

# Introducing Burp Extractor

## The Problem With Tokens and Scanning

Burp Suite's cookie jar is an incredibly handy tool that makes a penetration tester's life much easier in many situations. It makes for a no hassle way to reissue requests in Repeater as a different user, scan requests which were originally issued in a previous session, and other fun things. But what happens when you need to change more than just cookies? Sometimes an application requires an anti-CSRF token, an updated expiration time, or maybe a session is tracked in an Authorization header instead of cookies. Normally with Burp, this could cause a major headache and a match/replace might be used, or it could be worth struggling through setting up a macro. Both methods requiring a good amount of manual intervention.

## Burp Extractor as an Easy-to-Use Solution

Burp Extractor is intended to be a pain killer for these headaches. This extension allows penetration testers to select a string to extract from responses using a regular expression-based matching mechanism, and insert that string into any request with a similar matching mechanism, effectively making a cookie jar for whatever dynamic information the tester needs.

## Using Burp Extractor

If a request requires a value from a response, right click on that request and select "Send to Extractor". Then find a response where the client receives this value from, and select "Send to Extractor".

Intercept HTTP history WebSockets history Options

Logging of out-of-scope Proxy traffic is disabled

Filter: Hiding out of scope items; hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status	Length
7	http://localhost:8081	GET	/tokens/useToken?token=QRW...	✓		200	206
6	http://localhost:8081	GET	/tokens/getToken			200	232
4	http://localhost:8081	GET	/favicon.ico			404	1672
3	http://localhost:8081	GET	/javascripts/tokens.js			200	1316
1	http://localhost:8081	GET	/			200	582

Request Response

Raw Params Headers Hex

```
GET /tokens/useToken?token=QRWD1kvsWHQ6MpNAZ5GqhDZF2EaAIH3O HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: */*
Referer: http://localhost:8081/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

- Send to Spider
- Do an active scan
- Do a passive scan
- Send to Intruder Ctrl+I
- Send to Repeater Ctrl+R
- Send to Sequencer
- Send to Comparer
- Send to Decoder
- Show response in browser
- Request in browser
- Send to Extractor**
- Engagement tools

Go to the Extractor tab to view a Comparer-like interface, and select the request and response needed, then click "Go".

Target	Proxy	Spider	Scanner	Intruder	Repeater
Sequencer	Decoder	Comparer	Extender	Project options	User options
Alerts		Extractor	Logger++		

Selection

Select request to insert data into:

#	Length	Data
0	364	GET /tokens/getToken HTTP/1.1 Host: localhost 8...
2	352	GET /tokens/useToken?token=QRWD1kvsWHQ6...

Debug  
Paste  
Remove  
Clear

Select response to extract data from:

#	Length	Data
0	232	HTTP/1.1 200 OK:Powered-By: ExpressContent...
2	206	HTTP/1.1 200 OK:Powered-By: ExpressContent...

Go

Within the newly created tab, highlight the content of the request which needs to be replaced, and the content of the response which contains the value to be inserted. Adjust the scope as necessary, and click "Turn Extractor on".

Target	Proxy	Spider	Scanner	Intruder	Repeater
Sequencer	Decoder	Comparer	Extender	Project options	User options
Alerts		Extractor	Logger++		

Selection 2 x

Turn Extractor on ?

Value to insert:

Use suite scope  
 Use specified target host

Target host: localhost  Regex

Regex: (.useToken/?token=)[\*?]( HTTP/1.1/v/nHost.\*)

Request

```
GET
/tokens/useToken?token=QRWD1kvsWHQ6MpNAZ5GqbDZF2EaAIH30
HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebFit/537.36 (KHTML, like Gecko)
Chrome/67.0.3396.99 Safari/537.36
Accept: */*
Referer: http://localhost:8081/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close
```

Response

Use suite scope  
 Use specified target host

Target host: localhost  Regex

Regex: (.tion/: close/v/n/n)[\*?](\*)(\$)

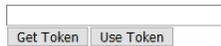
```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: text/html; charset=utf-8
Content-Length: 32
ETag: W/"20-APg2FS43FOPzUeV+d29FGnoMSR1"
Date: Fri, 27 Jul 2016 19:50:01 GMT
Connection: close
QRWD1kvsWHQ6MpNAZ5GqbDZF2EaAIH30
```

? < + > Type a search term 0 matches

Once turned on, Extractor will look for occurrences of the regex listed in the request and response panels, and extract or insert data appropriately. It will also update the "Value to insert" field with the newest value extracted.

Check out the demonstration below to see an example usage of the extension, and grab the extension off GitHub! Hopefully this tool will help ease some pains when dealing with web apps that use tokens or authentication that Burp is not always well-equipped to cope with.

## Use Some Tokens



The image shows a small, light-colored rectangular box. Inside the box, there is a text input field at the top, which is currently empty. Below the input field, there are two buttons side-by-side. The left button is labeled "Get Token" and the right button is labeled "Use Token". Both buttons have a light gray background and a thin border.

Example usage of Burp Extractor