Richtlinie zum „Software-Sprint“

B3scale – Projekttitel

Schlussbericht

Zuwendungsempfänger:

Annika Hannig

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS17S03 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei der Autorin.

Kurze Darstellung der Aufgabenstellung und Motivation

Seit Beginn der Covid-19 Pandemie wurde der Bedarf an Lösungen für den digitalen Remote-Unterricht kontinuierlich größer. BigBlueButton hat sich dabei als gute, nicht proprietäre und datenschutzfreundliche Lösung im Einsatz an Bildungs- und Schuleinrichtungen etabliert.

Eine besondere Herausforderung ist dabei ein effizienter Betrieb von großen Server-Clustern um wirtschaftlich konkurrenzfähig und somit eine attraktive Alternative zu bestehenden nicht-freien Angeboten zu bieten.

Bestehende Lösungen hatten sich dabei in der Praxis als instabil gezeigt oder hatten einen unzureichenden Funktionsumfang.

Die bisher am weitesten verbreitete Skalierungslösung „Scalelite“ ermöglichte zwar einen Betrieb von BigBlueButton-Clustern zur Last-Verteilung, allerdings musste für jede Einrichtung ein separates Cluster zur Verfügung gestellt werden. Dies führte dazu, dass viele Instanzen weitestgehend ungenutzt verblieben.

Für eine effiziente Auslastung mussten alle Einrichtungen das selbe Cluster ohne gegenseitige Interferenz nutzen können. Grundsätzlich sollte dabei allerdings die Kompatibilität mit Scalelite erhalten bleiben.

Weiter sollten für unterschiedliche Integrationen individuell einstellbare Optionen wie die hinterlegte Standard-Präsentation ermöglicht werden.

Eine weitere Anforderung war die Verteilung von Meetings anhand von Kriterien die für jedes Frontend einstellbar sein mussten um beispielsweise dedizierte Bereiche zeitweise zu reservieren.

B3scale selbst musste ebenfalls einen Cluster-Betrieb ermöglichen um Ausfallsicherheit zu gewährleisten.

Die Konfiguration sollte mittels einer API erfolgen um eine Automation des Betriebs zu ermöglichen.

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Kommunikation ist ein grundlegendes Bedürfnis des Menschen und ein fundamentaler Stützpfeiler unserer Gesellschaft.

Der Betrieb von Kommunikationsinfrastruktur ist etwas, was allen Menschen zu gute kommt. Dabei ist Offenheit der Schnittstellen und Offenheit der verwendeten Software wichtig um die Integrität und Sicherheit zu gewährleisten und Zugangshürden abzubauen.

B3scale ist dabei ein weiteres Werkzeug um die Verbreitung von offenen Lösungen wie BigBlueButton zu fördern. Dies wirkt zusätzlich einer Software-Monokultur, die anfällig für Schad- und Erpressungsoftware ist entgegen und fördert die europäische Datensuveränität.

Eine intelligentere Auslastung der Server ist nicht nur wirtschaftlich sondern auch mit Hinblick auf Umweltschutz und Green-IT wünschenswert.

Ausführliche Darstellung der Ergebnisse

B3scale wurde in der Architektur eines eines klassischen Proxy-LoadBalancers implementiert: HTTP-Anfragen werden entgegengenommen und dekodiert. Für das Routing wichtige Attribute wie die "MeetingID" oder die API-Resource werden dabei strukturiert. Mit dieser Anfrage wird nun das Gateway aufgerufen und der Ablauf einer Middleware-Kette angestoßen.

In einzelnen Middlewares wird die Anfrage aufbereitet und nach Bedarf verändert. So kann beispielsweise die Standard-Präsentation bei Erzeugung eines Raumes angepasst werden. Am Ende der Kette sitzt eine Handler-Middleware, welche die BigBlueButton-API implementiert. In den meisten Fällen bedeutet das, ein Backend auszuwählen und die Anfrage dort hin weiterzuleiten. Anfragen, die aus dem Zustand der Applikation direkt erzeugbar sind werden ohne hinzuziehen der BigBlueButton-Backends beantwortet. Dazu gehört unter anderem die Auflistung der derzeit laufenden Meetings im Cluster.

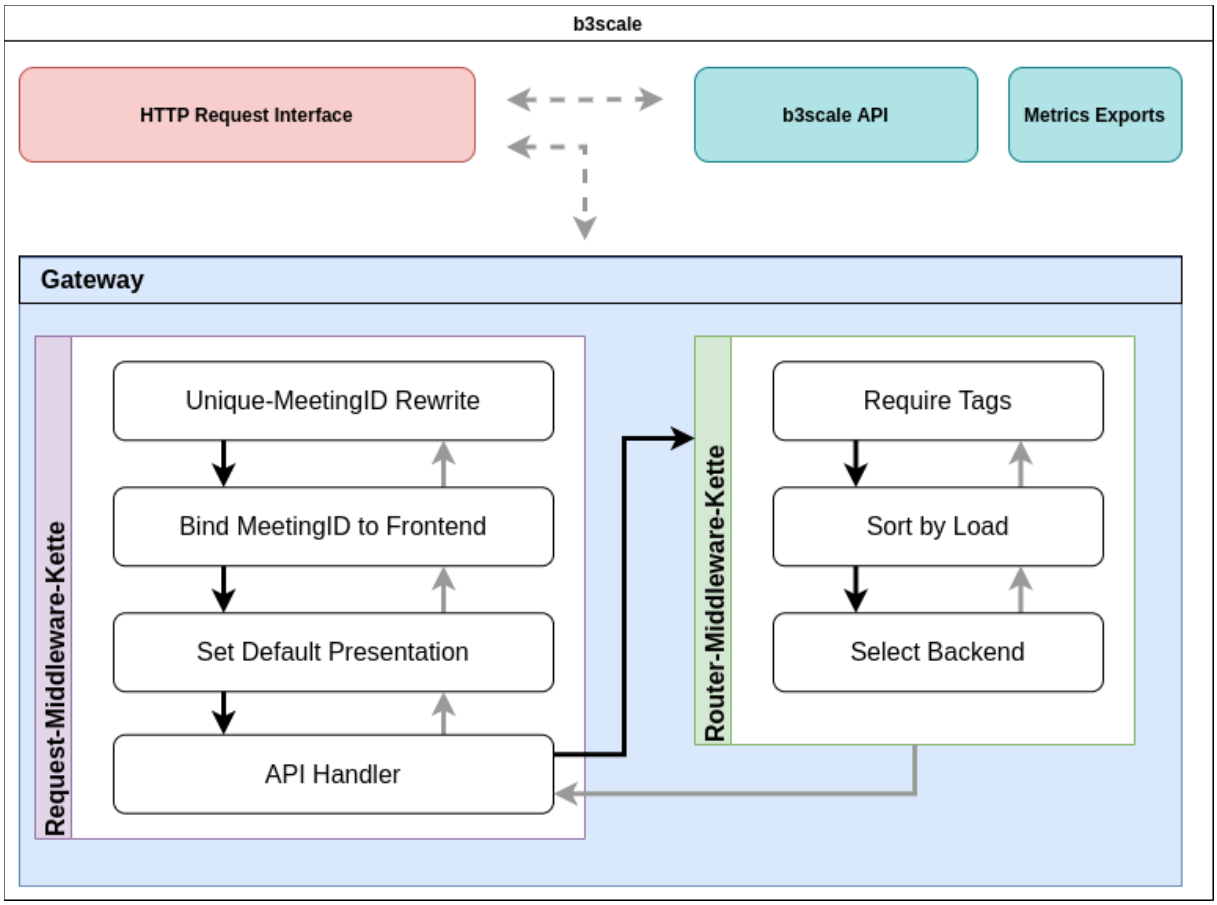


Abb. 1: Middleware-Ketten zur Verarbeitung einer BigBlueButton Anfrage.

Muss zur Beantwortung einer Anfrage auf ein BigBlueButton-Backend zugegriffen werden, so muss dies zunächst durch den Router ausgewählt werden. Die Anfrage wird dabei, analog zum Gateway, durch eine Middleware-Kette geleitet und BigBlueButton-Backends anhand von Kriterien selektiert oder verworfen.

Hier wird ebenfalls die Sortierung der Backends nach Last und die Partitionierung des Clusters mittels Tags durchgeführt.

Einzelne Middleware-Module, wie beispielsweise das zum Einstellen der Standard-Präsentation bei Raum-Erzeugung erfordern eine persistente Konfiguration zur Laufzeit des Clusters. B3scale stellt allen Middlewares eine Schnittstelle zur Speicherung und Abfrage von Einstellungen zur Verfügung. Diese Schnittstelle wird ebenfalls über die HTTP-API angeboten. Diese wurde nach dem Prinzip einer RESTful-API implementiert. Die Authentifikation erfolgt mittels JsonWebToken. Als Austauschformat wird JSON verwendet.

Genutzt wird diese API von der b3scalecli Terminal-Anwendung, um Backends und Frontends zu verwalten und Konfigurationsvariablen zu setzen.

Durch die API kann die Verwaltung des BigBlueButton-Clusters in Automatisations- und Verwaltungs-Lösungen integriert werden.

Periodische Aufgaben, wie das Abgleichen der Raum-Listen mit den Backends, oder deren Prüfung auf Erreichbarkeit werden nebenläufig, im Hintergrund, durchgeführt.

Durch die Anforderung des Cluster-Betriebs von b3scale, müssen diese Aufgaben auf die einzelnen Instanzen verteilt werden. Dazu wurde eine Job-Queue als Teil des gemeinsamen Zustandes implementiert. Alle Instanzen verarbeiten diese, allerdings kann jede Aufgabe nur durch jeweils eine Instanz gleichzeitig selektiert werden. Ausgenutzt wird dabei der Row-Level-Locking-Mechanismus der Postres-Datenbank.

Die Aufgaben werden dadurch auf die einzelnen b3scale-Instanzen aufgeteilt.

Um eine größtmögliche Stabilität bei der Verarbeitung der Aufgaben zu erreichen, wird die Job-Queue periodisch abgefragt.

Um den Zustand des BigBlueButton-Backends mit dem von b3scale synchron zu halten werden periodisch Meeting-Informationen von Backends abgefragt. Eine hohe Frequenz hätte dabei eine große Server-Last zur Folge. Mit einer niedrigen verliert im Laufe der Zeit die interne Zustandsrepräsentation und damit lastverteilungs-relevante Werte, wie die Anzahl Teilnehmender, ihre Genauigkeit.

Für das Erreichen einer möglichst exakten Abbildung des Backend-Zustandes im Cluster wird durch b3scale auf interne Ereignisse (Events) von BigBlueButton zurückgegriffen: Ein auf dem Backend-Server installierter Dienst (b3scalenoded) abonniert die von BigBlueButton produzierten Ereignisse auf dem lokal laufenden Redis-Cache und aktualisiert den geteilten Zustand des Clusters. Wird ein Raum betreten oder Verlassen, wird so diese Information unverzüglich dem Cluster zugänglich gemacht und kann so in Routing-Entscheidungen einfließen.

Dieser Dienst kann so gestartet werden, dass neue Backends automatisch am Cluster registriert werden. Dies erleichtert die Inbetriebnahme weiterer Instanzen.

Erfassung von Fehlern in Protokollen im Betrieb ist ebenfalls ein wichtiger Bestandteil von b3scale: Detailgrad und Ebene des Protokolls können eingestellt werden. Für die Ausgabe kann dabei entweder eine klassische, textoriertierte oder eine strukturierte Variante gewählt werden: Bei der strukturierten Protokollierung erfolgt die Ausgabe der Nachrichten in einem JSON-Format und kann dadurch im operativen Betrieb besser durch eine zentrale Protokollerfassung verarbeitet werden.

Zentral für den Betrieb von Clustern ist die permanente Erfassung der Zustände der einzelnen Systeme. B3scale stellt dazu über HTTP Metriken zur Verfügung,

die durch Datenbanken wie Prometheus (oder kompatibel) erfasst und als Zeitserien aufgearbeitet werden können.

Zu den exportierten Daten gehören sowohl Kennzahlen über die Applikations-Instanz selbst, wie Speicherverbrauch oder Threads, als auch die Anzahl von Meetings und Teilnehmenden im Cluster.

Alle Meilensteine, die für einen Produktivbetrieb von B3scale notwendig waren konnten erreicht werden. Zu diesem Zeitpunkt fehlt noch eine Unterstützung der in BigBlueButton integrierten Aufnahme-Funktion. Diese befindet sich aktuell noch in Entwicklung, wurde aber aufgrund datenschutzrechtlicher Bedenken mit niedriger Priorität verfolgt.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Die primäre Zielgruppe sind Betreibende von BigBlueButton-Clustern. Diese können bereits jetzt vom Einsatz von b3scale profitieren, wenn sie auf Aufnahmen verzichten können.

Die transparente Entwicklung in einem öffentlichen Repository, führte dazu, dass bereits früh Systemadministrierende auf das Projekt aufmerksam wurden und Probleme meldeten, die schnell behoben werden konnten.

Mögliche Weiterentwicklungen sind die Implementierung einer „Reverse-Proxy-Funktionalität“ zur Maskierung der einzelnen Server. Hier sind ggf. Änderungen an BigBlueButton selbst nötig. Die Integration der Aufnahme-Funktionen von BigBlueButton soll ebenfalls fertiggestellt werden.

Ein besonderer Lern-Erfolg war die effiziente Handhabung von Datenbank-Verbindungen und Transaktionen auf einer niedrigen Ebene in einer verteilten, hoch verfügbaren, Anwendung.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Einige Konzepte, zur optimalen Nutzung von Datenbank-Verbindungen und Transaktion wurden getestet und auf ihre Stabilität unter Last untersucht Verworfen wurde dabei die Nutzung von einer langlebigen Transaktion für jede HTTP Anfrage. Die Verbindungs-Allokation innerhalb der Anfragen-Verarbeitung wurde ebenfalls verworfen. Letztlich wurde eine Verbindungs-Allokation am Beginn der Anfrage und die Verwendung mehrerer möglichst kurzlebiger Transaktionen innerhalb der Anfragenverarbeitung gewählt.

Ein erster Prototyp einer Funktion zur Maskierung der einzelnen BigBlueButton-Instanzen („Reverse-Proxy“)des Clusters wurde implementiert, aber verworfen, weil sich die vollständige Integration als derzeit unpraktikabel ohne Änderungen am Code von BigBlueButton selbst herausgestellt hat.

Die erste Implementierung der Job-Queue nutzte einen Publish-Subscribe Mechanismus der die Einzelnen Instanzen über die Präsenz einer neuen Hintergrundaufgabe informierte. Dieser Mechanismus erwies sich allerdings unter Last als instabil und wurde durch eine periodische Abfrage ersetzt.

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

B3scale wurde von Anfang an in einem öffentlich einsehbaren Gitlab-Repository entwickelt: https://gitlab.com/infra.run/public/b3scale

Ein Umzug zu Github (https://github.com/b3scale/b3scale) zur Senkung der Partizipationshürde ist vorgesehen.

Weiter geplant sind veröffentlichte Container z.b. in der Github-Container-Registry.

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Im wesentlichen verlief alles nach Plan. Das regelmäßige veröffentlichen des Arbeitsfortschrittes im Chat des Prototypefundes hat dabei geholfen den Überblick über noch offene Aufgaben zu behalten.

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Die Entwicklung an Scalelite geht weiter. Die noch ausstehende Unterstützung für Recordings kann noch ein Hinderungsgrund für den Umstieg auf b3scale sein.