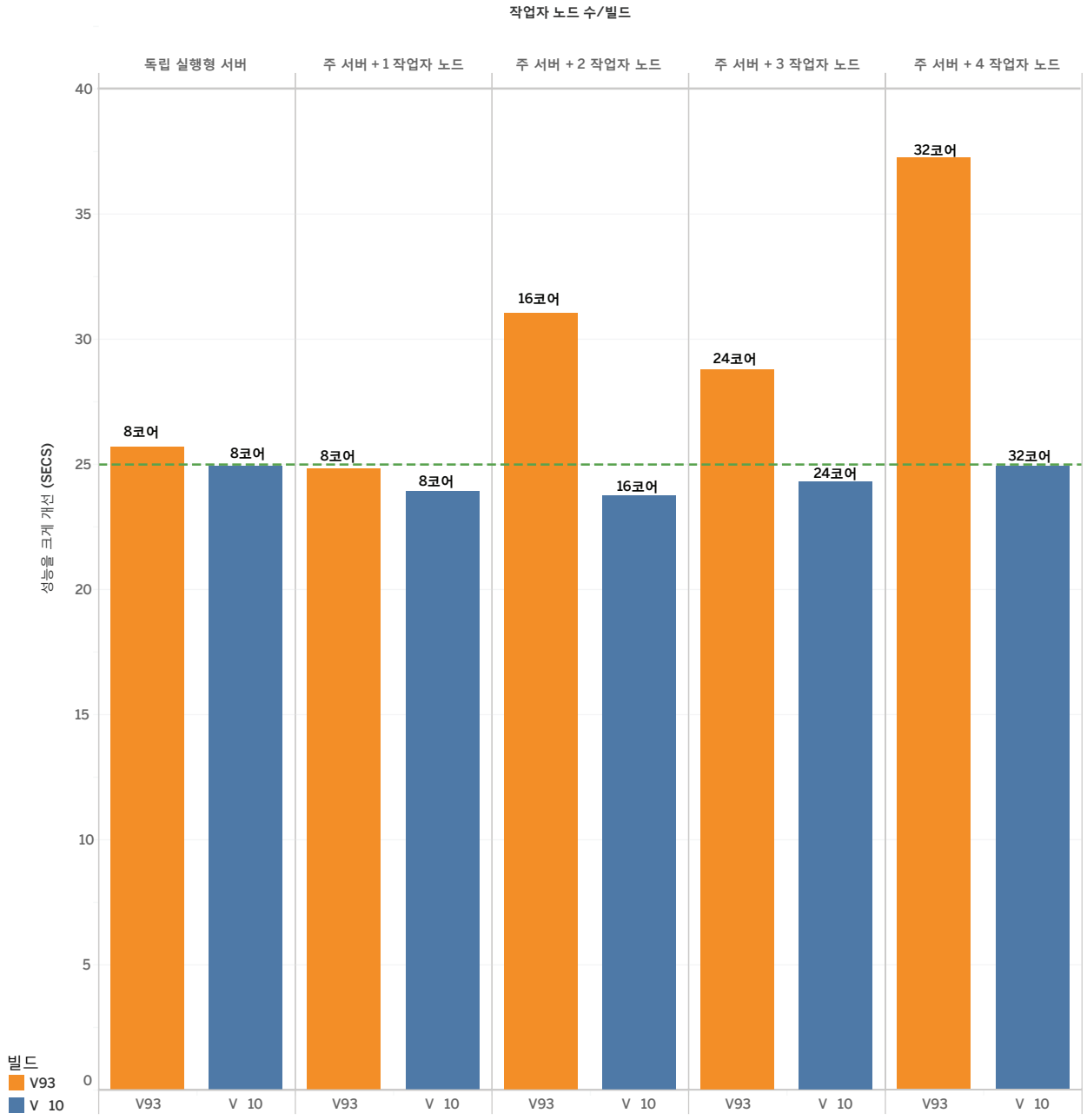

그림 10: 평균 시나리오 응답 시간(초) 비교: 9.3 및 10.0.1

그림 10에서 주황색 막대의 높이는 버전 9.3의 평균 시나리오 응답 시간을 나타냅니다. 파란색 막대의 높이는 버전 10.0.1의 동일한 메트릭을 나타냅니다. 버전 9.3을 호스팅한 클러스터에서 최종 사용자 로드를 증가시키면 응답 시간이 점점 길어짐을 확인했습니다. 반면 Tableau Server 10은 시스템에 가상 사용자를 계속 추가해도 더욱 일관된 응답 시간을 보여주었습니다. 이를 통해 Tableau Server 10이 Tableau Server 9.3에 비해 동일한 워크로드에서 최종 사용자에게 더 나은 성능과 응답성을 제공한다는 것을 알 수 있습니다.

로드가 증가하는 상태에서 Tableau Server 10의 상대적으로 일관되고 안정적인 응답 시간은 캐시 구성 요소에 대한 몇 가지 개선 사항에 기인한 결과였습니다. 특히 v10에서는 부트스트랩 응답 시나리오에 대한 캐싱을 최적화하기 위해 노력했습니다. 사용자 세션의 데이터를 초기화하고 캐시하기 위해 부트스트랩 시나리오가 맨 처음 수행됩니다. Tableau Server의 이전 버전에서는 캐싱 기능이 없어서 모든 후속 요청의 부트스트랩 데이터가 매우 유사하거나 동일한 경우에도 이를 계산하고 저장했을 것입니다. 버전 10에서는 이 시나리오에서 스마트 캐싱 작업을 수행합니다. 이러한 개선을 통해 Tableau Server 10은 뛰어난 응답 시간을 유지하면서 동시에 더 많은 사용자 로드를 처리할 수 있어 더욱 효율적입니다. 그런데 앞서 처리량 증가에서 확인한 대로 이제 서버는 더 많은 작업을 수행하고 있습니다.

위의 모든 결과는 Tableau Server가 최종 사용자의 분석 사용 증가에 따라 확장되는 방식을 관찰한 것이었습니다. 서버에 최종 사용자 워크로드가 더 증가하면 클러스터에 작업자 노드를 추가하여 충분한 용량을 제공함으로써 최종 사용자에게 양질의 서비스 환경을 제공해야 합니다.

다음 질문들은 백그라운더가 구성되는 위치와 방법에 따라 사용자에게 미치는 영향에 대해 중점적으로 다룹니다. 특히 백그라운더 프로세스가 있는 동일 컴퓨터에서 분석 서비스(VizQL Server)로 호스팅하는 경우와 백그라운더 프로세스를 다른 컴퓨터에 격리하는 경우 사용자에게 미치는 영향에 대해 살펴봅니다. 다음 섹션에서는 그 결과에 대해 설명합니다.

백그라운더 결과

먼저 기본 단일 서버 설치에서 백그라운더를 실행할 때의 영향을 수량화하고자 했습니다. 일반적으로 최종 사용자 및 확장성에 대해 이 시나리오가 미치는 영향은 무엇일까요? 단일 Tableau Server 배포에서 워크로드 실험을 실행하고 TPS를 기록했습니다. 그런 다음 작업자 노드를 추가하여 클러스터 토폴로지를 구축함으로써 클러스터에서의 워크로드를 테스트했습니다.

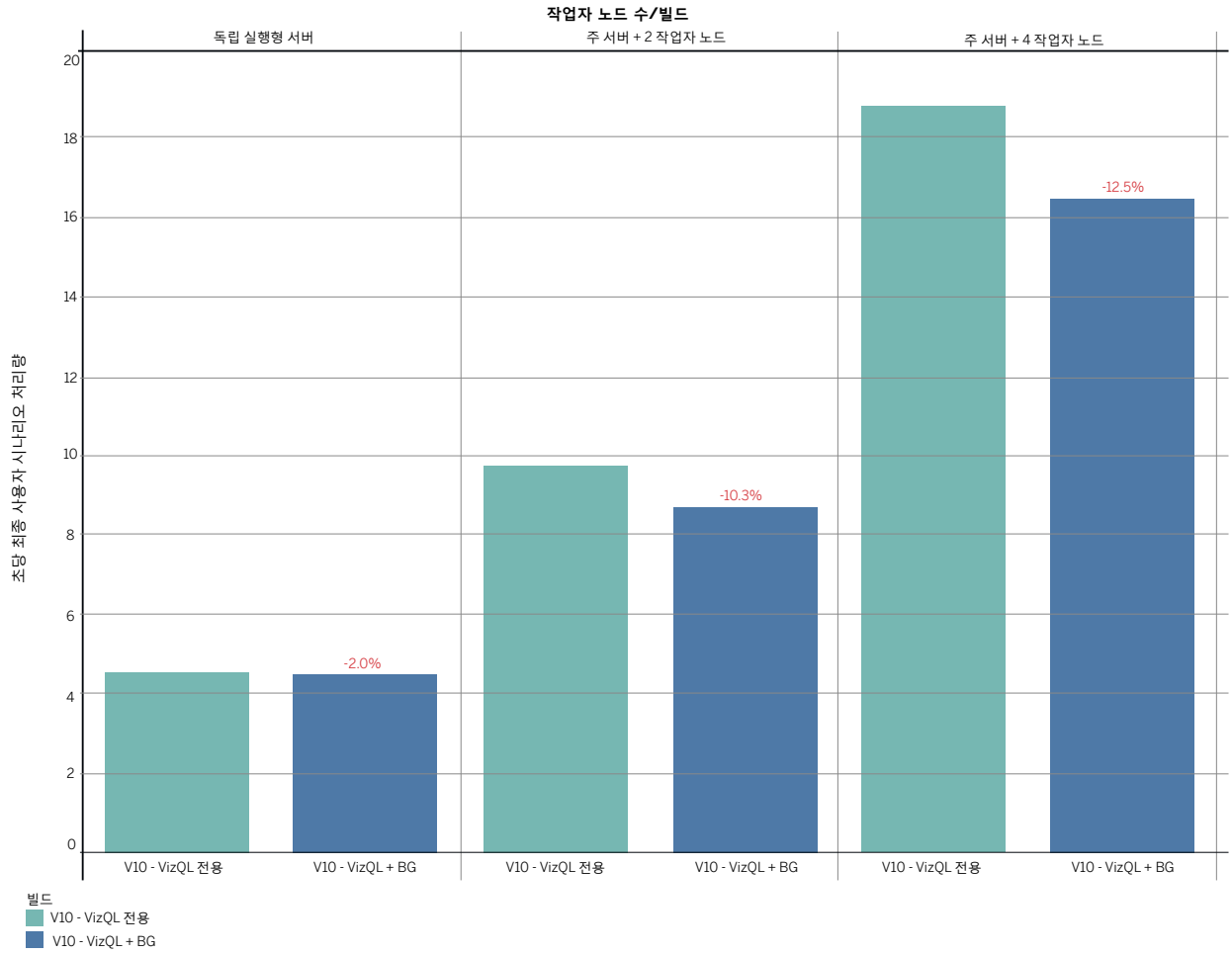

그림 11: 백그라운더와 VizQL 서비스를 함께 배치하는 경우 최종 사용자 확장성에 미치는 영향

그림 11에서는 2개의 테스트 시나리오 결과를 보여줍니다. 녹색 막대는 VizQL 전용 워크로드를 실행한 테스트 시나리오를 나타냅니다. VizQL 전용 워크로드는 최종 사용자 워크로드를 시뮬레이션합니다. 파란색 막대는 같은 클러스터에서 VizQL 워크로드에 고정된 양의 백그라운더 워크로드를 추가한 테스트 시나리오를 나타냅니다. 그런 다음 두 시나리오에서 최종 사용자 워크로드에 대한 TPS의 변화를 기록했습니다. VizQL Server와 백그라운더는 모두 컴퓨팅 위주의 워크로드이기 때문에 영향을 받을 것임을 알지만, 특정 워크로드에 대한 영향을 측정하고자 했습니다.

여기에서 우리는 전반적인 최종 사용자 시나리오 처리량이 2~12% 감소되었음을 확인했습니다. 이러한 감소는 최종 사용자에게 제공되는 클러스터 역량이 백그라운더 워크로드 처리의 영향을 받는다는 것을 보여줍니다. 이 데이터를 통해 클러스터가 시간 단위당 100개의 최종 사용자 시나리오를 처리하는 경우 백그라운더 로드를 계속 추가하면 처리 용량이 동일 시간 단위에서 88개의 사용자 시나리오로 감소된다고 추정할 수 있습니다. 이러한 처리량의 감소는 워크로드와 급격한 로드 증가, 하드웨어 한계, 인프라 변수 등 여러 요인에 따라 최종 사용자 확장성 및 서비스 품질에 상당한 영향을 미치는 것으로 해석될 수 있습니다.

이 테스트는 Tableau Server 워크로드 계획에 맞춰 하드웨어를 적절하게 지원하는 것이 중요함을 보여줍니다. 지원 역량이 부족하거나 제약이 있는 하드웨어에서 Tableau Server를 실행하면 성능 문제로 인해 처리량 감소, 백그라운더 작업 실패, 구독 알림의 지연 및 최종 사용자 오류가 발생할 수 있습니다. 이러한 상황이 발생하면 클러스터를 확장하여 백그라운더 워크로드를 전용 하드웨어에 격리하는 것이 좋습니다. 백그라운더 서비스를 격리하면 동일한 컴퓨터에 함께 배치되었을 때 같은 리소스를 놓고 경쟁할 컴퓨팅 기반 프로세스 문제가 해소됩니다.

백그라운더를 VizQL Server와 함께 배치했을 때와 자체 시스템에 격리했을 때 구독의 특정 워크로드에 대한 영향을 검토해 보겠습니다.

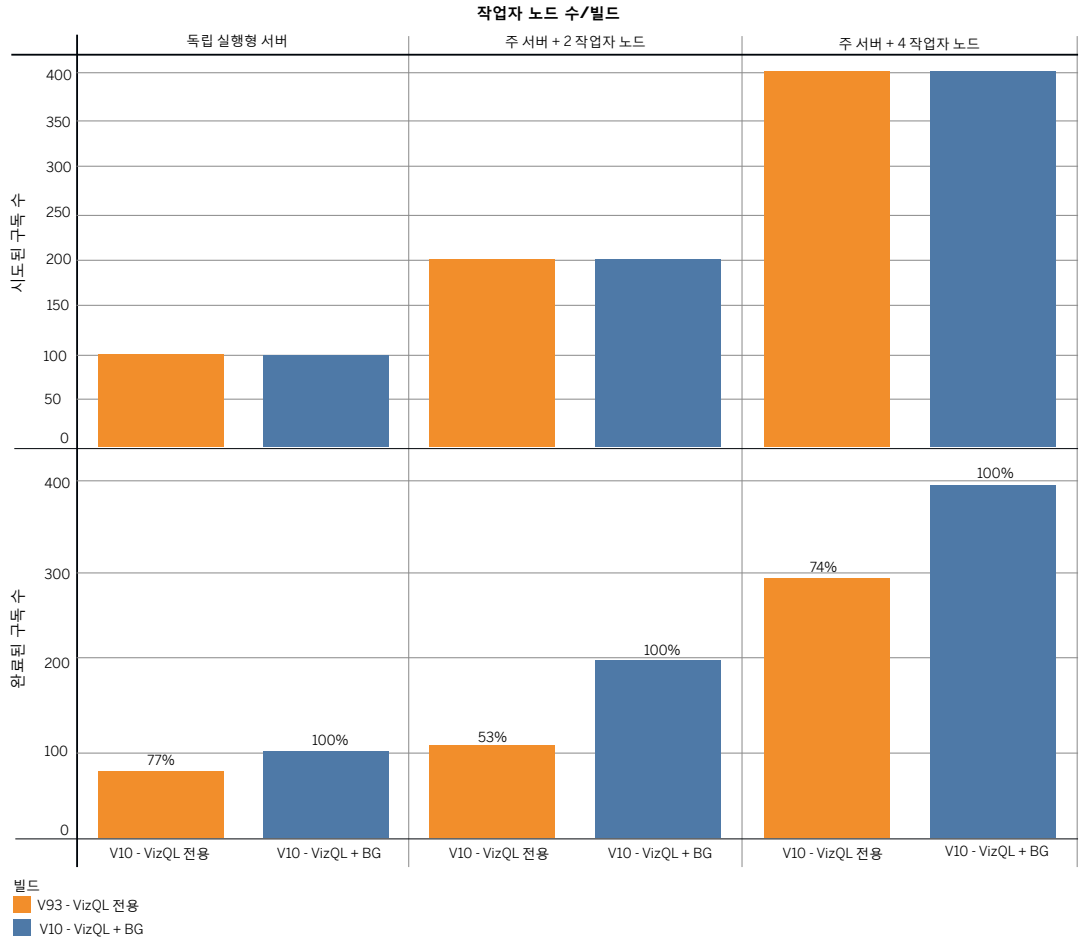

그림 12: 구독 알림 개선 사항

그림 12에서 각 패널은 Tableau Server 9.3(주황색) 및 10(파란색)의 클러스터 토폴로지를 보여줍니다. 여기에서 각 막대는 시도된 구독 알림 개수와 완료된 알림 개수를 나타냅니다. 각 반복 수행 중 0, 2, 4단계에서 작업자 노드 수를 늘렸으며, 각 작업자 노드에는 백그라운더 프로세스를 1개만 구성했습니다. (1 및 3 작업자 노드 테스트를 생략해도 추구한 결과의 정확도가 손상되지 않았습니다.) 백그라운더 프로세스를 작업자 노드당 하나로 제한하여 비교를 위한 일관된 측정 환경을 구축했습니다.

백그라운더 구독 로드를 증가시켰을 때 Tableau Server 10은 제출된 모든 작업을 완료할 수 있었습니다. 같은 테스트에서 Tableau Server 9.3은 포화되기 시작하여 일부 구독이 실패했습니다. 이러한 상황은 리소스가 부족하고 과도하게 로드된 토폴로지에서 악화될 수 있습니다. 또한 Tableau Server 10은 시스템에 작업자 노드가 추가됨에 따라 증가하는 구독 수를 처리할 수 있음을 확인했습니다. 위의 결과를 가져온 Tableau Server 10 개선 사항은 알림의 이미지 캐싱 도입으로 이루어졌습니다. 이 이미지 캐싱 기능은 백그라운더가 일정과 연결된 동일한 작업을 처리할 때 훨씬 작업을 적게 하도록 해줍니다. 동일한 일정에 동일한 작업을 실행하는 시나리오에서 Tableau Server 10은 첫 번째 실행에서 얻은 결과를 캐시하여 같은 워크로드의 후속 요청에 결과를 제공합니다. 즉, 같은 작업의 경우 더 효율적으로 완료됩니다.

동일한 일정에 동일한 통합 문서를 사용하는 구독의 경우 구독 완료 시간이 9.3과 비교하여 60~90% 단축된 것을 확인했습니다. 그림 13에서는 이러한 개선 사항을 보여줍니다.

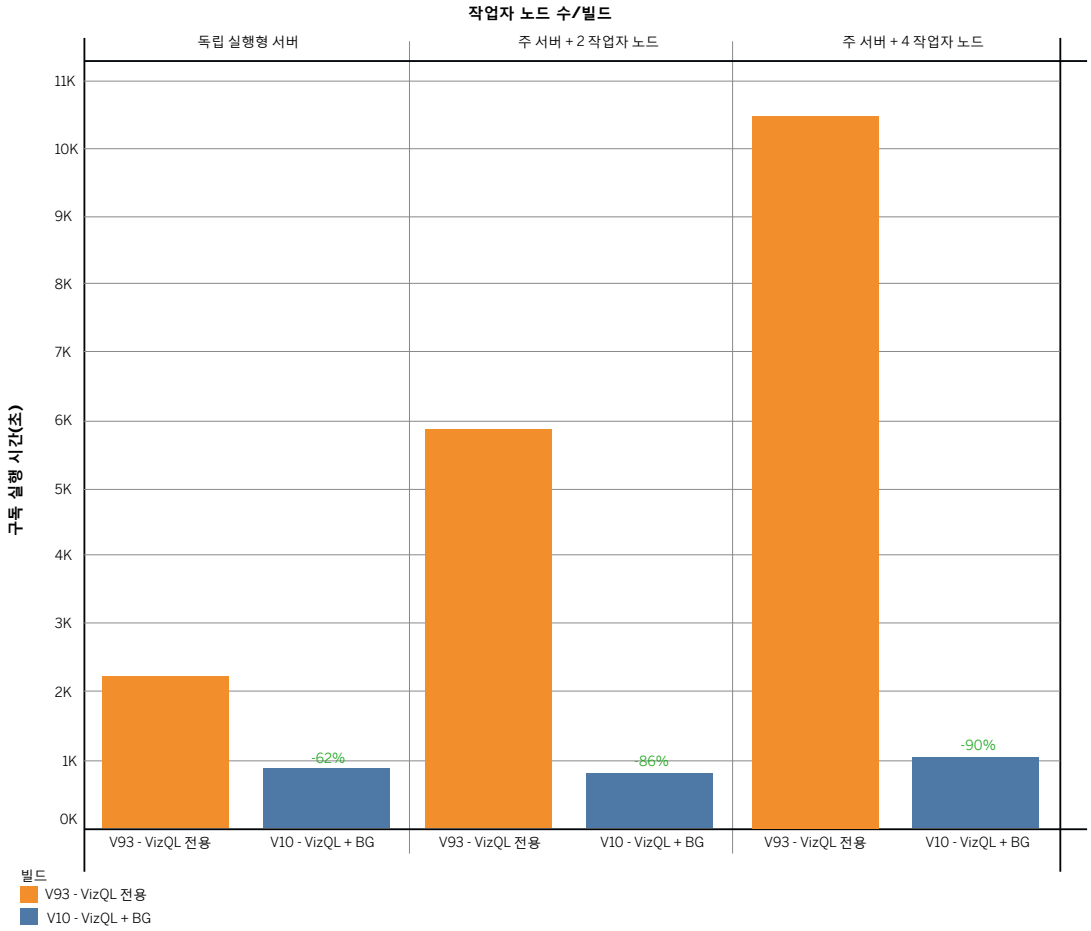

그림 13: 구독 실행 시간(완료까지 걸리는 시간): 9.3 및 10

이는 동일한 일정으로 동일한 통합 문서에 보내는 알림의 경우 완료되는 데 시간이 훨씬 덜 걸릴 것임을 의미합니다. 하지만 다른 통합 문서에 알림을 보내거나 사용자 필터가 설정된 경우 사용자에게 알림이 전달되기 전에 구독 처리 및 컴퓨팅 시간이 소요됩니다. 다양한 통합 문서와 사용자 필터를 사용하려면 Tableau Server에서 전체 사용자 비주얼라이제이션 프로세스를 실행해야 하는데, 여기에는 각 통합 문서나 필터링된 뷰에 대한 데이터 쿼리 및 데이터 시각화 프로세스가 포함됩니다.

구독의 이점은 비즈니스 사용자에게 원하는 데이터를 시기적절하게 제공한다는 것입니다. 효과적인 추출 새로 고침을 사용하면 조직에서 적절한 최신 데이터를 기반으로 올바른 결정을 내릴 수 있습니다. 백그라운더 프로세스는 이 2가지 주요 기능을 관라하므로 동일한 일정에 연결된 중복 작업과 같은 구독 일정을 계획할 때 캐시의 이점을 확실히 이용할 수 있습니다.

백그라운더 프로세스 격리

백그라운더 프로세스를 자체 작업자 노드에서 격리한 상태로 동일한 구독 실험을 수행했습니다. 그 결과 백그라운더 작업자 노드가 단일 4코어 시스템에서 400개의 구독을 완료할 수 있음을 확인했습니다. 이 수치는 단일 백그라운더 프로세스가 4대의 작업자 노드 컴퓨터에서 각 VizQL 작업자 노드와 함께 배치되었을 때 기록된 구독 수와 같습니다.

여기서 중요한 점은 백그라운더 프로세스 자체가 확장되는 동안 격리된 프로세스도 유사하게 확장되지만 최종 사용자 서비스 품질에는 영향을 주지 않는다는 것입니다. 백그라운더 프로세스는 단일 스레드 방식이며 가능한 한 신속하게 작업을 완료하도록 설계되었습니다. 이러한 디자인으로 백그라운더 프로세스는 처리할 작업이 있을 때 코어를 모두 사용합니다. 격리된 시스템에서는 백그라운더 프로세스의 공격적인 컴퓨팅 사용이 다른 사용자 관련 Tableau 서비스를 방해하지 않습니다.

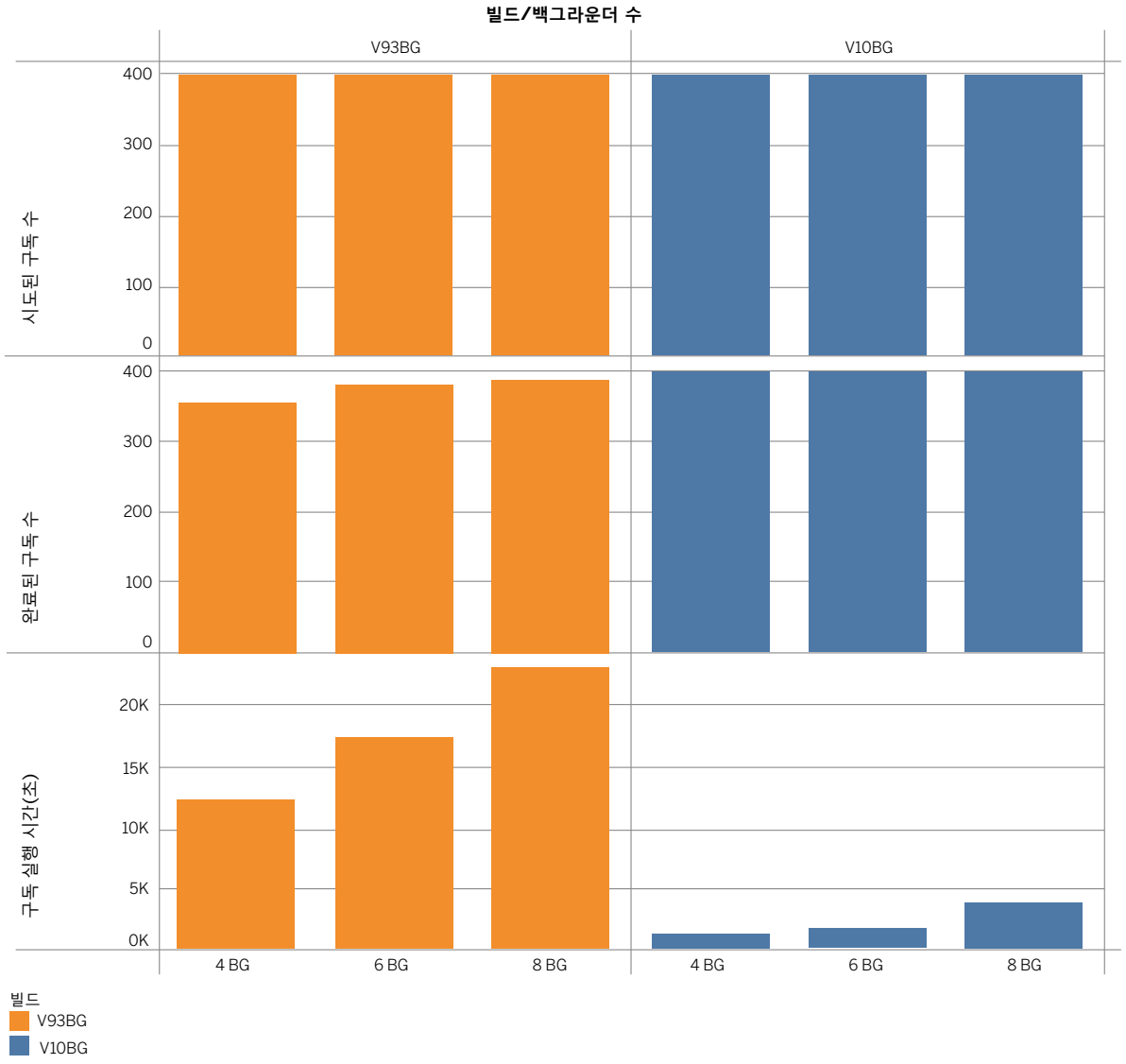

그림 14: 4코어 컴퓨터에 백그라운더 프로세스 추가: 9.3 및 10

그림 14에 표시된 대로 단일 4코어 컴퓨터에 백그라운더 프로세스를 추가하면 Tableau Server 10(파란색)이 9.3(주황색)과 비교하여 상당히 적은 시간 내에 구독을 완료할 수 있었습니다. 하지만 물리적 코어의 한계가 있는 컴퓨터에 계속 백그라운더 프로세스를 추가하면 부정적인 결과가 발생합니다. 표시된 대로 4코어로 제한된 컴퓨터에 8개의 백그라운더 프로세스를 추가하면 완료 시간이 느려집니다. 이는 백그라운더의 단일 스레드 디자인 때문입니다.

마지막으로 격리된 백그라운더 프로세스를 실행하는 작업자 노드에서 추출 새로 고침을 테스트한 실험에서는 Server 9.3과 Server 10이 비슷했음을 확인했습니다. 두 버전 모두 같은 시간 정도 내에 같은 개수의 추출 새로 고침을 완료했습니다. 추출 새로 고침을 위한 확장에서 고려할 중요한 사항은 추출 새로 고침이 충분한 성능을 내는 데 외부 데이터베이스에 대한 의존도가 높다는 것입니다. (이 테스트에서는 MS SQL Server에서 게시된 추출을 사용한 통합 문서를 사용하여 데이터를 새로 고쳤습니다.) 추출 새로 고침 성능과 확장은 데이터베이스 하드웨어 사양에 대한 의존도가 높습니다. 또한 조인 유형 및 실행되는 쿼리의 복잡성 등 데이터 특성이 확장에 영향을 줍니다. 이런 이유로 추출 새로 고침과 알림에 사용할 용량이 충분한지 확인해야만 시스템에 최종 사용자 로드가 급격히 증가하기 전에 작업을 완료할 수 있습니다.

백그라운더 고려 사항

백그라운더 프로세스는 추출 새로 고침, 구독 및 기타 예약된 백그라운드 작업과 관련된 많은 작업을 수행합니다. 이러한 작업을 사용량이 적은 시간에 실행되도록 예약하면 용량을 놓고 경쟁하는 상황이 발생하지 않습니다. 사용량이 적은 시간에 예약할 수 없을 경우에는 백그라운더 및 기타 사용자와 관련되지 않은 워크로드가 사용자 관련 프로세스와 함께 실행되는 데 필요한 용량을 계획하여 추가해야 합니다.

백그라운더는 가능한 한 빨리 작업을 완료해야 하기 때문에 각 프로세스가 전체 코어 용량을 사용하도록 설계되었습니다. 백그라운더 프로세스를 여러 개 실행할 때는 백그라운더 프로세스가 같은 컴퓨터에서 실행 중인 다른 서비스와 컴퓨팅 및 네트워크 리소스를 두고 경쟁할 수 있다는 사실을 고려해야 합니다.

모범 사례 – DIY 확장 테스트

여러분의 환경에서 고유한 워크로드로 로드 테스트를 실행하여 Tableau Server 확장성을 알아보려는 경우 다음과 같은 몇 가지 모범 사례를 참조할 수 있습니다.

1. **Tableau Server를 블랙박스로 취급하지 않아야 합니다.** 기존 로드 테스트는 테스트 중인 애플리케이션을 블랙박스로 취급하는 경우가 많습니다. 이는 배포를 조정/구성 또는 수정하지 않아도 로드 조건이 충족된다고 가정하는 것입니다. Tableau는 확장이 용이하도록 설계되었으며, Tableau 아키텍처에 대한 이해를 통해 확장성 테스트에 정보를 제공하여 상황에 맞는 결과를 유도할 수 있습니다.

2. **테스트에 올바른 도구를 선택해야 합니다.** Tableau Server는 리소스 중심의 복잡한 작업을 수행합니다. Tableau Server의 로드를 낮춰줄 수 있는 도구는 많습니다. Tableau에서 이러한 도구를 직접 지원하지는 않지만 사용하기 쉽고 운영 환경에 가장 잘 맞는 도구를 선택해야 합니다. 또한 로드 테스트를 수행할 때 사용 가능한 도구 선택 및 Tableau Server에 대한 적절한 전문 지식을 가지고 있어야 합니다. 이 테스트에서는 Tabjolt를 사용했습니다. TabJolt는 JMeter 기반의 포인트 앤 런(point and run) 테스트 도구로서 Tableau 같은 애드혹 분석 솔루션과 작동하면서 스크립트 유지 관리를 없애도록 만들어졌습니다.

3. **전형적인 통합 문서를 선택해야 합니다.** 성능 및 확장 문제의 원인 대부분은 사용되는 통합 문서가 모범 사례에 따라 작성되지 않았기 때문입니다. 통합 문서에서 단일 사용자 테스트의 응답 시간이 매우 느리면 로드 테스트 프로젝트를 착수하기 전에 통합 문서를 최적화해야 합니다. 성능이 떨어지는 대시보드를 운용 환경에 유지하지 않듯이 이를 테스트에 이용하지 않는 것이 좋습니다.

4. **기본값으로 시작해야 합니다.** 라이브 연결을 사용하여 통합 문서를 테스트할 때는, Tableau Server 9.0에 병렬화 기능이 도입되어 이전 버전의 Tableau Server에 배포되었던 VizQL 서버가 모두 필요하지 않을 수 있다는 점을 유념해야 합니다. 새로운 두 프로세스 기본 구성으로 시작하여 필요에 따라 점진적으로 확장하십시오.

실제 환경에서의 최적화 모범 사례

성능을 크게 개선하고 평균 응답 시간을 줄일 수 있는 모범 사례는 최적으로 설계된 시스템 외에도 여러 가지가 있습니다.

- 아름다움과 성능을 고려하여 통합 문서를 디자인합니다. 통합 문서가 느린 경우는 대부분 성능을 염두에 두지 않고 디자인했기 때문입니다. 단일 사용자 로드 시간이 느리다면 로드 양이 많은 상태에서 통합 문서 응답 시간도 느립니다. 분석 문화를 채택하면서 동시에 사용자가 멋지고 통찰력이 있으며 성능이 우수한 통합 문서를 디자인할 수 있도록 관리자가 도울 방법과 인력을 제공하면 확장 가능한 비주얼라이제이션을 구축하고 제공할 수도 있습니다. 효율적인 통합 문서 디자인은 성능이 우수하고 효율적인 대시보드 구축을 더욱 심도 있게 다루는 백서입니다.

- 총 응답 시간과 최종 사용자 환경은 여러 가지의 조합이지만 기본적으로 Tableau에서 처리하는 데 걸린 시간과 데이터 검색의 결합입니다. 백엔드 데이터베이스가 느리거나 쿼리 시간이 느리다면 비주얼라이제이션도 느릴 것입니다. 이는 데이터 전략에서 고려해야 할 중요한 요소입니다. 조직의 데이터 원본은 많은 경우 정리가 필요하고 공유되어야 합니다. 데이터가 비즈니스 사용자 생산성을 지원하는 방식으로 데이터를 제공하는 것이 중요합니다. 이는 데이터 최적화를 의미합니다. 예를 들어 색인이 생성된 테이블의 쿼리 속도를 높이려면 최적화된 조인과 적절한 집계가 보장되어야 합니다. 비주얼라이제이션과 성능의 우수성을 유지하려면 적합한 데이터 정리 프로세스를 갖추는 것이 중요합니다.

- Tableau 데이터 추출을 사용합니다. 데이터베이스의 쿼리 속도가 느리면 추출을 사용하여 쿼리 성능을 향상시킬 수 있습니다. 추출은 서버에 로컬로 저장되며 메모리에서 실행되므로 데이터베이스에 요청하지 않아도 사용자가 데이터에 빠르게 액세스할 수 있습니다. 또한 쉽게 필터링 및 집계될 수 있으며 사용자에게 행 수준 세부 정보가 필요하지 않을 때 적합합니다. 추출은 응답 시간을 크게 향상하며 사용자가 분석 흐름을 시작할 수 있습니다.

- 사용량이 적은 시간에 업데이트를 예약합니다. 데이터 원본은 실시간으로 업데이트되는 경우가 많지만 사용자는 일 단위 또는 주 단위로만 데이터가 필요합니다. 사용량이 적은 시간에 추출을 예약하면 데이터베이스와 Tableau Server 모두에서 사용량이 많은 시간에 수행되는 로드를 줄일 수 있습니다. 또한 코어 용량이 충분하면 기존 시스템에 백그라운더를 추가하거나 전용 하드웨어를 사용할 수 있습니다. 보다 신속하게 추출을 완료하려면 이 방법을 사용하는 것이 좋습니다.

- 사용량이 피크일 때는 '비용이 많이 드는' 작업을 피합니다. 게시 작업, 특히 대형 파일의 게시 작업은 리소스를 많이 사용합니다. 대개 간단한 요청으로 이러한 게시 작업 습관을 바꿀 수 있습니다. 사용자에게 사용량이 적은 시간에 게시하고 월요일 아침처럼 바쁜 시간을 피해달라고 요청하면 됩니다. 서버가 가장 많이 사용되는 시간을 알아보려면 관리자 뷰를 사용하여 실제 사용량을 기반으로 정책을 만드십시오. Tableau Server 10.0의 구성 방식에 따라, 고가용성을 위해 각 클러스터 노드에 수행되는 추출 복사도 게시 작업에 포함됩니다. 사용량이 적은 시간에 게시 작업을 수행하면 네트워크 대역폭을 극대화할 수 있습니다.

- 뷰를 캐시합니다. 다수의 사용자가 Tableau Server에 액세스하기 시작하면 공유 리소스 경합이 발생하기 때문에 처음에는 응답 시간이 느려집니다. 캐싱 기능을 설정하면, 시스템으로 들어오는 각 요청의 뷰가 캐시되어 다음에 동일한 대시보드를 보는 사용자에게 보다 빠르게 제공됩니다.

- Tableau Server 9.0에 도입된 캐시 서버 프로세스는 추출 새로 고침이 완료된 후에 공통 뷰의 이메일을 예약하면 시작할 수 있습니다. 이렇게 하면 이후에 보는 사용자의 경우 이전 요청에서 캐시된 데이터를 사용합니다. 정기적으로 많은 양의 트래픽이 발생하는 주요 비주얼라이제이션을 로드하는 자동화 도구 등 다른 접근법으로 캐시를 시작할 수도 있습니다. 사용자는 언제든지 외부 쿼리 캐시를 수동으로 무효화하여 데이터 원본에서 데이터를 새로 고칠 수 있습니다. 이렇게 하면 캐시가 다시 생성됩니다. 이러한 방법을 통해 특정 버전의 데이터가 캐시에 이미 있는지와 관계없이 항상 최신의 데이터 사본을 가져올 수 있습니다.

요약

Tableau Server 10은 모든 규모의 조직을 지원할 수 있도록 확장 가능한 엔터프라이즈급 플랫폼입니다. 온프레미스, 프라이빗 클라우드 또는 퍼블릭 클라우드에서 실행될 수 있으며, 작업자 노드 용량 추가에 따라 선형으로 확장할 수 있습니다. 각 환경에 자체적으로 고유한 특성과 구성을 가질 수 있지만 Tableau Server의 아키텍처를 사용하면 배포를 확장하여 사용자 요구를 충족할 수 있습니다.

확장성과 성능 이점이 다를 수 있고 특별히 모든 상황에 권장되지는 않지만 Tableau Server 10은 8~16코어 용량에서 25~100명의 사용자가 있는 팀이나 부서를 지원할 수 있습니다. 100~1000명의 사용자가 속한 팀을 고려하는 경우 사용량과 데이터 최신성 요구에 따라 16~24코어로 시작하는 것이 좋습니다. 사용자를 더 지원하거나 추가 백그라운드 로드가 필요한 경우 작업자 노드를 추가하여 32~64코어 이상으로 배포를 증가시켜서 더 많은 워크로드를 지원하는 클라우드 규모까지 확장할 수 있습니다.

Tableau 정보

Tableau를 사용하면 데이터를 비즈니스를 개선할 수 있는 실행 가능한 인사이트로 전환할 수 있습니다. 데이터가 저장된 위치나 형식에 상관없이 손쉽게 연결하고, 애드혹 분석을 빠르게 수행하여 숨겨진 기회를 발견하게 됩니다. 드래그 앤 드롭으로 고급 시각적 분석 기능을 갖춘 대화형 대시보드를 작성하여 조직 전체에서 공유하고, 팀원들이 데이터에 대한 고유한 관점을 탐색하도록 도와줍니다. 글로벌 기업부터 신생 기업 및 소규모 비즈니스에 이르기까지 전 세계 모든 사람들이 Tableau의 분석 플랫폼을 사용하여 데이터를 보고 이해하고 있습니다.

리소스

엔터프라이즈용 Tableau: IT 개요

서버 관리자 가이드

Tableau Server 10.0 고가용성: 규모에 맞는 필수 분석 기능 제공

Amazon Web Services에서의 Tableau