SWINBURNE UNIVERSITY OF TECHNOLOGY

COS80001
Cloud Computing Architect

Lecture 03 Networks

- **Caching with CloudFront**
- **Routing with Route53**

includes material from
ACA Modules 3 and 7

This week – Networks outside the VPC

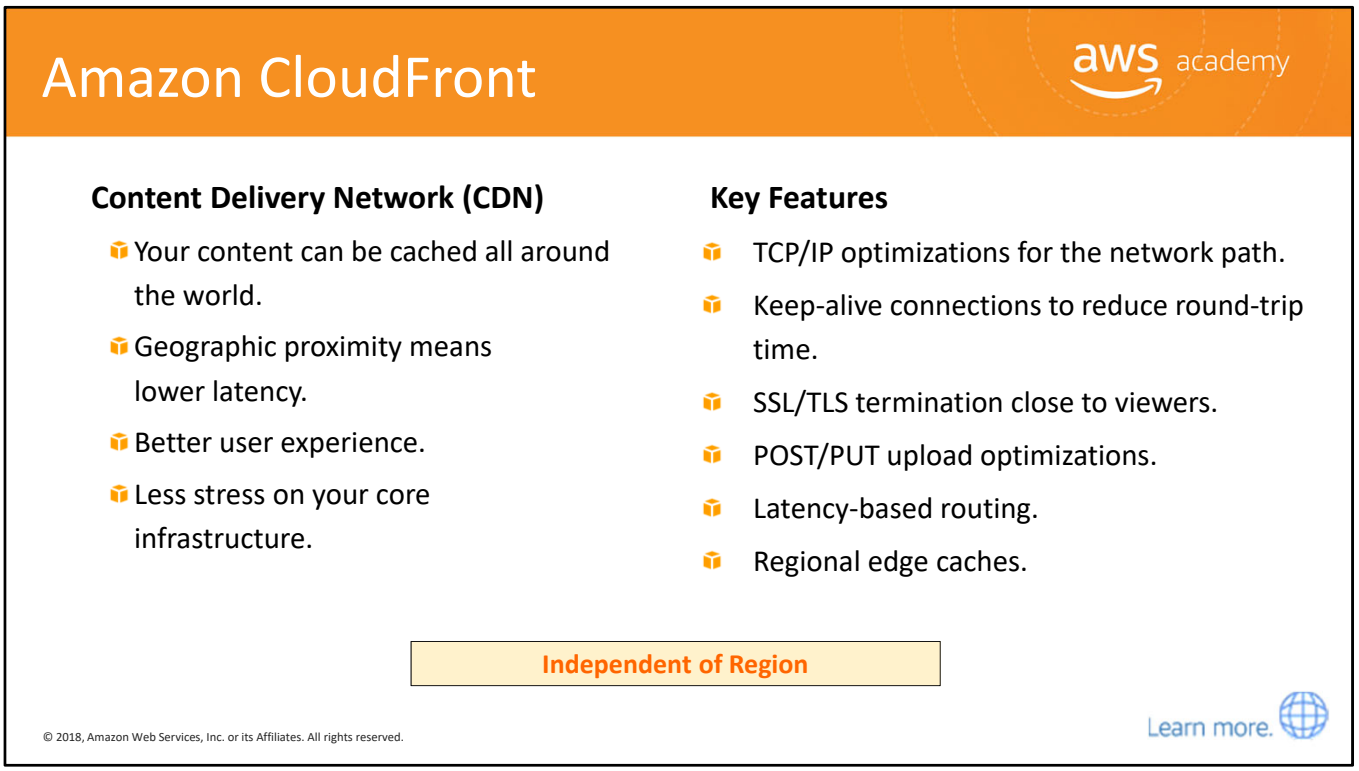
- **Web Caching**
  - **CloudFront**
- **Routing across Regions**
  - **Route53**

SWINBURNE UNIVERSITY OF TECHNOLOGY

Caching with Amazon CloudFront

SWINBURNE UNIVERSITY OF TECHNOLOGY

Now, let's discuss caching with Amazon CloudFront.



Amazon CloudFront is a web service that speeds the distribution of static and dynamic web content to users,—such as Hypertext Markup Language, or HTML, files; CSS files; JavaScript files; and image files. CloudFront is a content delivery network—or CDN—that delivers cached content through a worldwide network of data centers that are called edge locations. This geographic proximity means lower latency when you serve content to users. This makes it more cost effective and faster for a better user experience, as well as putting less stress on your core infrastructure.

Key features for CloudFront include:

- TCP/IP optimizations for the network path.
- Keep-alive connections to reduce round-trip time.
- SSL/TLS termination that is close to viewers.
- POST and PUT upload optimizations.
- Latency-based routing.
- And regional edge caches.

When a user requests content that you serve with CloudFront, the user is routed to the edge location that provides the lowest latency, or time delay. This process means that content is delivered to the user with the best possible performance. If

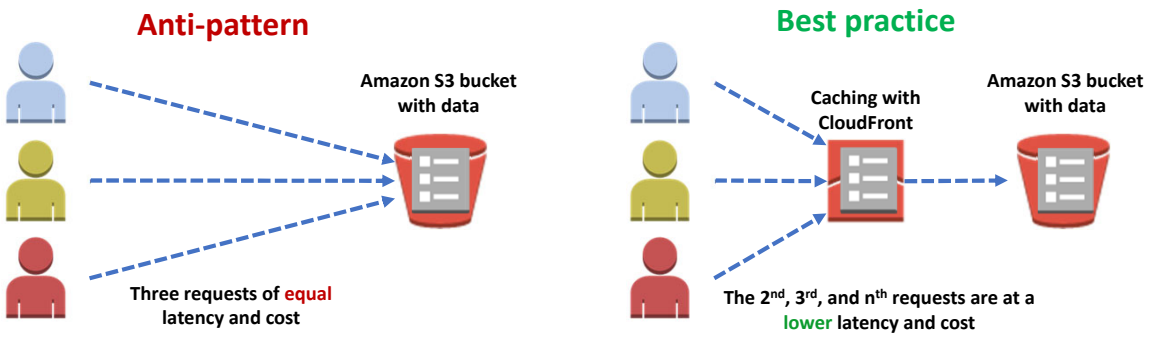
the content is already in the edge location with the lowest latency, CloudFront delivers it immediately. If the content is not currently in that edge location, CloudFront retrieves it from an Amazon S3 bucket or an HTTP server—for example, a web server—hat you have identified as the source for the definitive version of your content.

In addition, Amazon CloudFront has another type of edge location, which is called the regional edge cache. Regional edge caches further improves performance for your viewers. They can also help reduce the load on your origin resources, which minimizes the operational burden and the costs that are associated with scaling your origin resources. Regional edge caches are turned on by default for your CloudFront distributions, and you do not need to make any changes to your distributions to take advantage of this feature. There are also no additional charges to use this feature.

To learn more, go to the following page about CloudFront:
https://aws.amazon.com/cloudfront/details/

Best Practice: Caching

Implement caching at **multiple layers** of an architecture. It can **reduce cost and latency** and **increase performance** of applications.



It is an architectural best practice is to implement caching at multiple layers of an architecture to reduce cost and latency, and to increase the performance of applications. It is more cost-effective to distribute files from CloudFront than from an Amazon S3 bucket.

Caching is the process of temporarily storing data or files in an intermediary location between the requester and the permanent storage. The purpose of caching is to make responding to future requests faster and reduce network throughput.

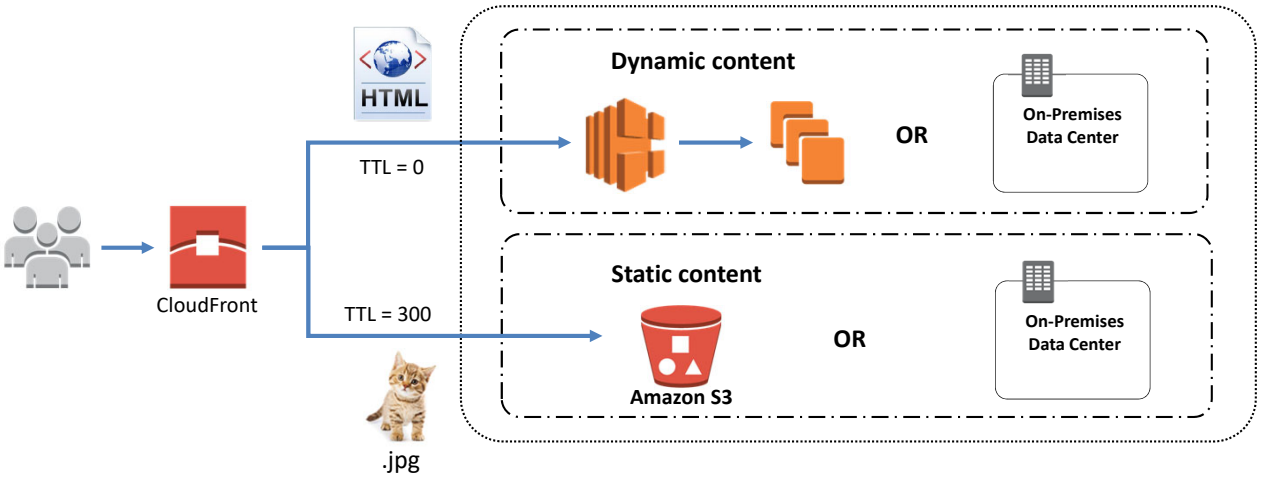
For instance, in the anti-pattern that is shown on the slide, your Amazon S3 bucket does not use a caching service. Three users request a file from one of your Amazon S3 buckets, one at a time. The file is delivered to each user in the same way. The result is that each request takes the same amount of time to complete. This process also incurs costs for the three separate times that the file is delivered for each request.

Let’s compare the process in the anti-pattern with a better pattern. In the best practice pattern, your infrastructure places Amazon CloudFront—which offers caching—in front of Amazon S3. In this scenario, the first request checks for the file in CloudFront. If the request does not find the file in CloudFront, it pulls the file from Amazon S3, and stores a copy of the file in CloudFront at the edge location that is closest to the user. The request then sends a copy of the file to the user who made the request. Now, when any other users request that file, it's

retrieved from the closer edge location in CloudFront. The request does not have to go to Amazon S3 to get the file.

The best practice pattern reduces both latency and cost. After the first request is complete, you no longer pay for the file to be transferred out of Amazon S3.

Cache Static and Reusable Content

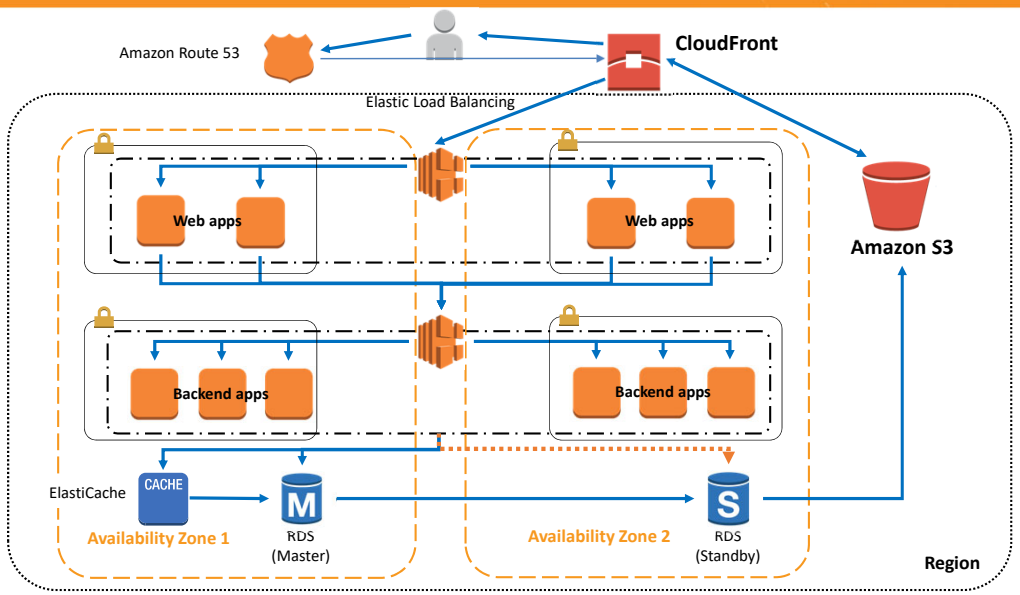


This slide shows another way to use Amazon CloudFront. In general, you only cache static content. However, dynamic or unique content affects the performance of your application. Depending on the demand, you might still get some performance gain by caching the dynamic or unique content in Amazon S3.

For example, if there's something that needs to be personalized, the time to live is zero. A time to live of zero tells CloudFront that each and every time it sees the dynamic content, it needs to go back to its origin because it will change a lot.

At the same time, the time to live can be set to 5 minutes or 24 hours, depending on how long the content is good for. CloudFront can pull content from Amazon S3 so that it can save the load back to an on-premises data center. When you save a load, you can have smaller instances and save money, and resources can perform more efficiently.

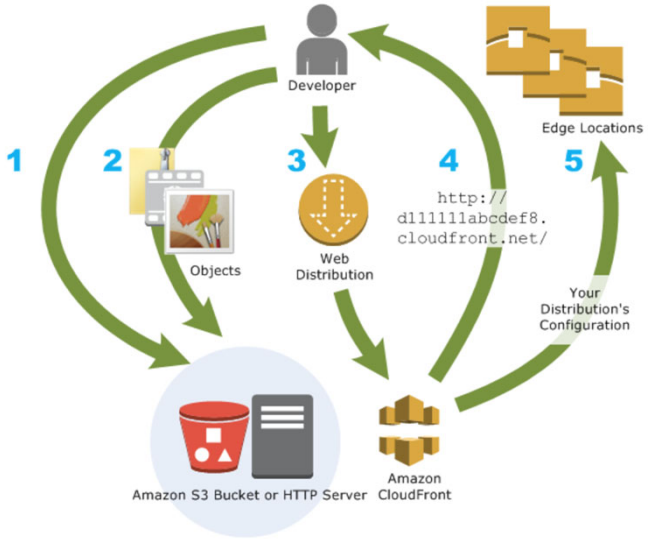
AWS Cloud Architecture: Web Hosting



Let's take a look at cloud architecture for web hosting. We have our Elastic Load Balancing load balancers in place with CloudFront. End users are directed to CloudFront via Amazon Route 53. The load balancers pull content and data from the Amazon S3 bucket, Amazon RDS, or Amazon ElastiCache. This can serve as a read replica if content is cached there.

This architecture diagram shows how you can use CloudFront in front of your hosting architecture to decrease the number of times CloudFront must redirect requests to the load balancer, and pull content.

How to Enable CloudFront?



How can you enable Amazon CloudFront?

First, you specify origin servers, like an Amazon S3 bucket or your own HTTP server. CloudFront gets your files from the origin servers, and the files will then be distributed from CloudFront edge locations all over the world.

Second, you upload your files to your origin servers. Your files, which are also known as objects, typically include webpages, images, and media files.

Third, you create a CloudFront distribution, which tells CloudFront which origin servers to get your files from when users request the files through your website or application. At the same time, you specify details, such as whether you want CloudFront to log all requests, and whether you want the distribution to be enabled as soon as it's created.

Fourth, CloudFront assigns a domain name to your new distribution. You can see the domain name in the CloudFront console. The domain name can also be returned in the response to a programmatic request, like a request from an application programming interface, or API.

Fifth, CloudFront sends your distribution's configuration—but not your content—to all of its edge locations. -collections of servers in geographically dispersed data centers where CloudFront caches copies of your objects.

CloudFront Enablement Options

Use a separate CNAME for static content.

- Static content cached, dynamic content straight from origin.
- Most efficient.
- More effort to set up and manage.

Point entire URL to CloudFront.

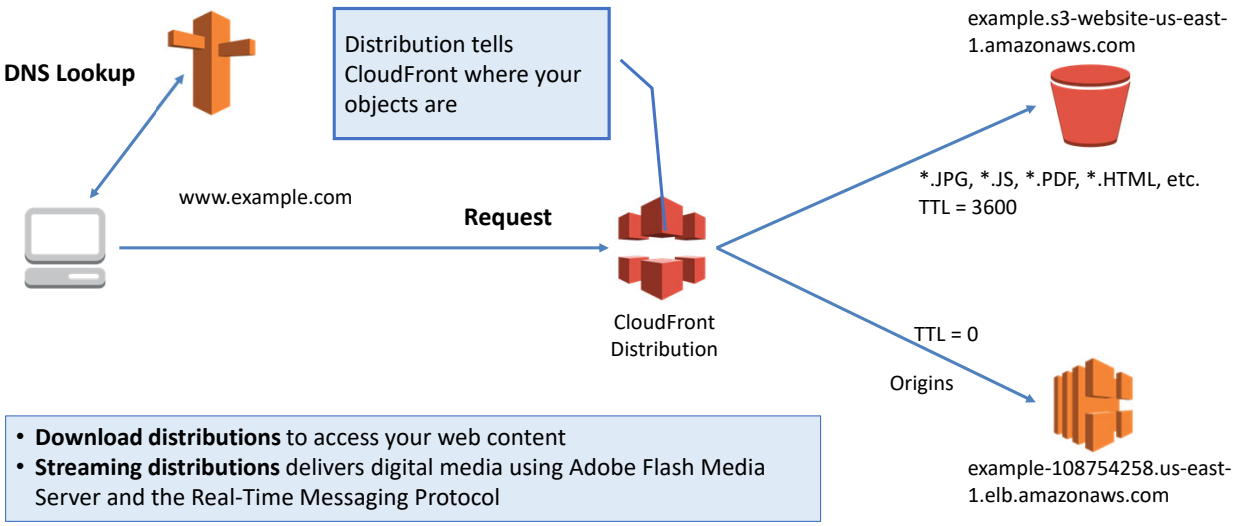
- Easiest to manage.
- Use URL patterns to stage dynamic content.
- ALL content goes through edge locations.

Amazon CloudFront has several options for enablement.

If you choose to use a separate Canonical Name Record—or CNAME—for static content, the static content is cached straight from the origin server. This process is the most efficient, but it takes more effort to set up and manage.

If you point the entire uniform resource locator—or URL—to CloudFront, it’s easier to manage. You can use URL patterns to stage dynamic content. All of the content goes through edge locations.

CloudFront Distributions



- **Download distributions** to access your web content
- **Streaming distributions** delivers digital media using Adobe Flash Media Server and the Real-Time Messaging Protocol

Now let's look at how the CloudFront distributions work. In this example, Amazon Route 53 resolves example.com on behalf of the client. A request is made to the CloudFront distribution. CloudFront looks at the Amazon S3 bucket for the content that it identifies as being stored statically. For any content that has a time to live of zero, CloudFront will go directly to the Elastic Load Balancing load balancer because it has to go back to the origin server to pull the content. In this way, CloudFront delivers both static and dynamic content.

There are two distribution types:

- Web distribution, which lets you access your web content in any combination of up to 10 Amazon S3 buckets or custom origin servers.
- And Real Time Messaging Protocol—or RTMP—distribution, which is always an Amazon S3 bucket.

CloudFront Speeds Up a Website

Use **cache control headers**.

- Cache control header set on your files identifies **static** and **dynamic** content.
- Delivering all your content using a single Amazon CloudFront distribution helps to ensure performance optimization on your entire website.

How does CloudFront speed up a website? Amazon CloudFront reads cache control headers to determine how frequently it needs to check the origin server for an updated version of that file. For an expiration period set to 0 seconds, Amazon CloudFront will revalidate every request with the origin server.

The cache control header that is set on your files identifies both static and dynamic content. The cache control header can even have custom headers within the CloudFront distribution graphical user interface—or GUI— within the console.

Delivering all your content by using a single Amazon CloudFront distribution helps to ensure performance optimization on your entire website.

Expiration Period

The expiration period is set by you.

- If your files don’t change often, set a long expiration period.

How long is a file kept at the edge location?

- Set **expiration period** by setting the cache control headers on your files in your origin.

What about an expiration period? You can set one. If your files don’t change very often, the best practice is to set a long expiration period, and implement a versioning system to manage updates to your files.

By default, if no cache control header is set, each edge location checks for an updated version of your file when it receives a request more than 24 hours after the previous time it checked the origin server for changes to that file.

How long is a file kept at the edge location?
You can set the expiration period by setting the cache control headers on your files in your origin server.

How to Expire Contents

Time to live (TTL)

- Fixed period of time (expiration period).
- Time period is set by you.
- GET request to origin from CloudFront will use `If-Modified-Since header`.

Change object name

- `Header-v1.jpg becomes Header-v2.jpg`.
- New name forces refresh.

Invalidate object

- Last resort: very inefficient and very expensive.

Learn more.

How can you set content to expire? There are three ways to retire cached content:
- Use time to live, or TTL
- Change the object name
- Or invalidate the object

The most preferred options are to use time to live and to change the object name. Using time to live is the easiest option if the replacement does not need to be immediate.

If you set the time to live for a particular origin server to 0, CloudFront will still cache the content from that origin server. It will then make a GET request with an If-Modified-Since header. This header allows the origin server to signal that CloudFront can continue to use the cached content if the content has not changed at the origin server.

Changing the object name requires more effort, but its effects are immediate. There might be some support for this option in some content management systems, or CMSs. Although you can update existing objects in a CloudFront distribution and use the same object names, it is not recommended. CloudFront distributes objects to edge locations only when the objects are requested, not when you put new or updated objects in your origin server. If you update an existing object in your origin server with a newer version that has the same name, an edge location won't get that new version from your origin server until the object is updated and requested.

Invalidating an object should be used sparingly for individual objects. It is a bad solution because the system must forcibly interact with all edge locations.

To learn more, go to the following page:
http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/RequestAndResponseBehaviorS3Origin.html