私塾在线《高级软件架构师实战培训阶段二》

——跟着cc学架构系列精品教程

Nginx优化补充-1

- 几点说明

1：Nginx默认的配置，已经是经过优化的了
2：对Nginx的优化，集中在对配置的微调上
3：如果调整配置效果不明显的话，增加机器，然后做负载均衡
4：在阶段一学习Ngnix的时候，已经系统的讲过Nginx的配置优化知识，这里再补充几个。

- 配置优化补充

1：tcp_nopush：告诉Nginx在一个数据包里发送所有头文件，可优化吞吐率，建议开启
2：tcp_nodelay：告诉Nginx不要缓存数据，而是一段一段的发送，可以考虑开启
3：client_header_timeout：设置请求头的超时时间，可以设置低些，如10秒
4：client_body_timeout：设置请求体的超时时间，可以设置低些，如10秒
5：reset_timeout_connection：关闭不响应的客户端连接，这将会释放那个客户端所占有的内存空间。建议开启
6：send_timeout：指定客户端的响应超时时间，如果在两次客户端读取操作之间，客户端没有读取任何数据，Nginx就会关闭连接，默认是60秒，可以设置小一些，如10秒。
7：缓存输出日志，如：access_log /logs/nginx/access.log main buffer=64k;
8：对于公共的静态资源，可以考虑放到Nginx里面

Nginx优化补充-2

- Nginx中配置使用二级域名

1：配置server_name，把二级域名配置进去，例如：

```
server_name www.sishuok.com *.sishuok.com;
```

2：判断是二级域名的，把信息保存到变量里面

```
if ( $host ~* (\b(?!www\b)\w+)\.\w+\.com) {
    set $subdomain $1;
    set $suburl $request_uri;
}
```

3：后面进行rewrite的时候，就可以使用这些变量了，比如：

```
location / {
    if ($suburl ~ "/$"){
        rewrite ^/(.*) /$subdomain/$suburl/index.html break;
    }
    proxy_pass   http://cc.com;
}
```

4：本地测试的时候，可以修改/etc/hosts文件

Varnish优化补充-1

- 后端发生变化，如何主动通知varnish更新缓存

思路：通过程序来发起PURGE的请求，使用Varnish的Http PURGE接口来清除缓存。

1：在管理服务器上，curl -X PURGE http://192.168.1.106:1111/a.jsp

2：远程管理，需要配置

（1）acl的配置里面，要把远程的ip加入，例如：

```
acl purgeallow {
        "127.0.0.1";
        "192.168.1.100"; }
```

（2）在vcl_recv里面，要判断是否可以PURGE，例如：

```
if (req.request == "PURGE") {
   if (client.ip ~ purgeallow) {
    return(lookup);   }
}
```

（3）在vcl_hit里面，判断如果是PURGE，就把ttl设置为0，也就是清除缓存，例如：

```
if (req.request == "PURGE") {
    set obj.ttl = 0s;
    error 200 "Purged.";}
```

（4）在vcl_miss里面，设置不用访问后台，例如：

```
if (req.request == "PURGE") {
    error 200 "Purged.";}
```

Varnish优化补充-2

3：Java应用里面，如何发起PURGE的请求

（1）使用Apache的HttpClient，添加需要的依赖包，如下：

```
<dependency>
    <groupId>commons-httpclient</groupId>
    <artifactId>commons-httpclient</artifactId>
    <version>3.1</version>
</dependency>
<dependency>
    <groupId>org.apache.httpcomponents</groupId>
    <artifactId>httpclient</artifactId>
    <version>4.4.1</version>
</dependency>
```

Varnish优化补充-3

（2）自定义PurgeMethod的类，用它来实现PURGE请求，如下：

```
class PurgeMethod extends HttpMethodBase {
    public PurgeMethod() {
        super();
        setFollowRedirects(true);
    }
    public PurgeMethod(String url) {
        super(url);
        setFollowRedirects(true);
    }
    public String getName() {
        return "PURGE";
    }
}
```

Varnish优化补充-4

（3）定义使用的方法，如下：

```
public void purge(String url) {
   HttpClient client = new HttpClient();
   HttpMethod method = new PurgeMethod(url);

   try {
        int status = 0;
        status = client.executeMethod(method);
        System.out.println("status===" + status);
   } catch (Exception e) {
        e.printStackTrace();
   } finally {
        method.releaseConnection();
   }
}
```

（4）然后就可以开始测试了

Varnish优化补充-5

- 页面片断缓存（ESI：Edge Side Includes ）基础

1：ESI是一种语言，用来包含在其他web页面中的web页面片断，可以认为他是一个通过HTTP实现的HTML包含语句

2：Varnish只实现了部分的ESI，基本的有：

（1）<esi:include src=""/>

（2）<esi:remove>

3：Varnish实现ESI的基本流程

- Varnish3开启对ESI的支持

在vcl_fetch里面，设置开启对ESI的支持，例如：

```
if (req.url == "/c.jsp") {
   set beresp.do_esi = true;
   set beresp.ttl = 20 m;
}
```

Varnish优化补充-6

- 输出Varnish的日志到文件

1：Varnish默认是通过内存共享的方式提供日志的，为了把日志输出到文件，而且要自定义日志的格式，常用的方法是：编写一个名为varnishncsa的shell脚本，并把此文件放到/etc/init.d目录下，大致的内容如下：

```
#!/bin/sh
if [ "$1" = "start" ];then
/usr/common/varnish/bin/varnishncsa -F '%t "%r" "%{VCL_Log:userUuid}x" %s %b
    "%{Referer}i" "%{User-agent}i"' -n /home/data/varnish/cache |
    /usr/common/apache/bin/rotatelogs /home/data/varnish/log/varnish.%Y.%m.%d.log
    86400 480 &
elif [ "$1" = "stop" ];then
     killall varnishncsa
else
     echo $0 "{start|stop}"
fi
```

Varnish优化补充-7

2：需要对这个文件授权，如chmod 7777 varnishncsa

3：要预先建好日志输出的文件夹

4：在Varnish的配置文件里面，就可以使用自定义的标记了，比如要输出key为userUuid的日志，形如：

```
std.log("userUuid:now req=="+req.http.cookie);
```

5：在启动varnish的时候要指定 -n，值跟上面的-n后面的值是一样的；如果这个时候想要 使用 varnishlog 之类的，也需要指定 -n，如：

```
./varnishd -f /usr/common/varnish/etc/varnish/default.vcl -s malloc,32M -T 127.0.0.1:2000 -a 0.0.0.0:1111 -n /home/data/varnish/cache
```

6：在varnish启动后，要启动varnishncsa，否则是不能输出到文件的，启动方式如：

```
/etc/init.d/varnishncsa start
```

7：接下来就可以测试了，要做一个输出Cookie里面包含userUuid的页面