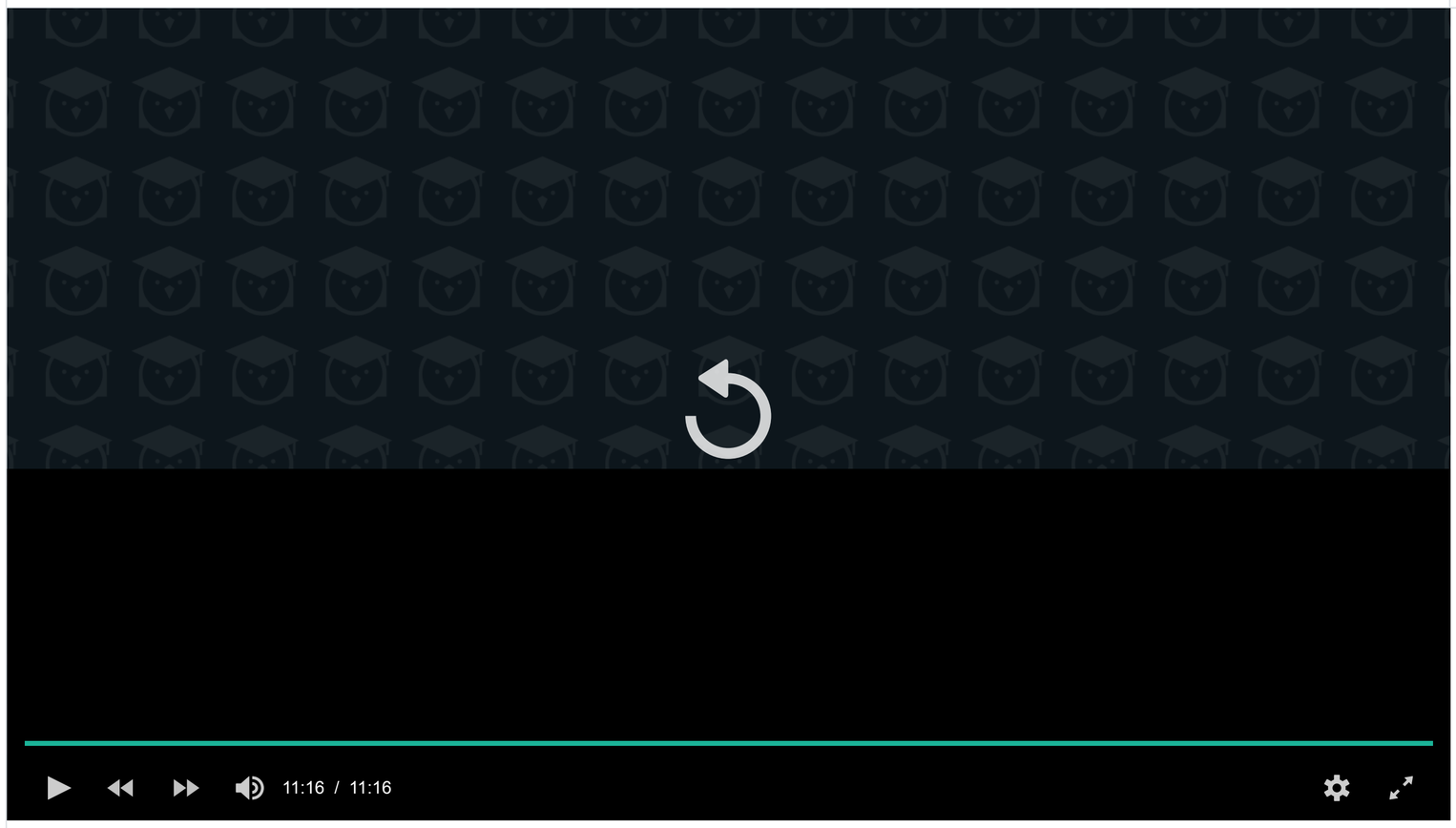
NGINX Web Server Deep Dive

Lecture: Content Compressions and Decompression



Much of what NGINX has configured for us by default is great, but there are some settings that we can adjust to make the server perform even better. One of the adjustments that we can make is to have NGINX compress resources before sending them to the client. This modification isn't free, as it does add some CPU overhead, but not enough to outweigh the benefits.

Note: the commands in this video are run as the `root` user.

Documentation For This Video

- NGINX `http_gzip` module
- NGINX `http_gunzip` module

Configuring and Enabling GZIP

We came across `gzip` when we first installed NGINX and looked at the default configuration, but it was commented out. The simplest way to enable compression for our responses is to uncomment that line in our `nginx.conf` file:

/etc/nginx/nginx.conf (partial)

```
    gzip on;
```

That change is simple enough, but there's more that we can do to make this better by using some of the other directives from the `gzip` module. By default, we're not going to compress the responses that we're getting from proxied servers or any piece of content that isn't HTML. Let's make some modifications to compress more of what we know we'll respond with:

/etc/nginx/nginx.conf (partial)

```
  gzip on;
  gzip_disable msie6;
  gzip_proxied no-cache no-store private expired auth;
  gzip_types text/plain text/css application/x-javascript application/javascript text/xml application/xml application/x
ml+rss text/javascript image/x-icon image/bmp image/svg+xml;
  gzip_min_length 1024;
  gzip_vary on;
```

Let's break down these directives:

- `gzip_disable` - Ensure that we're not sending compressed responses to older versions of Internet Explorer (hopefully no one is actually using these browsers)
- `gzip_proxied` - Specify that we only want to compress responses from proxied servers if we normally wouldn't cache them
- `gzip_types` - The `Content-Type`s that we will compress before sending the response
- `gzip_min_length` - Adjusting the minimum size of a file that we'll compress, the default is 20 bytes, but we're going to not compress resources that are less than one kilobyte
- `gzip_vary` - Adds a `Vary: Accept-Encoding` header. This tells intermediate caches (like CDNs) to treat the compressed and uncompressed version of the same resource as 2 separate entities

What About Decompression

We're handling on-the-fly compression now, but if one of our proxied servers happens to send us a pre-compressed response then we probably want to decompress it for clients that can't handle gzip. In this situation, we'll use the `gunzip` module, and we'll place this in the same area as our `gzip` directives:

/etc/nginx/nginx.conf

```
  gzip on;
  gzip_disable msie6;
  gzip_proxied no-cache no-store private expired auth;
  gzip_types text/plain text/css application/x-javascript application/javascript text/xml application/xml application/x
ml+rss text/javascript image/x-icon image/bmp image/svg+xml;
  gzip_min_length 1024;
  gzip_vary on;
  gunzip on;
```

Exceeded my Expectations | Room for Improvement

Mark Complete & Start Next Lesson

Flash Cards | Notes | Transcript | Downloads

Looks like you haven't created any flash cards for this deck yet.

Create My First Flash Card

Popular Decks | Popular Cards

Nginx Web Server Deep Dive

NGINX Web Server Deep Dive

NGINX Web Server Deep Dive