Front end SocketIO Events

(Version 2016-12-02)

Events that the client can send:

When emitting an event, the key names of the dictionary have to be the same as the documentation.
For example, in the login event below, inside the dictionary that will be passed along,
the key name for the user's ID must be "id",
the key name for the user's PS must be "ps".
So the full event emit will look like this:

```
socket.emit('login', {
   id: document.getElementById('#user_id').innerHTML,
   ps: document.getElementById('#user_ps').innerHTML
});
```

login: To login a user.
- form - Dictionary containing the ID and password.
  - id: User ID
  - ps: User password

register: To register a new user.
- form - Dictionary containing the new ID and password.
  - id: New user ID
  - ps: New user password

test_request: To request a new ZAP test against a given website.
- url - String representing the URL of the user's website.

test_cancel: To cancel an ongoing ZAP test.

Events that the client can receive:

When receiving an event from the server, the argument in the callback function can have any name desired. For example, in the login_reject event below, the argument name "error" is only there for consistency, and can be declared any way.
So the full event listener will look like this:

```
socket.on('login_reject', function (err_msg) {
   alert('Login Failed!\n', err_msg);
});
```

login_reject: When a user login fails**.**
    error - String representing the basic cause of failure.
login_success: When a user login succeeds.
    user - String representing userID.

register_reject: When a user registration fails.
    error - String representing the basic cause of failure.
register_success: When a user registration succeeds.

test_result: When a ZAP test is complete.
    data - Array of scanned vulnerabilities.
test_cancel_success: When a ZAP test has successfully been cancelled.

progress_zap_0_exec: When the server first spawns a new ZAP process.
progress_zap_1_spider: When ZAP performs a spider scan on the website.
progress_zap_2_active: When ZAP performs an active scan on each URL.
progress_zap_3_attack_done: When ZAP finishes the actual scan and prepares to finalize.
progress_zap_4_write_start: When ZAP starts writing a new report file.
progress_zap_5_write_done: When ZAP finishes writing the report file.
progress_zap_6_parse_start: When the server starts converting the report to JSON.
progress_zap_7_parse_done: When the server finishes converting the report to JSON.

error_zap_crash: When ZAP crashes in the server.
    error - String representing the basic cause of failure.
error_xmlparse_crash: When the parsing process crashes in the server.
    error - String representing the basic cause of failure.
error_invalid_url: When the given URL is invalid.

verify_token_reject: When the server fails to fetch the user's unique token.
    error - String representing the basic cause of failure
verify_token_success: When the server successfully fetches the user's unique token.

verify_request_fail: When the server fails to verify a new website for the user.
    error - String representing the basic cause of failure
verify_request_success: When the server successfully verifies the a new website for the user.