**Câu 2 ( cURL )** : Học viên dùng cURL để lấy token của DNA Center.

Thông tin server DNA Center bao gồm

Thông tin tài khoản là devnetuser/Cisco123! .

**Lưu ý** : username và password được mã hóa base64.

**Kết quả :**

Text

Description automatically generated

**Đáp án :**

**A picture containing text, screenshot, computer

Description automatically generated**

**[root@devops ~]#** curl --location --request POST 'https://sandboxdnac.cisco.com/dna/system/api/v1/auth/token' \

--header 'Authorization: Basic ZGV2bmV0dXNlcjpDaXNjbzEyMyE='

{"Token":"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiI2MGVjNGU0ZjRjYTdmOTIyMmM4MmRhNjYiLCJhdXRoU291cmNlIjoiaW50ZXJuYWwiLCJ0ZW5hbnROYW1lIjoiVE5UMCIsInJvbGVzIjpbIjVlOGU4OTZlNGQ0YWRkMDBjYTJiNjQ4ZSJdLCJ0ZW5hbnRJZCI6IjVlOGU4OTZlNGQ0YWRkMDBjYTJiNjQ4NyIsImV4cCI6MTYyNzcwNDU0NywiaWF0IjoxNjI3NzAwOTQ3LCJqdGkiOiI2OTNiYjJiYS00YmRhLTQyNjItYmVlNi1jMGQ4NDA5MGFmMjQiLCJ1c2VybmFtZSI6ImRldm5ldHVzZXIifQ.lrjVcHCj2kjbfy4jcnjoi6P9nYXv70Yp3YAatpZQyVk7kDr1UV8RpaJ-ai62OS3novo-jvhJc37k4jIs5L0aDobzTVaW2aZEF-bXkuK3KfS-5X6nefLcw-TrM\_b5atExN4bAmGd67xZ96dTmnw09tL1ftVmp0FtzVK3dhwnTKnvb6GycGEWF0PeuWpo1IpHvHP75pMXIIOp3Z2okZ7jAC\_utRoQJHMbe9IAqGotaXU9Pp5khe6Vs5EOhnsEOCUVrsjx76Nn1S1bcL2MenJaqofmyovQZBTpIQygNr0e9JFPza6sXryFqcSKMQM-sAEzo8-Vqk6I7fcD3uvh4og\_BcA"}

**[root@devops ~]#**

**-L, --location**

(HTTP/HTTPS) If the server reports that the requested page has moved to a different location (indicated with a Location: header and a 3XX response code), this option will make curl redo the request on the new place. If used together with -i, --include or -I, --head, headers from all requested pages will be shown. When authentication is used, curl only sends its credentials to the initial host. If a redirect takes curl to a different host, it won't be able to intercept the user+password. See also --location-trusted on how to change this. You can limit the amount of redirects to follow by using the --max-redirs option.

When curl follows a redirect and the request is not a plain GET (for example POST or PUT), it will do the following request with a GET if the HTTP response was 301, 302, or 303. If the response code was any other 3xx code, curl will re-send the following request using the same unmodified method.

[root@devops ~]# curl google.com

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">

<TITLE>301 Moved</TITLE></HEAD><BODY>

<H1>301 Moved</H1>

The document has moved

<A HREF="http://www.google.com/">here</A>.

</BODY></HTML>

[root@devops ~]# curl --head google.com

HTTP/1.1 301 Moved Permanently

Location: http://www.google.com/

Content-Type: text/html; charset=UTF-8

Date: Fri, 30 Jul 2021 10:14:19 GMT

Expires: Sun, 29 Aug 2021 10:14:19 GMT

Cache-Control: public, max-age=2592000

Server: gws

Content-Length: 219

X-XSS-Protection: 0

X-Frame-Options: SAMEORIGIN

[root@devops ~]#

**//** Khi thực hiện truy vấn tài nguyên của một máy chủ, sẽ có 1 số địa chỉ bị chuyển hướng bởi máy chủ còn được gọi là redirect. Nếu chúng ta thực hiện truy vấn trên trình duyệt thì việc này là tự động và trong suốt với người dùng. Nhưng với cURL thì chúng ta sẽ nhận được HTTP respone HTTP/1.1 301 Moved Permanently, để tự động chuyển hướng chúng ta thêm tùy chọn -L or –location.

**-X, --request <command>**

(HTTP) Specifies a custom request method to use when communicating with the HTTP server. The specified request will be used instead of the method otherwise used (which defaults to GET). Read the HTTP 1.1 specification for details and explanations. Common additional HTTP requests include PUT and DELETE, but related technologies like WebDAV offers PROPFIND, COPY, MOVE and more.

Normally you don't need this option. All sorts of GET, HEAD, POST and PUT requests are rather invoked by using dedicated command line options.

This option only changes the actual word used in the HTTP request, it does not alter the way curl behaves. So for example if you want to make a proper HEAD request, using -X HEAD will not suffice. You need to use the -I, --head option.

(FTP) Specifies a custom FTP command to use instead of LIST when doing file lists with FTP.

**//** Nếu không có tùy chọn này thì cURL mặc định sử dụng phương thức GET. Bởi vậy, chỉ định tùy chọn –request hoặc -X để sử dụng phương thức khác, ví dụ như POST, PUT, DELETE.

**-H, --header <header>**

(HTTP) Extra header to use when getting a web page. You may specify any number of extra headers. Note that if you should add a custom header that has thesame name as one of the internal ones curl would use, your externally set header will be used instead of the internal one. This allows you to make eventrickier stuff than curl would normally do. You should not replace internally set headers without knowing perfectly well what you're doing. Remove an internal header by giving a replacement without content on the right side of the colon, as in: -H "Host:". If you send the custom header with no-value then itsheader must be terminated with a semicolon, such as -H "X-Custom-Header;" to send "X-Custom-Header:".

curl will make sure that each header you add/replace is sent with the proper end-of-line marker, you should thus not add that as a part of the header content: do not add newlines or carriage returns, they will only mess things up for you.

See also the -A, --user-agent and -e, --referer options.

This option can be used multiple times to add/replace/remove multiple headers.

**//** Thêm header vào truy vấn, ví dụ chúng ta muốn 1 truy vấn nhưng cần xác thực user/password, thì tại tùy chọn -H hoặc –header sẽ cho chúng ta chèn vào thông tin này để thực hiện xác thực.

// Lưu ý: Các trường trong header chúng ta thêm vào phải được hỗ trợ bởi máy chủ.

// If you send the custom header with no-value then itsheader must be terminated with a semicolon, such as -H "X-Custom-Header;" to send "X-Custom-Header:". Câu này có nghĩa là nếu bạn muốn chèn vào 1 key mà có value rỗng thì chúng ta phải thêm cuối key đó dấu ;

**-G, --get**

When used, this option will make all data specified with -d, --data or --data-binary to be used in an HTTP GET request instead of the POST request that otherwise would be used. The data will be appended to the URL with a '?' separator.

If used in combination with -I, the POST data will instead be appended to the URL with a HEAD request.

If this option is used several times, only the first one is used. This is because undoing a GET doesn't make sense, but you should then instead enforce the alternative method you prefer.

**-u, --user <user:password>**

Specify the user name and password to use for server authentication. Overrides -n, --netrc and --netrc-optional.

If you just give the user name (without entering a colon) curl will prompt for a password.

If you use an SSPI-enabled curl binary and do NTLM authentication, you can force curl to pick up the user name and password from your environment by simply specifying a single colon with this option: "-u :".

If this option is used several times, the last one will be used.

**--url <URL>**

Specify a URL to fetch. This option is mostly handy when you want to specify URL(s) in a config file.

This option may be used any number of times. To control where this URL is written, use the -o, --output or the -O, --remote-name options.

**-d, --data <data>**

(HTTP) Sends the specified data in a POST request to the HTTP server, in the same way that a browser does when a user has filled in an HTML form and pressesthe submit button. This will cause curl to pass the data to the server using the content-type application/x-www-form-urlencoded. Compare to -F, --form.

**-d, --data**

is the same as --data-ascii. To post data purely binary, you should instead use the --data-binary option. To URL-encode the value of a form field you may use --data-urlencode.

If any of these options is used more than once on the same command line, the data pieces specified will be merged together with a separating &-symbol. Thus, using '-d name=daniel -d skill=lousy' would generate a post chunk that looks like 'name=daniel&skill=lousy'.

If you start the data with the letter @, the rest should be a file name to read the data from, or - if you want curl to read the data from stdin. The contents of the file must already be URL-encoded. Multiple files can also be specified. Posting data from a file named 'foobar' would thus be done with --data @foobar.

**-k, --insecure**

(SSL) This option explicitly allows curl to perform "insecure" SSL connections and transfers. All SSL connections are attempted to be made secure by using the CA certificate bundle installed by default. This makes all connections considered "insecure" fail unless -k, --insecure is used.

See this online resource for further details: <http://curl.haxx.se/docs/sslcerts.html>

NAME

curl - transfer a URL

SYNOPSIS

curl [options] [URL...]

DESCRIPTION

curl is a tool to transfer data from or to a server, using one of the supported protocols (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS,

POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMTP, SMTPS, TELNET and TFTP). The command is designed to work without user interaction.

curl offers a busload of useful tricks like proxy support, user authentication, FTP upload, HTTP post, SSL connections, cookies, file transfer resume, Metalink,

and more. As you will see below, the number of features will make your head spin!

curl is powered by libcurl for all transfer-related features. See libcurl(3) for details.

**HTTP GET request**

The first example is the most basic example which demonstrates a simple curl command that simulates a GET request for a website URL. This command will output the HTTP response of the URL in question.

**[root@devops ~]#** curl <https://dantri.com.vn>

**-I, --head**

(HTTP/FTP/FILE) Fetch the HTTP-header only! HTTP-servers feature the command HEAD which this uses to get nothing but the header of a document. When used on an FTP or FILE file, curl displays the file size and last modification time only.

// Truy vấn chỉ trả về thông tin của HTTP header

**[root@devops ~]#** curl **-I** https://dantri.com.vn

HTTP/1.1 200 OK

Server: nginx

Date: Fri, 30 Jul 2021 15:56:51 GMT

Content-Type: text/html; charset=utf-8

Connection: keep-alive

Vary: Accept-Encoding

Access-Control-Allow-Credentials: true

Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept,csrf-token

Access-Control-Allow-Methods: GET, POST, PUT, DELETE, OPTIONS

Access-Control-Allow-Origin: \*

asp-id: 5818

s-count: 225

r-count: 0

Age: 1

sv: 70

Strict-Transport-Security: max-age=31536000; includeSubDomains

iph: 0

DT-Id: dt-531465100 530981059

DT-CK: 15

Accept-Ranges: bytes

**[root@devops ~]#**

**-i, --include**

(HTTP) Include the HTTP-header in the output. The HTTP-header includes things like server-name, date of the document, HTTP-version and more...

**//** Đầu ra kết quả bao gồm có HTTP Respone Header

**[root@devops ~]#** curl http://google.com

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">

<TITLE>301 Moved</TITLE></HEAD><BODY>

<H1>301 Moved</H1>

The document has moved

<A HREF="http://www.google.com/">here</A>.

</BODY></HTML>

**[root@devops ~]#** curl --include http://google.com

HTTP/1.1 301 Moved Permanently

Location: http://www.google.com/

Content-Type: text/html; charset=UTF-8

Date: Fri, 30 Jul 2021 14:30:25 GMT

Expires: Sun, 29 Aug 2021 14:30:25 GMT

Cache-Control: public, max-age=2592000

Server: gws

Content-Length: 219

X-XSS-Protection: 0

X-Frame-Options: SAMEORIGIN

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">

<TITLE>301 Moved</TITLE></HEAD><BODY>

<H1>301 Moved</H1>

The document has moved

<A HREF="http://www.google.com/">here</A>.

</BODY></HTML>

**Saving the result of a curl command**

The -o and -O curl options are used to save the result of the curl command. The difference between both options is that -o will save the file with a predefined filename, which in this case is myfile.css. On the other hand, the -O option will save the file as its exisiting name, which is animate.min.css. An example of each scenario is shown below.

**[root@devops ~]#** curl **-O** http://speedtest.tele2.net/10MB.zip

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

100 10.0M 100 10.0M 0 0 2069k 0 0:00:04 0:00:04 --:--:-- 2374k

**[root@devops ~]#** curl **-o** buitansy.zip http://speedtest.tele2.net/10MB.zip

% Total % Received % Xferd Average Speed Time Time Time Current

Dload Upload Total Spent Left Speed

100 10.0M 100 10.0M 0 0 2272k 0 0:00:04 0:00:04 --:--:-- 2383k

[root@devops ~]# ls -al | egrep zip

-rw-r--r--. 1 root root 10485760 Jul 30 23:33 10MB.zip

-rw-r--r--. 1 root root 10485760 Jul 30 23:34 buitansy.zip

**[root@devops ~]#**

**-v, --verbose**

Makes the fetching more verbose/talkative. Mostly useful for debugging. A line starting with '>' means "header data" sent by curl, '<' means "header data" received by curl that is hidden in normal cases, and a line starting with '\*' means additional info provided by curl.

Note that if you only want HTTP headers in the output, -i, --include might be the option you're looking for.

If you think this option still doesn't give you enough details, consider using --trace or --trace-ascii instead.

This option overrides previous uses of --trace-ascii or --trace.

Use -s, --silent to make curl quiet.

// Hiển thị chi tiết hơn về truy vấn, bao gồm về certificate, HTTP Header request, HTTP Header respose…

**[root@devops ~]#** curl -v https://speedtest.net

\* About to connect() to speedtest.net port 443 (#0)

\* Trying 151.101.194.219...

\* Connected to speedtest.net (151.101.194.219) port 443 (#0)

\* Initializing NSS with certpath: sql:/etc/pki/nssdb

\* CAfile: /etc/pki/tls/certs/ca-bundle.crt

CApath: none

\* SSL connection using TLS\_ECDHE\_RSA\_WITH\_CHACHA20\_POLY1305\_SHA256

\* Server certificate:

\* subject: CN=\*.speedtest.net

\* start date: Jun 30 16:23:20 2021 GMT

\* expire date: Aug 01 16:23:19 2022 GMT

\* common name: \*.speedtest.net

\* issuer: CN=GlobalSign Atlas R3 DV TLS CA 2020,O=GlobalSign nv-sa,C=BE

> GET / HTTP/1.1

> User-Agent: curl/7.29.0

> Host: speedtest.net

> Accept: \*/\*

>

< HTTP/1.1 301 Moved Permanently

< Connection: close

< Content-Length: 0

< Server: Varnish

< Retry-After: 0

< Content-Type:

< Accept-Ranges: bytes

< Date: Fri, 30 Jul 2021 16:46:50 GMT

< Via: 1.1 varnish

< X-Served-By: cache-qpg1263-QPG

< X-Cache: MISS

< X-Cache-Hits: 0

< X-Timer: S1627663610.090004,VS0,VE0

< location: https://www.speedtest.net/

<

\* Closing connection 0

**[root@devops ~]#**

**-D, --dump-header <file>**

Write the protocol headers to the specified file.

This option is handy to use when you want to store the headers that an HTTP site sends to you. Cookies from the headers could then be read in a second curl invocation by using the -b, --cookie option! The -c, --cookie-jar option is however a better way to store cookies.

When used in FTP, the FTP server response lines are considered being "headers" and thus are saved there.

If this option is used several times, the last one will be used.