Unit-4

Internet and E-mail usage

Browser Error Codes

Introduction

When accessing a web server or application, every HTTP request that is received by a server is responded to with an HTTP status code. HTTP status codes are three-digit codes, and are grouped into five different classes. The class of a status code can be quickly identified by its first digit:

- 1xx: Informational
- 2xx: Success
- 3xx: Redirection
- 4xx: Client Error
- 5xx: Server Error

This guide focuses on identifying and troubleshooting the most commonly encountered HTTP **error** codes, i.e. 4xx and 5xx status codes, from a system administrator's perspective. There are many situations that could cause a web server to respond to a request with a particular error code–we will cover common potential causes and solutions.

Client and Server Error Overview

Client errors, or HTTP status codes from 400 to 499, are the result of HTTP requests sent by a user client (i.e. a web browser or other HTTP client). Even though these types of errors are client-related, it is often useful to know which error code a user is encountering to determine if the potential issue can be fixed by server configuration.

Server errors, or HTTP status codes from 500 to 599, are returned by a web server when it is aware that an error has occurred or is otherwise not able to process the request.

General Troubleshooting Tips

- When using a web browser to test a web server, refresh the browser after making server changes
- Check server logs for more details about how the server is handling the requests. For example, web servers such as Apache or Nginx produce two files called `access.log` and `error.log` that can be scanned for relevant information
- Keep in mind that HTTP status code definitions are part of a standard that is implemented by the application that is serving requests. This means that the actual status code that is returned depends on how the server software handles a particular error–this guide should generally point you in the right direction

Now that you have a high-level understanding of HTTP status codes, we will look at the commonly encountered errors.

400 Bad Request

The 400 status code, or *Bad Request* error, means the HTTP request that was sent to the server has invalid syntax.

Unit-4

Internet and E-mail usage

Here are a few examples of when a 400 Bad Request error might occur:

- The user's cookie that is associated with the site is corrupt. Clearing the browser's cache and cookies could solve this issue
- Malformed request due to a faulty browser
- Malformed request due to human error when manually forming HTTP requests (e.g. using `curl` incorrectly)

401 Unauthorized

The 401 status code, or an *Unauthorized* error, means that the user trying to access the resource has not been authenticated or has not been authenticated correctly. This means that the user must provide credentials to be able to view the protected resource.

An example scenario where a 401 Unauthorized error would be returned is if a user tries to access a resource that is protected by HTTP authentication, as in this Nginx tutorial. In this case, the user will receive a 401 response code until they provide a valid username and password (one that exists in the `.htpasswd` file) to the web server.

403 Forbidden

The 403 status code, or a *Forbidden* error, means that the user made a valid request but the server is refusing to serve the request, due to a lack of permission to access the requested resource. If you are encountering a 403 error unexpectedly, there are a few typical causes that are explained here.

File Permissions

403 errors commonly occur when the user that is running the web server process does not have sufficient permissions to read the file that is being accessed.

To give an example of troubleshooting a 403 error, assume the following situation:

- The user is trying to access the web server's index file, from `http://example.com/index.html`
- The web server worker process is owned by the `www-data` user
- On the server, the index file is located at `/usr/share/nginx/html/index.html`

If the user is getting a 403 Forbidden error, ensure that the `www-data` user has sufficient permissions to read the file. Typically, this means that the *other permissions* of the file should be set to *read*. There are several ways to ensure this, but the following command will work in this case:

```
sudo chmod o=r /usr/share/nginx/html/index.html
```

.htaccess

Another potential cause of 403 errors, often intentionally, is the use of an `.htaccess` file. The `.htaccess` file can be used to deny access of certain resources to specific IP addresses or ranges, for example.

If the user is unexpectedly getting a 403 Forbidden error, ensure that it is not being caused by your `.htaccess` settings.

Unit-4

Internet and E-mail usage

Index File Does Not Exist

If the user is trying to access a directory that does not have a default index file, and directory listings are not enabled, the web server will return a 403 Forbidden error. For example, if the user is trying to access `http://example.com/emptydir/`, and there is no index file in the `emptydir` directory on the server, a 403 status will be returned.

If you want directory listings to be enabled, you may do so in your web server configuration.

404 Not Found

The 404 status code, or a *Not Found* error, means that the user is able to communicate with the server but it is unable to locate the requested file or resource.

404 errors can occur in a large variety of situations. If the user is unexpectedly receiving a 404 Not Found error, here are some questions to ask while troubleshooting:

- Does the link that directed the user to your server resource have a typographical error in it?
- Did the user type in the wrong URL?
- Does the file exist in the correct location on the server? Was the resource was moved or deleted on the server?
- Does the server configuration have the correct document root location?
- Does the user that owns the web server worker process have privileges to traverse to the directory that the requested file is in? (Hint: directories require read and execute permissions to be accessed)
- Is the resource being accessed a symbolic link? If so, ensure the web server is configured to follow symbolic links

500 Internal Server Error

The 500 status code, or *Internal Server Error*, means that server cannot process the request for an unknown reason. Sometimes this code will appear when more specific 5xx errors are more appropriate.

This most common cause for this error is server misconfiguration (e.g. a malformed `.htaccess` file) or missing packages (e.g. trying to execute a PHP file without PHP installed properly).

502 Bad Gateway

The 502 status code, or *Bad Gateway* error, means that the server is a gateway or proxy server, and it is not receiving a valid response from the backend servers that should actually fulfill the request.

If the server in question is a reverse proxy server, such as a load balancer, here are a few things to check:

- The backend servers (where the HTTP requests are being forwarded to) are healthy
- The reverse proxy is configured properly, with the proper backends specified
- The network connection between the backend servers and reverse proxy server is healthy. If the servers can communicate on other ports, make sure that the firewall is allowing the traffic between them
- If your web application is configured to listen on a socket, ensure that the socket exists in the correct location and that it has the proper permissions

Unit-4

Internet and E-mail usage

503 Service Unavailable

The 503 status code, or *Service Unavailable* error, means that the server is overloaded or under maintenance. This error implies that the service should become available at some point.

If the server is not under maintenance, this can indicate that the server does not have enough CPU or memory resources to handle all of the incoming requests, or that the web server needs to be configured to allow more users, threads, or processes.

504 Gateway Timeout

The 504 status code, or *Gateway Timeout* error, means that the server is a gateway or proxy server, and it is not receiving a response from the backend servers within the allowed time period.

This typically occurs in the following situations:

- The network connection between the servers is poor
- The backend server that is fulfilling the request is too slow, due to poor performance
- The gateway or proxy server's timeout duration is too short

Conclusion

Now that you are familiar with the most common HTTP error codes, and common solutions to those codes, you should have a good basis for troubleshooting issues with your web servers or applications.

If you encounter any error codes that were not mentioned in this guide, or if you know of other likely solutions to the ones that were described, feel free to discuss them in the comments!