Linux必备下载命令之curl详解

字诀跳动

爱生活, 有理想, 善思考, 能沟通

文章目录

curl简介：

curl是一个非常实用的、用来与服务器之间传输数据的工具；支持的协议包括(DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, TELNET and TFTP)，curl设计为无用户交互下完成工作；curl提供了一大堆非常有用的功能，包括代理访问、用户认证、ftp上传下载、HTTP POST、SSL连接、cookie支持、断点续传...。

1、curl命令语法

```
curl [options] [URL...]
```

2、curl命令参数详解

由于linux curl功能十分强大，所以命令参数十分多，下表只筛选选出来的部分参数，更多参数请运行“man curl”命令查看。

3、常用选项分类

```
# 调试类
-v, --verbose                           输出信息
-q, --disable                           在第一个参数位置设置后 .curlrc 的设置直接失效
-K, --config FILE                       指定配置文件
-L, --location                          跟踪重定向 (H)

# CLI显示设置
-s, --silent                            Silent模式。不输出任务内容
-S, --show-error                        显示错误. 在选项 -s 中，当 curl 出现错误时将
-f, --fail                              不显示 连接失败时HTTP错误信息
```

```
-i, --include                          显示 response的header (H/F)
-I, --head                             仅显示 响应文档头
-l, --list-only                        只列出FTP目录的名称 (F)
-#, --progress-bar                     以进度条 显示传输进度

# 数据传输类
-X, --request [GET|POST|PUT|DELETE|…]  使用指定的 http method 例如 -X POST
-H, --header <header>                  设定 request里的header 例如 -H "Content-T
-e, --referer                          设定 referer (H)
-d, --data <data>                      设定 http body 默认使用 content-type appl
    --data-raw <data>                  ASCII 编码 HTTP POST 数据 (H)
    --data-binary <data>               binary 编码 HTTP POST 数据 (H)
    --data-urlencode <data>            url 编码 HTTP POST 数据 (H)
-G, --get                              使用 HTTP GET 方法发送 -d 数据 (H)
-F, --form <name=string>               模拟 HTTP 表单数据提交 multipart POST (H)
    --form-string <name=string>        模拟 HTTP 表单数据提交 (H)
-u, --user <user:password>             使用帐户，密码 例如 admin:password
-b, --cookie <data>                    cookie 文件 (H)
-j, --junk-session-cookies             读取文件中但忽略会话cookie (H)
-A, --user-agent                       user-agent设置 (H)

# 传输设置
-C, --continue-at OFFSET               断点续转
-x, --proxy [PROTOCOL://]HOST[:PORT]   在指定的端口上使用代理
-U, --proxy-user USER[:PASSWORD]       代理用户名及密码

# 文件操作
-T, --upload-file <file>               上传文件
-a, --append                           添加要上传的文件 (F/SFTP)

# 输出设置
-o, --output <file>                    将输出写入文件，而非 stdout
-O, --remote-name                      将输出写入远程文件
-D, --dump-header <file>               将头信息写入指定的文件
-c, --cookie-jar <file>                操作结束后，要写入 Cookies 的文件位置
```

4 常用curl实例

抓取页面内容到一个文件中

```
curl -o home.html  http://www.sina.com.cn
```

> 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

```
curl -O http://www.mydomain.com/linux/index.html
```

模拟用户登录

```
# 此参数相当于设置http头 Authorization:
curl --user user:password http://blog.mydomain.com/login.php
# 使用用户名、密码认证，此参数会覆盖“-n”、“--netrc”和“--netrc-optional”选项
```

模拟表单信息，模拟登录，保存cookie信息

```
curl -c ./cookie_c.txt -F log=aaaa -F pwd=****** http://blog.mydomain.com/logi
```

模拟表单信息，模拟登录，保存头信息

```
curl -D ./cookie_D.txt -F log=aaaa -F pwd=****** http://blog.mydomain.com/logi
```

> -c(小写)`产生的cookie和`-D里面的cookie是不一样的

使用cookie文件

```
curl -b ./cookie_c.txt  http://blog.mydomain.com/wp-admin
```

断点续传，`-C(大写的)`

```
curl -C -O http://www.sina.com.cn
```

传送数据

最好用登录页面测试，因为你传值过去后，回抓数据，你可以看到你传值有没有成功

```
curl -d log=aaaa  http://blog.mydomain.com/login.php
```

显示抓取错误 `-f`

```
curl -f http://www.sina.com.cn/asdf
curl: (22) The requested URL returned error: 404
curl http://www.sina.com.cn/asdf
```

伪造来源地址，有的网站会判断，请求来源地址 `-e`

```
curl -e http://localhost http://www.sina.com.cn
```

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

```
curl -x 10.10.90.83:80 -o home.html http://www.sina.com.cn
```

比较大的东西，我们可以分段下载

```
curl -r 0-100 -o img.part1 http://mydomian.cn/thumb/xxx.jpg
  % Total    % Received % Xferd  Average Speed  Time    Time    Time  Current
                                 Dload  Upload  Total  Spent    Left  Speed
100  101  100  101    0    0  1926      0 --:--:-- --:--:-- --:--:--    0

curl -r 100-200 -o img.part2 http://mydomian.cn/thumb/xxx.jpg
  % Total    % Received % Xferd  Average Speed  Time    Time    Time  Current
                                 Dload  Upload  Total  Spent    Left  Speed
100  101  100  101    0    0  3498      0 --:--:-- --:--:-- --:--:--  98k

curl -r 200- -o img.part3 http://mydomian.cn/thumb/xxx.jpg
  % Total    % Received % Xferd  Average Speed  Time    Time    Time  Current
                                 Dload  Upload  Total  Spent    Left  Speed
100 13515  100 13515    0    0  154k      0 --:--:-- --:--:-- --:--:--  280k
ll |grep img.part
```

用的时候，把他们cat一下就OK了,`cat img.part* >img.jpg`

不显示下载进度信息 `-s`

```
curl -s -o aaa.jpg
```

显示下载进度条 `-#`

```
curl -# -O  http://www.mydomain.com/linux/25002_3.html
######################################################################## 100.0
```

通过ftp下载文件

```
curl -u 用户名:密码 -O http://blog.mydomain.com/demo/curtain/bbstudy_files/style
% 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
curl -O ftp://xukai:test@192.168.242.144:21/www/focus/enhouse/index.php
  % Total    % Received % Xferd  Average Speed  Time    Time    Time  Current
                                 Dload  Upload  Total  Spent    Left  Speed
100 87518  100 87518    0    0  2312k      0 --:--:-- --:--:-- --:--:-- 11.5M
```

通过ftp上传

```
curl -T xukai.php ftp://xukai:test@192.168.242.144:21/www/focus/enhouse/
  % Total    % Received % Xferd  Average Speed  Time    Time    Time  Current
                                 Dload  Upload  Total  Spent    Left  Speed
100 87518    0    0  100 87518      0  2040k --:--:-- --:--:-- --:--:-- 8901k
```

用法收集

快速用法（配合sed/awk/grep）

```
$curl http: //mydomain.net
```

下载保存

```
$curl http://mydomain.net > index.html
$curl -o index.html http://mydomain.net
$curl -O http://mydomain.net/target.tar.gz
```

GET

```
$curl http://www.yahoo.com/login.cgi?user=nickname&password=12345
```

POST

```
$curl -d "user=nickname&password=12345" http://www.yahoo.com/login.cgi
```

POST 文件

```
$curl -F upload= $localfile  -F $btn_name=$btn_value http://mydomain.net/~zzh/u
```

通过代理

```
$curl -x 123.45.67.89:1080 -o page.html http://mydomain.net
```

保存cookie

```
$curl -x 123.45.67.89:1080 -o page1.html -D cookie0001.txt http://mydomain.net
```

使用cookie

```
$curl -x 123.45.67.89:1080 -o page1.html -D cookie0002.txt -b cookie0001.txt ht
```

模仿浏览器

```
$curl -A "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)" -x123.45.67.89:10
```

伪造referer

```
$curl -A "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0)" -x123.45.67.89:10
```

高级下载功能

循环下载

```
$curl -O http://mydomain.net/~zzh/screen[1-10].JPG
```

循环（匹配）下载

```
$curl -O http://mydomain.net/~{zzh,nick}/[001-201].JPG  # >like zzh/001.JPG
```

循环（引用）下载

```
$curl -o #2_#1.jpg http://mydomain.net/~{zzh,nick}/[001-201].JPG # like >001_zz
```

断点续传

```
$curl -c -O http://mydomain.net/~zzh/screen1.JPG
```

分块下载

```
$curl -r  0 -10240  -o "zhao.part1"  http://mydomain.net/~zzh/zhao1.mp3 &\
$curl -r 10241 -20480  -o "zhao.part1"  http://mydomain.net/~zzh/zhao1.mp3 &\
$curl -r 20481 -40960  -o "zhao.part1"  http://mydomain.net/~zzh/zhao1.mp3 &\
$curl -r 40961 - -o  "zhao.part1"  http://mydomain.net/~zzh/zhao1.mp3
...
$cat zhao.part* > zhao.mp3
```

5、curl语法及选项整理

curl（7.29.0）所支持的选项（options）参数如下

```
在以下选项中，(H) 表示仅适用 HTTP/HTTPS ，(F) 表示仅适用于 FTP
    --anyauth       选择 "any" 认证方法 (H)
-a, --append        添加要上传的文件 (F/SFTP)
    --basic         使用HTTP基础认证（Basic Authentication）(H)
    --cacert FILE  CA 证书，用于每次请求认证 (SSL)
    --capath DIR    CA 证书目录 (SSL)
-E, --cert CERT[:PASSWD] 客户端证书文件及密码 (SSL)
    --cert-type TYPE 证书文件类型 (DER/PEM/ENG) (SSL)
    --ciphers LIST  SSL 秘钥 (SSL)
    --compressed    请求压缩 (使用 deflate 或 gzip)
-K, --config FILE  指定配置文件
    --connect-timeout SECONDS  连接超时设置
-C, --continue-at OFFSET  断点续转
-b, --cookie STRING/FILE  Cookies字符串或读取Cookies的文件位置 (H)
-c, --cookie-jar FILE  操作结束后，要写入 Cookies 的文件位置 (H)
    --create-dirs  创建必要的本地目录层次结构
    --crlf          在上传时将 LF 转写为 CRLF
    --crlfile FILE  从指定的文件获得PEM格式CRL列表
-d, --data DATA    HTTP POST 数据 (H)
    --data-ascii DATA  ASCII 编码 HTTP POST 数据 (H)
    --data-binary DATA  binary 编码 HTTP POST 数据 (H)
    --data-urlencode DATA  url 编码 HTTP POST 数据 (H)
    --delegation STRING GSS-API 委托权限
    --digest        使用数字身份验证 (H)
    --disable-eprt  禁止使用 EPRT 或 LPRT (F)
    --disable-epsv  禁止使用 EPSV (F)
-D, --dump-header FILE  将头信息写入指定的文件
    --egd-file FILE  为随机数据设置EGD socket路径(SSL)
    --engine ENGINGE  加密引擎 (SSL). "--engine list" 指定列表
-f, --fail          连接失败时不显示HTTP错误信息 (H)
-F, --form CONTENT  模拟 HTTP 表单数据提交（multipart POST） (H)
```

```
    --form-string STRING  模拟 HTTP 表单数据提交 (H)
    --ftp-account DATA  帐户数据提交 (F)
    --ftp-alternative-to-user COMMAND  指定替换 "USER [name]" 的字符串 (F)
    --ftp-create-dirs  如果不存在则创建远程目录 (F)
    --ftp-method [MULTICWD/NOCWD/SINGLECWD] 控制 CWD (F)
    --ftp-pasv      使用 PASV/EPSV 替换 PORT (F)
-P, --ftp-port ADR  使用指定 PORT 及地址替换 PASV (F)
    --ftp-skip-pasv-ip 跳过 PASV 的IP地址 (F)
    --ftp-pret      在 PASV 之前发送 PRET (drftpd) (F)
    --ftp-ssl-ccc  在认证之后发送 CCC (F)
    --ftp-ssl-ccc-mode ACTIVE/PASSIVE  设置 CCC 模式 (F)
    --ftp-ssl-control ftp 登录时需要 SSL/TLS (F)
-G, --get           使用 HTTP GET 方法发送 -d 数据  (H)
-g, --globoff       禁用的 URL 队列 及范围使用 {} 和 []
-H, --header LINE  要发送到服务端的自定义请求头 (H)
-I, --head           仅显示响应文档头
-h, --help           显示帮助
-0, --http1.0       使用 HTTP 1.0 (H)
    --ignore-content-length  忽略 HTTP Content-Length 头
-i, --include       在输出中包含协议头 (H/F)
-k, --insecure       允许连接到 SSL 站点，而不使用证书 (H)
    --interface INTERFACE  指定网络接口／地址
-4, --ipv4           将域名解析为 IPv4 地址
-6, --ipv6           将域名解析为 IPv6 地址
-j, --junk-session-cookies 读取文件中但忽略会话cookie (H)
    --keepalive-time SECONDS  keepalive 包间隔
    --key KEY       私钥文件名 (SSL/SSH)
    --key-type TYPE 私钥文件类型 (DER/PEM/ENG) (SSL)
    --krb LEVEL     启用指定安全级别的 Kerberos (F)
    --libcurl FILE  命令的libcurl等价代码
    --limit-rate RATE  限制传输速度
-l, --list-only     只列出FTP目录的名称 (F)
    --local-port RANGE  强制使用的本地端口号
-L, --location       跟踪重定向 (H)
    --location-trusted 类似 --location 并发送验证信息到其它主机 (H)
-M, --manual         显示全手动
    --mail-from FROM  从这个地址发送邮件
    --mail-rcpt TO  发送邮件到这个接收人(s)
    --mail-auth AUTH  原始电子邮件的起始地址
    --max-filesize BYTES  下载的最大文件大小 (H/F)
    --max-redirs NUM  最大重定向数 (H)
-m, --max-time SECONDS  允许的最多传输时间
    --metalink       处理指定的URL上的XML文件
    --negotiate     使用 HTTP Negotiate 认证 (H)
-n, --netrc         必须从 .netrc 文件读取用户名和密码
    --netrc-optional 使用 .netrc 或 URL; 将重写 -n 参数
    --netrc-file FILE  设置要使用的 netrc 文件名
-N, --no-buffer     禁用输出流的缓存
    --no-keepalive  禁用 connection 的 keepalive
    --no-sessionid  禁止重复使用 SSL session-ID (SSL)
    --noproxy       不使用代理的主机列表
    --ntlm           使用 HTTP NTLM 认证 (H)
-o, --output FILE  将输出写入文件，而非 stdout
    --pass PASS     传递给私钥的短语 (SSL/SSH)
    --post301       在 301 重定向后不要切换为 GET 请求 (H)
    --post302       在 302 重定向后不要切换为 GET 请求 (H)
```

```
    --post303       在 303 重定向后不要切换为 GET 请求 (H)
-#, --progress-bar  以进度条显示传输进度
    --proto PROTOCOLS  启用/禁用 指定的协议
    --proto-redir PROTOCOLS  在重定向上 启用/禁用 指定的协议
-x, --proxy [PROTOCOL://]HOST[:PORT] 在指定的端口上使用代理
    --proxy-anyauth 在代理上使用 "any" 认证方法 (H)
    --proxy-basic  在代理上使用 Basic 认证  (H)
    --proxy-digest  在代理上使用 Digest 认证 (H)
    --proxy-negotiate 在代理上使用 Negotiate 认证 (H)
    --proxy-ntlm    在代理上使用 NTLM 认证 (H)
-U, --proxy-user USER[:PASSWORD]  代理用户名及密码
    --proxy1.0 HOST[:PORT]  在指定的端口上使用 HTTP/1.0 代理
-p, --proxytunnel  使用HTTP代理 (用于 CONNECT)
    --pubkey KEY    公钥文件名 (SSH)
-Q, --quote CMD    在传输开始前向服务器发送命令 (F/SFTP)
    --random-file FILE  读取随机数据的文件 (SSL)
-r, --range RANGE  仅检索范围内的字节
    --raw          使用原始HTTP传输，而不使用编码 (H)
-e, --referer      Referer URL (H)
-J, --remote-header-name 从远程文件读取头信息 (H)
-O, --remote-name  将输出写入远程文件
    --remote-name-all 使用所有URL的远程文件名
-R, --remote-time  将远程文件的时间设置在本地输出上
-X, --request COMMAND  使用指定的请求命令
    --resolve HOST:PORT:ADDRESS  将 HOST:PORT 强制解析到 ADDRESS
    --retry NUM  出现问题时的重试次数
    --retry-delay SECONDS 重试时的延时时长
    --retry-max-time SECONDS  仅在指定时间段内重试
-S, --show-error    显示错误. 在选项 -s 中，当 curl 出现错误时将显示
-s, --silent        Silent模式。不输出任务内容
    --socks4 HOST[:PORT]  在指定的 host + port 上使用 SOCKS4 代理
    --socks4a HOST[:PORT]  在指定的 host + port 上使用 SOCKSa 代理
    --socks5 HOST[:PORT]  在指定的 host + port 上使用 SOCKS5 代理
    --socks5-hostname HOST[:PORT] SOCKS5 代理，指定用户名、密码
    --socks5-gssapi-service NAME  为gssapi使用SOCKS5代理服务名称
    --socks5-gssapi-nec  与NEC Socks5服务器兼容
-Y, --speed-limit RATE  在指定限速时间之后停止传输
-y, --speed-time SECONDS  指定时间之后触发限速. 默认 30
    --ssl          尝试 SSL/TLS (FTP, IMAP, POP3, SMTP)
    --ssl-reqd      需要 SSL/TLS (FTP, IMAP, POP3, SMTP)
-2, --sslv2        使用 SSLv2 (SSL)
-3, --sslv3        使用 SSLv3 (SSL)
    --ssl-allow-beast 允许的安全漏洞，提高互操作性(SSL)
    --stderr FILE  重定向 stderr 的文件位置. - means stdout
    --tcp-nodelay  使用 TCP_NODELAY 选项
-t, --telnet-option OPT=VAL  设置 telnet 选项
    --tftp-blksize VALUE  设备 TFTP BLKSIZE 选项 (必须 >512)
-z, --time-cond TIME  基于时间条件的传输
-1, --tlsv1        使用 => TLSv1 (SSL)
    --tlsv1.0      使用 TLSv1.0 (SSL)
    --tlsv1.1      使用 TLSv1.1 (SSL)
    --tlsv1.2      使用 TLSv1.2 (SSL)
    --trace FILE    将 debug 信息写入指定的文件
    --trace-ascii FILE  类似 --trace 但使用16进度输出
    --trace-time    向 trace/verbose 输出添加时间戳
    --tr-encoding  请求压缩传输编码 (H)
```

```
-T, --upload-file FILE  将文件传输（上传）到指定位置
    --url URL       指定所使用的 URL
-B, --use-ascii     使用 ASCII/text 传输
-u, --user USER[:PASSWORD]  指定服务器认证用户名、密码
    --tlsuser USER  TLS 用户名
    --tlspassword STRING TLS 密码
    --tlsauthtype STRING  TLS 认证类型 (默认 SRP)
    --unix-socket FILE    通过这个 UNIX socket 域连接
-A, --user-agent STRING  要发送到服务器的 User-Agent (H)
-v, --verbose       显示详细操作信息
-V, --version       显示版本号并退出
-w, --write-out FORMAT  完成后输出什么
    --xattr         将元数据存储在扩展文件属性中
-q                 .curlrc 如果作为第一个参数无效
```

编辑于 2020-12-12 20:54

cURL HTTP 下载工具

写下你的评论...

还没有评论，发表第一个评论吧

文章被以下专栏收录

linux云计算云原生自动化运维
linux云原生高可用自动化运维,k8s,CICD

推荐阅读

linux之curl命令

3分钟短文 | Linux 使用curl发

使用 curl 从命令行访问互联网 | Linux 中国

Linux... 发表于Linux

curl命令 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curl为下载工具。作为一款强力工具，curl支持包括HTTP、HTTPS、f…

入门小站 发表于入门小站

起post请求的4个常用方式

引言cURL是一种命令行实用程序，用于使用一种受支持的协议，从远程服务器传输数据，或将数据传输到远程服务器。默认情况下，已安装在macOS和大多数Linux发行版上。 开发人员可以使用cURL来…

程序员小助... 发表于程序员小助...