Burp intruder attack types

https://www.sjoerdlangkemper.nl/2017/08/02/burp-intruder-attack-types/

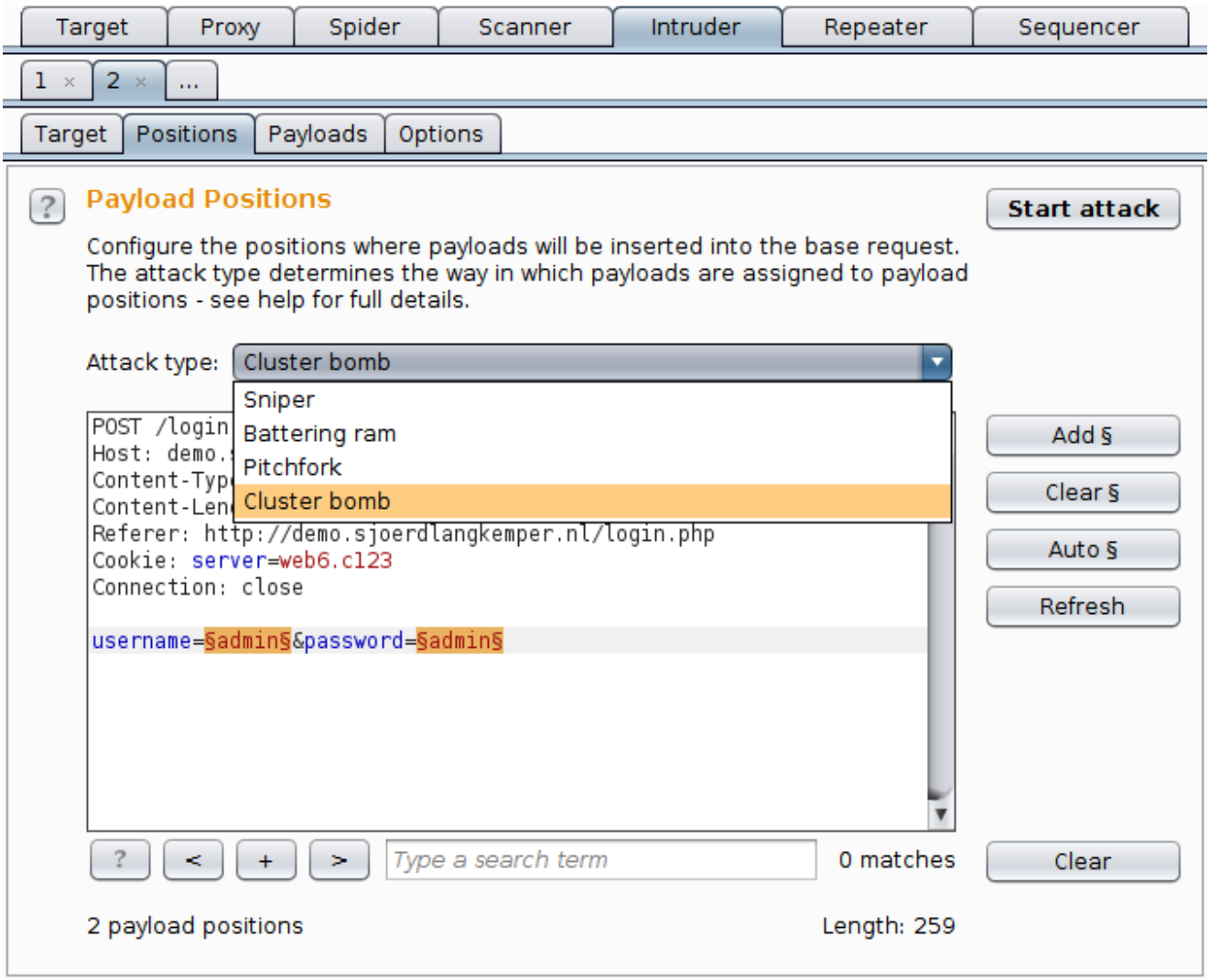
Burp is an intercepting proxy that can be used to test web sites. It has a fuzzing feature called *intruder* that can replace parameters in a request with values from one or more payload lists. It has several attack types that determine how the payloads are used in the request parameters. This post explains how the different attack types work.

Intruder introduction

Burp Intruder makes it possible to perform a number of automatically modified requests. For example, you can perform a brute-force attack by configuring the intruder with a login request and lists with usernames and passwords. There are several ways to configure an intruder attack:

- the base request, as shown on the Positions tab. This is the template for the request.
- the attack type, on the Positions tab, determines the way payloads are put in positions and is the subject of this post.
- the positions within the requests, also shown on the Positions tab. The positions are marked using § characters. Anything between two § characters is replaced by a payload.
- the payload sets on the Payload tab contain the data that is inserted into the positions. Each payload set has some way to generate payloads, which are strings to use in the request.

After clicking the “Start attack” button, the intruder will perform a number of requests, replacing the marked positions with payloads in each request. Exactly which payloads it puts in which position depends on the attack type. This also determines how many requests it will perform.



Sniper

The sniper attack uses only one payload set, and it replaces only one position at a time. It loops through the payload set, first replacing only the first marked position with the payload and leaving all other positions to their original value. After its done with the first position, it continues with the second position.

This attack type is most useful when fuzzing, for example to find XSS or SQL injection. The payload is tried in each position while leaving the other parameters intact, making a successful request more likely.

The sniper attack

- replaces one position at a time,
- uses one payload set, regardless of the number of positions,
- uses the original values for all positions that have no payload,
- does *positions* × *payloads* requests.

For example, consider a URL with two positions. First, the first position is replaced by values from the payload set and the second position is left alone. After all values are exhausted, the second position is used and the first position is left alone.

Original URL
/search.php?cat_id=§123§&q=§hello§

Payload set 1
456

Requests
/search.php?cat_id=456&q=hello
/search.php?cat_id= &q=hello
/search.php?cat_id=123&q=456
/search.php?cat_id=123&q=

Battering ram

The battering ram attack type places the same payload value in all positions. It uses only one payload set. It loops through the payload set and replaces all positions with the payload value.

The battering ram

- uses one payload set, regardless of the number of positions,
- replaces all positions with the same payload,
- does as many requests as there are payloads in the payload set.

Using the same example URL with the two positions, you can see that the same payload is put in all positions.

Original URL
/search.php?cat_id=§123§&q=§hello§

Payload set 1
456

Requests
/search.php?cat_id=456&q=456
/search.php?cat_id= &q=

Pitchfork

The pitchfork attack type uses one payload set for each position. It places the first payload in the first position, the second payload in the second position, and so on.

It then loops through all payload sets at the same time. The first request uses the first payload from each payload set, the second request uses the second payload from each payload set, and so on.

This attack type is useful if you have data items that belong together. For example, you have usernames with corresponding passwords and want to know whether they work with this web application. In this case you want to replace both the username and the password in the login request. Load the usernames in the first payload set, and the corresponding passwords in the second payload set. Now only one request for each username/password combination is done.

The pitchfork attack

- uses as many payload sets as there are positions,
- replaces each position with its respective payload,
- does as many requests as the maximum payload set size,
- first payload set goes into first position, etc.

In this example you can see that it uses the first payload from each set in the first request, and the second payload from each set in the second request.

Original URL
/search.php?cat_id=§123§&q=§hello§

Payload set 1
456

Requests
/search.php?cat_id=456&q=aaa
/search.php?cat_id= &q=richard

Payload set 2
aaa
richard

Cluster bomb

The cluster bomb attack tries all different combinations of payloads. It still puts the first payload in the first position, and the second payload in the second position. But when it loops through the payload sets, it tries all combinations.

This attack type is useful for a brute-force attack. Load a list of commonly used usernames in the first payload set, and a list of commonly used passwords in the second payload set. The cluster bomb attack will then try all combinations.

Note that the number of requests can grow very quickly. If you have 100 usernames and 100 passwords, this attack will perform 10,000 requests. This becomes exponentially worse when using more positions, so this attack is only feasible with a relatively small number of payloads and positions.

The cluster bomb attack

- makes combinations with all payload sets,
- does *payloads__positions* requests,
- first payload set goes into first position, etc.

The cluster bomb tries all possible combinations, while still keeping the first payload set in the first position and the second payload set in the second position.

Original URL
/search.php?cat_id=§123§&q=§hello§

Payload set 1
456

Payload set 2
aaa
richard

Requests
/search.php?cat_id=456&q=aaa
/search.php?cat_id= &q=aaa
/search.php?cat_id=456&q=richard
/search.php?cat_id= &q=richard