Unit -4

BIG DATA ANALYSTICS:

Big data analytics is the often complex process of examining big data to uncover information
-- such as hidden patterns, correlations, market trends and customer preferences
-- that can help organizations make informed business decisions.
On a broad scale, data analytics technologies and techniques give organizations a way to analyze data sets and gather new information. Business intelligence (BI) queries answer basic questions about business operations and performance.

Clustering in Big Data:

Clustering is a popular unsupervised method and an essential tool for Big Data Analysis. Clustering can be used either as a pre-processing step to reduce data dimensionality before running the learning algorithm, or as a statistical tool to discover useful patterns within a dataset

Properties of Clustering :

1. Clustering Scalability: Nowadays there is a vast amount of data and should be dealing with huge databases. In order to handle extensive databases, the clustering algorithm should be scalable. Data should be scalable, if it is not scalable, then we can't get the appropriate result which would lead to wrong results.

2. High Dimensionality: The algorithm should be able to handle high dimensional space along with the data of small size.

3. Algorithm Usability with multiple data kinds: Different kinds of data can be used with algorithms of clustering. It should be capable of dealing with different types of data like discrete, categorical and interval-based data, binary data etc.

4. Dealing with unstructured data: There would be some databases that contain missing values, and noisy or erroneous data. If the algorithms are sensitive to such data then it may lead to poor quality clusters. So it should be able to handle unstructured data and give some structure to the data by organising it into groups of similar data objects. This makes the job of the data expert easier in order to process the data and discover new patterns.

5. Interpretability: The clustering outcomes should be interpretable, comprehensible, and usable. The interpretability reflects how easily the data is understood.

Clustering Methods:

The clustering methods can be classified into the following categories:

- Partitioning Method
- Hierarchical Method
- Density-based Method
- Grid-Based Method
- Model-Based Method
- Constraint-based Method

Advantages of Cluster Analysis:

- It can help identify patterns and relationships within a dataset that may not be immediately obvious.
- It can be used for exploratory data analysis and can help with feature selection.
- It can be used to reduce the dimensionality of the data.
- It can be used for anomaly detection and outlier identification.
- It can be used for market segmentation and customer profiling.

Disadvantages of Cluster Analysis:

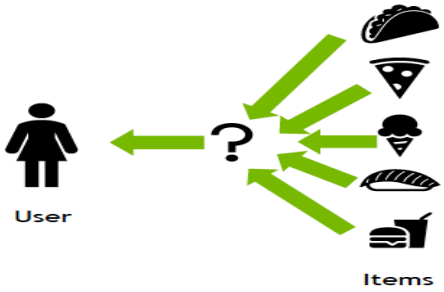
- It can be sensitive to the choice of initial conditions and the number of clusters.
- It can be sensitive to the presence of noise or outliers in the data.
- It can be difficult to interpret the results of the analysis if the clusters are not well-defined.
- It can be computationally expensive for large datasets.

The results of the analysis can be affected by the choice of clustering algorithm used.
It is important to note that the success of cluster analysis depends on the data, the goals of the analysis, and the ability of the analyst to interpret the results.

CLASSIFICATION OF BID DATA RECOMMENDATION SYSTEM :

A recommendation system (or recommender system) is a class of machine learning that uses data to help predict, narrow down, and find what people are looking for among an exponentially growing number of options.

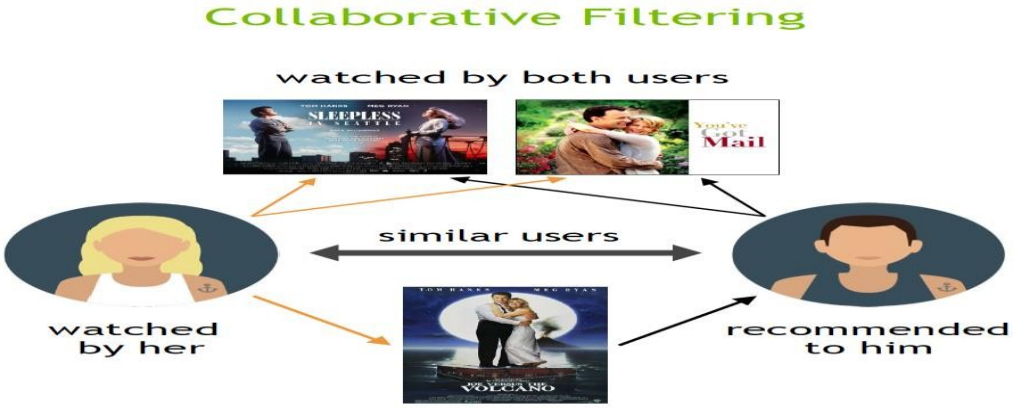
A recommendation system is an artificial intelligence or AI algorithm, usually associated with machine learning, that uses Big Data to suggest or recommend additional products to consumers. These can be based on various criteria, including past purchases, search history, demographic information, and other factors. Recommender systems are highly useful as they help users discover products and services they might otherwise have not found on their own. Recommender systems are trained to understand the preferences, previous decisions, and characteristics of people and products using data gathered about their interactions. These include impressions, clicks, likes, and purchases. Because of their capability to predict consumer interests and desires on a highly personalized level, recommender systems are a favorite with content and product providers. They can drive consumers to just about any product or service that interests them, from books to videos to health classes to clothing.



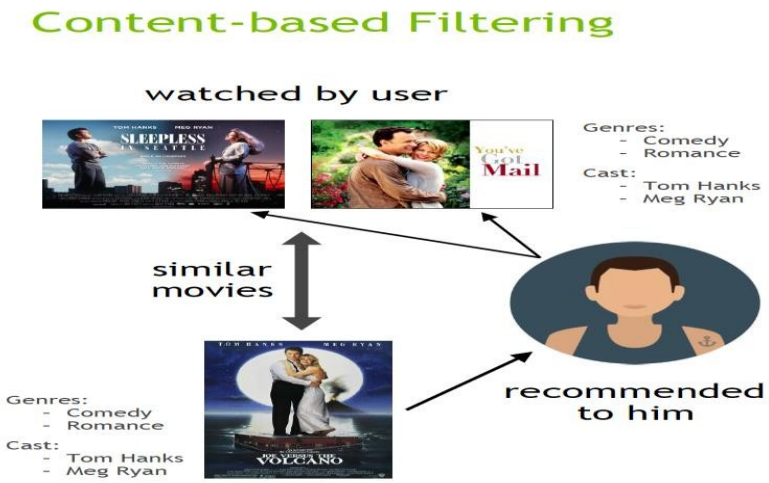
Types of Recommendation Systems
While there are a vast number of recommender algorithms and techniques, most fall into these broad categories:

- collaborative filtering,
- content filtering
- context filtering.

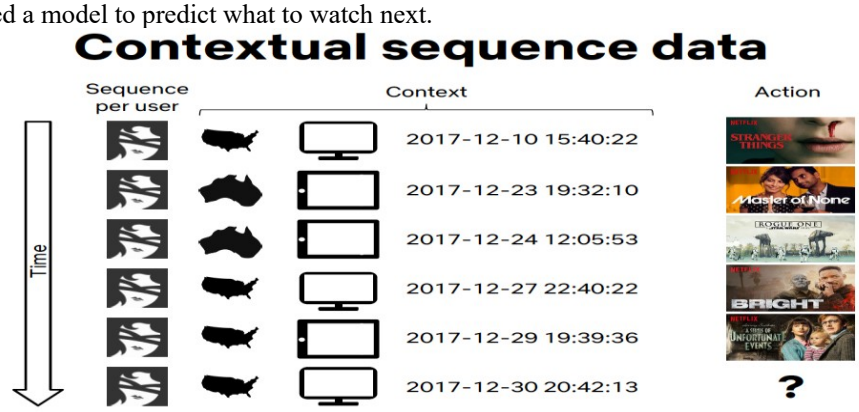
Collaborative filtering algorithms recommend items (this is the filtering part) based on preference information from many users (this is the collaborative part). This approach uses similarity of user preference behavior, given previous interactions between users and items, recommender algorithms learn to predict future interaction. These recommender systems build a model from a user's past behavior, such as items purchased previously or ratings given to those items and similar decisions by other users. The idea is that if some people have made similar decisions and purchases in the past, like a movie choice, then there is a high probability they will agree on additional future selections. For example, if a collaborative filtering recommender knows you and another user share similar tastes in movies, it might recommend a movie to you that it knows this other user already likes.



Content filtering, by contrast, uses the attributes or features of an item (this is the content part) to recommend other items similar to the user's preferences. This approach is based on similarity of item and user features, given information about a user and items they have interacted with (e.g. a user's age, the category of a restaurant's cuisine, the average review for a movie), model the likelihood of a new interaction. For example, if a content filtering recommender sees you liked the movies You've Got Mail and Sleepless in Seattle, it might recommend another movie to you with the same genres and/or cast such as Joe Versus the Volcano.



Hybrid recommender systems combine the advantages of the types above to create a more comprehensive recommending system.
Context filtering includes users' contextual information in the recommendation process. Netflix spoke at NVIDIA GTC about making better recommendations by framing a recommendation as a contextual sequence prediction. This approach uses a sequence of contextual user actions, plus the current context, to predict the probability of the next action. In the Netflix example, given one sequence for each user—the country, device, date, and time when they watched a movie—they trained a model to predict what to watch next.



BENEFITS OF RECOMMENDATION SYSTEMS

Recommender systems are a critical component driving personalized user experiences, deeper engagement with customers, and powerful decision support tools in retail, entertainment, healthcare, finance, and other industries. On some of the largest commercial platforms, recommendations account for as much as 30% of the revenue. A 1% improvement in the quality of recommendations can translate into billions of dollars in revenue.
Companies implement recommender systems for a variety of reasons, including:

- Improving retention. By continuously catering to the preferences of users and customers, businesses are more likely to retain them as loyal subscribers or shoppers. When a customer senses that they're truly understood by a brand and not just having information randomly thrown at them, they're far more likely to remain loyal and continue shopping at your site.
- Increasing sales. Various research studies show increases in upselling revenue from 10-50% resulting from accurate 'you might also like' product recommendations. Sales can be increased with recommendation system strategies as simple as adding matching product recommendations to a purchase confirmation; collecting information from abandoned electronic shopping carts; sharing information on 'what customers are buying now'; and sharing other buyers' purchases and comments.
- Helping to form customer habits and trends. Consistently serving up accurate and relevant content can trigger cues that build strong habits and influence usage patterns in customers.

- Speeding up the pace of work. Analysts and researchers can save as much as 80% of their time when served tailored suggestions for resources and other materials necessary for further research.
- Boosting cart value. Companies with tens of thousands of items for sale would be challenged to hard code product suggestions for such an inventory. By using various means of filtering, these ecommerce titans can find just the right time to suggest new products customers are likely to buy, either on their site or through email or other means.

Multimedia **Introduction:**

Multimedia:
Multimedia is usually recorded and played, displayed, or accessed by information content processing devices, such as computerized and electronic devices, but can also be part of a live performance. Multimedia devices are electronic media devices used to store and experience multimedia content.

Streaming Media:
Streaming media is multimedia that is constantly received by and presented to an end-user while being delivered by a provider.
Its verb form, "to stream", refers to the process of delivering media in this manner; the term refers to the delivery method of the medium rather than the medium itself .

Types of streaming media are Live Streaming, Video Streaming

Live Streaming refers to content delivered live over the Internet, requires a camera for the media, an encoder to digitize the content, a media publisher, and a content delivery network to distribute and deliver the content. In streaming video and audio, the traveling information is a stream of data from a server.

The decoder is a stand-alone player or a plugin that works as part of a Web browser. The server, information stream and decoder work together to let people watch live or prerecorded broadcasts.

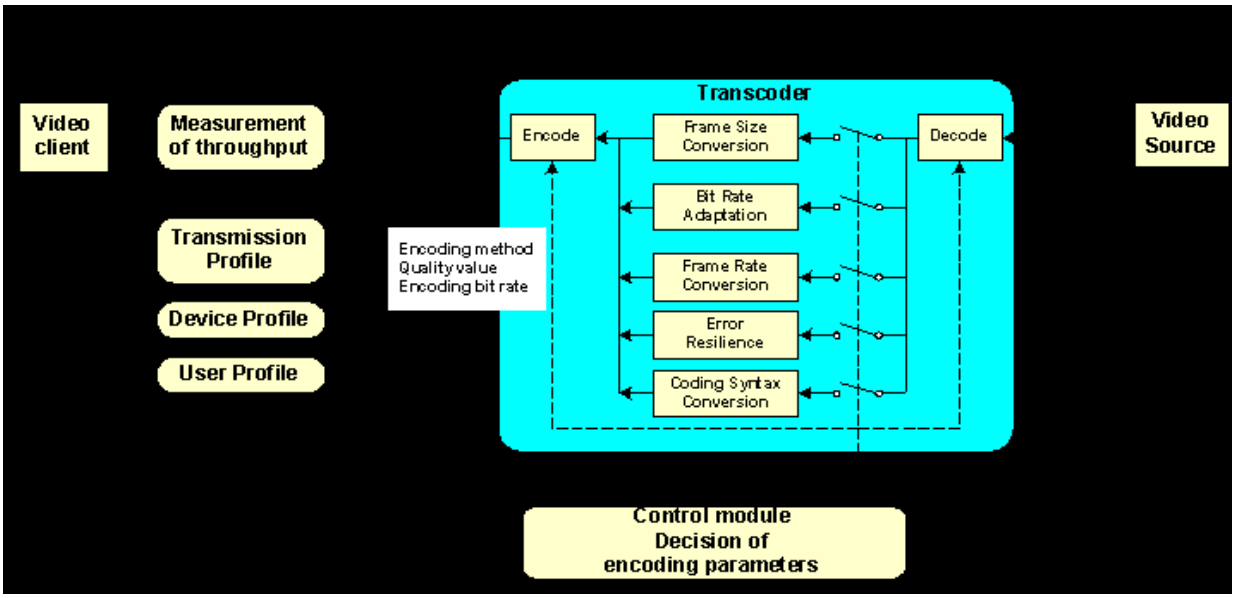
Most streaming videos don't fill the whole screen on a computer. Instead, they play in a smaller frame or window.

If you stretch many streaming videos to fill your screen, you'll see a drop in quality. For this reason, streaming video and audio use protocols that allow the transfer of data in real time.

They break files into very small pieces and send them to a specific location in a specific order. These protocols include:
1. Real-time transfer protocol (RTP)
2. Real-time streaming protocol (RTSP)
3. Real-time transport control protocol (RTCP)

Users expect powerful and stable functions for multimedia videos stability is of greatest importance.
To provide appropriate multimedia files for diversified terminal units.

In the transcoding mode, multimedia files are transcoded dynamically, in order to be applicable to the device side according to the terminal environment.
This mode needs to consider the real-time problem, especially for H.264/SVC coding, as the time required for transcoding causes difficulties for real-time streaming. Although SVC is applicable to varying bandwidth networks due to its multilayer architecture, how to provide a multimedia hierarchy that is suitable for dynamic environment variations according to the terminal unit is an interesting research project.



Basic Modules of the Video Transcoding System

Transmission profile is responsible for monitoring the dynamic condition of the transmission channel, such as effective channel bandwidth, channel error rate, etc.
Device profile describes the capability of the device, such as screen size, processing power, etc.

User profile describes the user preference.

Here the svc transcoding controller transcodes the appropriate video file according to
the mobile device parameters.
based on cloud computing introduces the new concept that uses map reduce to separate the video content into different clips.

The losses in bandwidth can be reduced by this approach. svc plays an important role in this method and
also provides the different formats of the video files with the cloud environment to simulate the overall network environment.
In this schema user wants to download a particular file from the cloud server first the user should register with cloud server, if already exists the device profile then directly get the appropriate video file according
to the parameters of the mobile device.

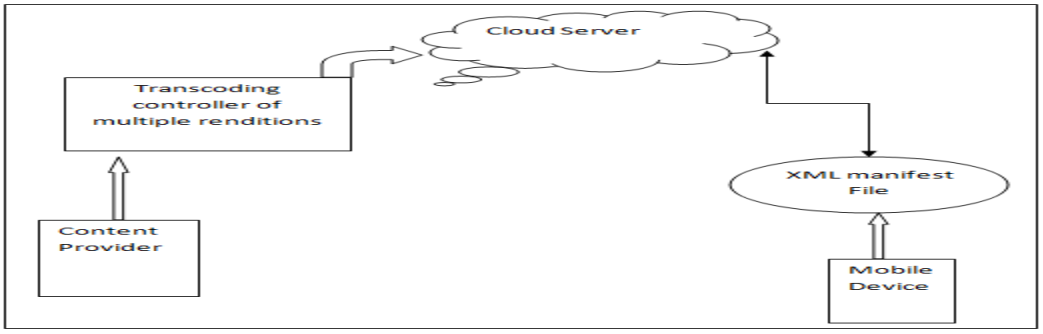
Any mobile device using this service for the first time the cloud will not provide such a profile. so there should be an additional profile examination to provide the necessary information about the mobile device.

Through this functionality the mobile device can generate a schema and send to the profile agent.
The profile agent determines the required parameters and then sends to the NDAMM for identification.
It determines the user profile of the mobile device and then sends to the svc transcoding controller for increasing the efficiency of the video streaming according to the parameters of the mobile device.

Case Study: Performance Evaluation
This Concept mainly tells about how to provide different renditions of the video file while request being sent by the mobile device for varying bandwidth networks. Here we maintain the multiple renditions of single source file in cloud server according to the network device we send the appropriate video files

The proposed system has the following modules:

- Parameter Calculation
- Cloud Service
- Bandwidth settings
- Transcoding
- Adaptive Streaming



Structure of Cloud based Mobile Streaming

A. Parameter Calculation For parameter calculation we can send the mobile device in formation in the form XML manifest file. In this we can set the network parameters according to the device.

There are three types of bandwidths namely existing, average and standard deviation values to calculate the current bandwidth. This type of parameter form is maintained then we can send the device parameters to the network estimation module and device aware Bayesian prediction module for relevant prediction. Here in the xml manifest file we can set the mobile network parameters and the send to the Cloud server then it detects by the server and send the appropriate Video file to the mobile device according to the parameters of the mobile device

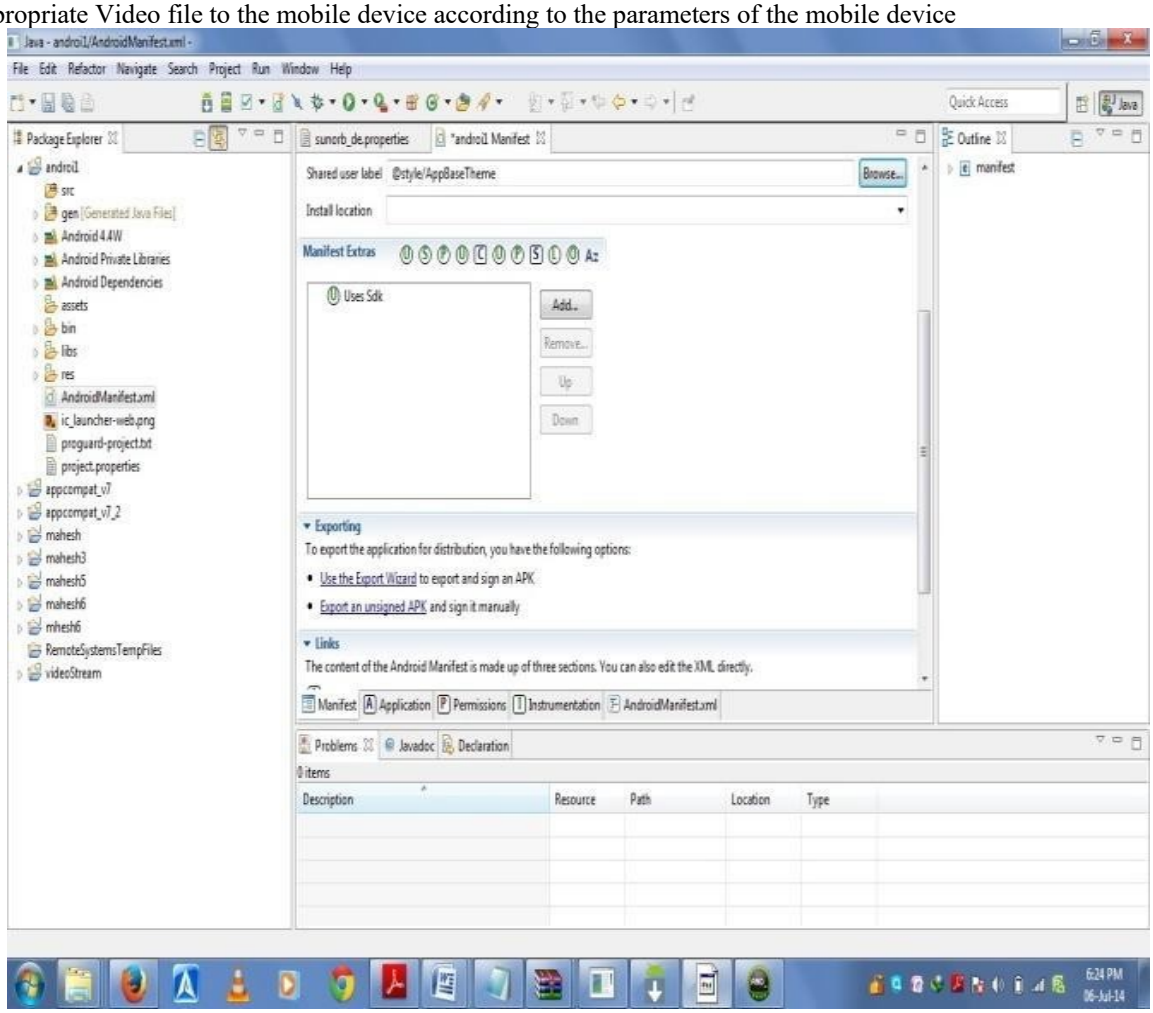


Fig. 3. Android XML Manifest File

B. Cloud Service

In this module we can maintain different video formats of the single video file then we can store on to the cloud

database. Whenever the mobile device sends the parameters to the cloud server the server can detect the user profiles of the particular mobile device and can send the required video file to the mobile device. The use highly scalable, reliable, secure, fast and inexpensive infrastructure.

This service mainly aims to maximize the benefits of the user. The video file was stored according to the bitrates, resolution and frame rate, bandwidth of the width, height, and standard deviation, decoding and encoding formats.

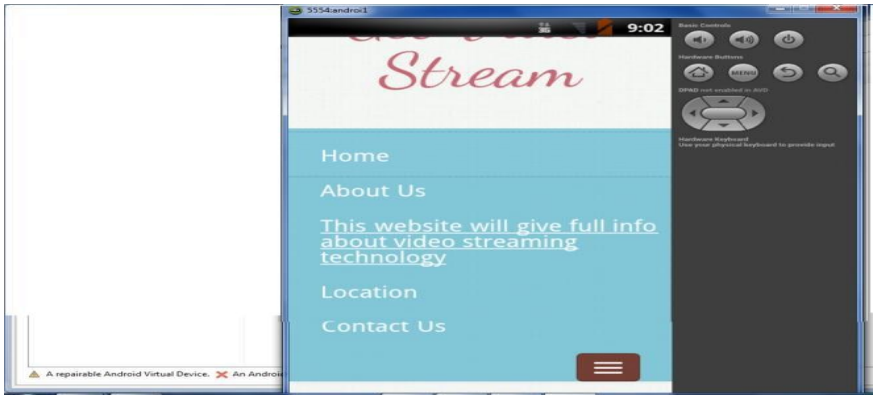
C. Bandwidth Settings

In this module we can estimate the video process according to the frame rate, bit rate and resolution. To decode or encode the video file according to the parameters of the mobile device. Device feature can find by power consumption, device model, device network in order to conform to the real time requirements of the mobile device. In order to conform to the real time requirements of the mobile multimedia, this study adopted Bayesian theory to infer weather the video features are conformed to the decoding action. the inference module was based on the following two conditions: the LCD brightness does not always change this hypothesis aims at a hardware energy evaluation. The literature states that the TFT LCD energy consumption accounts for about 20%-45% of the total power consumption for different terminal hardware environments. Although the overall power can be reduced effectively by adjusting the LCD, with multimedia services, users are sensitive to brightness; they dislike video brightness that repeatedly changes.

D. Transcoding

In this module the transcoding work can be done by the cloud server efficiently according to the bandwidth and network environment. Scalable video coding is an improvement over traditional H.264/MPEG-4 AVC coding, as it has higher coding flexibility. It is characterized by temporal scalability, spatial scalability and SNR scalability, allowing video transmissions to be more adaptable to heterogeneous network bandwidth. Transcoding can be done by the cloud server instantly in this service and moreover this method is very convenient for users to get more benefits.

E. Adaptive Streaming



Adaptive Video Streaming

A good dynamic communication mechanism can reduce the bandwidth needs and the power consumption of the device resulting from excessive packet transmission, and transmission frequency can be determined according to the bandwidth based on such dynamic decision making when the network bandwidth difference exceeds a triple standard deviation, this indicates the present network is unstable. The overall communication frequency shall incline to frequency to avoid errors; however, when the network bandwidth difference is less than a triple standard deviation, the current network is still in a stable state, and the influence on bandwidth difference can be corrected gradually. Ultimately we will get the service very effectively in this method.

Case Study: Live Video Streaming App,

A proof of concept (POC) is an early product version between the design and main development phases. POC has become a common way for startups and established businesses to test whether the idea they have is actually going to work since POC demonstrates that the project can be done. A proof of concept also creates a starting point for the development of the project as a whole.

Businesses need a proof of concept when there is no guarantee that the technical result is achievable due to the leverage of complex architecture and new technologies, what's relevant to video streaming mobile applications. Thus, by developing a POC, developers, and stakeholders get evidence that the project is viable.

Our challenge

Develop the proof of concept of a video streaming application with the basic functionality of a social media application.
To achieve this goal, we needed to:
Implement the following mobile screens and use cases:

- Sign up / Sign in. Users can sign up/sign in to the system.
- View profile. Users can view their and other users' profile data.
- Edit profile. Users can edit their profile data, such as name, avatar, and bio.
- Search. Users can search for other users by name and follow them.
- Start streaming. Users can start real-time video streaming.
- View streamings list. Users can view the list of active streams.
- Join the stream. Users can participate in the streaming of another user as a viewer.

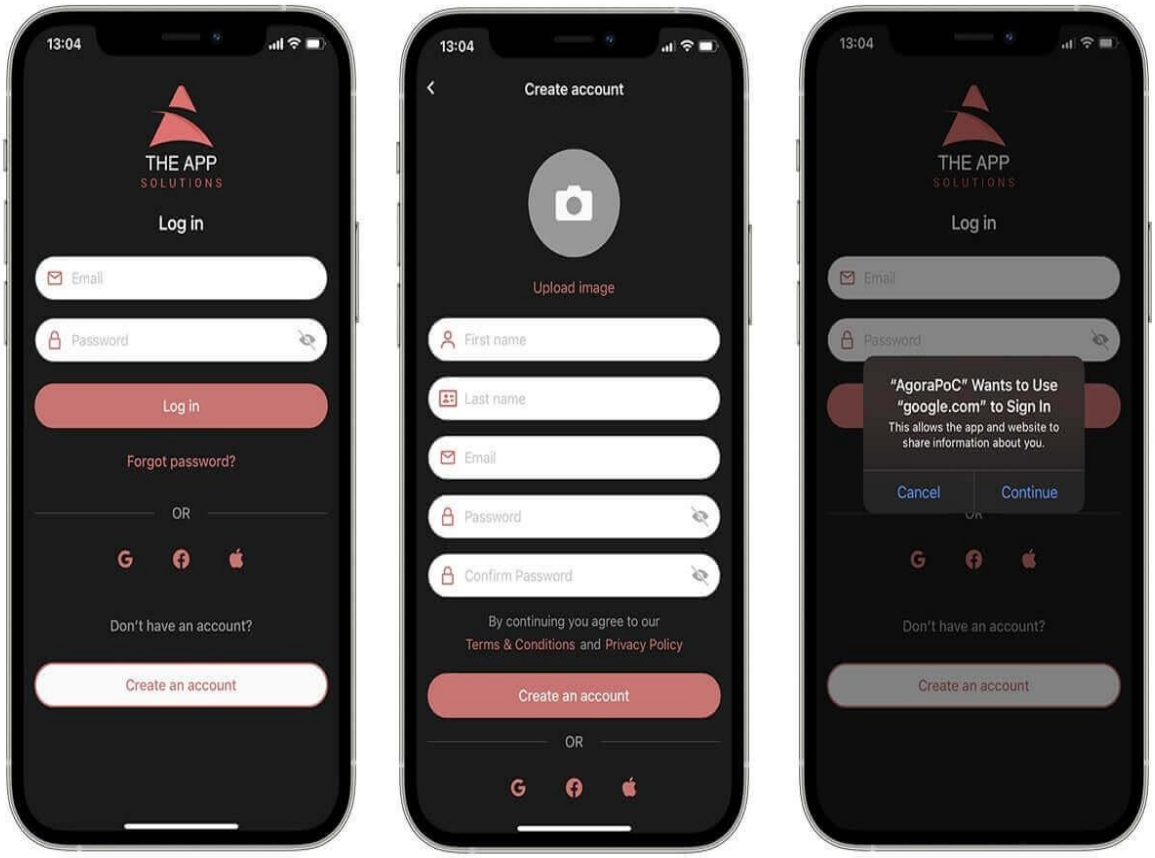
Integrate several authorization methods, such as:

- Email and Password
- Google authorization
- Facebook authorization
- Apple authorization

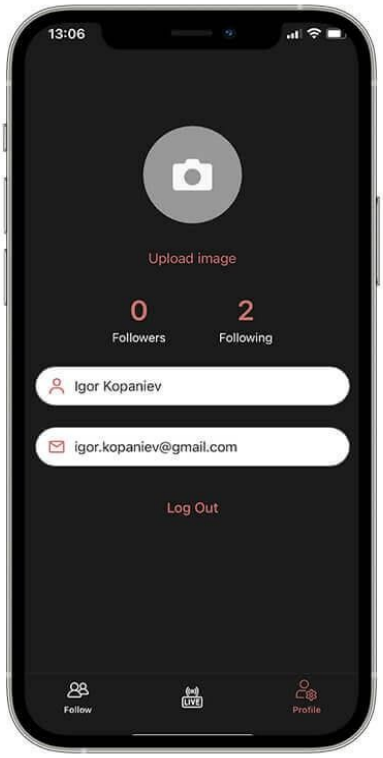
Our Solution - Video Streaming App Proof of Concept

We developed a proof of concept of a video streaming application with the basic functionality of a social media app to show off our tech expertise in live broadcasting and demonstrate how such a project may look.
Implemented features:

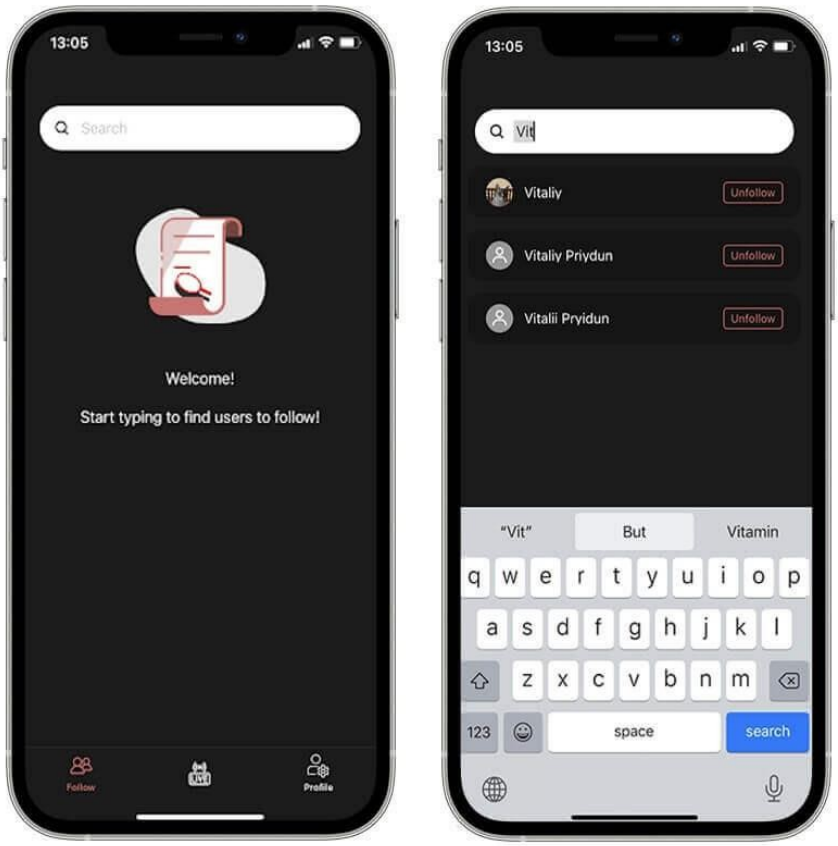
- Sign-in/Sign-up via email and password, Facebook, Google, and Apple ID.



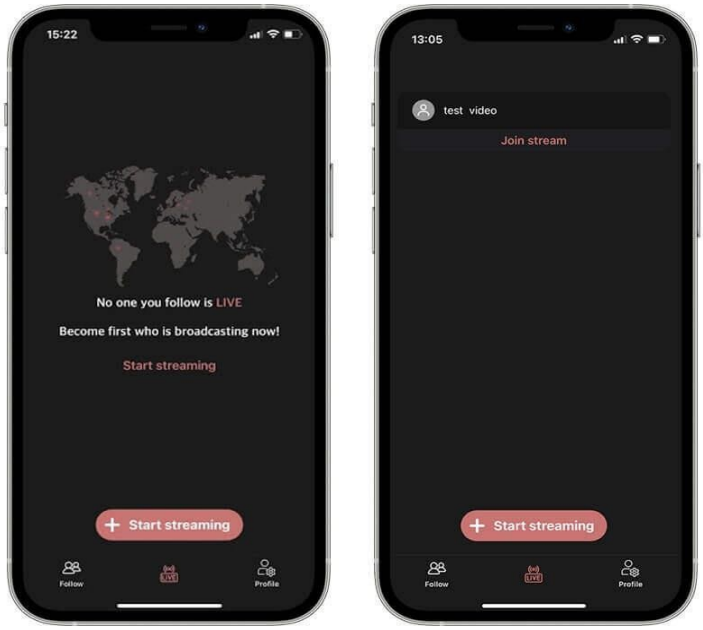
- User Profile



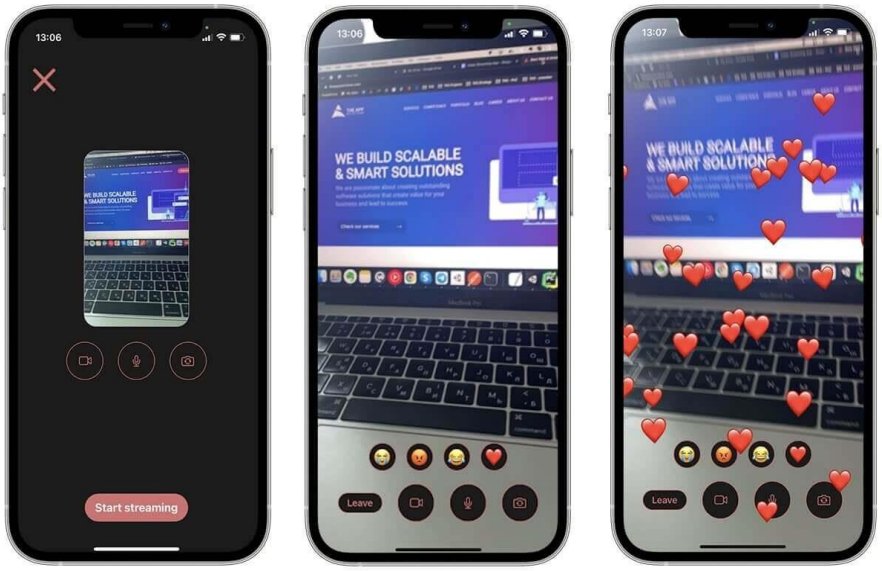
- Search for followers, follow and unfollow functionality



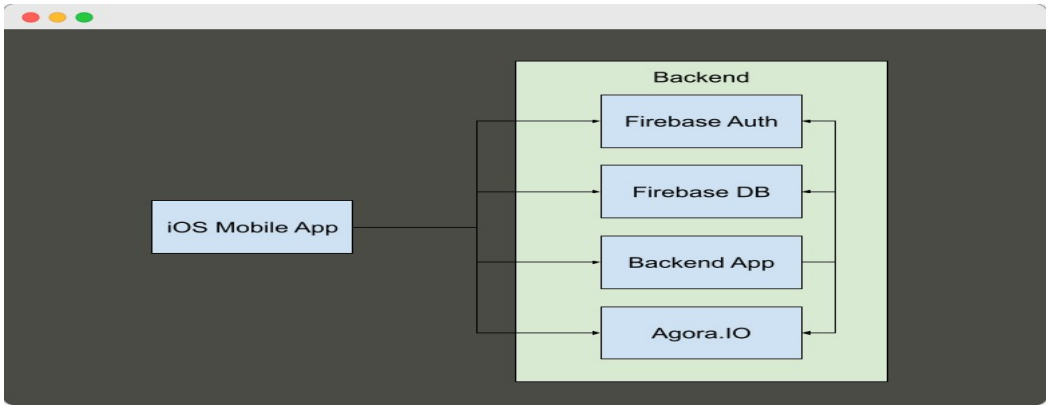
- View the list of active video streams



- Broadcasting videos to subscribers and receiving reactions



HIGH-LEVEL ARCHITECTURE VISION



TECH STACK

- Swift for iOS application
- Firebase Real-time BD supports direct connectivity from mobile and web platforms as well as backend applications
- Firebase for user authentication and authorization, data and image storing
- Google Cloud Platform for hosting app's back-end
- Python for application's back-end
- Agora.IO, a SaaS for video broadcasting and participating in video streaming

CONTRIBUTORS

- iOS developer Vitalii Pryidun
- Backend developer Ihor Kopaniev
- DevOps support Vasily Lebediev

HOW WE DEVELOPED A STREAMING APP PROOF OF CONCEPT

CORE

We built the app's POC using MVP+Router+Configurator architecture, including MVVM+Combine for lists, etc.
We made DI using ServiceLocator Singleton, which is a factory of abstract services.

Main services

- Keychain for saving JWT and Apple sign-in credentials.
- Network, AuthorizedNetwork, TokenProvider, APIErrorParser for executing network requests. All requests have to conform to ***APIRequestProtocol*** or ***APIAuthorizedRequestProtocol*** for requests that include a token into headers.
- TokenProvider for fetching a token from the keychain and refreshing it via Firebase if needed. If your app has to refresh a token using a backend request, go to Core/Networking/TokenProvider and rewrite this service to restore the token manually.
- FirebaseManager for authentication using email+password, verification ID, social media, password reset, logout, etc.
- FirebaseDatabaseManager for obtaining followers list, fetching users, etc.
- FirebaseStorage for setting and fetching an avatar.
- AuthService is just a stub for validating Firebase JWT tokens. If your back-end requires JWT verification, insert a validation request into the validate method.
- SearchService for fetching users with input from the search field.
- FollowService for following/unfollowing the user fetched with SearchService.
- UserService for updating user profile (name etc.).
- StreamService for fetching a token to join agora channel, notifying back-end about start/end of the channel, subscribing to user reactions, sending reactions, etc.

OUR RESULTS

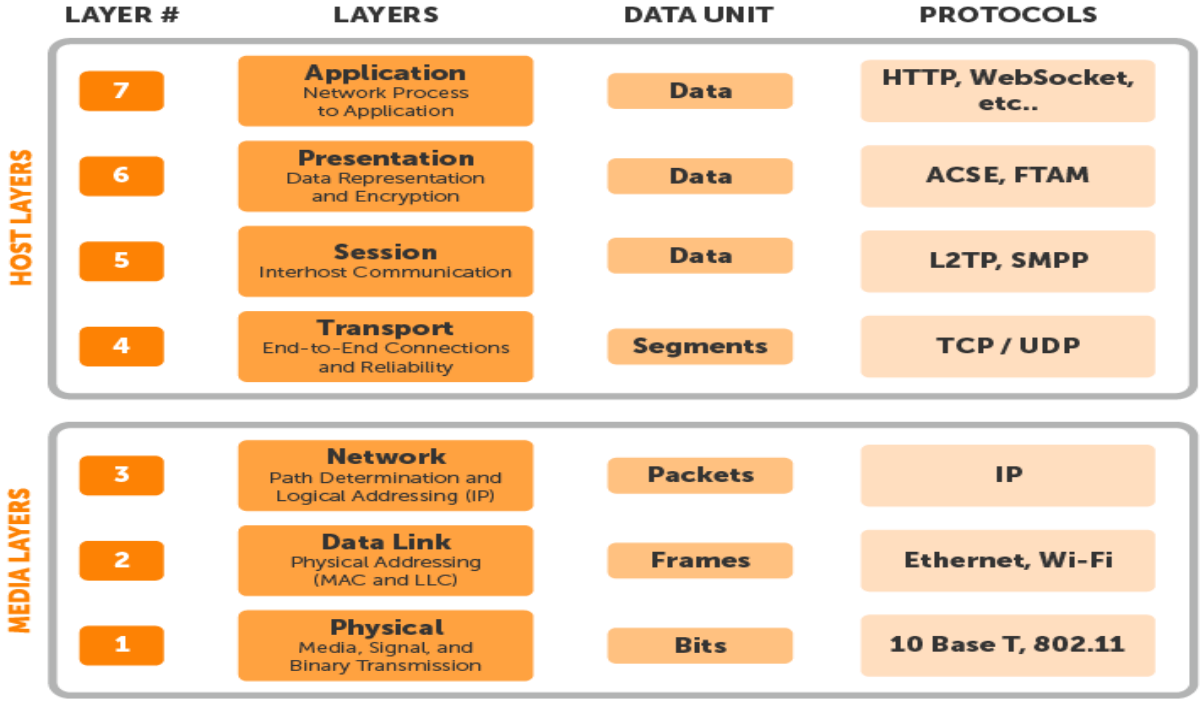
The development of the video streaming app proof of concept gave us the following expertise:

- We integrated video streaming functionality to the POC using Agora.IO SaaS.
- We implemented the authentication and authorization by Firebase Authentication.
- We worked with Firebase Realtime Database, which supports direct connectivity with end-users applications (like mobile, web, etc.) server-side applications.
- We optimized the development process by applying ready-to-use Firebase functionality.
- As a result, we showcased our expertise in video streaming app development.

Case Study: Streaming Protocols

PROTOCOL?

A protocol is a set of rules governing how data travels from one communicating system to another. These are layered on top of one another to form a protocol stack. That way, protocols at each layer can focus on a specific function and cooperate with each other. The lowest layer acts as a foundation, and each layer above it adds complexity.
You've likely heard of an IP address, which stands for Internet Protocol. This protocol structures how devices using the internet communicate. The Internet Protocol sits at the network layer. It's typically overlaid by the Transmission Control Protocol (TCP) at the transport layer, as well as the Hypertext Transfer Protocol (HTTP) at the application layer.



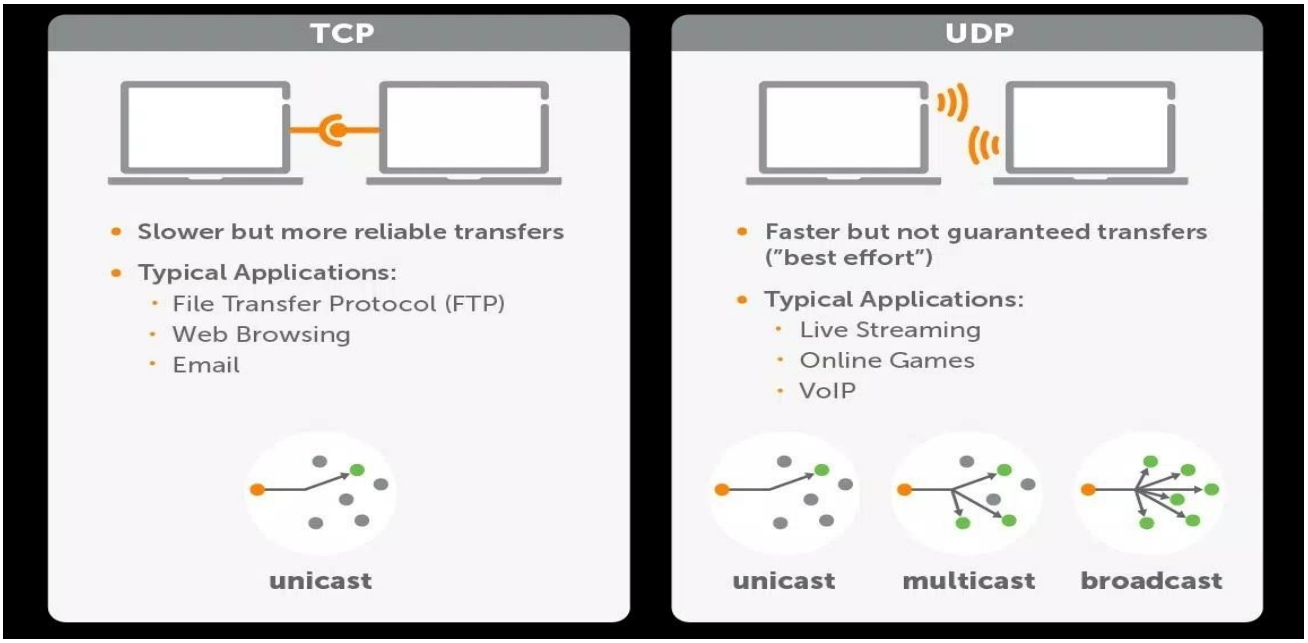
The seven layers — which include physical, data link, network, transport, session, presentation, and application — were defined by the International Organization for Standardization's (IS0's) Open Systems Interconnection model, as depicted above.

WHAT IS A STREAMING PROTOCOL?

Each time you watch a live stream or video on demand, video streaming protocols are used to deliver data over the internet. These can sit in the application, presentation, and session layers.
Online video delivery uses both streaming protocols and HTTP-based protocols. Streaming protocols like Real-Time Messaging Protocol (RTMP) transport video using dedicated streaming servers, whereas HTTP-based protocols rely on regular web servers to optimize the viewing experience and quickly scale. Finally, emerging HTTP-based technologies like Apple's Low-Latency HLS seek to deliver the best of both options by supporting low-latency streaming at scale.

UDP VS. TCP: A QUICK BACKGROUND

User Datagram Protocol (UDP) and Transmission Control Protocol (TCP) are both core components of the internet protocol suite, residing in the transport layer. The protocols used for streaming sit on top of these. UDP and TCP differ in terms of quality and speed, so it's worth taking a closer look.



Adapted from https://microchipdeveloper.com/tcpip:tcp-vs-udp

The primary difference between UDP and TCP hinges on the fact that TCP requires a three-way handshake when transporting data. The initiator (client) asks the accepter (server) to start a connection, the accepter responds, and the initiator acknowledges the response and maintains a session between either end. For this reason, TCP is quite reliable and can solve for packet loss and ordering. UDP, on the other hand, starts without requiring any handshake. It transports data regardless of any bandwidth constrains, making it speedier and riskier. Because UDP doesn't support retransmissions, packet ordering, or error-checking, there's potential for a network glitch to corrupt the data en route.
Protocols like Secure Reliable Transport (SRT) often use UDP, whereas protocols like HTTP Live Streaming (HLS) use TCP.

UDP vs. TCP Deep Dive

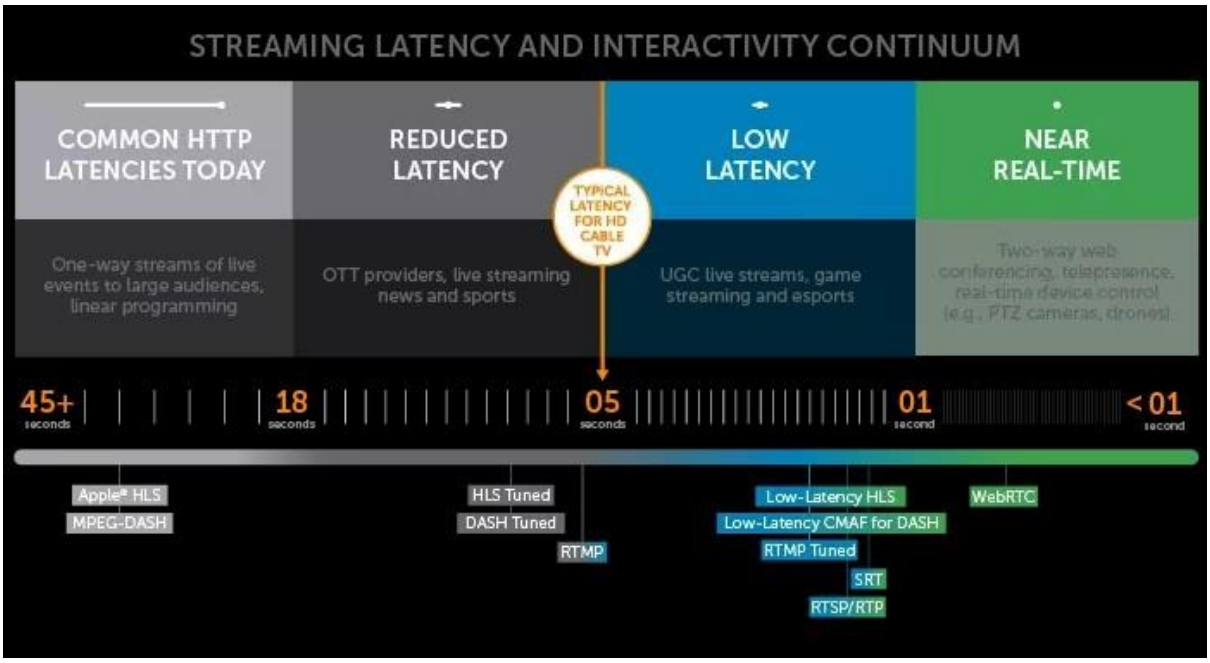
What Are the Most Common Protocols for Video Streaming?

Video Streaming Protocols Comparison in 2022

- Real-Time Messaging Protocol (RTMP)
- Real-Time Streaming Protocol (RTSP)
- HTTP Live Streaming (HLS)
- Low-Latency HLS
- Dynamic Adaptive Streaming over HTTP (MPEG-DASH)
- Low-Latency CMAF for DASH
- Microsoft Smooth Streaming
- Adobe HDS (HTTP Dynamic Streaming)
- SRT (Secure Reliable Transport)
- WebRTC (Web Real-Time Communications)

Traditional Video Streaming Protocols

Traditional streaming protocols, such as RTSP and RTMP, support low-latency streaming. But they aren't natively supported on most endpoints (e.g., browsers, mobile devices, computers, and televisions). Today, these streaming formats work best for transporting video between and IP camera or encoder and a dedicated media server.



As shown above, RTMP delivers video at roughly the same pace as a cable broadcast — in just over five seconds. RTSP/RTP is even quicker at around two seconds. Both formats achieve such speed by transmitting the data using a firehose approach rather than requiring local download or caching. But because very few players support RTMP and RTSP, they aren't optimized for great viewing experiences at scale. Many broadcasters choose to transport live streams to the media server using a stateful protocol like RTMP. From there, they can transcode it into an HTTP-based technology for multi-device delivery.

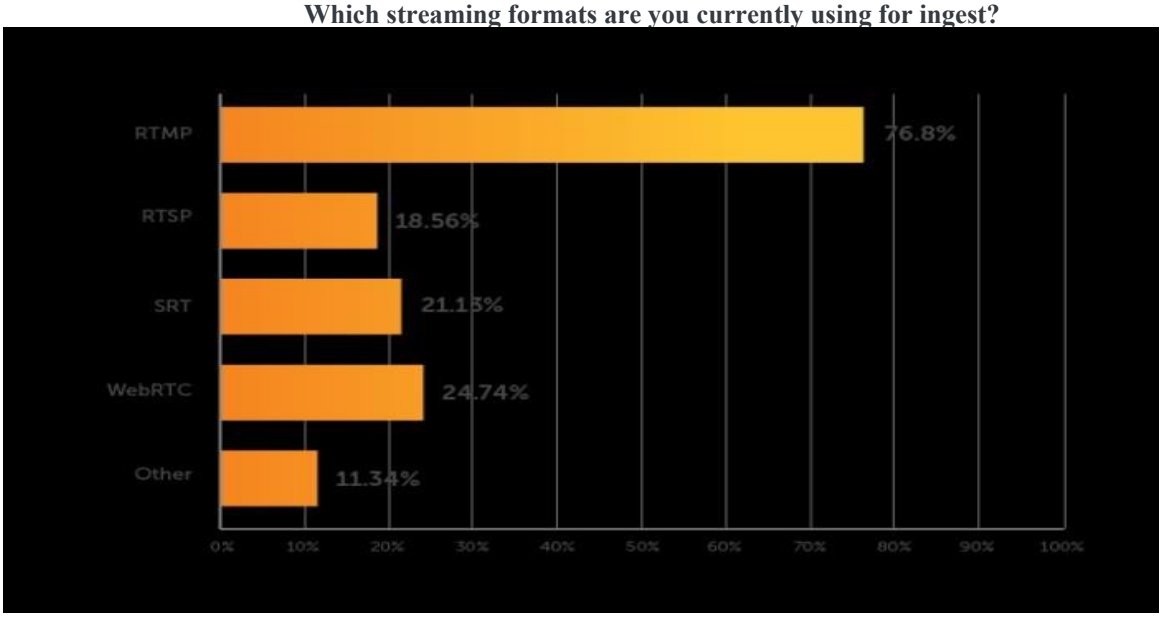
Adobe RTMP

Adobe designed the RTMP specification at the dawn of streaming. The protocol could transport audio and video data between a dedicated streaming server and the Adobe Flash Player. Reliable and efficient, this worked great for live streaming. But open standards and adaptive bitrate streaming eventually edged RTMP out. The writing on the wall came when Adobe announced the death of Flash — which officially ended in 2020.

While Flash's end-of-life date was overdue, the same cannot be said for using RTMP for video contribution. RTMP encoders are still a go-to for many content producers, even though the proprietary protocol has fallen out of favor for last-mile delivery.

In fact, in our ***2021 Video Streaming Latency Report***, more than 76% of content distributors indicated they use RTMP for ingest.

Which streaming formats are you currently using for ingest?



- **Video Codecs:** H.264, VP8, VP6, Sorenson Spark®, Screen Video v1 & v2
- **Audio Codecs:** AAC, AAC-LC, HE-AAC+ v1 & v2, MP3, Speex, Opus, Vorbis
- **Playback Compatibility:** Not widely supported (Flash Player, Adobe AIR, RTMP-compatible players)
- **Benefits:** Low-latency and requires no buffering
- **Drawbacks:** Not optimized for quality of experience or scalability
- **Latency**: 5 seconds
- **Variant Formats**: RTMPT (tunneled through HTTP), RTMPE (encrypted), RTMPTE (tunneled and encrypted), RTMPS (encrypted over SSL), RTMFP (travels over UDP instead of TCP)

RTMP Explained

RTSP/RTP

Like RTMP, RTSP/RTP describes an old-school technology used for video contribution. RTSP and RTP are often used interchangeably. But to be clear: RTSP is a presentation-layer protocol that lets end users command media servers via pause and play capabilities, whereas RTP is the transport protocol used to move said data.
Android and iOS devices don't have RTSP-compatible players out of the box, making this another protocol that's rarely used for playback. That said, RTSP remains standard in many surveillance and closed-circuit television (CCTV) architectures. Why? The reason is simple. RTSP support is still ubiquitous in IP cameras.

- **Video Codecs:** H.265 (preview), H.264, VP9, VP8
- **Audio Codecs:** AAC, AAC-LC, HE-AAC+ v1 & v2, MP3, Speex, Opus, Vorbis
- **Playback Compatibility:** Not widely supported (Quicktime Player and other RTSP/RTP-compliant players, VideoLAN VLC media player, 3Gpp-compatible mobile devices)
- **Benefits:** Low-latency and supported by most IP cameras
- **Drawbacks:** No longer used for video delivery to end users
- **Latency:** 2 seconds
- **Variant Formats:** The entire stack of RTP, RTCP (Real-Time Control Protocol), and RTSP is often referred to as RTSP

RTSP Explained

Adaptive HTTP-Based Streaming Protocols

Streams deployed over HTTP are not technically "streams." Rather, they're progressive downloads sent via regular web servers. Using adaptive bitrate streaming, HTTP-based protocols deliver the best video quality and viewer experience possible — no matter the connection, software, or device. Some of the most common HTTP-based protocols include MPEG-DASH and Apple's HLS.

Apple HLS

Since Apple is a major player in the world of internet-connected devices, it follows that Apple's HLS protocol rules the digital video landscape. For one, the protocol supports adaptive bitrate streaming, which is key to viewer experience. More importantly, a stream delivered via HLS will play back on the majority of devices — thereby ensuring accessibility to a large audience.
HLS support was initially limited to iOS devices such as iPhones and iPads, but native support has since been added to a wide range of platforms. All Google Chrome browsers, as well as Android, Linux, Microsoft, and MacOS devices, can play streams delivered using HLS.

NEVER MISS AN HLS UPDATE

- **Video Codecs:** H.265, H.264
- **Audio Codecs:** AAC-LC, HE-AAC+ v1 & v2, xHE-AAC, Apple Lossless, FLAC
- **Playback Compatibility:** Great (All Google Chrome browsers; Android, Linux, Microsoft, and MacOS devices; several set-top boxes, smart TVs, and other players)
- **Benefits:** Adaptive bitrate and widely supported
- **Drawbacks:** Quality of experience is prioritized over low latency
- **Latency:** 6-30 seconds (lower latency only possible when tuned)
- **Variant Formats:** Low-Latency HLS (see below), PHLS (Protected HTTP Live Streaming)

HLS Explained

LOW-LATENCY HLS

Low-Latency HLS (LL-HLS) is the latest and greatest technology when it comes to low-latency streaming. The proprietary protocol promises to deliver sub-three-second streams globally. It also offers backward compatibility to existing clients.

In other words, it's designed to deliver the same simplicity, scalability, and quality as HLS — while significantly shrinking the latency. At Wowza, we call this combination the streaming trifecta.

Even so, successful deployments of Low-Latency HLS require integration from vendors across the video delivery ecosystem. Support is still lacking, and large-scale deployments of Low-Latency HLS are few and far between.

- **Playback Compatibility:** Any players that aren't optimized for Low-Latency HLS can fall back to standard (higher-latency) HLS behavior
    - HLS-compatible devices include MacOS, Microsoft, Android, and Linux devices; all Google Chrome browsers; several set-top boxes, smart TVs, and other players
- **Benefits:** Low latency, scalability, and high quality… Oh, and did we mention backward compatibility?
- **Drawbacks:** As an emerging spec, vendors are still implementing support
- **Latency:** 2 seconds or less

More About Low-Latency HLS

MPEG-DASH

MPEG-DASH is a vendor-independent alternative to HLS. Basically, with DASH you get a non-proprietary option that ensures the same scalability and quality. But because Apple tends to prioritize its own tech stack, support for DASH plays second fiddle in the slew of Apple devices out there.

- **Video Codecs:** Codec-agnostic
- **Audio Codecs**: Codec-agnostic
- **Playback Compatibility:** Good (All Android devices; most post-2012 Samsung, Philips, Panasonic, and Sony TVs; Chrome, Safari, and Firefox browsers)
- **Benefits:** Vendor-independent, international standard for adaptive bitrate
- **Drawbacks:** Not supported by iOS or Apple TV
- **Latency:** 6-30 seconds (lower latency only possible when tuned)
- **Variant Formats:** MPEG-DASH CENC (Common Encryption)

MPEG-DASH Explained

Low-Latency CMAF for DASH

Low-latency CMAF for DASH is another emerging technology for speeding up HTTP-based video delivery. Although it's still in its infancy, the technology shows promise in delivering superfast video at scale by using shorter data segments. That said, many vendors have prioritized support for Low-Latency HLS over that of low-latency CMAF for DASH.

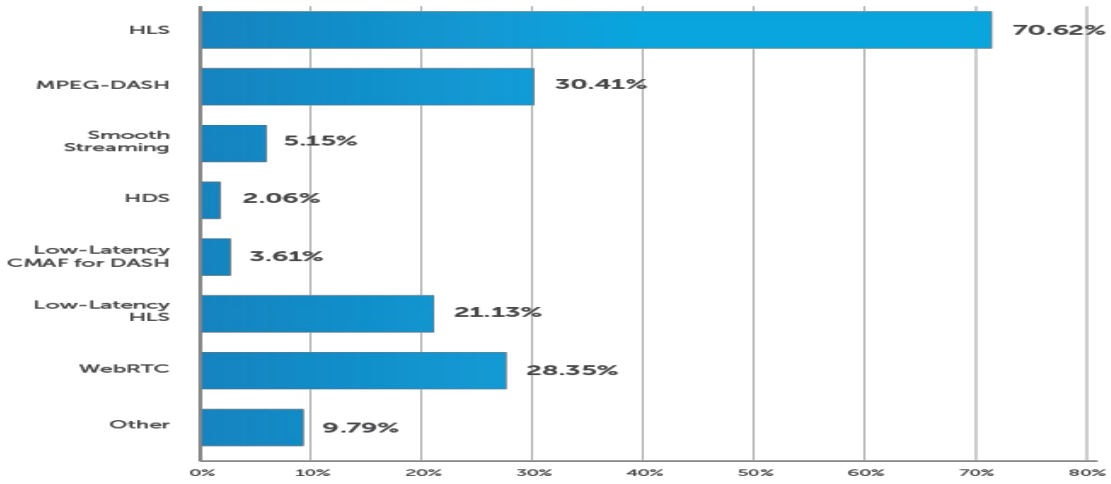
- **Playback Compatibility:** Any players that aren't optimized for low-latency CMAF for DASH can fall back to standard (higher-latency) DASH behavior
- **Benefits:** Low latency meets HTTP-based streaming
- **Drawbacks:** As an emerging spec, vendors are still implementing support
- **Latency:** 3 seconds or less

CMAF Explained

Microsoft Smooth Streaming

Microsoft developed Microsoft Smooth Streaming in 2008 for use with Silverlight player applications. It enables adaptive delivery to all Microsoft devices. The protocol can't compete with other HTTP-based formats and is falling out of use. In fact, in our *2021 Video Streaming Latency Report*, only 5 percent of respondents were using Smooth Streaming.

Which streaming formats are you currently using?



- **Video Codecs:** H.264, VC-1
- **Audio Codecs:** AAC, MP3, WMA
- **Playback Compatibility:** Good (Microsoft and iOS devices, Xbox, many smart TVs, Silverlight player-enabled browsers)
- **Benefits:** Adaptive bitrate and supported by iOS
- **Drawbacks:** Proprietary technology, doesn't compete with HLS and DASH
- **Latency:** 6-30 seconds (lower latency only possible when tuned)

Adobe HDS

HDS was developed for use with Flash Player as the first adaptive bitrate protocol. Because Flash is no more, it's also slowly dying. Don't believe us? Just take a look at the graph above.

- **Video Codecs:** H.264, VP6
- **Audio Codecs:** AAC, MP3
- **Playback Compatibility:** Not widely supported (Flash Player, Adobe AIR)
- **Benefits:** Adaptive bitrate technology for Flash
- **Drawbacks:** Proprietary technology with lacking support
- **Latency:** 6-30 seconds (lower latency only possible when tuned)

New Technologies

Last but not least, new technologies like WebRTC and SRT promise to change the landscape. Similar to low-latency CMAF for DASH and Apple Low-Latency HLS, these protocols were designed with latency in mind.

SRT

This open-source protocol is recognized as a proven alternative to proprietary transport technologies — helping to deliver reliable streams, regardless of network quality. It competes directly with RTMP and RTSP as a first-mile solution, but it's still being adopted as encoders, decoders, and players add support.

From recovering lost packets to preserving timing behavior, SRT was designed to solve the challenges of video contribution and distribution across the public internet. And it's quickly taking the industry by storm. One interactive use case for which SRT proved instrumental was the 2020 virtual NFL draft. The NFL used this game-changing technology to connect 600 live feeds for the first entirely virtual event.

- **Video Codecs:** Codec-agnostic
- **Audio Codecs:** Codec-agnostic
- **Playback Compatibility:** Limited (VLC Media Player, FFPlay, Haivision Play Pro, Haivision Play, Larix Player, Brightcove)
- **Benefits:** High-quality, low-latency video over suboptimal networks
- **Drawbacks:** Not widely supported for video playback
- **Latency:** 3 seconds or less, tunable based on how much latency you want to trade for packet loss

SRT Explained

WebRTC

As the speediest technology available, WebRTC delivers near-instantaneous voice and video streaming to and from any major browser. It can also be used end-to-end and thus competes with ingest and delivery protocols. The framework was designed for pure chat-based applications, but it's now finding its way into more diverse use cases.

Scalability remains a challenge with WebRTC, though, so you'll need to use a solution like Wowza's Real-Time Streaming at Scale feature to overcome this. The solution deploys WebRTC across a custom CDN to provide near-limitless scale. This allows broadcasters to reach a million viewers with sub-500 ms delivery — a once impossible feat.

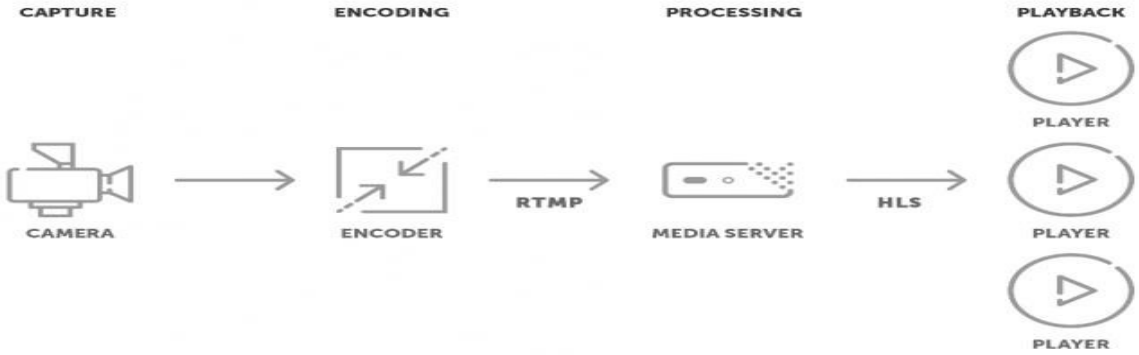
Workflow: Real-Time Streaming at Scale for Wowza Video

- **Video Codecs:** H.264, VP8, VP9
- **Audio Codecs:** Opus, iSAC, iLBC
- **Playback Compatibility:** Chrome, Firefox, and Safari support WebRTC without any plugin
- **Benefits:** Super fast and browser-based
- **Drawbacks:** Designed for video conferencing and not scale
- **Latency:** Sub-500-millisecond delivery

WebRTC Explained

Considerations When Choosing a Streaming Protocol

Selecting the right media streaming protocol starts with defining what you're trying to achieve. Latency, playback compatibility, and viewing experience can all be impacted. What's more, content distributors don't always stick with the same protocol from capture to playback. Many broadcasters use RTMP to get from the encoder to server and then transcode the stream into an adaptive HTTP-based format.



Media streaming protocols differ in the following areas:

- First-mile contribution vs. last-mile delivery
- Playback support
- Encoder support
- Scalability
- Latency
- Quality of experience (adaptive bitrate enabled, etc.)
- Security

By prioritizing the above considerations, it's easy to narrow down what's best for you.

Contribution vs . Delivery

RTMP and SRT are great bets for first-mile contribution, while both DASH and HLS lead the way when it comes to playback. On the flip side, RTMP has fallen entirely out of favor for delivery, and HLS isn't an ideal ingest format. That's why most content distributors rely on a media server or cloud-based video platform to transcode their content from one protocol to another.

Playback Support

What's the point of distributing a stream if viewers can't access it? Lacking playback support is the reason RTMP no longer plays a role in delivery to end users. And ubiquitous playback support is the reason why HLS is the most popular protocol today.

Encoder Support

The inverse of playback support is encoder support. RTMP maintains a stronghold despite its many flaws due to the prevalence of RTMP encoders already out there. Similarly, RTSP has stayed relevant in the surveillance industry because it's the protocol of choice for IP cameras.
WebRTC is unique in that it can be used for browser-based publishing and playback without any additional technologies, enabling simple streaming for use cases that don't require production-quality encoders and cameras.

Scalability

HLS is synonymous with scalability. The widely-supported HTTP-based protocol leverages web servers to reach any device worth reaching today. But what it delivers in scalability, it lacks in terms of latency. That's because latency and scalability have traditionally been at odds with one another. New technologies like Real-Time Streaming at Scale, however, resolve this polarity.

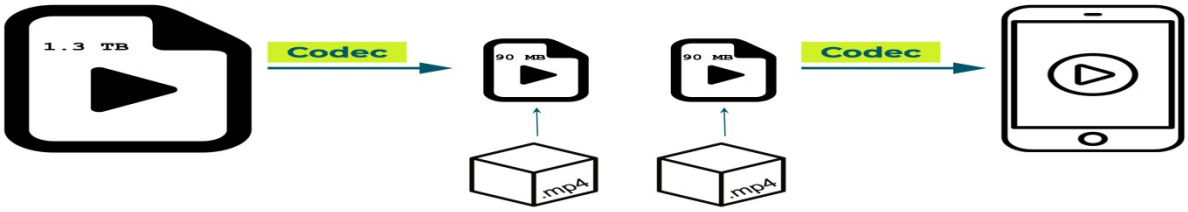
Latency

Low-Latency HLS, low-latency CMAF for DASH, and WebRTC were all designed with speedy delivery in mind. Anyone deploying interactive video environments should limit themselves to one of these three delivery protocols.

Case Study: Video Transcoding APP.

VIDEO FILE?

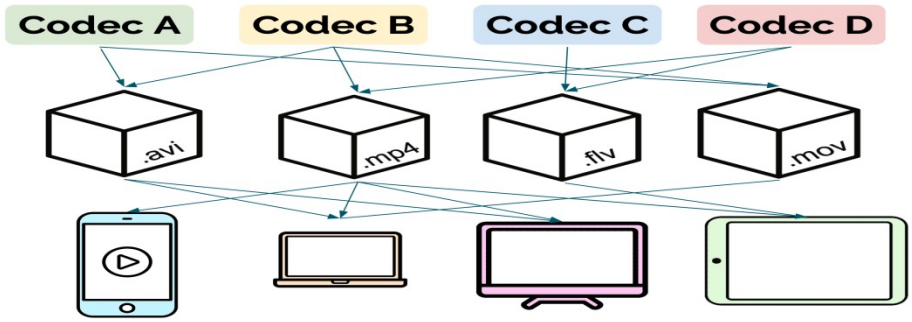
To discuss the problems surrounding compatibility and bandwidth, we must first explain what a video file is.
When a video is first recorded by a phone or camera, the raw video data is too large to store locally, much less send over the web. For example, an hour of 1080p 60fps uncompressed video is around ***1.3 terabytes.***
To bring the size of this raw data down to a more manageable size, the data must be compressed. The software that is used to compress this data is called a codec *(a combination of the words coder and decoder)*. A codec applies an algorithm to compress video data, ***encoding*** it so that it can be easily stored and sent. Once compressed, the data is packaged into a file format, called a container. Containers have extensions you may have seen, like .mp4 or .mov.
When playing a video this process is reversed. A media player opens the container, and the same codec is used to ***decode*** the video data and display it on the device.



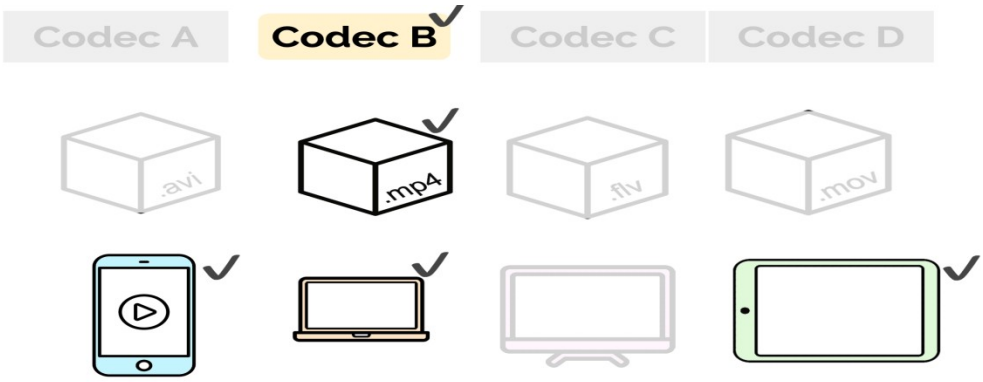
Encoded videos are decoded with the same codec for playback

THE PROBLEM OF COMPATIBILITY

The first problem that businesses face is that there are dozens of different codecs, containers, and video players, each with their own strengths and weaknesses. Unfortunately, they are not all compatible with each other. Using the wrong codec or container could mean that some users can't play certain videos on their device. Therefore a decision must be made as to which codec and container will be used to package a video.



Usually this decision will be made based on the characteristics of the codec and the types of devices or media players that a business expects their users to have. Once they have made this decision, they need to convert any video files they have using the codec and container they have decided on.



Businesses select a codec and container format to optimize compatibility

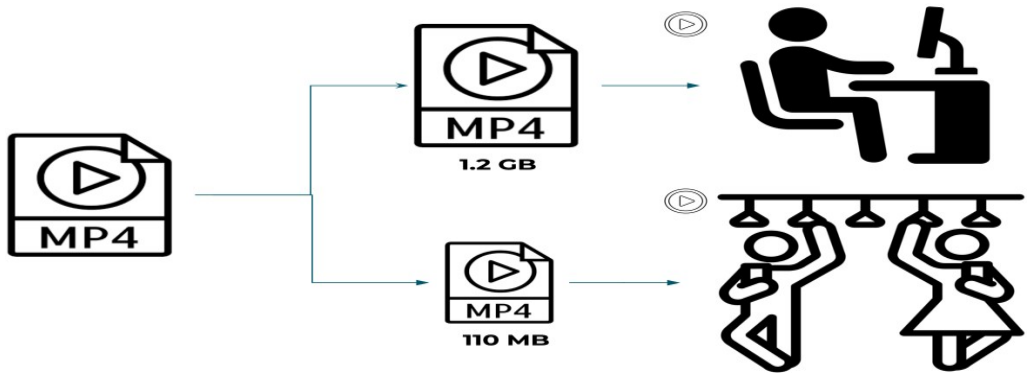
However, now they need to answer a second question: how much should they compress their videos?

THE PROBLEM OF BANDWIDTH

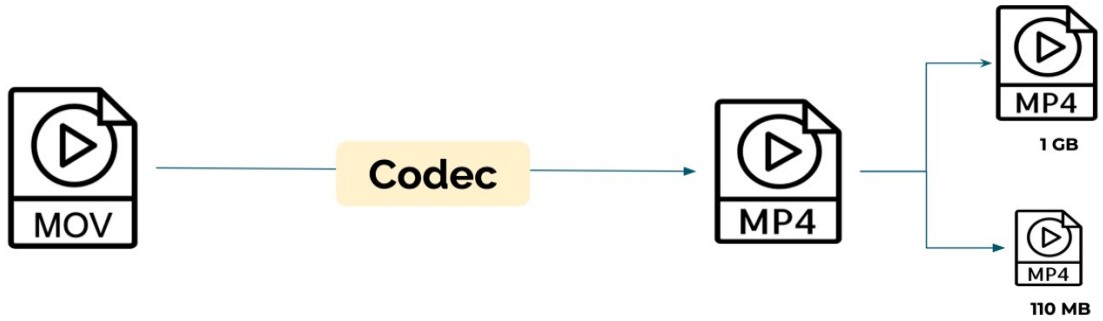
Generally speaking, there is a negative correlation between the level of compression and the quality of the resultant video. The more a video file is compressed, the greater the decrease in visual fidelity. This means that the less a video is compressed, the more of its original quality is preserved and the larger the file size. However, not all users will have the bandwidth to quickly download larger, higher quality files.

Consider for instance, the difference in download speed that will be available to a user on a fiber internet connection in their office, and a user on 3G mobile connection going through a subway tunnel as they both attempt to download the same video. The person in their office will have a smooth experience, whereas the person in the tunnel may have a choppy experience, if their video plays at all.

For this reason, businesses will usually create multiple versions of the same video at different rates of compression, and thus different file sizes. Modern media players detect users' bandwidth, and deliver the video file most appropriate for speed of their connection. Smaller, more compressed videos will be delivered to users with less available bandwidth, while users with stronger connections will be served higher quality videos.



VIDEO TRANSCODING AND IT 'S INHERENT CHALLENGES



To recap, businesses that deliver video will use a ***codec*** to convert their videos into a single ***container*** format, compressed to ***multiple file sizes***.

This process of conversion is called video transcoding. The process of ingesting a video file and transcoding it to specific formats and file sizes is coordinated and facilitated by a transcoding pipeline.

Every business that delivers video files on the internet will consider how they are going to handle this process of transcoding their videos. Unfortunately, the video transcoding process possesses its own inherent challenges that need to be addressed when negotiating a transcoding solution.

First, transcoding large video files takes a ***very long time***. Transcoding a single 60 minute HD video can take anywhere from two to six hours, and sometimes more.

Second, transcoding is demanding on memory and CPU. A video transcoding process will happily eat all the CPU and memory that are thrown at it.

Transcoding on a single local machine may be a viable option for individuals that only transcode a few videos per month. However, businesses that have regular video demand will not find this to be a feasible option, and will generally choose between one of two professional transcoding solutions.

2.5 EXISTING VIDEO TRANSCODING SOLUTIONS

Broadly considered, professional video transcoding solutions will fall into two categories: customized solutions that are developed and maintained in-house and third party commercial software

Custom transcoding

Examples: Amazon EC2, Digital Ocean Droplet, bare metal

Advantages:

- Cheaper than third party products
- Maximum control over transcoding setting

Trade-offs:

- Requires technical expertise
- Unused compute power
- Error prone
- Slow without expertise and optimizations

Some businesses choose to build their own video transcoding farm in-house. In this case, a development team is needed to build custom transcoding pipeline and deploy it to bare metal servers or cloud infrastructure (e.g. Amazon EC2, Digital Ocean Droplet).
Once deployed, this option is cheaper than third-party software, and it provides businesses with maximum control over what formats, codecs, and video settings they support, as well as any additional transformations they want to make to their videos.
This option requires a high amount of technical expertise both to develop and scale effectively: video transcoding can be both slow and error prone without significant engineering attention. As we'll see later, provisioning servers for video transcoding can also result in periods wherein compute power is going unused.
Building a custom transcoding farm is best-suited for video platforms such as Netflix and YouTube. Companies that ingest and stream millions of hours of video each week can build their own engineering operations around this task to minimize the trade-offs.

Commercial software

Examples: Zencoder, Amazon MediaConvert

Advantages:

- Fine-grained transcoding options
- Tuned for speed and reliability
- Less technical expertise

Trade-offs:

- Expensive
- Complicated settings may be overkill for many businesses and individuals

A second solution is to use commercial video transcoding software. Services such as Amazon MediaConvert and Zencoder offer powerful cloud-based transcoding.
These services provide comprehensive solutions for businesses that need to support multiple formats, codecs, and devices. Because these services specialize in video, they will be tuned for speed and adept at handling the error cases that typically accompany transcoding jobs.
However, the number of transcoding options many of these services provide may be overkill for smaller businesses that have fewer input and output requirements. And, as one might expect, these services tend to be a more expensive option.
Transcoding services are a good fit for media production companies (e.g. HBO, BBC) that produce massive amounts of video content that lands on many devices, and don't want to build large software engineering teams around video transcoding.

Solutions compared

Let's briefly review the options outlined above.

	Speed	Technical Expertise	Control	Cost
Private servers	LOW/MED	HIGH	HIGHEST	MED/HI
Transcoding as a Service	HIGH	LOW/MED	MED/HI	HIGHEST

Custom pipelines provide the highest level of control over inputs and outputs. The bottleneck is technical expertise - to really get the best performance and cost from this model, a business needs to build a dedicated software engineering team around video transcoding.

By contrast, businesses can outsource their video transcoding to third-party services. These services provide the benefits of speed and control, but at a higher cost.

These options are sufficient for some businesses, but the Bento team saw an opportunity here. Is it possible to build a fast, low-cost transcoding pipeline, suited for businesses that don't have video expertise and won't need a plethora of video options?

We felt that the answer was yes, and that the solution might lie in serverless technology.

CLOUD APPLICATION BENCHMARKING & TUNING

INTRODUCTION: -

Multi-tier applications deployed in cloud computing environments can experience rapid changes in their workloads, which is also the reason why cloud computing has been preferred over traditional in-house resources.

To ensure market readiness of such applications, adequate resources need to be provisioned so that the applications can meet the demands of specified workload levels and at the same time ensure that service level agreements are met.

BENCHMARKING OF CLOUD APPLICATIONS

Benchmarking of cloud applications is important or the following reasons:

Provisioning and capacity planning: The process of provisioning and capacity planning for cloud applications involves determining the amount of computing, memory and network resources to provision for the application. Benchmarking can help in comparing alternative deployment architectures and choosing the best and most cost-effective deployment architecture that can meet the application performance requirements.

Ensure proper utilization of resources: Benchmarking can help in determining the utilization of computing, memory and network resources for applications and identify resources which are either under-utilized or over-provisioned and hence save deployments costs.

Market readiness of applications: Performance of art application depends on the characteristics of the workloads it experiences. Different types of workloads can dead to different performance for the same application. To ensure the market readiness of an application it is important to model all types of workloads the application can experience and benchmark the application with such workloads.

The steps involved in benchmarking of cloud applications are described below:

Trace Collection/Generation
Workload Modeling
Workload Specification
Synthetic Workload Generation
User Emulation vs Aggregate Workloads

Trace Collection/Generation:- The first step in benchmarking cloud applications is to collect/generate traces of real application workloads. For generating a trace of workload, the application is instrumented to log information such as the requests submitted by the users, the time-stamps of the requests, etc.

Workload Modeling:- Workload modeling involves creation of mathematical models that can be used for generation of synthetic workloads. Workloads of applications are often recorded as traces of workload related events such as arrival of requests along with the time-stamps, details about the users requesting the services, etc. Analysis of such traces can provide insights into the workload characteristics which can be used for formulating mathematical models for the workloads.

Workload Specification:- Since the workload models of each class of cloud computing applications can have different workload attributes, a Workload Specification Language (WSL) is often used for specification of application workloads.

WSL can provide a structured way for specifying the workload attributes that are critical to the performance of the applications. WSL can be used by synthetic workload generators for generating workloads with slightly varying the characteristics.

This can be used to perform sensitivity analysis of the application performance to the workload attributes by generating synthetic workloads.

1) **Synthetic Workload Generation:-** Synthetic workloads are used for benchmarking cloud applications. An important requirement for a synthetic workload generator is that the generated workloads should be representative of the real workloads and should preserve the important

characteristics of real workloads such as inter-session and intra-session intervals, etc. **There are two approaches to synthetic workload generation:**

- **Empirical approach:** In this approach traces of applications are sampled and replayed to generate the synthetic workloads. The empirical approach lacks flexibility as the real traces obtained from a particular system are used for workload generation which may not well represent the workloads on other systems with different configurations and load conditions.
- **Analytical approach:** This approach uses mathematical models to define the workload characteristics that are used by a synthetic workload generator. Analytical approach is flexible and allows generation of workloads with different characteristics by varying the workload model attributes. With the analytical approach it is possible to modify the workload model parameters one at a time and investigate the effect on application performance to measure the application sensitivity to different parameters.

2) **User Emulation vs Aggregate Workloads:-** The commonly used techniques for workload generation are user emulation and aggregate workload generation.

User Emulation

- In User Emulation, each user is emulated by a separate thread that mimics the actions of a user by alternating between making requests and lying idle.
- The attributes for workload generation in the user emulation method include think time, request types, inter-request dependencies, for instance.
- User emulation allows fine grained control over modeling the behavioral aspects or' the users interacting with the system under test, however, it does not allow controlling the exact time instants at which the requests arrive the system.

Aggregate workload generation

- Allows specifying the exact time instants at which the requests should arrive the system under test. However, there is no notion of an individual user in aggregate workload generation, therefore, it is not possible to use this approach when dependencies between requests need to be satisfied.

Dependencies can be of two types **inter-request** and **data dependencies**. An inter-request dependency exists when the current request depends on the previous request, whereas a data dependency exists when the current requests requires input data which is obtained from the response of the previous request.

WORKLOAD CHARACTERSTICS

INTRODUCTION : -

Each class of multi-tier applications can have their own characteristic workloads. A successful performance evaluation methodology should be able to accurately model the application workloads. Workload modeling involves creation of mathematical models that can be used for generation of synthetic workloads.

CHARACTERISTICS OF APPLICATION WORKLOADS INCLUDE :

- **Session:** A set of successive requests submitted by a user constitute a session.
- **Inter-Session Interval:** Inter-session interval is the time interval between successive sessions.
- **Think Time:** In a session, a user submits a series of requests in succession. The time interval between two successive requests is called think time. Think time is the inactive period between subsequent requests in a session. It is the time taken by the user to review the response of a request and decide what the next request should be.

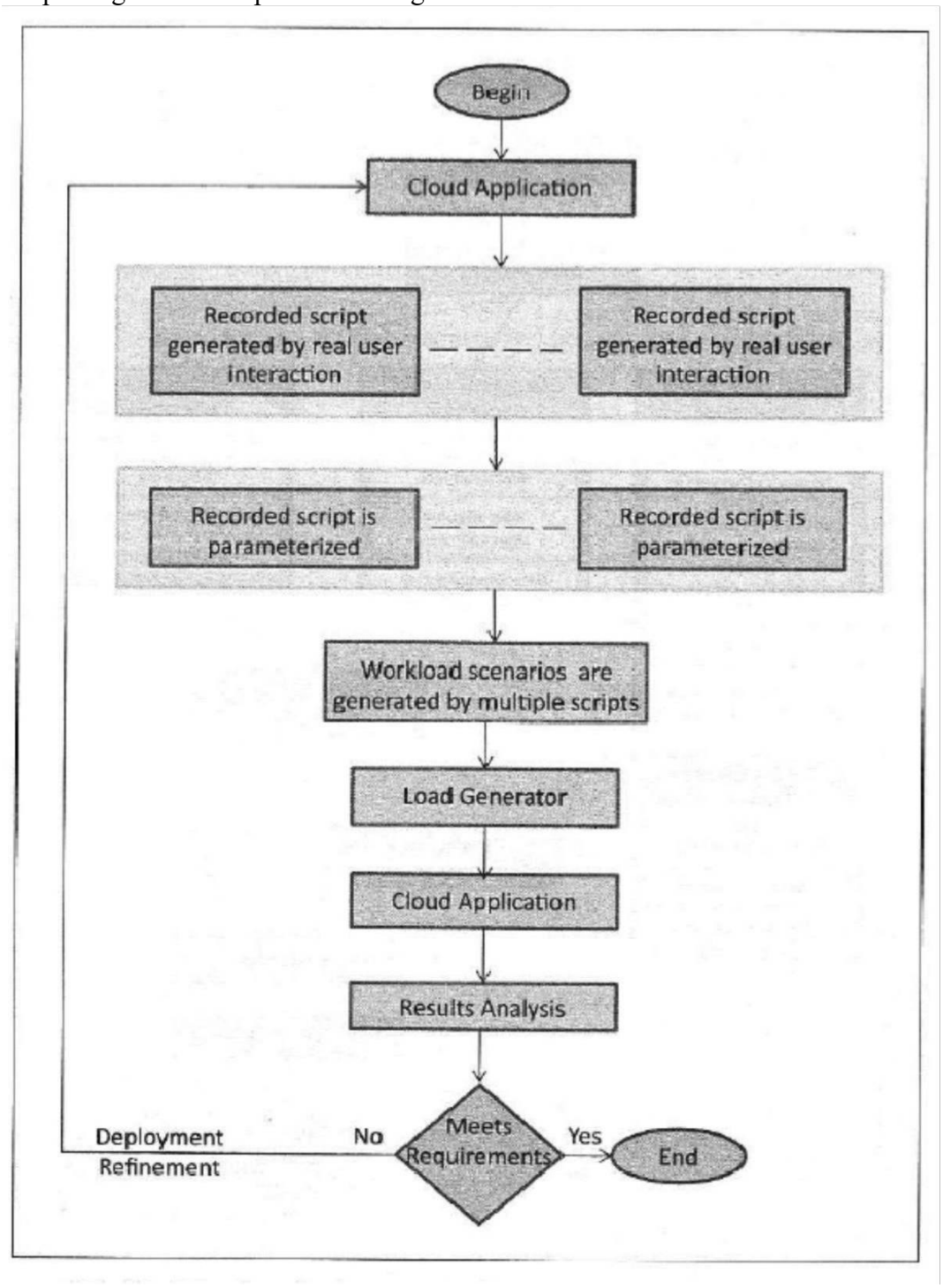
- **Session Length**: The number of requests submitted by a user in a session is called the session length.
- **Workload Mix:** Workload mix defines the transitions between different pages of an application and the proportion in which the pages are visited.

Multi-tier cloud applications can experience rapid changes in their workloads. Workload characteristics of an application may also vary widely based on the kind of user interactions with an application.

An e-Commerce application, for example, can experience database write-intensive workloads during the times of a year when a large number of user's are purchasing products online. Whereas, the same application can have a read-intensive workload mix when users are only browsing for products.

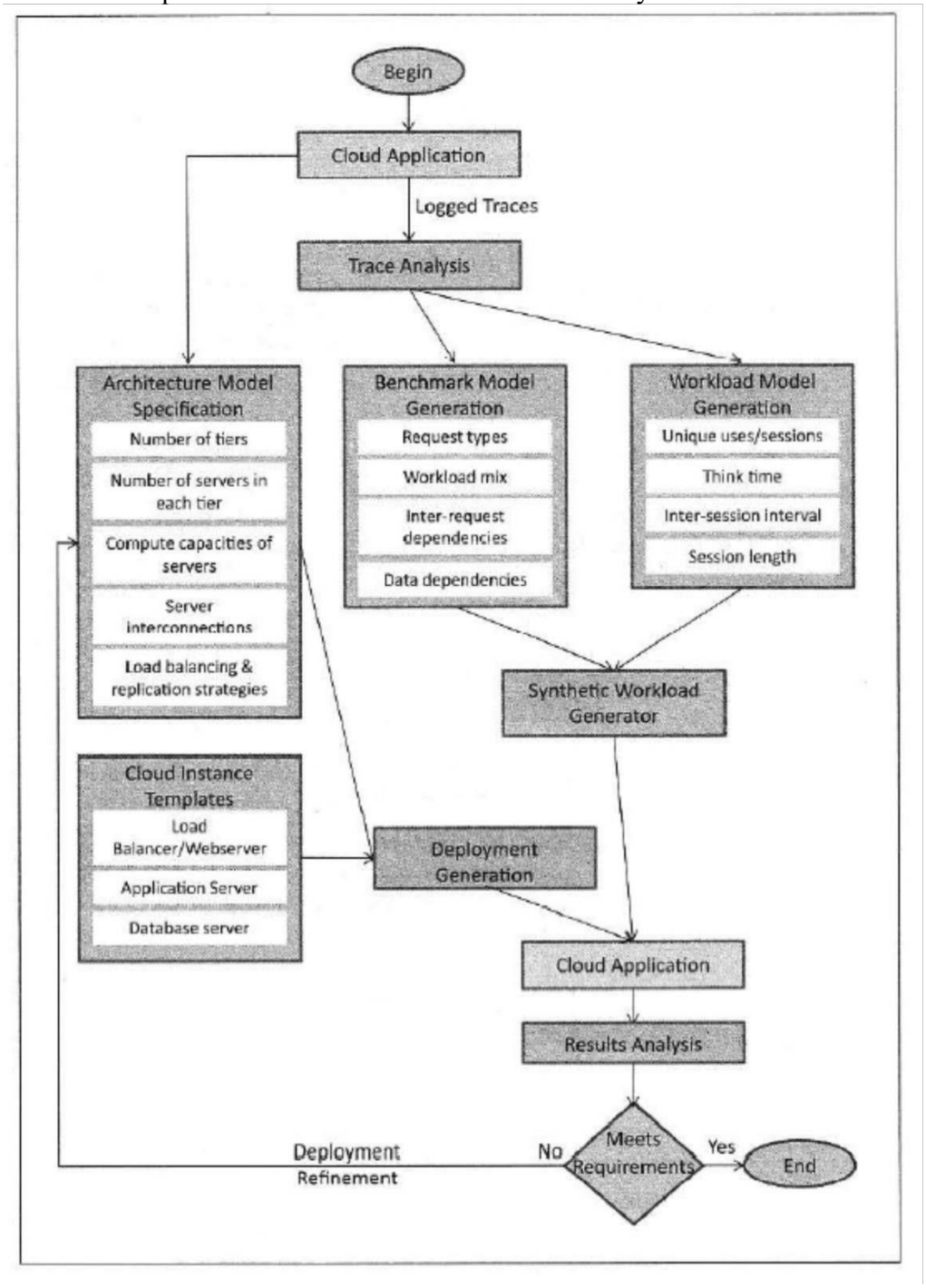
TRADITIONAL PERFORMANCE EVALUATION WORKFLOW

Figure 11.1: Traditional performance evaluation workflow based on a semi- automated approach. A real user has to first manually interact with the application to record scripts and then parameterize recorded scripts to generate scripts for creating virtual users.



AUTOMATED PERFORMANCE EVALUATION WORKFLOW

Figure I I .2: Automated performance evaluation workflow used by recent tools such as GT-CAT.



Benchmarking may be performed for the following reasons:

System design: comparing different systems, system components, or applications. For commercial products, benchmarking may provide data to aid a purchase decision, specifically the *price/performance* ratio of the available options. In some cases, results from published *industry benchmarks* can be used, which avoids the need for customers to execute the benchmarks themselves.

Tuning: testing tunable parameters and configuration options, to identify those that are worth further investigation with the production workload.

Development: for both *non-regression testing* and *limit investigations* during product development. Non-regression testing may be an automated battery of performance tests that run regularly, so that any performance regression can be discovered early and quickly matched to the product change. For limit

investigations, benchmarking can be used to drive products to their limit during development, in order to identify where engineering effort is best spent to improve product performance.
Capacity planning: determining system and application limits for capacity planning, either to provide data for modeling performance, or to find capacity limits directly.
Troubleshooting: to verify that components can still operate at maximum performance, for example, testing maximum network throughput between hosts to check whether there may be a network issue.
Marketing: determining maximum product performance for use by marketing (also called *benchmarketing*).

Application Performance Metrics:

Cloud Performance Metrics
Cloud Performance Testing Metrics
Cloud Security Metrics
Cloud Network Metrics
Cloud Cost Metrics

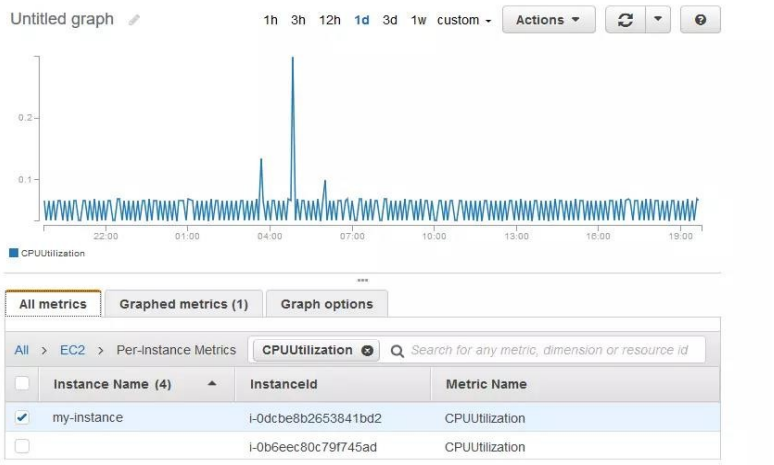
Cloud Performance Metrics

Cloud-based infrastructure, applications, and other components generate metrics companies can use to measure the reliability and operational excellence of their cloud services.

1. Uptime or availability

    This metric measures the percentage of time a service or system is available to serve customer requests. Downtime is the opposite. Uptime increases your chances of retaining customers and generating revenue.

2. CPU utilization



Credit: Amazon CloudWatch

CPU utilization measures the percentage of compute units you use. Tracking it will reveal if a CPU is throttling performance because of under- or over-utilization.

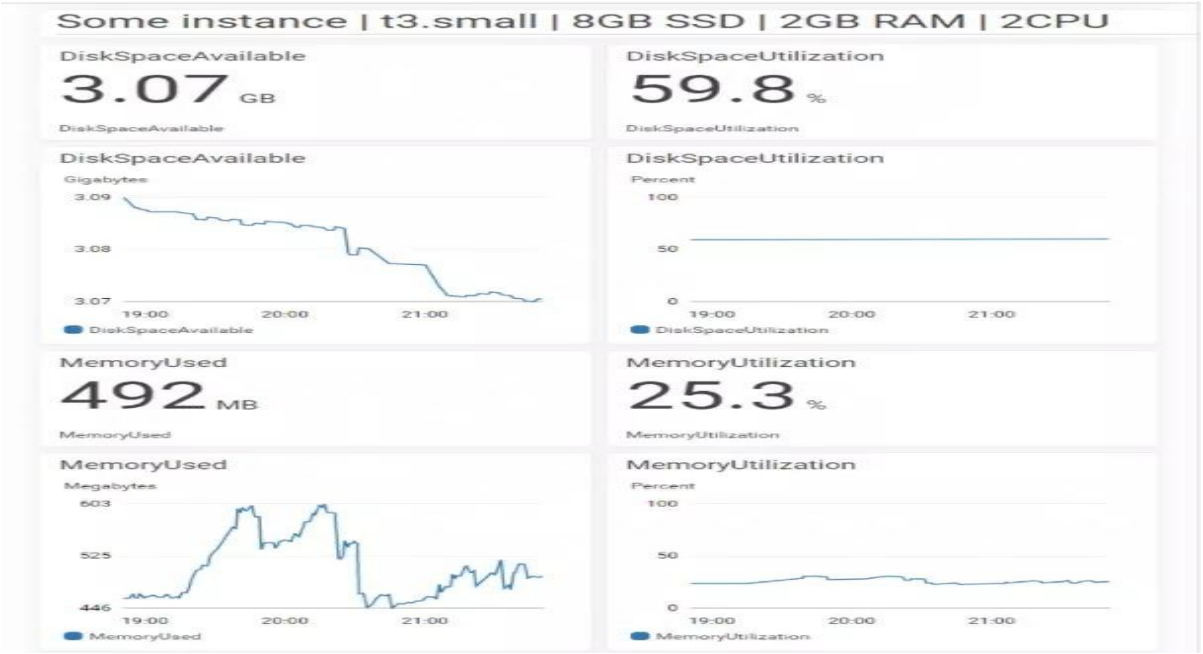
3. Memory utilization

    Memory utilization helps to measure memory usage in public, private, and hybrid cloud environments. A consistently high memory utilization may require you to scale up your memory capacity to ensure smooth performance.

4. Requests per minute

    Requests per minute tell how many requests a cloud-based application receives each minute. It is crucial to monitor how and when users access the app, so you can scale your cloud resources to meet demand, ensuring optimal performance.

5. Disk utilization



Credit: Letswp

Disk utilization enables you to track the disk volume on a node's storage capacity to tell if it is sufficient for your workloads. Typical storage metrics include IOPS and throughput. The IOPS metric describes the number of reads and writes per second, whereas throughput measures the amount of data transferred from and to storage in bytes per second (bps).

6. Average time to acknowledge

   Average time to acknowledge refers to the average time your application takes to begin a response to a request. If acknowledgement times are slow, then there may be a load balancer issue, or the app is struggling with underprovisioning and other latency issues.

7. Latency

   Latency measures the time between when a customer sends a request (request time) and when the cloud provider sends back a response (response time). High latency can negatively impact productivity. Your cloud provider's backend servers, web server dependencies, and network problems could all lead to increased latency.

8. The error rate

   The error rate measures how often a request results in an error. You can troubleshoot issues like improperly configured access credentials by identifying the types of errors the system generates.

9. Swap usage

   Swap usage refers to the amount of disk space devoted to holding data that should be in memory. High swap usage degrades application performance and defeats the purpose of in-memory caching.

10. Mean time between failure (MTBF)

    Mean Time Between Failure (MTBF) refers to the average time a repairable cloud component works before failing. It will help you understand why systems fail, so you can identify repair methods that improve MTBF and be better equipped to tolerate failures.

11. Mean time to repair

    Mean Time to Repair helps measure the average time it takes to repair a failed cloud component and get it back in service. Analyzing the MTTR will help you understand how long it takes your company to restore service after a failure. A shorter MTTR increases your chances of retaining clients and meeting SLAs.

Cloud Performance Testing Metrics

There are a variety of metrics to gauge the robustness of cloud infrastructure and the applications it hosts. They include:

12. Capacity test metrics
Capacity test metrics indicate the maximum load amount or traffic your cloud system can handle without throttling performance in production.
13. Target infrastructure metrics
Targeted infrastructure metrics enable you to isolate and fix problem areas of a specific layer or application component.
14. Stress testing metrics
Stress testing metrics help gauge the stability and responsiveness of your cloud environment and its components under high loads.
15. Load testing metrics
Load testing metrics enable engineering to check how cloud resources perform when multiple users try to access and use them simultaneously.
16. Failover test metrics
Failover test metrics measure a system's ability to call up additional cloud resources to handle heavy or peak loads.
17. Latency test metrics
Latency test metrics help you determine the time it takes your cloud resources to transfer data messages between two points on the network.
18. Soak test metrics
Soak test metrics are indicators of your cloud system's resilience during prolonged periods of heavy traffic.
Overall, cloud performance test metrics will help verify if your system will perform efficiently in a production environment.

Cloud Security Metrics

Keeping track of security and compliance KPIs is particularly challenging in the cloud's dynamic computing environment. Yet, it is possible. To mitigate threats, monitor metrics such as:
19. Pateched/unpatched know vulnerabilities
Patched/unpatched known vulnerabilities will indicate how timely and adequately patch cloud security risks in your system — or if you leave them open for too long.
20. Request per minute metrics
Requests per minute metrics not only measures cloud performance but also risks. A high number of requests per minute may indicate an ongoing threat, such as a Distributed Denial of Service (DDOS) attack.
21. Peer-to-peer-sharing metrics
Peer-to-peer file-sharing metrics help monitor changes in the number of files downloaded or shared through authorized means. An increase may indicate a compromised cloud security posture.
22. Other important security threats
New user accounts that delete multiple users
A computer instance that starts and stops programmatically could be a warning sign of an ongoing or inevitable attack.
Temporary security credentials that last a long time
Employee access credentials that are reassessed frequently
Cloud activity that deletes CloudTrail logs (in Amazon Web Services)
A sudden spike in "super user" usage levels
Data on violations, compliance score, and resolution progress are examples of compliance metrics.
By monitoring security and compliance metrics, you can prevent your cloud system from leaking confidential business information, customer data, and damaging your reputation.

Cloud Networking Metrics

Credit: Opsview

It was fairly straightforward to fix network issues when apps were hosted over a Local Area Network (LAN). It takes greater attention to identify the cause of network issues in the cloud.

Here are two crucial cloud networking metrics to keep an eye on:

23. Network capacity

Network capacity is the maximum data transit rate possible between a source and destination through the most congested hop in the application delivery path.

Available capacity measures the actual amount of network resources available to applications, while Utilized capacity is a strong indicator of network performance degradation. Both metrics will help you determine the root cause of service degradation.

24. Packet loss

Packet loss is a measure of the percentage loss of network packets between the source and destination. Packet loss can cause latency and network congestion when an internet protocol retransmits the data. Track this metric to make sure your system **doesn't** drop **users'** requests, resulting in customer frustration.

Network metrics provide a good indication of the kind of customer experience your organization provides. While modern networking technologies can easily handle small packet losses and jitter, sustained network problems can cause customers to unsubscribe from your service.

Cloud Cost Metrics

Cost-conscious teams treat cloud costs as a first-class metric. Engineers can then build cost-effective solutions while finance optimizes cloud costs without hindering innovation.

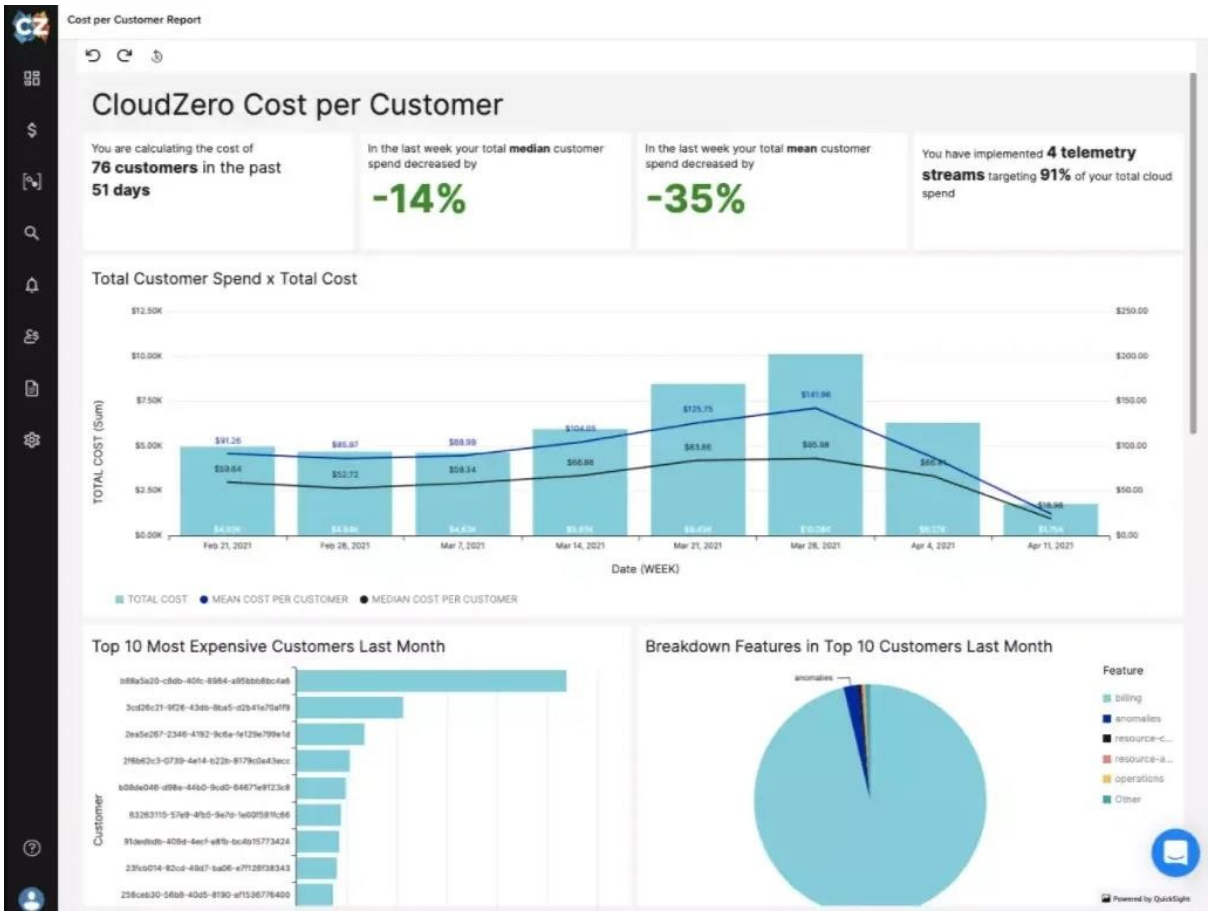
But it is not enough to collect high-level cost metrics that are difficult to link to actual business activity. Instead, monitor unit costs that relate to specific business activities, such as:

25. Cost per feature

Cost per feature is a measure of the amount you spend to release and support a particular product feature. You can track which customers use it, when, and how often. You can also use this metric to calculate how much you need to charge for the feature to turn a profit.

26. Cost per customer

The average cost per customer does not tell you how much you spend on specific customers. But with cost per customer, you can tell your most expensive customers, so you can adjust pricing across different customer segments to improve margins.



CloudZero aligns cloud costs to key business metrics, such as cost per customer or product feature. Our Cost Per Customer report allows teams to see how individual customers drive their cloud spend and how much specific customers cost their business. With cloud cost intelligence, companies can make informed engineering, business, and pricing that ensure profitability. Click here to learn more.

27. Cost per team

Cost per team indicates whether your team is as efficient as others in the industry or as you'd like. It is difficult for distributed teams to track their cloud costs without a robust tool.

28. Cost per deployment

Cost per deployment allows your engineering and finance teams to visualize how much a given deployment project costs from beginning to end. As you collect insight, you will be better able to allocate cloud resources and lower waste.

29. Kubernetes cost metrics

Cost metrics in containerized applications and Kubernetes clusters are difficult to monitor using most cloud cost management tools.

accurate view of how much it costs to run a microservice, support a customer, deliver a product, and more when your workloads rely on both containerized and traditional services. Click here to learn more.

30. Multi-cloud cost metrics

Multi-cloud cost metrics are generated across different clouds, making it tough to collect, enrich, and present them to various stakeholders in a language they understand.

Design Considerations for a benchmarking Methodology

TYPES OF BENCHMARKING

There are many different types of benchmarking that fall into three primary categories: internal, competitive, and strategic.

INTERNAL BENCHMARKING

If other teams or organizations within your company have established best practices in processes similar to yours, internal benchmarking involves analyzing what they are doing so you can find areas where you can improve and be more efficient.
For example, you could compare the performance of one warehousing and shipping site against another warehousing and shipping site. The site with superior performance simply needs to share their processes and procedures so that the entire company benefits from increased performance.

Competitive benchmarking

This type of benchmarking is a comparison of products, services, processes, and methods of your direct competitors. This type gives you insight into your position within your industry and what you may need to do to increase productivity.
For example, you can compare the customer satisfaction of a competitor's product to yours. If your competitor is getting better customer reviews, you need to analyze what the difference is and figure out how to improve the quality of your product.

Strategic benchmarking

Use this type of benchmarking when you need to look beyond your own industry to identify world-class performance and best practices so you can look for ways to adapt their methods to your procedures and processes.
For example, Kellog began using multi-industry benchmarks in 2019 to reduce cost, boost revenue, and see how they compare with peers. The benchmarks date back to 2004 when a group of corporations, government organizations and consultants partnered with the American Productivity & Quality Center to develop thousands of standards for business processes—from purchasing inventory to managing supply chains. Today, Kellog's global business services unit generates cost savings that represent about 6% to 7% of its annual run rate thanks to their benchmarking process.

Design Considerations for a Benchmarking Methodology

8 steps in the benchmarking process

1. Select a subject to benchmark
What to benchmark is just as important as how to benchmark it. Executives and other senior management should be involved in deciding which processes are critical to the company's success. Prioritize the processes based on which metrics are most important to all stakeholders, with an emphasis on processes or functions that are easily quantifiable. After prioritizing, select and define the measures you want to collect.
2. Decide which organizations or companies you want to benchmark
Determine if you are going to benchmark processes within your own company, a competitor, or a company outside of your industry.
It may be hard to collect all the data you want if you benchmark a direct competitor. So you should select several different organizations to study in order to get the data you need. Gather information from several sources to get the most detailed information about the organization you select to study.
3. Document your current processes
Map out your current processes so you can identify areas that need improvement and more easily compare against the chosen organization.
4. Collect and analyze data
This step is important—but it can prove difficult when you are trying to gather data from a competitor because a lot of that information may be confidential. Gather information through research, interviews, casual conversations with contacts from the other companies, and with formal interviews or questionnaires.
You can also collect secondary information from websites, reports, marketing materials, and news articles. However, secondary information may not be as reliable.
After you have collected enough data, get all stakeholders together to analyze the data.
5. Measure your performance against the data you've collected

Look at the data you've collected side by side with the metrics you gathered from your analysis of your own processes. You may want to layer your performance metrics on top of your process diagrams or map out your competitor's processes to more easily see where you're falling behind.
As you analyze the comparisons, try to identify what causes the gaps in your process. For example, do you have enough people and are they sufficiently trained to perform assigned tasks? Perhaps there are multiple steps that can be automated or combined to streamline workflow. Brainstorm ideas to effectively and efficiently fill those gaps.
6. Create a plan
Create a plan to implement agreed-on changes that you have identified as being the best to close performance gaps. Implementation requires total buy-in from the top down. Your plan must include clearly defined goals and should be written with the company's culture in mind to help minimize any pushback you may get from employees.
7. Implement the changes
Closely monitor the changes and employee performance. If new processes are not running smoothly as expected, identify areas that need to be tweaked. Make sure all employees understand their jobs, are well trained, and have the expertise to complete their assigned tasks.
Document all processes and make sure all employees have access to documentation and instructions so that all are on the same page working toward the same goal.
8. Repeat the process
After successfully implementing a new process, it's time to find other ways to improve. The benchmarking process is one of continual improvement and iteration. Review the new processes you've implemented and see if there are any changes that need to be made. If everything is running smoothly, look to other areas or more ambitious projects that you may want to benchmark and start the process again.
When you correctly implement and follow the continuous practice of benchmarking, your company will grow, and you will keep up with (or even surpass) your competitors.

BENCHMARKING TOOLS:

TOOLS FOR BENCHMARKING CLOUD COMPUTING :

- Cloud benchmarking tools: CloudBench, CloudPerf, and CloudScreener
- Cloud monitoring tools: CloudWatch, Datadog, and New Relic
- Cloud performance testing tools: LoadRunner, JMeter, and Gatling
- Performance benchmarking tools: PerfKitBenchmarker
- Cross-platform processor benchmarking tools: Geekbench 3
- Open source benchmarks: Iometer, Phoronix Test Suite

DEPLOYMENT PROTOTYPING, LOAD TESTING & BOTTLENECK DETECTION CASE STUDY,

- To identify performance bottlenecks in a web application by using Azure Load Testing.
- You'll create a load test for a sample Node.js application.

The sample application consists of a Node.js web API, which interacts with a NoSQL database.
You'll deploy the web API to Azure App Service web apps and use Azure Cosmos DB as the database.

- Deploy the sample app.
- Create and run a load test.
- Identify performance bottlenecks in the app.
- Remove a bottleneck.
- Rerun the load test to check performance improvements.
- Prerequisites

An Azure account with an active subscription. If you don't have an Azure subscription, create a free account before you begin.
Azure CLI version 2.2.0 or later. Run az --version to find the version that's installed on your computer. If you need to install or upgrade the Azure CLI, see How to install the Azure CLI.
Visual Studio Code. If you don't have it, download and install it.
Git. If you don't have it, download and install it.
Deploy the sample app
Before you can load test the sample app, you have to get it deployed and running. Use Azure CLI commands, Git commands, and PowerShell commands to make that happen.
Open Windows PowerShell, sign in to Azure, and set the subscription:
Azure CLICopy
az login
az account set --subscription<your-Azure-Subscription-ID>
Clone the sample application's source repo:
PowerShellCopy
git clone https://github.com/Azure-Samples/nodejs-appsvc-cosmosdb-bottleneck.git
The sample application is a Node.js app that consists of an Azure App Service web component and an Azure Cosmos DB database. The repo includes a PowerShell script that deploys the sample app to your Azure subscription. It also has an Apache JMeter script that you'll use in later steps.
Go to the Node.js app's directory and deploy the sample app by using this PowerShell script:
PowerShellCopy
cd nodejs-appsvc-cosmosdb-bottleneck
.\deploymentscript.ps1
Tip
You can install PowerShell on Linux/WSL or macOS.
After you install it, you can run the previous command as pwsh ./deploymentscript.ps1.
At the prompt, provide:
Your Azure subscription ID.
A unique name for your web app.
A location. By default, the location is eastus. You can get region codes by running the Get-AzLocation command.
Important
For your web app's name, use only lowercase letters and numbers. Don't use spaces or special characters.
After deployment finishes, go to the running sample application by opening https://<yourappname>.azurewebsites.net in a browser window.
Now that you have the application deployed and running, you can run your first load test against it.

CONFIGURE AND CREATE THE LOAD TEST

In this section, you'll create a load test by using a sample Apache JMeter test script.
The sample application's source repo includes an Apache JMeter script named *SampleApp.jmx*. This script makes three API calls to the web app on each test iteration:
add: Carries out a data insert operation on Azure Cosmos DB for the number of visitors on the web app.
get: Carries out a GET operation from Azure Cosmos DB to retrieve the count.
lasttimestamp: Updates the time stamp since the last user went to the website.
Note
The sample Apache JMeter script requires two plugins: Custom Thread Groups and Throughput Shaping Timer. To open the script on your local Apache JMeter instance, you need to install both plugins. You can use the Apache JMeter Plugins Manager to do this.

CREATE THE AZURE LOAD TESTING RESOURCE

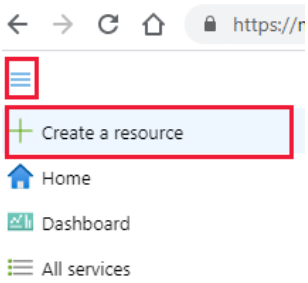
The Azure load testing resource is a top-level resource for your load-testing activities. This resource provides a centralized place to view and manage load tests, test results, and related artifacts.

If you already have a load testing resource, skip this section and continue to Create a load test.

If you don't yet have an Azure load testing resource, create one now:

Sign in to the Azure portal by using the credentials for your Azure subscription.

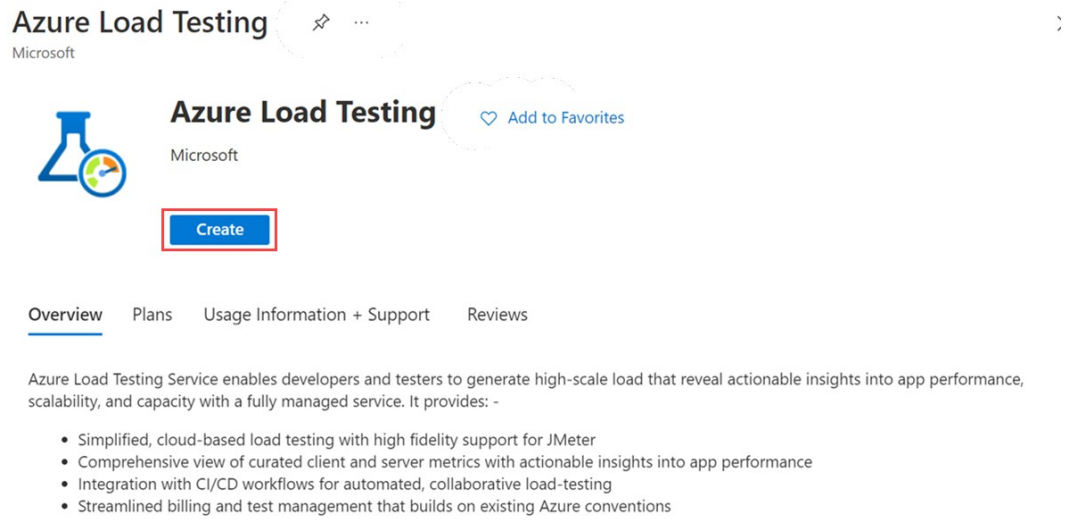
Select the menu button in the upper-left corner of the portal, and then select **+ Create a resource**.



Use the search bar to find **Azure Load Testing**.

Select Azure Load Testing.

On the Azure Load Testing pane, select Create.



Provide the following information to configure your new Azure Load Testing resource:

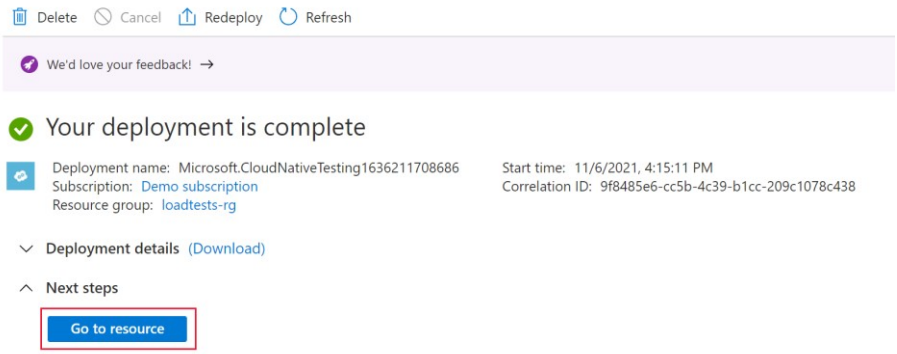
Field	Description
Subscription	Select the Azure subscription that you want to use for this Azure Load Testing resource.
Resource group	Select an existing resource group. Or select **Create new**, and then enter a unique name for the new resource group.
Name	Enter a unique name to identify your Azure Load Testing resource. The name can't contain special characters, such as \/""[]:|<>+=;,?*@&, or whitespace. The name can't begin with an underscore (_), and it can't end with a period (.) or a dash (-). The length must be 1 to 64 characters.
Location	Select a geographic location to host your Azure Load Testing resource. This location also determines where the test engines are hosted and where the JMeter client requests originate from.

Note

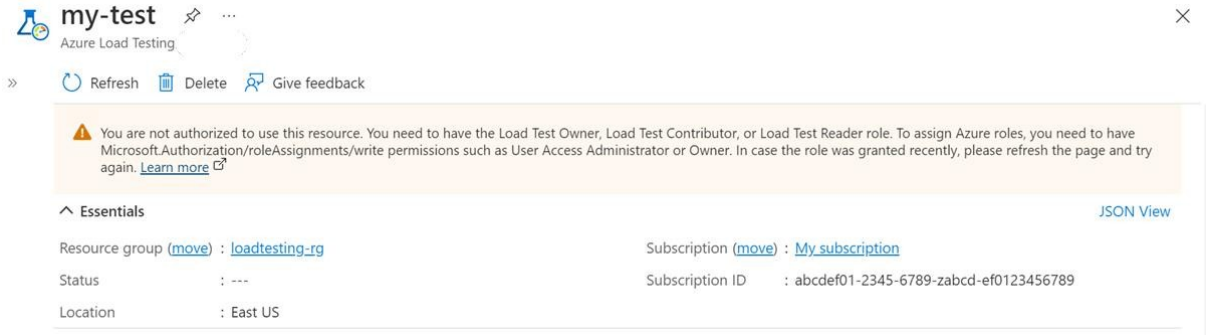
Optionally, you can configure more details on the **Tags** tab. Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups.

After you're finished configuring the resource, select **Review + Create**.

Review all the configuration settings and select **Create** to start the deployment of the Azure Load Testing resource.
When the process has finished, a deployment success message appears.
To view the new resource, select **Go to resource**.

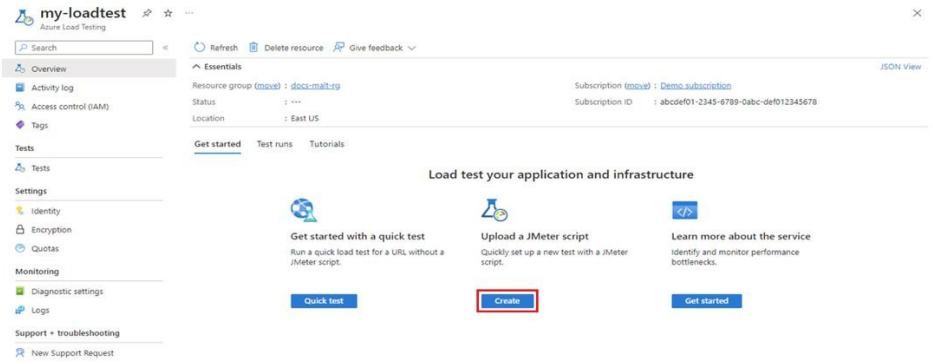


Optionally, [manage access to your Azure Load Testing resource].
Azure Load Testing uses role-based access control (RBAC) to manage permissions for your resource. If you encounter this message, your account doesn't have the necessary permissions to manage tests.

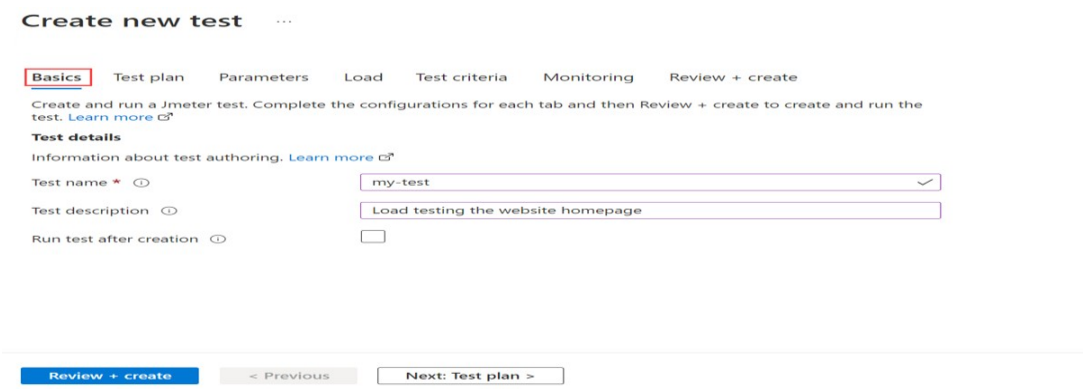


CREATE A LOAD TEST

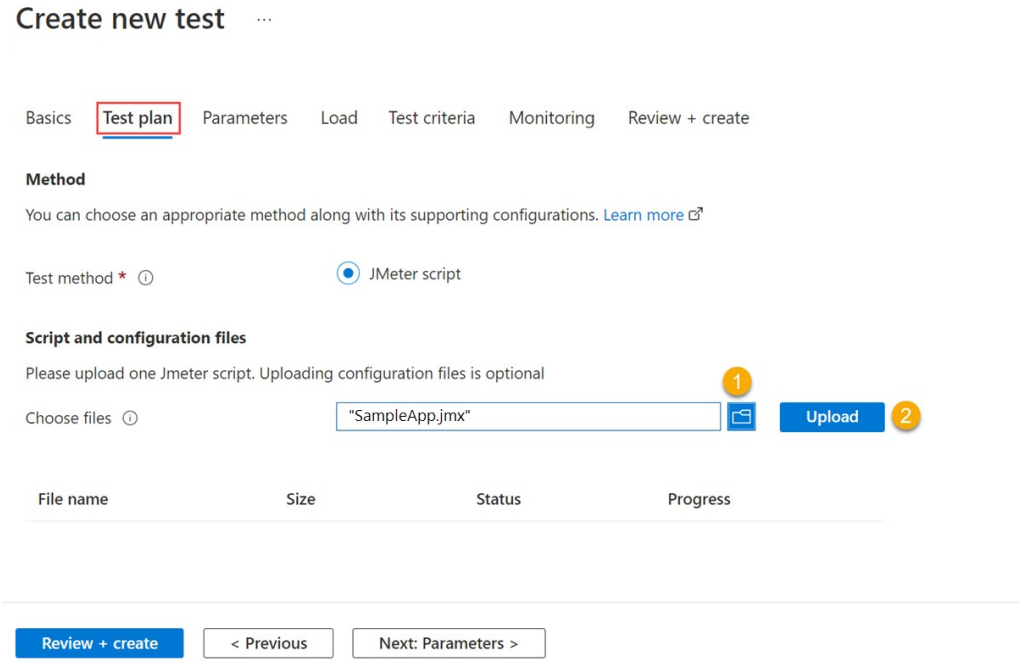
Next, you create a load test in your load testing resource for the sample app. You create the load test by using an existing JMeter script in the sample app repository.
Go to your load testing resource, and select **Create** on the **Overview** page.



On the **Basics** tab, enter the **Test name** and **Test description** information. Optionally, you can select the **Run test after creation** checkbox to automatically start the load test after creating it.



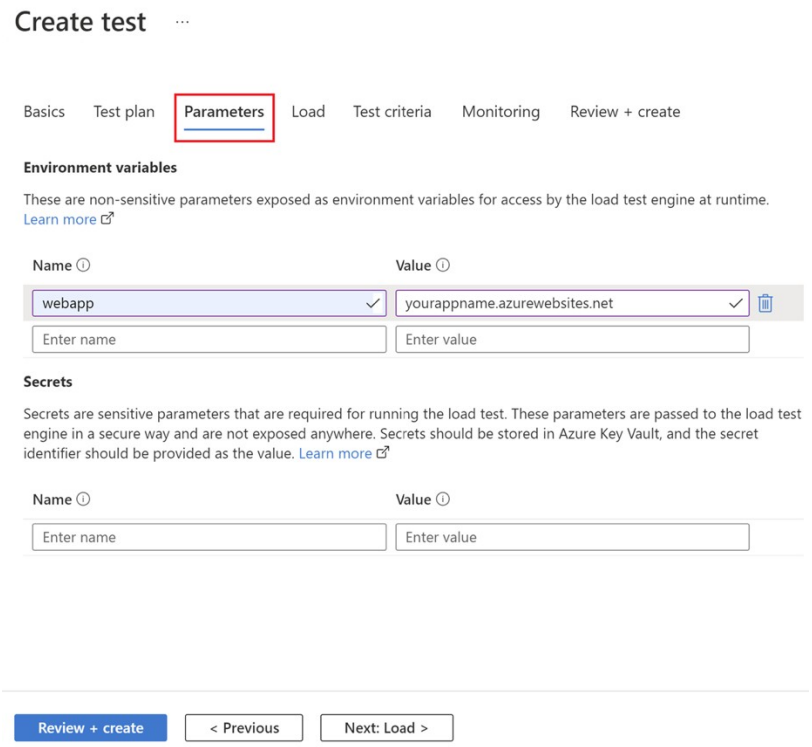
On the **Test plan** tab, select the **JMeter script** test method, and then select the *SampleApp.jmx* test script from the cloned sample application directory. Next, select **Upload** to upload the file to Azure and configure the load test.



Optionally, you can select and upload additional Apache JMeter configuration files or other files that are referenced in the JMX file. For example, if your test script uses CSV data sets, you can upload the corresponding *.csv* file(s).

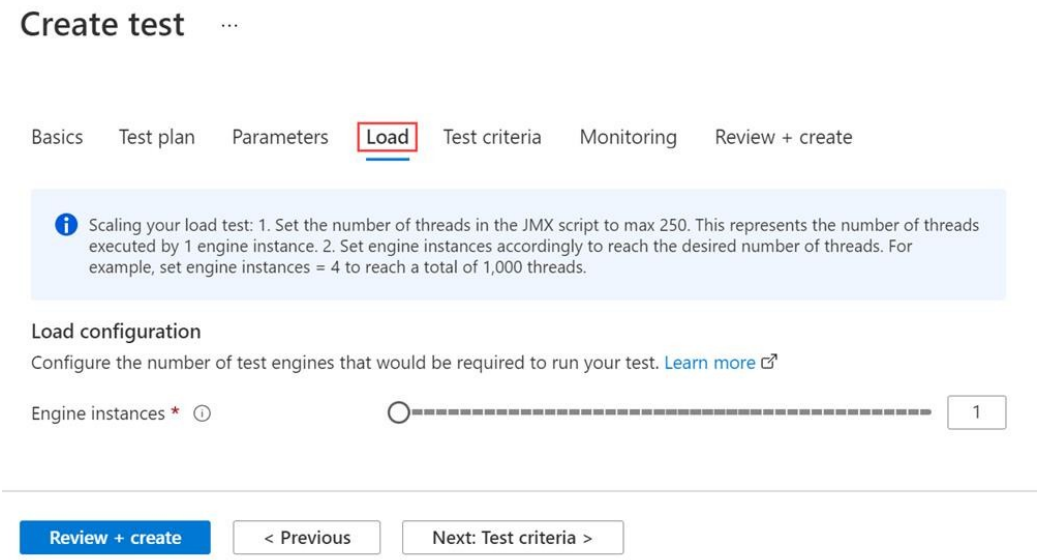
On the **Parameters** tab, add a new environment variable. Enter *webapp* for the **Name** and *<yourappname>.azurewebsites.net* for the **Value**. Replace the placeholder text <yourappname> with the name of the newly deployed sample application. Don't include the https:// prefix.

The Apache JMeter test script uses the environment variable to retrieve the web application URL. The script then invokes the three APIs in the web application.

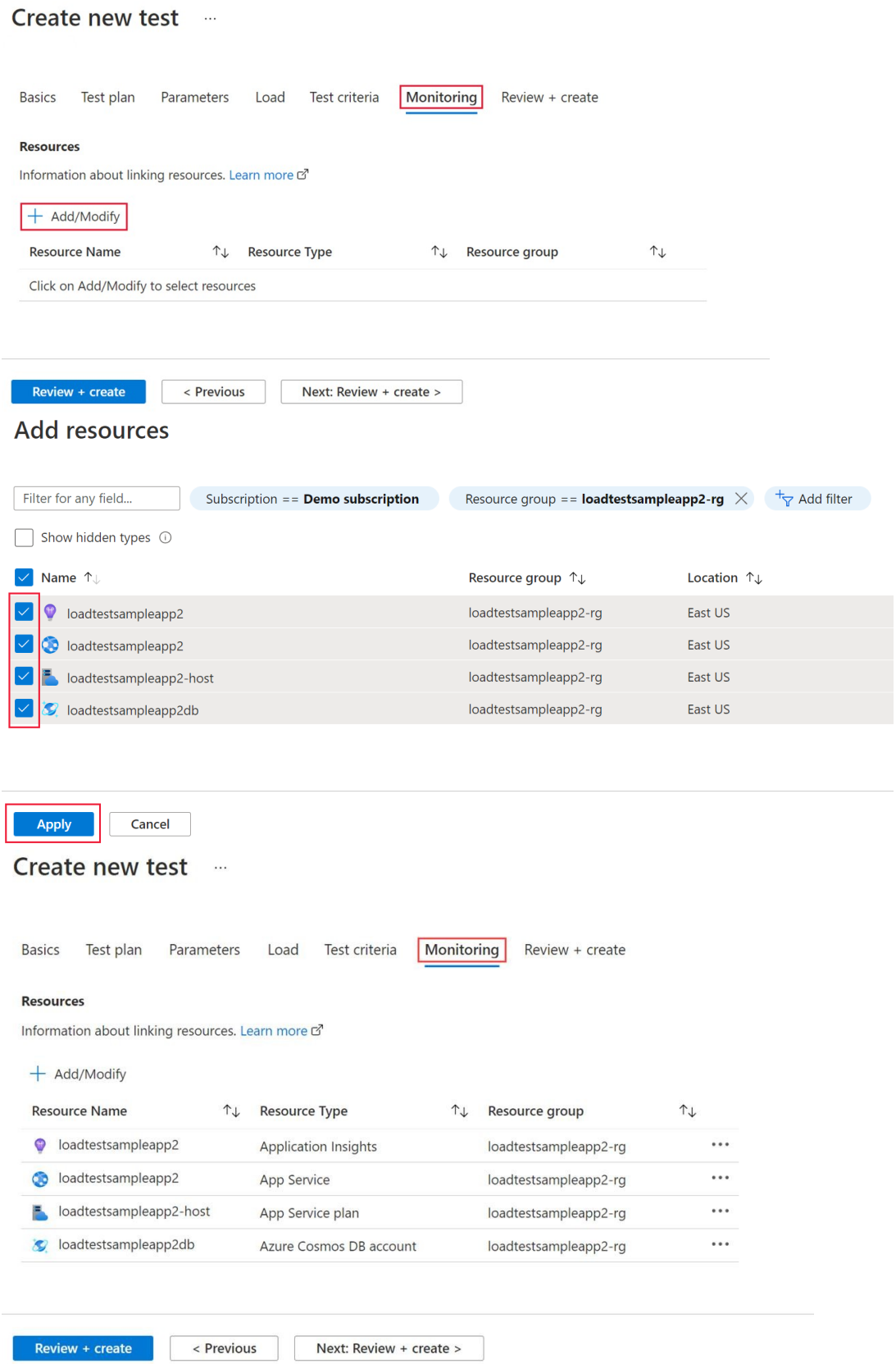


On the **Load** tab, configure the following details. You can leave the default value for this tutorial.

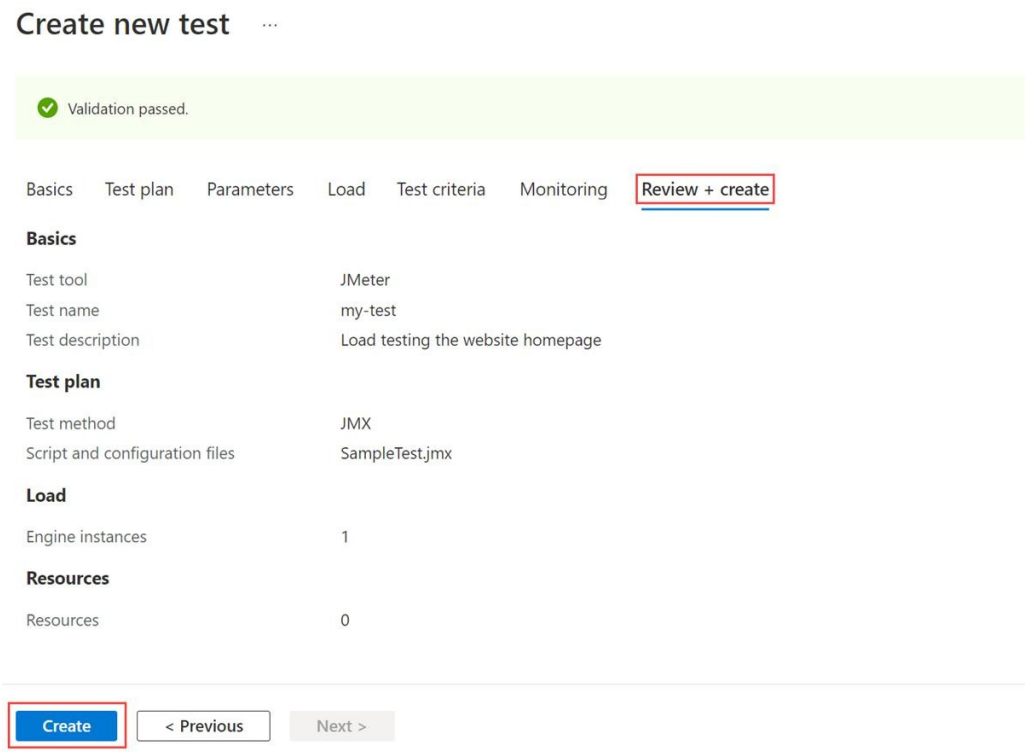
Setting	Value	Description
Engine instances	1	The number of parallel test engines that run the Apache JMeter script.



On the **Monitoring** tab, specify the application components that you want to monitor with the resource metrics. Select **Add/modify** to manage the list of application components.



Select **Review + create**, review all settings, and select **Create**.



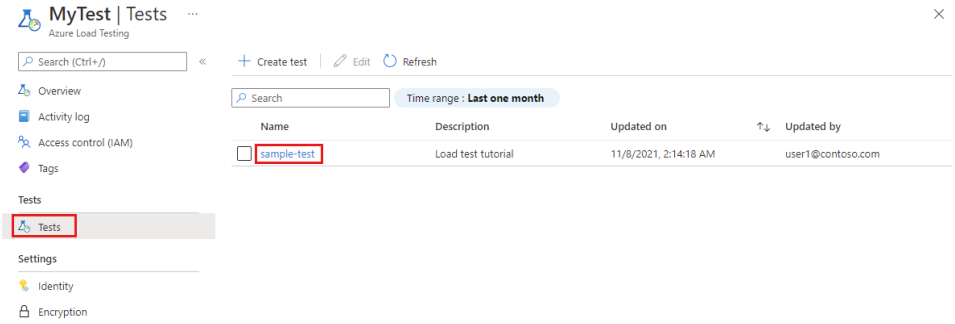
Note

You can update the test configuration at any time, for example to upload a different JMX file. Choose your test in the list of tests, and then select **Edit**.

Run the load test in the Azure portal

In this section, you'll use the Azure portal to manually start the load test that you created previously. If you checked the **Run test after creation** checkbox, the test will already be running.

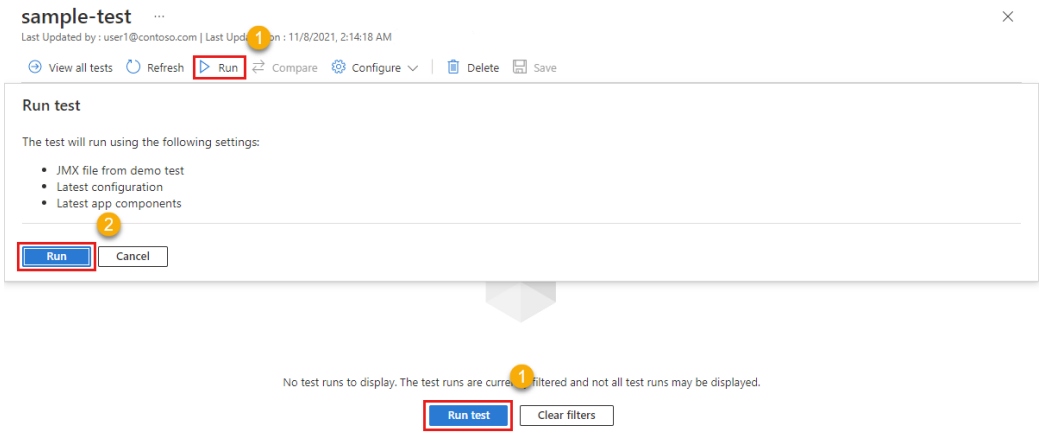
Select **Tests** to view the list of tests, and then select the test that you created.



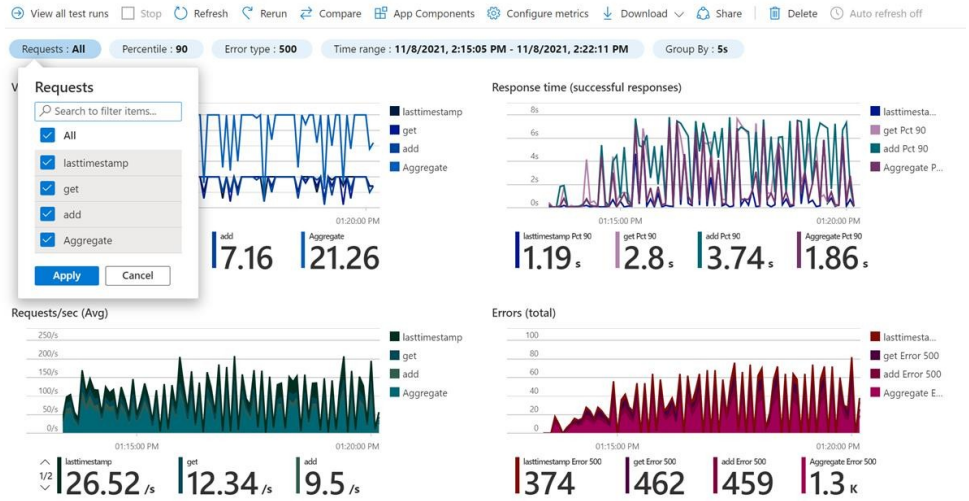
Tip

You can use the search box and the **Time range** filter to limit the number of tests.

On the test details page, select **Run** or **Run test**. Then, select **Run** on the **Run test** confirmation pane to start the load test.



Azure Load Testing begins to monitor and display the application's server metrics on the dashboard. You can see the streaming client-side metrics while the test is running. By default, the results refresh automatically every five seconds.



You can apply multiple filters or aggregate the results to different percentiles to customize the charts.

Tip

You can stop a load test at any time from the Azure portal by selecting **Stop**.

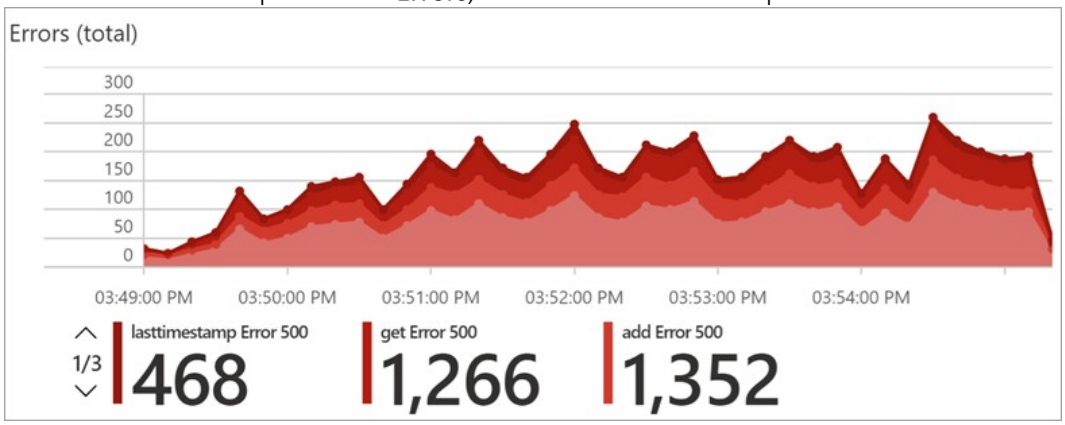
Wait until the load test finishes fully before you proceed to the next section.

Identify performance bottlenecks

In this section, you'll analyze the results of the load test to identify performance bottlenecks in the application. Examine both the client-side and server-side metrics to determine the root cause of the problem.

First, look at the client-side metrics. You'll notice that the 90th percentile for the **Response time** metric for the add and get API requests is higher than it is for the lasttimestamp API.

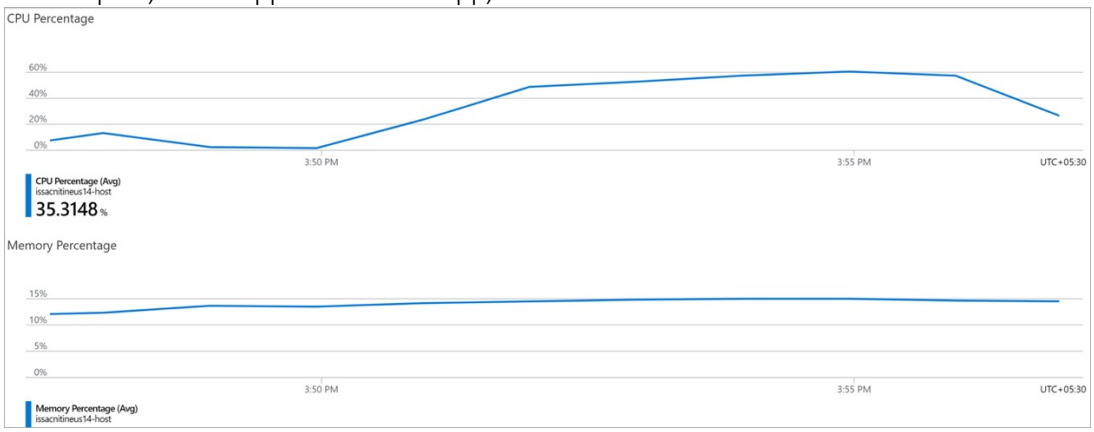
You can see a similar pattern for **Errors**, where the lasttimestamp API has fewer errors than the other APIs.



The results of the add and get APIs are similar, whereas the lasttimestamp API behaves differently. The cause might be database related, because both the add and get APIs involve database access.

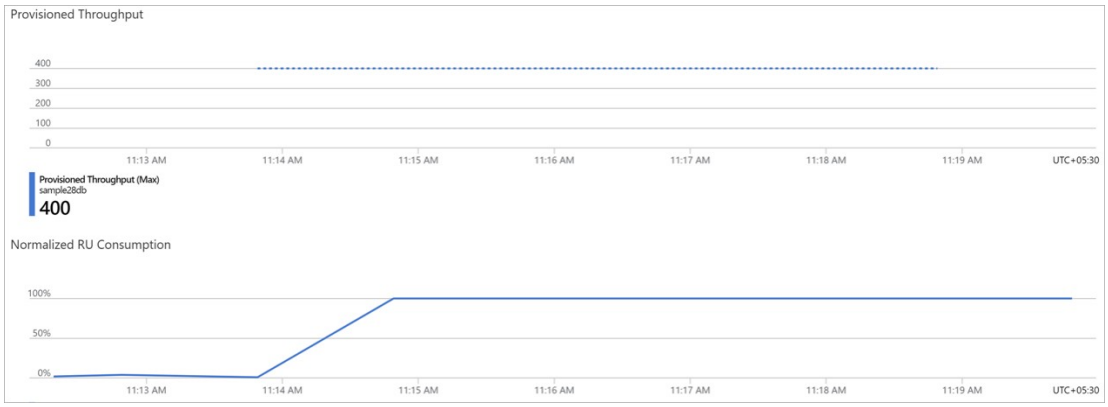
To investigate this bottleneck in more detail, scroll down to the **Server-side metrics** dashboard section.

The server-side metrics show detailed information about your Azure application components: Azure App Service plan, Azure App Service web app, and Azure Cosmos DB.



In the metrics for the Azure App Service plan, you can see that the **CPU Percentage** and **Memory Percentage** metrics are within an acceptable range.

Now, look at the Azure Cosmos DB server-side metrics.



Notice that the **Normalized RU Consumption** metric shows that the database was quickly running at 100% resource utilization. The high resource usage might have caused database throttling errors. It also might have increased response times for the add and get web APIs.

You can also see that the **Provisioned Throughput** metric for the Azure Cosmos DB instance has a maximum throughput of 400 RUs. Increasing the provisioned throughput of the database might resolve the performance problem.

Increase the database throughput

In this section, you'll allocate more resources to the database, to resolve the performance bottleneck.

For Azure Cosmos DB, increase the database RU scale setting:

Go to the Azure Cosmos DB resource that you provisioned as part of the sample application deployment.

Select the **Data Explorer** tab.

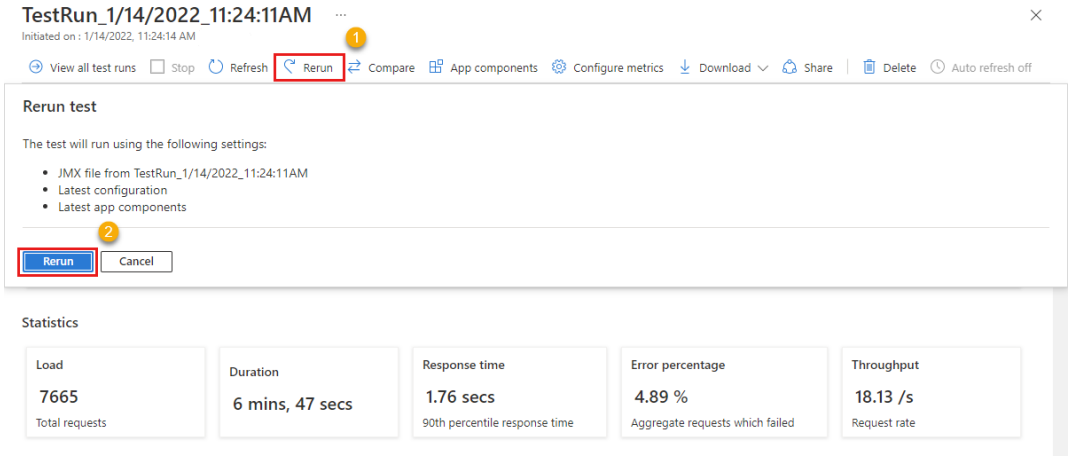
Select **Scale & Settings**, and update the throughput value to **1200**.

Select **Save** to confirm the changes.

Validate the performance improvements

Now that you've increased the database throughput, rerun the load test and verify that the performance results have improved:

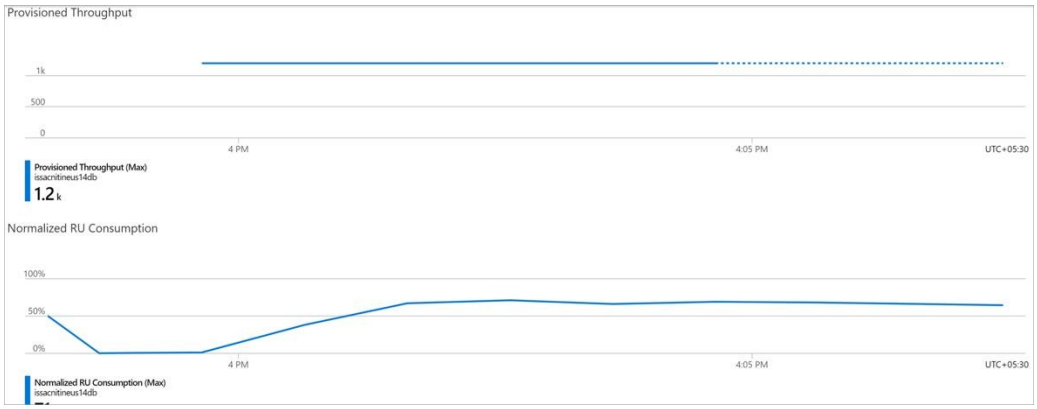
On the test run dashboard, select **Rerun**, and then select **Rerun** on the **Rerun test** pane.



You'll see a new test run entry with a status column that cycles through the **Provisioning**, **Executing**, and **Done** states. At any time, select the test run to monitor how the load test is progressing.

After the load test finishes, check the **Response time** results and the **Errors** results of the client-side metrics.

Check the server-side metrics for Azure Cosmos DB and ensure that the performance has improved.



The Azure Cosmos DB **Normalized RU Consumption** value is now well below 100%.

Now that you've changed the scale settings of the database, you see that:

The response time for the add and get APIs has improved.

The normalized RU consumption remains well under the limit.

As a result, the overall performance of your application has improved.

Clean up resources

Important

You can reuse the Azure Load Testing resource that you created for other Azure Load Testing tutorials and how-to articles.

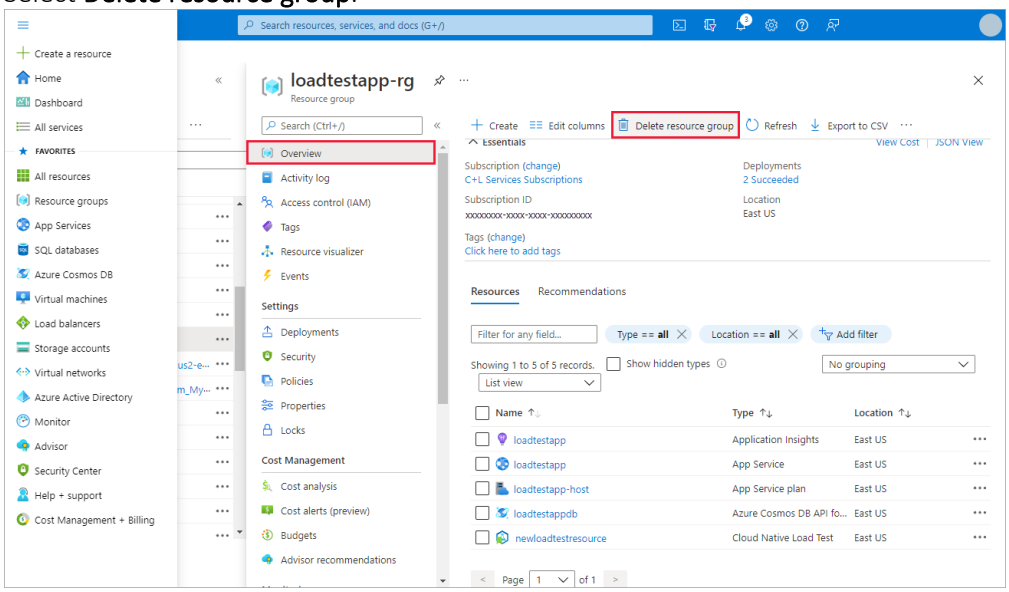
If you don't plan to use any of the resources that you created, delete them so you don't incur any further charges. If you've deployed the sample application in a different resource group, you might want to repeat the following steps.

To delete resources by using the Azure portal:

Select the menu button in the upper-left corner, and then select **Resource groups**.

From the list, select the resource group that you created.

Select **Delete resource group**.



Enter the resource group name. Then select **Delete**.

To delete resources by using the Azure CLI, enter the following command:

Azure CLICopy

az group delete --name<yourresourcegroup>

Remember, deleting the resource group deletes all of the resources within it

A Case Study on Benchmarking and Performance studies of MapReduce / HadoopFramework on Blue Waters Supercomputer

MapReduce is an emerging and widely used programming model for large-scale data parallel applications thatrequire to process large amount of raw data.

There are severalimplementations of MapReduce framework, among which ApacheHadoop is the most commonly used and open source implementation. These frameworks are rarely deployed on supercomputers asmassive as Blue Waters. We want to evaluate how such massiveHPC resource can help solving large-scale data analytics, data-mining problems using MapReduce / Hadoop framework.

In this Case Study we present our studies and detailed performanceanalysis of MapReduce / Hadoop framework on Blue Waters Super-computer. We have used standard popular MapReduce benchmarksuite that represents wide range of MapReduce applications withvarious computation and data densities. Also, we are planning touse Intel HiBenchHaddopBenchmark Suite in future.

We identifyfew factors that significantly affect the performance of MapReduce / Hadoop and shed light on few alternatives that can improve theoverall performance of MapReduce techniques on the system.

The results we have obtained strengthen our belief in possibility of using massive specialized supercomputers to tacklebig data problems. We demonstrate the initial performance of the MapReduce / Hadoop framework with encouraging results andwe are confident that the massive traditional High PerformanceComputing resource can be useful in tackling the big-data researchchallenges and in solving large-scale data analytics, dataminingproblems.

Map Phase:
In the map phase, the input data ispartitioned into input splits and assigned to Map tasksassociated with processing nodes in the cluster. The Maptask typically executes on the same node containing its assigned partition of data in the cluster. These Map tasks Phasesperform user-specified computations on each input key-value pair from the partition of input data assigned to thetask, and generate a set of intermediate results for each key.

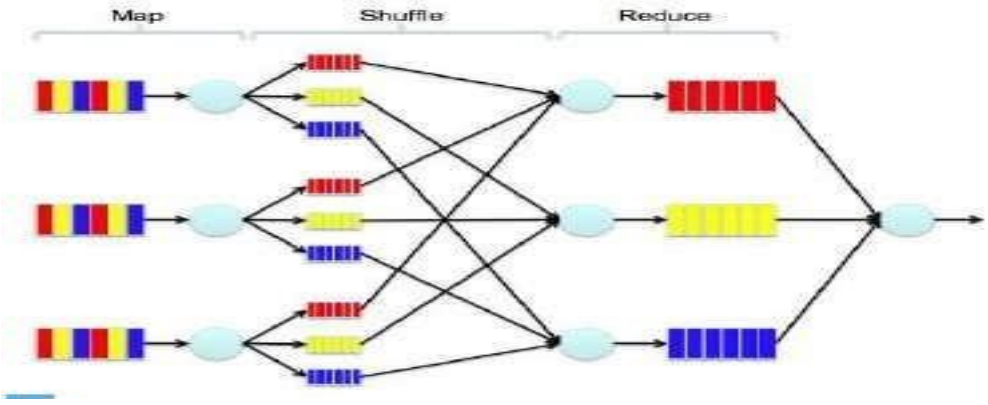


Fig. 1: MapReduce

The shuffle and sort phase:
The shuffle and sort phasesorts the intermediate data generated by each Map taskfrom other nodes and divides this data into regions to beprocessed by the reduce tasks. This phase also distributesthis data as needed to nodes where the Reduce tasks willexecute.

Reduce Phase:
In reduce phase, the data divided by shuffle and sort phase is processed. The Reduce tasks performadditional user-specified operations on the intermediate datapossibly merging values associated with a key to a smallerset of values to produce the output data.All Map tasks must complete prior to the shuffle and sortand reduce phases. The number of Reduce tasks does notneed to be the same as the number of Map tasks. For morecomplex data processing procedures, multiple MapReducecalls may be linked together in sequence.The MapReduce programming model is also becomingpopular in scientific computing, where scientists need tofrequently analyse a large volume of experimental and

simulation data. Such data analysis is often implemented asindependent tasks that can be expressed as mapping operations in MapReduce. For example, in genome sequencing,the matching of large number of sequences against a hugecollection of known sequences can be considered as mappingof similarity function to pair of sequences. Similarly, forthe post-processing of simulation data, the tasks can beexpressed as MapReduce, where a single program is runmultiple times with different input parameters. MapReduceis a simple and scalable approach that enables scientists toachieve simulation results from large-scale data.

In this Case Study we conduct initial benchmarking and performance results of MapReduce framework on the BlueWaterspetascale system. We have briefly described thechallenges of using Mapreduce / Hadoop framework onHigh Performance Computing (HPC) platforms. We haveused Apache Hadoop, the most popular and commonlyused MapReduce framework. However, there is no official / formal support for Hadoop or related stack on the BlueWaters system.

Experimental Setup

In this section we describe the benchmarking environmentalong with the configuration setting we used for Hadoopdeployment on Blue Waters system and we detail some of the challenges faced in deploying Apache Hadoop softwarestack on the Blue Waters system.

Benchmarking Environment

We have used JYC, the Test and Development System(TDS) as well as Blue Waters system for our experiments.Blue Waters has 22640 XE and 4224 XK nodes while JYCconsists of 76 XE and 8 XK nodes. The file system is Lustrewhich is shared across all the nodes.Resource management and scheduling is handled byTorque. So each job may or may not get different nodes n the systems and at different network location. We haveintegrated Yarn with the existing resource management andscheduling software.
We useccmrunsupported by ClusterCompatibility Mode on Cray systems to properly launch theMapReduce / Hadoop workload on the system.

MapReduce and Hadoop Settings

We have used an Open Source distribution of ApacheHadoop stack 2.3.0. The node manager resource memoryis set to 52 GB which is approximately 80% of memoryavailable on a single compute node. The value for
cpu-coresis set to 32, virtual core to physical core ratio is set to 2and virtual memory to physical memory ratio is set to 2.The memory per container is set to 2 GB, therefore we canhave 25 number of containers per node. The heap sizes formap task and reduce task are set to 1.6 GB and 3.2 GBrespectively. The detailed information on how to set theseparameters is available at and we have followed theseinstructions. We have used same settings on both JYC andBlue Waters systems.

MyHadoop

MyHadoop is a framework used for configuringHadoop on traditional HPC resources using the standard job scheduling and resource manager software. User canrun Hadoop codes on the HPC resource without havingroot privileges using myHadoop. It supports a regular non-persistent mode where the local file system on each computenode is used as the data directory for the Hadoop DistributedFile System (HDFS), and also a persistent mode where theHDFS can be hosted on a shared file system such as Lustreor GPFS. We have used myHadoop version 2.1.0.

Yarn

MapReduce / Hadoop workloads are executed on standard Hadoop cluster with the help of resource managementand scheduling entirely handled by Hadoop framework. Incontrast, the resouce management and scheduling is alwayshandled by special type of dedicated software or tool LikeTorque, Moab or PBS. While, a typical HPC resource hasseveral different users with various types of workloads,Hadoop workload is similar in nature. Each job that runson HPC system can get different node configurations, canbe placed in various toplogy configurations or can getdifferent node types depends upon the type of hardwareconfiguration, available queues and scheduling policies. Thechanges in the standard Hadoop cluster are very rare in termsof node configuration or node placements in the topology.We have integrated Yarn [with the Torque scheduler thatis currently available on the system.