参考：<http://blog.csdn.net/foxman209/article/details/6278093/>

Curl常用命令大全

对于windows用户如果用Cygwin模拟unix环境的话，里面没有带curl命令，要自己装，所以建议用Gow来模拟，它已经自带了curl工具，安装后直接在cmd环境中用curl命令就可，因为路径已经自动给你配置好了。

[**Linux**](http://lib.csdn.net/base/linux) curl是一个利用URL规则在命令行下工作的文件传输工具。它支持文件的上传和下载，所以是综合传输工具，但按传统，习惯称url为下载工具。  
  
　　一，curl命令参数，有好多我没有用过，也不知道翻译的对不对，如果有误的地方，还请指正。  
  
　　-a/--append 上传文件时，附加到目标文件  
  
　　-A/--user-agent <string>  设置用户代理发送给服务器  
  
　　- anyauth   可以使用“任何”身份验证方法  
  
　　-b/--cookie <name=string/file> cookie字符串或文件读取位置  
  
　　- basic 使用HTTP基本验证  
  
　　-B/--use-ascii 使用ASCII /文本传输  
  
　　-c/--cookie-jar <file> 操作结束后把cookie写入到这个文件中  
  
　　-C/--continue-at <offset>  断点续转  
  
　　-d/--data <data>   HTTP POST方式传送数据  
  
　　--data-ascii <data>  以ascii的方式post数据  
  
　　--data-binary <data> 以二进制的方式post数据  
  
　　--negotiate     使用HTTP身份验证  
  
　　--digest        使用数字身份验证  
  
　　--disable-eprt  禁止使用EPRT或LPRT  
  
　　--disable-epsv  禁止使用EPSV  
  
　　-D/--dump-header <file> 把header信息写入到该文件中  
  
　　--egd-file <file> 为随机数据(SSL)设置EGD socket路径  
  
　　--tcp-nodelay   使用TCP\_NODELAY选项  
  
　　-e/--referer 来源网址  
  
　　-E/--cert <cert[:passwd]> 客户端证书文件和密码 (SSL)  
  
　　--cert-type <type> 证书文件类型 (DER/PEM/ENG) (SSL)  
  
　　--key <key>     私钥文件名 (SSL)  
  
　　--key-type <type> 私钥文件类型 (DER/PEM/ENG) (SSL)  
  
　　--pass  <pass>  私钥密码 (SSL)  
  
　　--engine <eng>  加密引擎使用 (SSL). "--engine list" for list  
  
　　--cacert <file> CA证书 (SSL)  
  
　　--capath <directory> CA目录 (made using c\_rehash) to verify peer against (SSL)  
  
　　--ciphers <list>  SSL密码  
  
　　--compressed    要求返回是压缩的形势 (using deflate or gzip)  
  
　　--connect-timeout <seconds> 设置最大请求时间  
  
　　--create-dirs   建立本地目录的目录层次结构  
  
　　--crlf          上传是把LF转变成CRLF  
  
　　-f/--fail          连接失败时不显示http错误  
  
　　--ftp-create-dirs 如果远程目录不存在，创建远程目录  
  
　　--ftp-method [multicwd/nocwd/singlecwd] 控制CWD的使用  
  
　　--ftp-pasv      使用 PASV/EPSV 代替端口  
  
　　--ftp-skip-pasv-ip 使用PASV的时候,忽略该IP地址  
  
　　--ftp-ssl       尝试用 SSL/TLS 来进行ftp数据传输  
  
　　--ftp-ssl-reqd  要求用 SSL/TLS 来进行ftp数据传输  
  
　　-F/--form <name=content> 模拟http表单提交数据  
  
　　-form-string <name=string> 模拟http表单提交数据  
  
　　-g/--globoff 禁用网址序列和范围使用{}和[]  
  
　　-G/--get 以get的方式来发送数据  
  
　　-h/--help 帮助  
  
　　-H/--header <line>自定义头信息传递给服务器  
  
　　--ignore-content-length  忽略的HTTP头信息的长度  
  
　　-i/--include 输出时包括protocol头信息  
  
　　-I/--head  只显示文档信息  
  
　　从文件中读取-j/--junk-session-cookies忽略会话Cookie  
  
　　- 界面<interface>指定网络接口/地址使用  
  
　　- krb4 <级别>启用与指定的安全级别krb4  
  
　　-j/--junk-session-cookies 读取文件进忽略session cookie  
  
　　--interface <interface> 使用指定网络接口/地址  
  
　　--krb4 <level>  使用指定安全级别的krb4  
  
　　-k/--insecure 允许不使用证书到SSL站点  
  
　　-K/--config  指定的配置文件读取  
  
　　-l/--list-only 列出ftp目录下的文件名称  
  
　　--limit-rate <rate> 设置传输速度  
  
　　--local-port<NUM> 强制使用本地端口号  
  
　　-m/--max-time <seconds> 设置最大传输时间  
  
　　--max-redirs <num> 设置最大读取的目录数  
  
　　--max-filesize <bytes> 设置最大下载的文件总量  
  
　　-M/--manual  显示全手动  
  
　　-n/--netrc 从netrc文件中读取用户名和密码  
  
　　--netrc-optional 使用 .netrc 或者 URL来覆盖-n  
  
　　--ntlm          使用 HTTP NTLM 身份验证  
  
　　-N/--no-buffer 禁用缓冲输出  
  
　　-o/--output 把输出写到该文件中  
  
　　-O/--remote-name 把输出写到该文件中，保留远程文件的文件名  
  
　　-p/--proxytunnel   使用HTTP代理  
  
　　--proxy-anyauth 选择任一代理身份验证方法  
  
　　--proxy-basic   在代理上使用基本身份验证  
  
　　--proxy-digest  在代理上使用数字身份验证  
  
　　--proxy-ntlm    在代理上使用ntlm身份验证  
  
　　-P/--ftp-port <address> 使用端口地址，而不是使用PASV  
  
　　-Q/--quote <cmd>文件传输前，发送命令到服务器  
  
　　-r/--range <range>检索来自HTTP/1.1或FTP服务器字节范围  
  
　　--range-file 读取（SSL）的随机文件  
  
　　-R/--remote-time   在本地生成文件时，保留远程文件时间  
  
　　--retry <num>   传输出现问题时，重试的次数  
  
　　--retry-delay <seconds>  传输出现问题时，设置重试间隔时间  
  
　　--retry-max-time <seconds> 传输出现问题时，设置最大重试时间  
  
　　-s/--silent静音模式。不输出任何东西  
  
　　-S/--show-error   显示错误  
  
　　--socks4 <host[:port]> 用socks4代理给定主机和端口  
  
　　--socks5 <host[:port]> 用socks5代理给定主机和端口  
  
　　--stderr <file>  
-t/--telnet-option <OPT=val> Telnet选项设置  
  
　　--trace <file>  对指定文件进行debug  
  
　　--trace-ascii <file> Like --跟踪但没有hex输出  
  
　　--trace-time    跟踪/详细输出时，添加时间戳  
  
　　-T/--upload-file <file> 上传文件  
  
　　--url <URL>     Spet URL to work with  
  
　　-u/--user <user[:password]>设置服务器的用户和密码  
  
　　-U/--proxy-user <user[:password]>设置代理用户名和密码  
  
　　-v/--verbose  
  
　　-V/--version 显示版本信息  
  
　　-w/--write-out [format]什么输出完成后  
  
　　-x/--proxy <host[:port]>在给定的端口上使用HTTP代理  
  
　　-X/--request <command>指定什么命令  
  
　　-y/--speed-time 放弃限速所要的时间。默认为30  
  
　　-Y/--speed-limit 停止传输速度的限制，速度时间'秒  
  
　　-z/--time-cond  传送时间设置  
  
　　-0/--http1.0  使用HTTP 1.0  
  
　　-1/--tlsv1  使用TLSv1（SSL）  
  
　　-2/--sslv2 使用SSLv2的（SSL）  
  
　　-3/--sslv3         使用的SSLv3（SSL）  
  
　　--3p-quote      like -Q for the source URL for 3rd party transfer  
  
　　--3p-url        使用url，进行第三方传送  
  
　　--3p-user       使用用户名和密码，进行第三方传送  
  
　　-4/--ipv4   使用IP4  
  
　　-6/--ipv6   使用IP6  
  
　　-#/--progress-bar 用进度条显示当前的传送状态  
  
　　-a/--append 上传文件时，附加到目标文件  
  
　　-A/--user-agent <string>  设置用户代理发送给服务器  
  
　　- anyauth   可以使用“任何”身份验证方法  
  
　　-b/--cookie <name=string/file> cookie字符串或文件读取位置  
  
　　- basic 使用HTTP基本验证  
  
　　-B/--use-ascii 使用ASCII /文本传输  
  
　　-c/--cookie-jar <file> 操作结束后把cookie写入到这个文件中  
  
　　-C/--continue-at <offset>  断点续转  
  
　　-d/--data <data>   HTTP POST方式传送数据  
  
　　--data-ascii <data>  以ascii的方式post数据  
  
　　--data-binary <data> 以二进制的方式post数据  
  
　　--negotiate     使用HTTP身份验证  
  
　　--digest        使用数字身份验证  
  
　　--disable-eprt  禁止使用EPRT或LPRT  
  
　　--disable-epsv  禁止使用EPSV  
  
　　-D/--dump-header <file> 把header信息写入到该文件中  
  
　　--egd-file <file> 为随机数据(SSL)设置EGD socket路径  
  
　　--tcp-nodelay   使用TCP\_NODELAY选项  
  
　　-e/--referer 来源网址  
  
　　-E/--cert <cert[:passwd]> 客户端证书文件和密码 (SSL)  
  
　　--cert-type <type> 证书文件类型 (DER/PEM/ENG) (SSL)  
  
　　--key <key>     私钥文件名 (SSL)  
  
　　--key-type <type> 私钥文件类型 (DER/PEM/ENG) (SSL)  
  
　　--pass  <pass>  私钥密码 (SSL)  
  
　　--engine <eng>  加密引擎使用 (SSL). "--engine list" for list  
  
　　--cacert <file> CA证书 (SSL)  
  
　　--capath <directory> CA目录 (made using c\_rehash) to verify peer against (SSL)  
  
　　--ciphers <list>  SSL密码  
  
　　--compressed    要求返回是压缩的形势 (using deflate or gzip)  
  
　　--connect-timeout <seconds> 设置最大请求时间  
  
　　--create-dirs   建立本地目录的目录层次结构  
  
　　--crlf          上传是把LF转变成CRLF  
  
　　-f/--fail          连接失败时不显示http错误  
  
　　--ftp-create-dirs 如果远程目录不存在，创建远程目录  
  
　　--ftp-method [multicwd/nocwd/singlecwd] 控制CWD的使用  
  
　　--ftp-pasv      使用 PASV/EPSV 代替端口  
  
　　--ftp-skip-pasv-ip 使用PASV的时候,忽略该IP地址  
  
　　--ftp-ssl       尝试用 SSL/TLS 来进行ftp数据传输  
  
　　--ftp-ssl-reqd  要求用 SSL/TLS 来进行ftp数据传输  
  
　　-F/--form <name=content> 模拟http表单提交数据  
  
　　-form-string <name=string> 模拟http表单提交数据  
  
　　-g/--globoff 禁用网址序列和范围使用{}和[]  
  
　　-G/--get 以get的方式来发送数据  
  
　　-h/--help 帮助  
  
　　-H/--header <line>自定义头信息传递给服务器  
  
　　--ignore-content-length  忽略的HTTP头信息的长度  
  
　　-i/--include 输出时包括protocol头信息  
  
　　-I/--head  只显示文档信息  
  
　　从文件中读取-j/--junk-session-cookies忽略会话Cookie  
  
　　- 界面<interface>指定网络接口/地址使用  
  
　　- krb4 <级别>启用与指定的安全级别krb4  
  
　　-j/--junk-session-cookies 读取文件进忽略session cookie  
  
　　--interface <interface> 使用指定网络接口/地址  
  
　　--krb4 <level>  使用指定安全级别的krb4  
  
　　-k/--insecure 允许不使用证书到SSL站点  
  
　　-K/--config  指定的配置文件读取  
  
　　-l/--list-only 列出ftp目录下的文件名称  
  
　　--limit-rate <rate> 设置传输速度  
  
　　--local-port<NUM> 强制使用本地端口号  
  
　　-m/--max-time <seconds> 设置最大传输时间  
  
　　--max-redirs <num> 设置最大读取的目录数  
  
　　--max-filesize <bytes> 设置最大下载的文件总量  
-M/--manual  显示全手动  
  
　　-n/--netrc 从netrc文件中读取用户名和密码  
  
　　--netrc-optional 使用 .netrc 或者 URL来覆盖-n  
  
　　--ntlm          使用 HTTP NTLM 身份验证  
  
　　-N/--no-buffer 禁用缓冲输出  
  
　　-o/--output 把输出写到该文件中  
  
　　-O/--remote-name 把输出写到该文件中，保留远程文件的文件名  
  
　　-p/--proxytunnel   使用HTTP代理  
  
　　--proxy-anyauth 选择任一代理身份验证方法  
  
　　--proxy-basic   在代理上使用基本身份验证  
  
　　--proxy-digest  在代理上使用数字身份验证  
  
　　--proxy-ntlm    在代理上使用ntlm身份验证  
  
　　-P/--ftp-port <address> 使用端口地址，而不是使用PASV  
  
　　-Q/--quote <cmd>文件传输前，发送命令到服务器  
  
　　-r/--range <range>检索来自HTTP/1.1或FTP服务器字节范围  
  
　　--range-file 读取（SSL）的随机文件  
  
　　-R/--remote-time   在本地生成文件时，保留远程文件时间  
  
　　--retry <num>   传输出现问题时，重试的次数  
  
　　--retry-delay <seconds>  传输出现问题时，设置重试间隔时间  
  
　　--retry-max-time <seconds> 传输出现问题时，设置最大重试时间  
  
　　-s/--silent静音模式。不输出任何东西  
  
　　-S/--show-error   显示错误  
  
　　--socks4 <host[:port]> 用socks4代理给定主机和端口  
  
　　--socks5 <host[:port]> 用socks5代理给定主机和端口  
  
　　--stderr <file>  
  
　　-t/--telnet-option <OPT=val> Telnet选项设置  
  
　　--trace <file>  对指定文件进行debug  
  
　　--trace-ascii <file> Like --跟踪但没有hex输出  
  
　　--trace-time    跟踪/详细输出时，添加时间戳  
  
　　-T/--upload-file <file> 上传文件  
  
　　--url <URL>     Spet URL to work with  
  
　　-u/--user <user[:password]>设置服务器的用户和密码  
  
　　-U/--proxy-user <user[:password]>设置代理用户名和密码  
  
　　-v/--verbose  
  
　　-V/--version 显示版本信息  
  
　　-w/--write-out [format]什么输出完成后  
  
　　-x/--proxy <host[:port]>在给定的端口上使用HTTP代理  
  
　　-X/--request <command>指定什么命令  
  
　　-y/--speed-time 放弃限速所要的时间。默认为30  
  
　　-Y/--speed-limit 停止传输速度的限制，速度时间'秒  
  
　　-z/--time-cond  传送时间设置  
  
　　-0/--http1.0  使用HTTP 1.0  
  
　　-1/--tlsv1  使用TLSv1（SSL）  
  
　　-2/--sslv2 使用SSLv2的（SSL）  
  
　　-3/--sslv3         使用的SSLv3（SSL）  
  
　　--3p-quote      like -Q for the source URL for 3rd party transfer  
  
　　--3p-url        使用url，进行第三方传送  
  
　　--3p-user       使用用户名和密码，进行第三方传送  
  
　　-4/--ipv4   使用IP4  
  
　　-6/--ipv6   使用IP6  
  
　　-#/--progress-bar 用进度条显示当前的传送状态  
  
　　二，常用curl实例  
  
　　1，抓取页面内容到一个文件中  
  
　　[root@krlcgcms01 mytest]# curl -o home.html  http://blog.51yip.com  
  
　　[root@krlcgcms01 mytest]# curl -o home.html  http://blog.51yip.com  
  
　　2，用-O（大写的），后面的url要具体到某个文件，不然抓不下来。我们还可以用正则来抓取东西  
  
　　[root@krlcgcms01 mytest]# curl -O  
  
　　[root@krlcgcms01 mytest]# curl -O  
  
　　3，模拟表单信息，模拟登录，保存cookie信息  
  
　　[root@krlcgcms01 mytest]# curl -c ./cookie\_c.txt -F log=aaaa -F pwd=\*\*\*\*\*\* http://blog.51yip.com/wp-login.[**PHP**](http://lib.csdn.net/base/php)  
  
　　[root@krlcgcms01 mytest]# curl -c ./cookie\_c.txt -F log=aaaa -F pwd=\*\*\*\*\*\* http://blog.51yip.com/wp-login.[**php**](http://lib.csdn.net/base/php)  
  
　　4，模拟表单信息，模拟登录，保存头信息  
  
　　[root@krlcgcms01 mytest]# curl -D ./cookie\_D.txt -F log=aaaa -F pwd=\*\*\*\*\*\* http://blog.51yip.com/wp-login.php  
  
　　[root@krlcgcms01 mytest]# curl -D ./cookie\_D.txt -F log=aaaa -F pwd=\*\*\*\*\*\* http://blog.51yip.com/wp-login.php  
  
　　-c(小写)产生的cookie和-D里面的cookie是不一样的。  
  
　　5，使用cookie文件  
  
　　[root@krlcgcms01 mytest]# curl -b ./cookie\_c.txt  http://blog.51yip.com/wp-admin  
  
　　[root@krlcgcms01 mytest]# curl -b ./cookie\_c.txt  http://blog.51yip.com/wp-admin  
  
　　6，断点续传，-C(大写的)  
  
　　[root@krlcgcms01 mytest]# curl -C -O  
  
　　7，传送数据,最好用登录页面[**测试**](http://lib.csdn.net/base/softwaretest)，因为你传值过去后，curl回抓数据，你可以看到你传值有没有成功  
  
　　[root@krlcgcms01 mytest]# curl -d log=aaaa  http://blog.51yip.com/wp-login.php  
  
　　[root@krlcgcms01 mytest]# curl -d log=aaaa  http://blog.51yip.com/wp-login.php  
  
　　8，显示抓取错误，下面这个例子，很清楚的表明了。  
  
　　[root@krlcgcms01 mytest]# curl -f http://blog.51yip.com/asdf  
  
　　curl: (22) The requested URL returned error: 404  
  
　　[root@krlcgcms01 mytest]# curl http://blog.51yip.com/asdf  
  
　　<HTML><HEAD><TITLE>404,not found</TITLE>  
  
　　。。。。。。。。。。。。  
  
　　[root@krlcgcms01 mytest]# curl -f http://blog.51yip.com/asdf  
  
　　curl: (22) The requested URL returned error: 404  
  
　　[root@krlcgcms01 mytest]# curl http://blog.51yip.com/asdf  
  
　　<HTML><HEAD><TITLE>404,not found</TITLE>  
  
　　。。。。。。。。。。。。  
  
　　9，伪造来源地址，有的网站会判断，请求来源地址。  
  
　　[root@krlcgcms01 mytest]# curl -e http://localhost http://blog.51yip.com/wp-login.php  
  
　　[root@krlcgcms01 mytest]# curl -e http://localhost http://blog.51yip.com/wp-login.php  
  
　　10，当我们经常用curl去搞人家东西的时候，人家会把你的IP给屏蔽掉的,这个时候,我们可以用代理  
  
　　[root@krlcgcms01 mytest]# curl -x 24.10.28.84:32779 -o home.html http://blog.51yip.com  
  
　　[root@krlcgcms01 mytest]# curl -x 24.10.28.84:32779 -o home.html http://blog.51yip.com  
11，比较大的东西，我们可以分段下载  
  
　　[root@krlcgcms01 mytest]# curl -r 0-100 -o img.part1 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100   101  100   101    0     0    105      0 --:--:-- --:--:-- --:--:--     0  
  
　　[root@krlcgcms01 mytest]# curl -r 100-200 -o img.part2 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100   101  100   101    0     0     57      0  0:00:01  0:00:01 --:--:--     0  
  
　　[root@krlcgcms01 mytest]# curl -r 200- -o img.part3 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100  104k  100  104k    0     0  52793      0  0:00:02  0:00:02 --:--:-- 88961  
  
　　[root@krlcgcms01 mytest]# ls |grep part | xargs du -sh  
  
　　4.0K    one.part1  
  
　　112K    three.part3  
  
　　4.0K    two.part2  
  
　　[root@krlcgcms01 mytest]# curl -r 0-100 -o img.part1 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100   101  100   101    0     0    105      0 --:--:-- --:--:-- --:--:--     0  
  
　　[root@krlcgcms01 mytest]# curl -r 100-200 -o img.part2 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100   101  100   101    0     0     57      0  0:00:01  0:00:01 --:--:--     0  
  
　　[root@krlcgcms01 mytest]# curl -r 200- -o img.part3 http://blog.51yip.com/wp-  
  
　　content/uploads/2010/09/compare\_varnish.jpg  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　100  104k  100  104k    0     0  52793      0  0:00:02  0:00:02 --:--:-- 88961  
  
　　[root@krlcgcms01 mytest]# ls |grep part | xargs du -sh  
  
　　4.0K    one.part1  
  
　　112K    three.part3  
  
　　4.0K    two.part2  
  
　　用的时候，把他们cat一下就OK了,cat img.part\* >img.jpg  
  
　　12，不会显示下载进度信息  
  
　　[root@krlcgcms01 mytest]# curl -s -o aaa.jpg   
  
　　13，显示下载进度条  
  
　　[root@krlcgcms01 mytest]# curl -# -O   
  
　　######################################################################## 100.0%  
  
　　14,通过ftp下载文件  
  
　　[zhangy@BlackGhost ~]$ curl -u 用户名:密码 -O http://blog.51yip.com/demo/curtain/bbstudy\_files/style.css  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　101  1934  101  1934    0     0   3184      0 --:--:-- --:--:-- --:--:--  7136  
  
　　[zhangy@BlackGhost ~]$ curl -u 用户名:密码 -O http://blog.51yip.com/demo/curtain/bbstudy\_files/style.css  
  
　　% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current  
  
　　Dload  Upload   Total   Spent    Left  Speed  
  
　　101  1934  101  1934    0     0   3184      0 --:--:-- --:--:-- --:--:--  7136  
  
　　或者用下面的方式  
  
　　[zhangy@BlackGhost ~]$ curl -O ftp://用户名:密码@ip:port/demo/curtain/bbstudy\_files/style.css  
  
　　[zhangy@BlackGhost ~]$ curl -O ftp://用户名:密码@ip:port/demo/curtain/bbstudy\_files/style.css  
  
　　15，通过ftp上传  
  
　　[zhangy@BlackGhost ~]$ curl -T test.sql ftp://用户名:密码@ip:port/demo/curtain/bbstudy\_files/  
  
　　[zhangy@BlackGhost ~]$ curl -T test.sql ftp://用户名:密码@ip:port/demo/curtain/bbstudy\_files/  
原文出处：http://[**linux**](http://lib.csdn.net/base/linux).chinaitlab.com/command/830656.html

<http://www.5678520.com/kaiwangdian/130.html>

<http://www.5678520.com/kaiwangdian/129.html>

<http://www.5678520.com/kaiwangdian/128.html>

<http://www.5678520.com/kaiwangdian/127.html>

<http://www.5678520.com/kaiwangdian/126.html>

<http://www.lianzhiwei.com/News/389/20122116.html>

<http://www.lianzhiwei.com/News/389/20122115.html>

<http://www.lianzhiwei.com/News/389/20122114.html>

<http://www.lianzhiwei.com/News/389/20122113.html>

<http://www.lianzhiwei.com/News/389/20122112.html>