Burp Suite: Intruder

By :- NAITRO 07

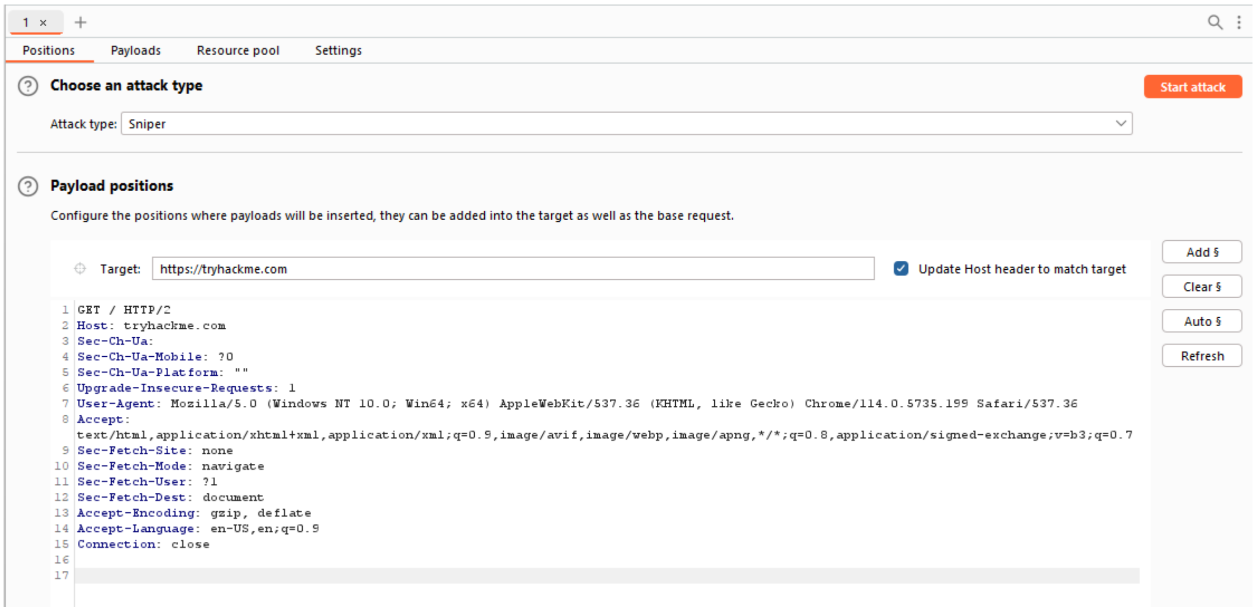
What is Intruder

Intruder is Burp Suite's built-in **fuzzing tool** that allows for automated request modification and repetitive testing with variations in input values. By using a captured request (often from the Proxy module), Intruder can send multiple requests with slightly altered values based on user-defined configurations. It serves various purposes, such as brute-forcing login forms by substituting username and password fields with values from a wordlist or performing fuzzing attacks using *wordlists to test subdirectories,* endpoints, or virtual hosts.

Intruder's functionality is comparable to command-line tools like **Wfuzz** or **ffuf**.

However, it's important to note that while Intruder can be used with Burp Community Edition, it is rate-limited, significantly reducing its speed compared to ***Burp Professional***. This limitation often leads security practitioners to rely on other tools for fuzzing and brute-forcing. Nonetheless, Intruder remains a valuable tool and is worth learning how to use it effectively.

Let's explore the Intruder interface:



Choose an attack type

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: https://tryhackme.com

✓ Update Host header to match target

```
GET / HTTP/2
Host: tryhackme.com
Sec-Ch-Ua:
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: ""
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.199 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close


```

The initial view of Intruder presents a simple interface where we can select our target. This field will already be populated if a request has been sent from the Proxy (using `Ctrl + I` or right-clicking and selecting *"Send to Intruder"*).

There are four sub-tabs within Intruder:

- **Positions:** This tab allows us to select an attack type (which we will cover in a future task) and configure where we want to insert our payloads in the request template.
- **Payloads:** Here we can select values to insert into the positions defined in the **Positions** tab. We have various payload options, such as loading items from a wordlist. The way these payloads are inserted into the template depends on the attack type chosen in the **Positions** tab. The **Payloads** tab also enables us to modify Intruder's behavior regarding payloads, such as defining pre processing rules for each payload (e.g., adding a prefix or suffix, performing match and replace, or skipping payloads based on a defined regex).
- **Resource Pool:** This tab is not particularly useful in the Burp Community Edition. It allows for resource allocation among various automated tasks in Burp Professional. Without access to these automated tasks, this tab is of limited importance.
- **Settings:** This tab allows us to configure attack behavior. It primarily deals with how Burp handles results and the attack itself. For instance, we can flag

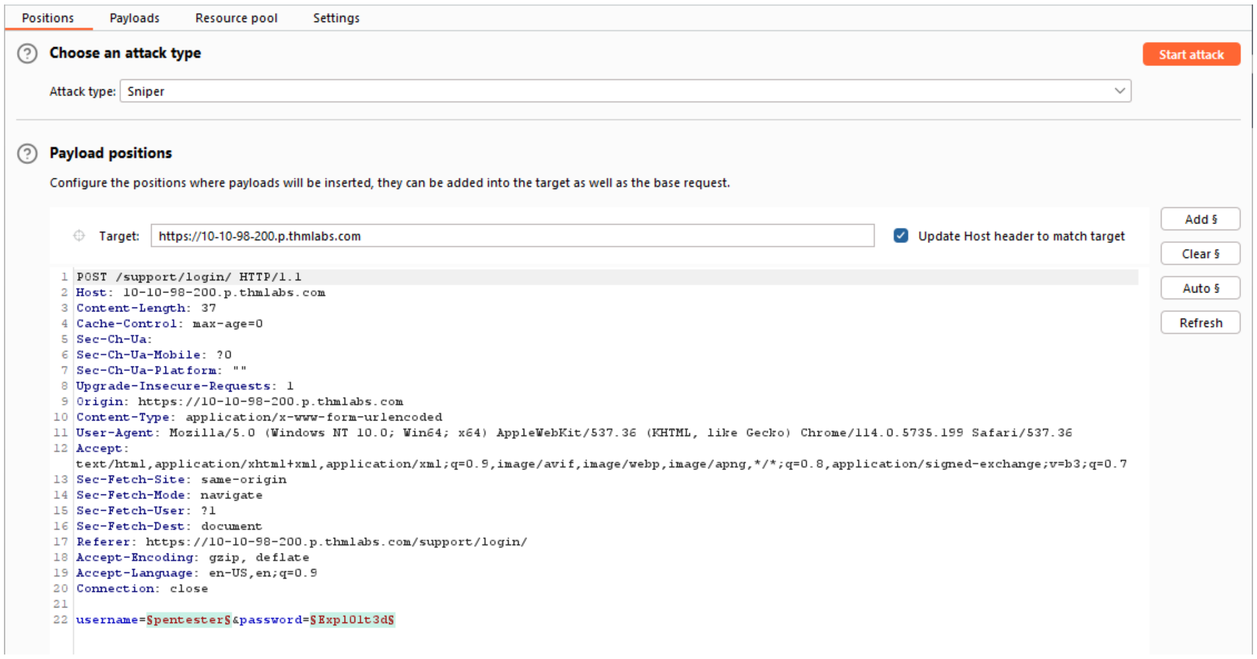
requests containing specific text or define Burp's response to redirect (3xx) responses.

Note: The term "fuzzing" refers to the process of testing functionality or existence by applying a set of data to a parameter. For example, fuzzing for endpoints in a web application involves taking each word in a wordlist and appending it to a request URL (e.g., `http://MACHINE_IP/WORD_GOES_HERE`) to observe the server's response.

Positions

When using Burp Suite Intruder to perform an attack, the first step is to examine the positions within the request where we want to insert our payloads. These positions inform Intruder about the locations where our payloads will be introduced (as we will explore in upcoming tasks).

Let's navigate to the Positions tab:



Choose an attack type

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: https://10-10-98-200.p.thmlabs.com

[✓] Update Host header to match target

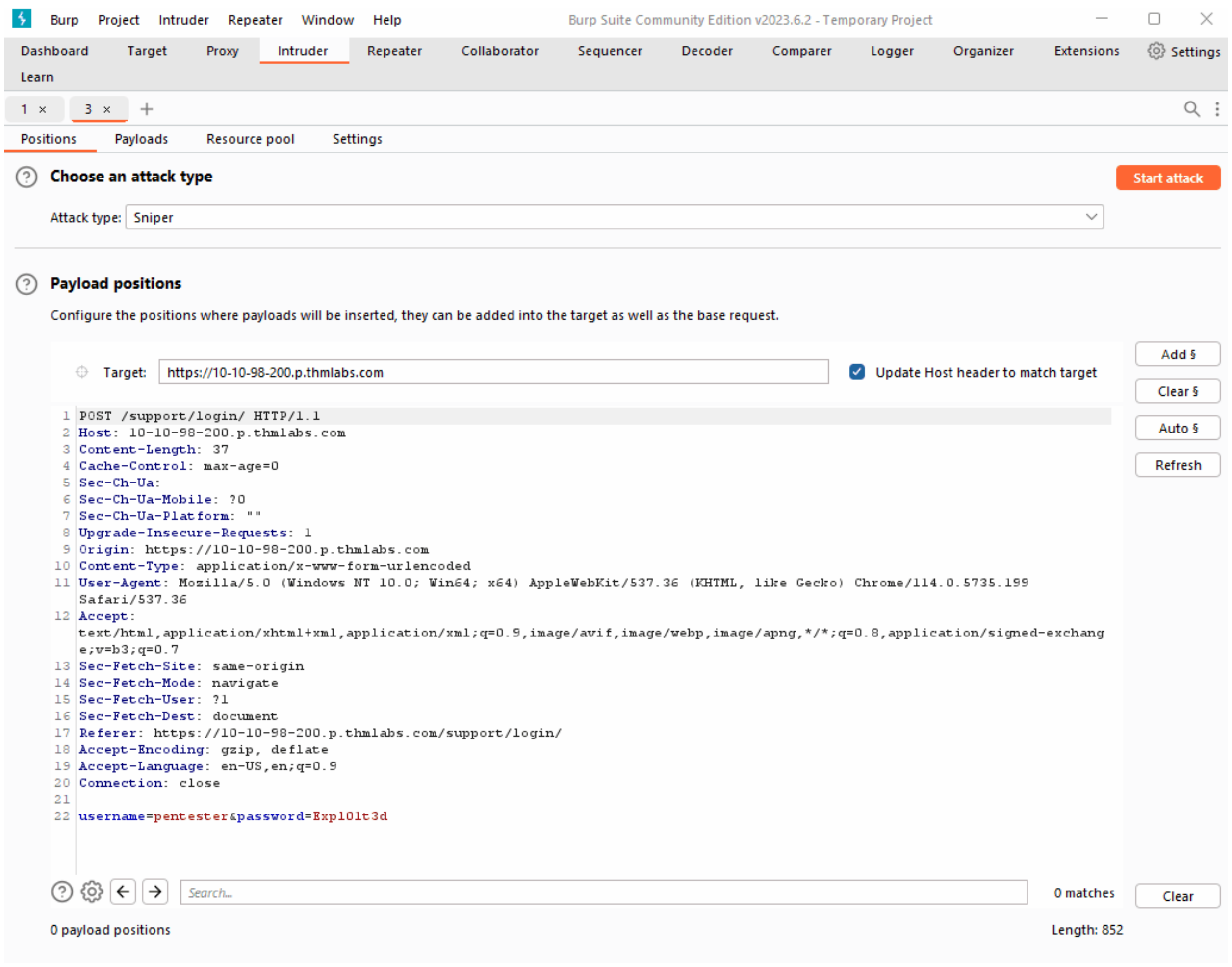
```
POST /support/login/ HTTP/1.1
Host: 10-10-98-200.p.thmlabs.com
Content-Length: 37
Cache-Control: max-age=0
Sec-Ch-Ua:
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: ""
Upgrade-Insecure-Requests: 1
Origin: https://10-10-98-200.p.thmlabs.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.199 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://10-10-98-200.p.thmlabs.com/support/login/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

username=§pentester§&password=§Expl01t3d§
```

Notice that Burp Suite automatically attempts to identify the most probable positions where payloads can be inserted. These positions are highlighted in green and enclosed by section marks (`§`).

On the right-hand side of the interface, we find the following buttons: `Add §`, `Clear §`, and `Auto §`:

- The `Add §` button allows us to define new positions manually by highlighting them within the request editor and then clicking the button.
- The `Clear §` button removes all defined positions, providing a blank canvas where we can define our own positions.
- The `Auto §` button automatically attempts to identify the most likely positions based on the request. This feature is helpful if we previously cleared the default positions and want them back.

The following GIF demonstrates the process of adding, clearing, and automatically reselecting positions:



Choose an attack type

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: https://10-10-98-200.p.thmlabs.com

[✓] Update Host header to match target

```
POST /support/login/ HTTP/1.1
Host: 10-10-98-200.p.thmlabs.com
Content-Length: 37
Cache-Control: max-age=0
Sec-Ch-Ua:
Sec-Ch-Ua-Mobile: ?0
Sec-Ch-Ua-Platform: ""
Upgrade-Insecure-Requests: 1
Origin: https://10-10-98-200.p.thmlabs.com
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/114.0.5735.199
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchang
e;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: https://10-10-98-200.p.thmlabs.com/support/login/
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
Connection: close

username=pentester&password=Expl01t3d
```

Search... 0 matches

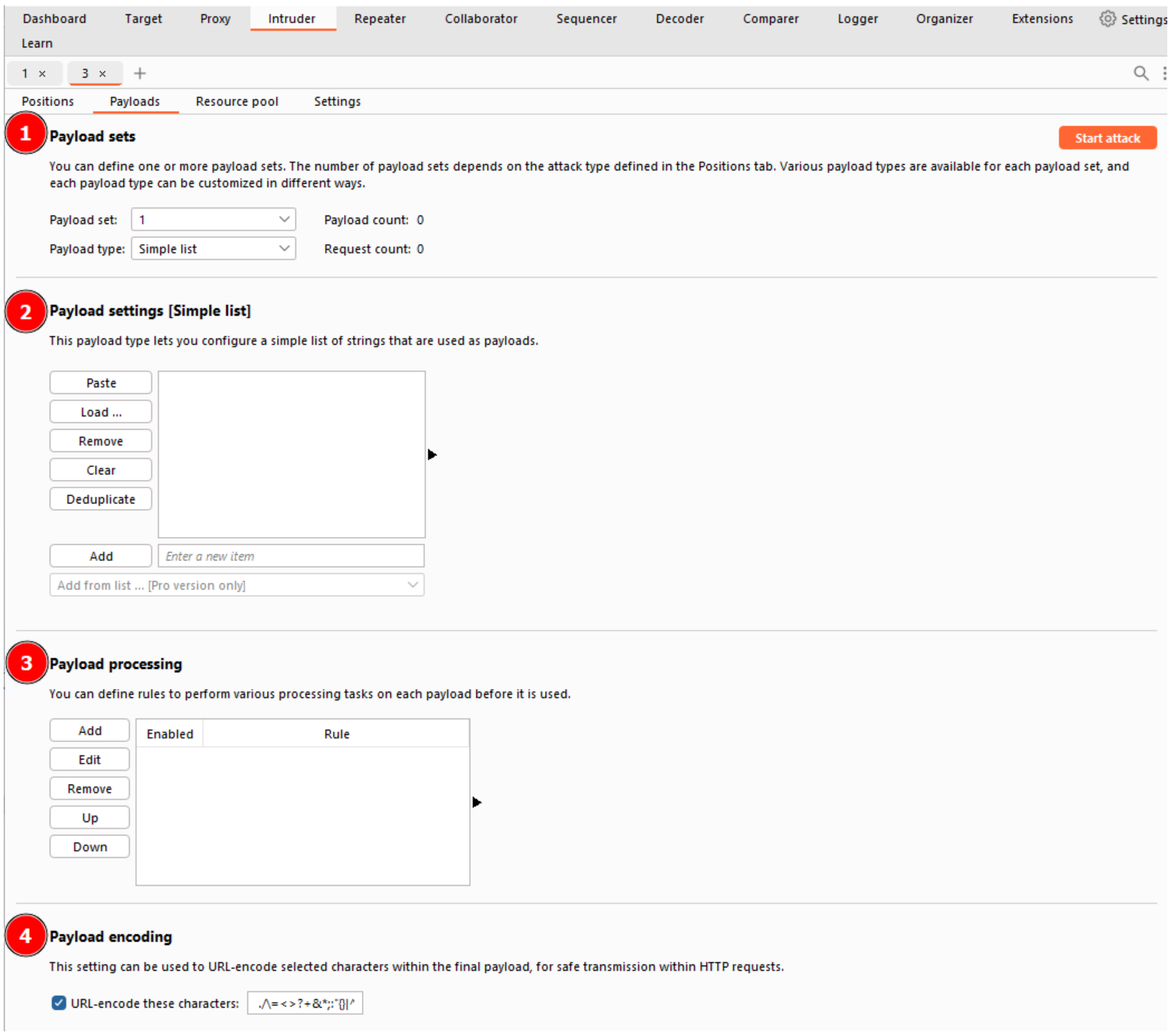
0 payload positions

Length: 852

Take some time to familiarize yourself with adding, clearing, and auto-selecting positions using the Burp Suite Intruder interface.

Payloads

In the **Payloads** tab of Burp Suite Intruder, we can create, assign, and configure payloads for our attack. This sub-tab is divided into four sections:



Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1
Payload count: 0
Payload type: Simple list
Request count: 0

Payload settings [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Payload encoding

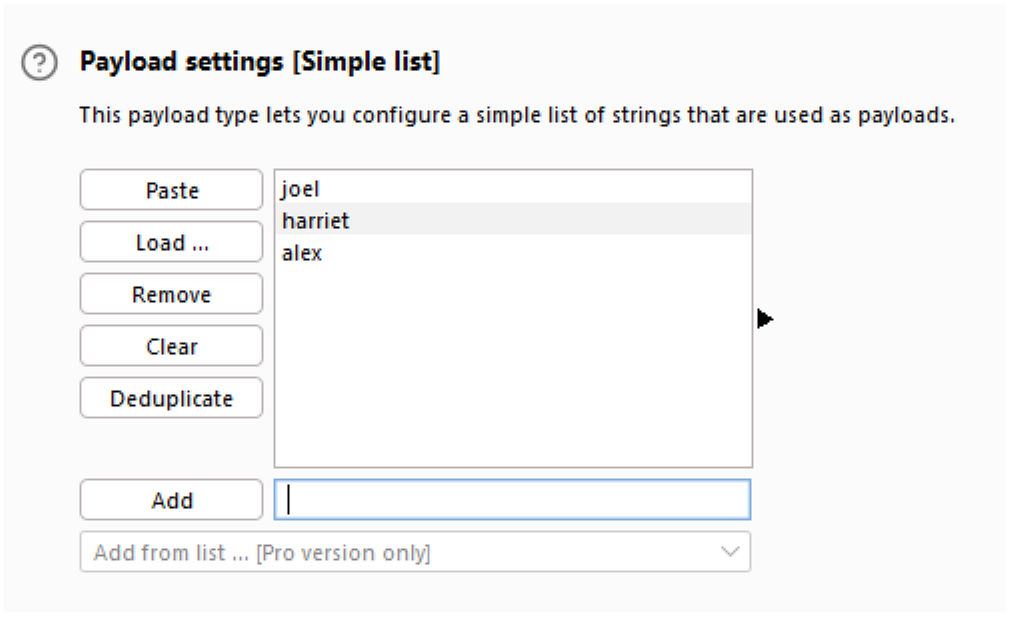
This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.

✓ URL-encode these characters: ./\=<>?+&*;:"{}|^

1. **Payload Sets:**
    - This section allows us to choose the position for which we want to configure a payload set and select the type of payload we want to use.

- When using attack types that allow only a single payload set (Sniper or Battering Ram), the "Payload Set" dropdown will have only one option, regardless of the number of defined positions.
- If we use attack types that require multiple payload sets (Pitchfork or Cluster Bomb), there will be one item in the dropdown for each position.
- ***Note:*** When assigning numbers in the "Payload Set" dropdown for multiple positions, follow a top-to-bottom, left-to-right order. For example, with two positions (`username=§pentester§&password=§Expl01ted§`), the first item in the payload set dropdown would refer to the username field, and the second item would refer to the password field.

2. **Payload settings:**
    - This section provides options specific to the selected payload type for the current payload set.
    - For example, when using the "Simple list" payload type, we can manually add or remove payloads to/from the set using the **Add** text box, **Paste** lines, or **Load** payloads from a file. The **Remove** button removes the currently selected line, and the **Clear** button clears the entire list. Be cautious with loading huge lists, as it may cause Burp to crash.
    - Each payload type will have its own set of options and functionality. Explore the options available to understand the range of possibilities.


Payload settings [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

joel
harriet
alex

3. **Payload Processing:**
   - In this section, we can define rules to be applied to each payload in the set before it is sent to the target.
   - For example, we can capitalize every word, skip payloads that match a regex pattern, or apply other transformations or filtering.
   - While you may not use this section frequently, it can be highly valuable when specific payload processing is required for your attack.
4. **Payload Encoding:**
   - The section allows us to customize the encoding options for our payloads.
   - By default, Burp Suite applies URL encoding to ensure the safe transmission of payloads. However, there may be cases where we want to adjust the encoding behavior.
   - We can override the default URL encoding options by modifying the list of characters to be encoded or unchecking the "URL-encode these characters" checkbox.

By leveraging these sections, we can create and customize payload sets to suit the specific requirements of our attacks. This level of control allows us to finely tune our payloads for effective testing and exploitation.

Introduction to Attack Types

The **Positions** tab of Burp Suite Intruder has a dropdown menu for selecting the attack type. Intruder offers **four attack types**, each serving a specific purpose. Let's explore each
of them:

1. **Sniper:** The **Sniper** attack type is the default and most commonly used option. It cycles through the payloads, inserting one **payload** at a time into each position defined in the request. Sniper
attacks iterate through all the
 payloads in a linear fashion, allowing for precise and focused testing.
2. **Battering ram**: The **Battering ram** attack type differs from Sniper in that it sends all payloads simultaneously, each payload inserted into its respective position. This attack type is
useful when testing for race conditions or when payloads need to be sent concurrently.
3. **Pitchfork:** The **Pitchfork attack** type enables the simultaneous testing of multiple positions with different payloads. It allows the tester to define multiple payload sets, each associated with a specific position in the request. **Pitchfork** attacks are effective when there are distinct parameters that need separate testing.
4. **Cluster bomb:** The **Cluster bomb attack type** combines the **Sniper and Pitchfork** approaches. It performs a Sniper-like attack on each position but simultaneously tests all payloads from each
set. This attack type is useful when multiple positions have different payloads, and we want to test them all together.

Each attack type has its advantages and is suitable for different testing scenarios. Understanding their differences helps us select the appropriate attack type based on the testing objectives.

Sniper

The **Sniper** attack type is the default and most commonly used attack type in Burp Suite Intruder. It is particularly effective for single-position attacks, such as ***password brute-force or fuzzing for API*** endpoints. In a Sniper attack, we provide a set of payloads, which can be a wordlist or a range of numbers, and Intruder inserts each payload into each defined position in the request.

Let's refer to our example template from before:

Example Positions

```
POST /support/login/ HTTP/1.1
Host: MACHINE_IP
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:80.0)
Gecko/20100101 Firefox/80.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.
9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 37
Origin: http://MACHINE_IP
Connection: close
Referer: http://MACHINE_IP/support/login/
Upgrade-Insecure-Requests: 1

username=§pentester§&password=§Expl01ted§
```

In this example, we have two positions defined for the `username` and `password` body parameters. In a Sniper attack, Intruder takes each payload from the payload set and substitutes it into each defined position in turn.

Assuming we have a wordlist with three words: `burp`, `suite`**, and** `intruder`, Intruder would generate six requests:

Request Number	Request Body
1	`username=burp&password=Expl01ted`
2	`username=suite&password=Expl01ted`

3	`username=intruder&password=Expl01ted`
4	`username=pentester&password=burp`
5	`username=pentester&password=suite`
6	`username=pentester&password=intruder`

Observe how Intruder starts with the first position (`username`) and substitutes each payload into it, then moves to the second position (`password`) and performs the same substitution with the payloads. The total number of requests made by Intruder Sniper can be calculated as

`requests = numberOfWords * numberOfPositions`.

The Sniper attack type is beneficial when we want to perform tests with **single-position attacks**, utilizing different payloads for each position. It allows for precise testing and analysis of different
payload variations.

Battering Ram

The **Battering ram** attack type in Burp Suite Intruder differs from Sniper in that it places the same payload in every position simultaneously, rather than substituting each payload into each position in turn.

Let's refer back to our previous example template:

Example Positions

```
POST /support/login/ HTTP/1.1
    Host: MACHINE_IP
    User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:8
0.0) Gecko/20100101 Firefox/80.0
    Accept: text/html,application/xhtml+xml,application/xml;q
=0.9,image/webp,*/*;q=0.8
    Accept-Language: en-US,en;q=0.5
    Accept-Encoding: gzip, deflate
    Content-Type: application/x-www-form-urlencoded
```

```
Content-Length: 37
Origin: http://MACHINE_IP
Connection: close
Referer: http://MACHINE_IP/support/login/
Upgrade-Insecure-Requests: 1

username=§pentester§&password=§Expl01ted§
```

Using the **Battering Ram attack** type with the same wordlist from before (`burp`, `suite`, and `intruder`), Intruder would generate three requests:

Request Number	Request Body
1	`username=burp&password=burp`
2	`username=suite&password=suite`
3	`username=intruder&password=intruder`

As shown in the table, each payload from the wordlist is inserted into every position for each request made. In a **Battering Ram attack,** the same payload is thrown at every defined position simultaneously, providing a brute-force-like approach to testing.

The **Battering Ram attack** type is useful when we want to test the same payload against multiple positions at once without the need for sequential substitution.

In the upcoming tasks, we will explore further configurations and settings related to Intruder's Battering Ram attack type and examine its applications in different scenarios.

Pitchfork

The **Pitchfork** attack type in Burp Suite Intruder is similar to having multiple Sniper attacks running simultaneously. While Sniper uses one payload set to test all positions simultaneously, Pitchfork utilises one payload set per position (**up to a maximum of 20**) and iterates through them all simultaneously.

To better understand Pitchfork, let us **revisit our brute-force** example, but this time with two wordlists:

1. The first wordlist contains usernames: `joel`, `harriet`, and `alex`.
2. The second wordlist contains passwords: `J03l`, `Emma1815`, and `Sk1ll`.

We can use these two lists to perform a Pitchfork **attack on the login form**. Each request made during the attack would look like this:

Request Number	Request Body
1	`username=joel&password=J03l`
2	`username=harriet&password=Emma1815`
3	`username=alex&password=Sk1ll`

As shown in the table, **Pitchfork** takes the first item from each list and substitutes them into the request, one per position. It then repeats this process for the next request by taking the second item from each list and substituting it into the template. Intruder continues this iteration until one or all of the lists run out of items. It's important to note that Intruder stops testing as soon as one of the lists is complete. Therefore, in Pitchfork attacks, it is ideal for the payload sets to have the same length. If the lengths of the payload sets differ, Intruder will only make requests until the shorter list is exhausted, and the remaining items in the longer list will not be tested.

The Pitchfork attack type is especially useful when conducting **credential-stuffing attacks** or when multiple positions require separate payload sets. It allows for simultaneous testing of multiple positions with different payloads.

Cluster Bomb

The **Cluster bomb** attack type in Burp Suite Intruder allows us to choose multiple payload sets, one per position (up to a maximum of 20). Unlike Pitchfork, where all payload sets are tested simultaneously, Cluster bomb iterates through each payload set individually, ensuring that every possible combination of payloads is tested.

To illustrate the Cluster bomb attack type, let's use the same wordlists as before:

- **Usernames:** `joel`, `harriet`, and

  `alex`.
- **Passwords:** `J03l`, `Emma1815`, and

  `Sk1ll`.

In this example, let's assume that we don't know which password belongs to which user. We have three users and three passwords, but the **mappings** are unknown. In this case, we can use a Cluster bomb attack to try every combination of values. The request table for our username and password positions would look like this:

Request Number	Request Body
1	`username=joel&password=J03l`
2	`username=harriet&password=J03l`
3	`username=alex&password=J03l`
4	`username=joel&password=Emma1815`
5	`username=harriet&password=Emma1815`
6	`username=alex&password=Emma1815`
7	`username=joel&password=Sk1ll`
8	`username=harriet&password=Sk1ll`
9	`username=alex&password=Sk1ll`

As shown in the table, the Cluster bomb attack type iterates through every combination of the provided payload sets. It tests every possibility by substituting each value from each payload set into the corresponding position in the request.

Cluster bomb attacks can generate a significant amount of traffic as it tests every combination. The number of requests made by a Cluster **bomb attack can be calculated by multiplying the number of lines in each payload set together.** It's important to be cautious when using this attack type, especially when dealing with large payload sets.
Additionally, when using Burp Community and its Intruder rate-limiting, the

execution of a Cluster bomb attack with a moderately sized payload set can take a significantly longer time.

The Cluster bomb attack type is particularly useful for credential brute-forcing scenarios where the mapping between usernames and passwords is unknown.

Thanks NAITRO 07