ClientDeniedByServerConfiguration

Client denied by server configuration

This error means that the access to the directory on the file system was denied by an Apache configuration.

Apache HTTP server 2.4 notes

The 2.4 release introduced significant changes to the authorization and authentication process. Users of that release are encouraged to read this link to migrate their older config files.

Before you start

Before attempting to alter any existing config file, please take note of the full file system path for which access is being denied, and the IP or hostname of the client:

```
[<date here>] [error] [client ::1] client denied by server configuration: /var/www/example.com/
```

Using the correct path in the directory block for the following examples is essential to solving this problem. In this case, a client from the local machine (::1) is being denied access to /var/www/example.com .

Troubleshooting

First, remember "Directory" permissions propagate to subdirectories by default.

The possible causes are:

- Access was denied due to an explicit deny (2.2) directive or require (2.4) directive in a directory block or .htaccess file.

```
DocumentRoot /var/www/example.com
```

2.2:

```
<Directory /var/www/example.com>
  Order deny,allow
  Deny from all
</Directory>
```

2.4:

```
<Directory /var/www/example.com>
  Require all denied
</Directory>
```

In the above examples, using the following configuration will resolve the issue:

2.2:

```
<Directory /var/www/example.com>
  Order allow,deny
  Allow from all
</Directory>
```

2.4:

```
<Directory /var/www/example.com>
  Require all granted
</Directory>
```

- An attempt to access a directory outside of the DocumentRoot defined by an alias without a corresponding directory block.

```
DocumentRoot /var/www/example.com

Alias /foo /var/www/foo
```

Solution (2.2):

```
<Directory /var/www/foo>
  Order allow,deny
  Allow from all
</Directory>
```

Solution (2.4):

```
<Directory /var/www/foo>
  Require all granted
</Directory>
```

- Proxying to a service with no explicit access in a location block.

```
ProxyPass /foo/ http://internal.example.com:8900/

ProxyPassReverse /foo/ http://internal.example.com:8900/
```

Solution (2.2):

```
<Location /foo>
  Order allow,deny
  Allow from all
</Location>
```

Solution (2.4):

```
<Location /foo>
  Require all granted
</Location>
```

- A PUT request was received; a 403 is the default response. Access can be granted with limitexcept (2.2) or mod_allowmethods (2.4).
- A mix of allow (2.2) and require (2.4) directives while using apache HTTPD 2.4, used in the same or separate directory blocks. The new 2.4 directives should be used exclusively, and the mod_access_compat module should be unloaded by commenting out the LoadModule directive.

```
<Directory /var/www/example.com>
  Order allow,deny
  Allow from all
  Require all granted
</Directory>
```

The solution:

```
<Directory /var/www/example.com>
  Require all granted
</Directory>
```

- Using mod_security with an explicit directive to deny access. Altering or commenting out the offending directives from that module will resolve the issue.
- Using a bandwidth or rate limiting module such as mod_evasive, mod_limitipconn or mod_bw. A capable firewall is far more efficient at limiting traffic bursts, and abusive clients.

Words of caution

The following configuration may be included in your apache HTTPD configuration; its purpose is to prevent unauthorized access to the root of the file system. Under no condition should it be altered. Instead, the existing directory block for the full file system path should be altered, or a new one should be created if it was not already present.

2.2:

```
<Directory />
  Order deny,allow
  Deny from all
</Directory>
```

2.4:

```
<Directory />
  Require all denied
</Directory>
```

Restricting access a little further

If granting full access to the resource in question is not an option, specific IP addresses, partial IP addresses, network masks and CIDR specifications can be used with the allow and require directives.