

So while that's happening I want to talk about the architecture of CloudFront and to do that, I'm going to click on this "Architecture" button and just step through this architecture at a really high level. So at the bottom of this diagram, we've got the different types of origins that we can use. We've got an EC2 instance that's running in a virtual private cloud, an S3 bucket, and a server that's running in a corporate data center. Now the common aspects for all of these three is that they need to be publicly accessible. CloudFront is a public global service, but it always needs to be able to communicate with the origin server that it's connected to. Essentially, an EC2 instance needs a public IP address, a server in a corporate data center also needs a static, public IP address or DNS name and an S3 bucket needs to be configured to be publicly accessible so it can't have any restrictions not allowing certain IP address ranges or preventing it being accessed if you're not using a VPC endpoint, it needs to be a normal S3 bucket that is accessible from the public internet. At the other end of the architecture, we've got customers and our customers essentially browse to the DNS name that a CloudFront distribution has and I'll show you how that works momentarily once this has finished creating. Our customer browses to this DNS name and CloudFront returns the location of a close edge location, the one that CloudFront thinks is going to provide the best performance. And so, at this point, the customer is directly communicating with the edge location asking for one or more objects. It's probably going to start with index.html, but then eventually in the example I'm using in this lesson, it's going to trying to load those different cat pictures. The one from the U.K., the one from Holland, the one from the U.S., and the one from Canada. Essentially, we're talking directly to an edge location. Now, if the edge location does have a cached copy of that content, then it's going to deliver it directly, and that process is known as a cache hit or a cache hit. If it doesn't have a copy of that content what it's going to do is it's going to first check a regional cache and a regional cache you can think of as a big edge location that operates almost like a level two cache—a second level of cache in the system. Now, if the content doesn't exist on this regional cache, which is bigger so it has more storage, it can store more cached content. But if it doesn't have a copy of that content then what happens is a process called origin fetch and that's the process where the edge location pulls a copy of that content from the origin server. So essentially that content is delivered with this green line all the way from our origin server, which is the S3 bucket, to this local edge location. Now the second that the first bite of that content reaches the edge location it starts being delivered to the costumer but in addition, a secondary copy of that data is transferred to the regional cache, which works with all of these local edge locations. Why that's important is that from now on, if a customer requests the same objects then they're going to be delivered from that local edge location. But if for some reason, the same customer or a different customer attempts to access these objects from a similar location, so not the same edge location, but another one in the same area, maybe a different edge location in the same country. If that edge location doesn't have a cached copy of the content rather than having to go all the way back to the origin server, it can consult this regional cache and that can then be delivered from the regional cache to the edge location. So these make CloudFront even more efficient. Now, over time, what will happen is if customers are accessing the same objects over and over again, then they're always going to be delivered from these local edge locations but at some point, if the content isn't accessed, it's going to expire. It's going to age, and it's going to be removed from these edge locations. So generally these edge locations are only storing regularly access data. The cached contents of these regional caches can be more than these edge locations. So you tend to find if you got objects that are accessed on a regular basis, and then for whatever reason, you have no customers accessing them for a certain period of time then they will be cached on the regional cache but not these local ones and that's a perfectly normal process. Objects themselves can have their own TTL or time to live values, which dictates to CloudFront how long they should be cached for. Examples might be one minute, five minutes, a day, you can specify that on a per object basis, but also a CloudFront distribution can have a distribution default TTL value. So that's again, something to keep in mind. And that, in essence, is how CloudFront works and at an associate level, you don't really need to know any more detail than this. What I do want to attempt to demonstrate at this point, this is still deploying, but I want to attempt to see if it's deployed to enough edge locations to just quickly demonstrate how it works. I'm going to click on the distribution. I'm going to try to locate the domain name that's associated with this distribution and look, it's already got one allocated, which is this random string .cloudfront.net and that is the default domain name that every CloudFront distribution gets. It will be different for every distribution that you create and if I close this tab that goes direct to the S2 bucket and open this in a new tab, I should find that the content is loaded from the nearest edge location. So I'm going to attempt to do that now and there we go, our CloudFront distribution, while it is still deploying, it is already accessible. So I've already loaded all of these objects from a nearby edge location. Now what actually happened is because the edge location is just deployed it wouldn't have access to this content and so it would have had to do an origin fetch and that's fine. That all happens in the background. If I keep refreshing this over and over again, we can see that it loads almost immediately and that's for two reasons. One, my local browser probably has a cache, but also any content that's being delivered is being delivered directly now from these edge locations, which are a lot closer to my location in the bucket which is in North Virginia in the U.S. I know for a fact that I have an edge location nearby me in Australia. So this content will be delivered from this nearby edge location. Okay, so just a couple of additional things that I want to talk about before this finishes deploying. This is probably going to take longer than this lesson to deploy and that's okay but I do want to cover a number of additional facts that you'll need to understand for the exam.

By default, **every CloudFront distribution does come with this default domain name and this works both for HTTP as well as HTTPS. I can change this to HTTPS and it's now secure. You are able to specify your own alternative domain names, and you can do that by editing the distribution and then looking in this alternate domain name box.** So I could register a domain and call it globalcats.com if that DNS name was still available and if I had selected that in that box and clicked on, "Yes, edit" that would be applied to the CloudFront distribution and then inside Route 53 I'd need to create this domain, create a record, for example, www or just the globalcats.com apex domain and then just point it at this CloudFront distribution. Now, if I do want to do that, **I'll also need to create an SSL certificate. It used to be that you only had to create an SSL certificate if you wanted to use HTTPS and also wanted to use your own custom domain name. But now, whatever alternative domain name you set in this box, you need to create an SSL certificate within ACM and that proves your ownership of that domain.** So keep that in mind for the exam. I'll be talking about ACM elsewhere in the course, but essentially you can go ahead and request or import certificate with ACM, specify that here, make sure you've got a matching cname, click on "Yes," it will update the distribution, and then you can browse to this custom cname.

Another thing I want to mention is that **by default, CloudFront is a publicly accessible content delivery network. You are able to define that you want a distribution to be private and that's configured inside behaviors. So I select the default behavior, I'll click on "Edit" and if you want to configure this, you need to scroll down.** I'll need to tick this yes box and then I'll need to specify a list of trusted signers. Remember that term for the exam**. If you ever hear the term trusted signers, then you know that it's a private CloudFront distribution.** That might get you one or two answers in the exam without knowing the intricacies of that. So if you **hear trusted signers, you know it's a private distribution.** Now, as soon as you tick yes here, it means you won't be able to access any of the objects in this distribution using anonymous or unauthenticated identities. You won't be able to simply browse to it from your web browser. In order to access it when it's restricted, you'll need to use signed URLs or signed cookies, which I've already talked about in the S3 part of this section. This is just the equivalent, so using signed URLs but you're generating them for use with CloudFront. **So as soon as you restrict a behavior inside a distribution you'll no longer be able to access it publicly. You'll need to use the signed URLs**. Now if I did tick this what would happen is the behavior and because this is the default behavior it means the whole distribution would become private. But what's important to understand is that **if I did make this a private distribution, I would still be able to access the S3 bucket bypassing CloudFront without using the signed URLs. Now you can restrict it. You can make it so that an S3 bucket can only be accessed via CloudFront and that's what we're going to talk about in the next lesson**.