
Offensive Web Testing Framework Documentation

Release MacinOWTF

OWTF Team

Aug 25, 2021

Contents

1	Installation	3
1.1	Prerequisites	3
1.2	Installation	3
1.3	Advanced Installation	4
2	owtf	5
2.1	owtf package	5
3	Configuration	29
3.1	Database Configuration	29
3.2	Framework Configuration (Optional)	29
4	Usage	31
4.1	Starting OWTF	31
4.2	Using Sessions	31
4.3	Managing Targets	32
4.4	Understanding Plugins	32
4.5	Analyzing results	36
4.6	Managing Workers	43
4.7	Controlling Worklist	45
5	Troubleshooting	47
6	Want Help or Request a feature?	49
	Python Module Index	51
	Index	53



Contents:

1.1 Prerequisites

There are few packages which are mandatory before you proceed

- Git client: `sudo apt-get install git`
- Python 2.7, installed by default in most systems

1.2 Installation

There are two ways in which you can proceed:

1.2.1 Manual Installation

Manual installation of OWTF is nothing but cloning the repo and running the owtf setup.

```
git clone https://github.com/owtf/owtf.git
cd owtf/
python setup.py install
```

1.2.2 Docker

Docker automates the task of setting up owtf doing all the bootstrapping it needs. Just make sure that you have docker and docker-compose installed and run:

```
docker-compose up
```

- If you wish to override the environment variables for docker setup, use the file named `owtf.env`

1.3 Advanced Installation

If your distro is not officially supported in the install script, the following packages might not have been installed. So please make sure you atleast have the mandatory packages installed. Almost all the packages can be obtained using package manager of any major distro.

1.3.1 Mandatory

- [Postgresql](#)

1.3.2 Optional Packages

- [Tor](#) (For Botnet mode)
- [Proxychains](#) (For Botnet mode)

1.3.3 Optional Tools

- [Curl](#)
- [Arachni](#)
- [w3af](#)
- [Skipfish](#)
- [Dirbuster](#)

2.1 owtf package

2.1.1 Subpackages

owtf.api package

Subpackages

owtf.api.handlers package

Submodules

owtf.api.handlers.base module

owtf.api.handlers.config module

owtf.api.handlers.health module

owtf.api.handlers.index module

owtf.api.handlers.misc module

owtf.api.handlers.plugin module

owtf.api.handlers.report module

`owtf.api.handlers.session` module

`owtf.api.handlers.targets` module

`owtf.api.handlers.transactions` module

`owtf.api.handlers.work` module

Module contents

Submodules

`owtf.api.main` module

`owtf.api.reporter` module

`owtf.api.routes` module

`owtf.api.utils` module

`owtf.api.utils`

class `owtf.api.utils.VersionMatches` (*api_version*)

Bases: `tornado.routing.Matcher`

Matches path by *version* regex.

match (*request*)

Matches current instance against the request.

Parameters **request** (*httputil.HTTPServerRequest*) – current HTTP request

Returns a dict of parameters to be passed to the target handler (for example, `handler_kwargs`, `path_args`, `path_kwargs` can be passed for proper `~.web.RequestHandler` instantiation). An empty dict is a valid (and common) return value to indicate a match when the argument-passing features are not used. `None` must be returned to indicate that there is no match.

Module contents

`owtf.cli` package

Submodules

`owtf.cli.main` module

Module contents

`owtf.db` package

Submodules

`owtf.db.database` module

`owtf.db.models` module

Module contents

`owtf.filesrv` package

Submodules

`owtf.filesrv.handlers` module

`owtf.filesrv.main` module

`owtf.filesrv.routes` module

Module contents

`owtf.http` package

Submodules

`owtf.http.requester` module

`owtf.http.transaction` module

Module contents

`owtf.lib` package

Submodules

`owtf.lib.cli_options` module

`owtf.lib.cli_options`

Main CLI processing machine

`owtf.lib.cli_options.parse_options` (*cli_options*, *valid_groups*, *valid_types*)

Main arguments processing for the CLI

Parameters

- **cli_options** (*dict*) – CLI args Supplied by user
- **valid_groups** (*list*) – Plugin groups to chose from
- **valid_types** (*list*) – Plugin types to chose from

Returns**Return type**

`owtf.lib.cli_options.usage(error_message)`

Display the usage message describing how to use owtf.

Parameters `error_message` (*str*) – Error message to display

Returns None

Return type None

owtf.lib.exceptions module**owtf.lib.exceptions**

Declares the framework exceptions and HTTP errors

exception `owtf.lib.exceptions.APIError` (*status_code=500, log_message=None, *args, **kwargs*)

Bases: `tornado.web.HTTPError`

Equivalent to `RequestHandler.HTTPError` except for in name

exception `owtf.lib.exceptions.DBIntegrityException` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.DatabaseNotRunningException`

Bases: `exceptions.Exception`

exception `owtf.lib.exceptions.FrameworkAbortException` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.FrameworkException` (*value*)

Bases: `exceptions.Exception`

exception `owtf.lib.exceptions.InvalidActionReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidConfigurationReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidErrorReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidMappingReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidMessageReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidParameterType` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidSessionReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidTargetReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidTransactionReference` (*value*)

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidUrlReference (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidWorkReference (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.InvalidWorkerReference (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.PluginAbortException (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.PluginException`

Bases: `exceptions.Exception`

exception `owtf.lib.exceptions.PluginsAlreadyLoaded`

Bases: `owtf.lib.exceptions.PluginException`

`load_plugins()` called twice.

exception `owtf.lib.exceptions.PluginsDirectoryDoesNotExist`

Bases: `owtf.lib.exceptions.PluginException`

The specified plugin directory does not exist.

exception `owtf.lib.exceptions.UnreachableTargetException (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

exception `owtf.lib.exceptions.UnresolvableTargetException (value)`

Bases: `owtf.lib.exceptions.FrameworkException`

`owtf.lib.exceptions.api_assert (condition, *args, **kwargs)`

Assertion to fail with if not condition Asserts that condition is True, else raises an `APIError` with the provided args and kwargs :type condition: bool

owtf.lib.filelock module

owtf.lib.filelock

Implementation of a simple cross-platform file locking mechanism. This is a modified version of code retrieved on 2013-01-01 from <http://www.evanfosmark.com/2009/01/cross-platform-file-locking-support-in-python>. The original code was released under the BSD License, as is this modified version. Modifications in this version:

- Tweak docstrings for sphinx.
- Accept an absolute path for the protected file (instead of a file name relative to cwd).
- Allow timeout to be None.
- Fixed a bug that caused the original code to be NON-threadsafe when the same FileLock instance was shared by multiple threads in one process. (The original was safe for multiple processes, but not multiple threads in a single process. This version is safe for both cases.)
- Added `purge()` function.
- Added `available()` function.
- Expanded API to mimic threading.Lock interface: - `__enter__` always calls `acquire()`, and therefore blocks if `acquire()` was called previously. - `__exit__` always calls `release()`. It is therefore a bug to call `release()` from within a context manager. - Added `locked()` function. - Added blocking parameter to `acquire()` method

taken from <https://github.com/ilastik/lazyflow/blob/master/lazyflow/utility/fileLock.py> # original version from <http://www.evanfosmark.com/2009/01/cross-platform-file-locking-support-in-python/>

```
class owtf.lib.filelock.FileLock(protected_file_path,          timeout=None,          delay=1,
                                lock_file_contents=None)
```

Bases: object

A file locking mechanism that has context-manager support so you can use it in a `with` statement. This should be relatively cross compatible as it doesn't rely on `msvcrt` or `fcntl` for the locking.

exception FileLockException

Bases: `exceptions.Exception`

acquire (*blocking=True*)

Acquire the lock, if possible. If the lock is in use, and *blocking* is False, return False. Otherwise, check again every *self.delay* seconds until it either gets the lock or exceeds *timeout* number of seconds, in which case it raises an exception.

Parameters **blocking** (*bool*) – File blocked or not

Returns True if lock is acquired, else False

Return type *bool*

available ()

Returns True iff the file is currently available to be locked.

Returns True if lockfile is available

Return type *bool*

locked ()

Returns True iff the file is owned by THIS FileLock instance. (Even if this returns false, the file could be owned by another FileLock instance, possibly in a different thread or process).

Returns True if file owned by Filelock instance

Return type *bool*

purge ()

For debug purposes only. Removes the lock file from the hard disk.

release ()

Get rid of the lock by deleting the lockfile. When working in a *with* statement, this gets automatically called at the end.

Returns None

Return type None

owtf.lib.owtf_process module

Module contents

owtf.managers package

Submodules

owtf.managers.command_register module

owtf.managers.config module

owtf.managers.error module

owtf.managers.mapping module

owtf.managers.plugin module

owtf.managers.poutput module

owtf.managers.resource module

owtf.managers.session module

owtf.managers.target module

owtf.managers.transaction module

owtf.managers.url module

owtf.managers.worker module

owtf.managers.worklist module

Module contents

owtf.plugin package

Submodules

owtf.plugin.plugin_handler module

owtf.plugin.plugin_helper module

owtf.plugin.plugin_params module

owtf.plugin.scanner module

Module contents

owtf.plugin

owtf.protocols package

Submodules

owtf.protocols.smb module

owtf.protocols.smtp module

owtf.protocols.smtp

Description: This is the OWTF SMTP handler, to simplify sending emails.

Module contents

owtf.proxy package

Submodules

owtf.proxy.cache_handler module

owtf.proxy.cache_handler

Inbound Proxy Module developed by Bharadwaj Machiraju (blog.tunnelshade.in) as a part of Google Summer of Code 2013

class `owtf.proxy.cache_handler.CacheHandler` (*cache_dir, request, cookie_regex, blacklist*)
Bases: `object`

This class will be used by the request handler to either load or dump to cache. Main things that are done here :-
* The request_hash is generated here * The file locks are managed here * .rd files are created here

calculate_hash (*callback=None*)

Based on blacklist boolean the cookie regex is used for filtering of cookies in request_hash generation. However the original request is not tampered.

Parameters `callback` – Callback function

Returns

Return type

create_response_object ()

Create a proxy response object from cache file

Returns

Return type

dump (*response*)

This function takes in a HTTPResponse object and dumps the request and response data. It also creates a .rd file with same file name

Note: This is used by transaction logger

Parameters `response` – The proxy response

Returns

Return type

`load()`

This is the function which is called for every request. If file is not found in cache, then a file lock is created for that and a None is returned.

Returns Load a transaction from cache

Return type

class `owtf.proxy.cache_handler.DummyObject`

Bases: `object`

This class is just used to create a fake response object

`owtf.proxy.cache_handler.request_from_cache(file_path)`

A fake request object is created with necessary attributes

Parameters `file_path (str)` – The file path for the cache file

Returns

Return type

`owtf.proxy.cache_handler.response_from_cache(file_path)`

A fake response object is created with necessary attributes

Parameters `file_path (str)` – The file path for the cache file

Returns

Return type

owtf.proxy.gen_cert module

owtf.proxy.gen_cert

Inbound Proxy Module developed by Bharadwaj Machiraju (blog.tunnelshade.in) as a part of Google Summer of Code 2013

`owtf.proxy.gen_cert.gen_signed_cert(domain, ca_cert, ca_key, ca_pass, certs_folder)`

This function takes a domain name as a parameter and then creates a certificate and key with the domain name(replacing dots by underscores), finally signing the certificate using specified CA and returns the path of key and cert files. If you are yet to generate a CA then check the top comments

Parameters

- **domain** (*str*) – domain for the cert
- **ca_cert** (*str*) – ca.crt file path
- **ca_key** (*str*) – ca.key file path
- **ca_pass** (*str*) – Password for the certificate
- **certs_folder** (*str*) –

Returns Key and cert path

Return type *str*

owtf.proxy.main module

owtf.proxy.proxy module

owtf.proxy.socket_wrapper module

owtf.proxy.socket_wrapper

`owtf.proxy.socket_wrapper.starttls` (*socket, domain, ca_cert, ca_key, ca_pass, certs_folder, success=None, failure=None, io=None, **options*)

Wrap an active socket in an SSL socket.

Taken from <https://gist.github.com/weaver/293449/4d9f64652583611d267604531a1d5f8c32ac6b16>.

Parameters

- **socket** –
- **domain** –
- **ca_cert** –
- **ca_key** –
- **ca_pass** –
- **certs_folder** –
- **success** –
- **failure** –
- **io** –
- **options** –

Returns

Return type

owtf.proxy.tor_manager module

owtf.proxy.tor_manager

TOR manager module developed by Marios Kourtesis <name.surname@gmail.com>

class `owtf.proxy.tor_manager.TOR_manager` (*args*)

Bases: `object`

authenticate ()

This function is handling the authentication process to TOR control connection.

Returns

Return type

static is_tor_running ()

Check if tor is running

Returns True if running, else False

Return type *bool*

static msg_configure_tor()

static msg_start_tor(self)

open_connection()

Opens a new connection to TOR control

Returns

Return type

renew_ip()

Sends an NEWNYM message to TOR control in order to renew the IP address

Returns True if IP is renewed, else False

Return type *bool*

run()

Starts a new TOR_control_process which will renew the IP address.

Returns

Return type

tor_control_process()

This will run in a new process in order to renew the IP address after certain time.

Returns None

Return type None

owtf.proxy.transaction_logger module

Module contents

owtf.shell package

Submodules

owtf.shell.async_subprocess module

owtf.shell.blocking_shell module

owtf.shell.interactive_shell module

owtf.shell.pexpect_shell module

Module contents

owtf.utils package

Submodules

owtf.utils.app module

owtf.utils.app

```
class owtf.utils.app.Application(*args, **kwargs)
    Bases: tornado.web.Application
```

owtf.utils.commands module

owtf.utils.commands

```
owtf.utils.commands.get_command(argv)
    Format command to remove directory and space-separated arguments.

    Params list argv Arguments for the CLI.

    Returns Arguments without directory and space-separated arguments.

    Return type list
```

owtf.utils.error module

owtf.utils.error

The error handler provides a centralised control for aborting the application and logging errors for debugging later.

```
owtf.utils.error.abort_framework(message)
    Abort the OWTF framework.

    Warning If it happens really early and framework.core.Core has not been instantiated yet,
    sys.exit() is called with error code -1
```

Parameters `message` (*str*) – Descriptive message about the abort.

Returns full message explaining the abort.

Return type *str*

```
owtf.utils.error.user_abort(level, partial_output="")
    This function handles the next steps when a user presses Ctrl-C
```

Parameters

- **level** (*str*) – The level which was aborted
- **partial_output** (*str*) – Partial output generated by the command or plugin

Returns Message to present to the user

Return type *str*

```
owtf.utils.error.get_option_from_user(options)
    Give the user options to select
```

Parameters `options` (*str*) – Set of available options for the user

Returns The different options for the user to choose from

Return type *str*

```
class owtf.utils.error.SentryProxy(sentry_client)
    Bases: object

    Simple proxy for sentry client that logs to stderr even if no sentry client exists.

    capture_exception(exc_info=None, **kwargs)

owtf.utils.error.get_sentry_client(sentry_key="")
owtf.utils.error.log_and_exit_handler(signum, frame)
owtf.utils.error.setup_signal_handlers()
    Setup the handlers
```

owtf.utils.file module

owtf.utils.file

```
class owtf.utils.file.FileOperations
    Bases: object

    static codecs_open(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static create_missing_dirs(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static dump_file(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static make_dirs(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static mkdir(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static open(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

    static rm_tree(*args, **kwargs)
        Call the original function while checking for errors. If owtf_clean parameter is not explicitly passed or if it is set to True, it force OWTF to properly exit.

owtf.utils.file.catch_io_errors(func)
    Decorator on I/O functions. If an error is detected, force OWTF to quit properly.

owtf.utils.file.clean_temp_storage_dirs(owtf_pid)
    Rename older temporary directory to avoid any further confusions.

    Returns

    Return type None

owtf.utils.file.cleanup_target_dirs(target_url)
    Cleanup the directories for the specific target
```

Returns None

Return type None

`owtf.utils.file.create_output_dir_target(target_url)`

Creates output directories for the target URL

Parameters `target_url` (*str*) – The target URL

Returns None

Return type None

`owtf.utils.file.create_temp_storage_dirs(owtf_pid)`

Create a temporary directory in /tmp with pid suffix.

Returns

Return type None

`owtf.utils.file.directory_access(path, mode)`

Check if a directory can be accessed in the specified mode by the current user.

Parameters

- **path** (*str*) – Directory path.
- **mode** (*str*) – Access type.

Returns Valid access rights

Return type *str*

`owtf.utils.file.get_dir_worker_logs()`

Returns the output directory for the worker logs

Returns Path to output directory for the worker logs

Return type *str*

`owtf.utils.file.get_file_as_list(filename)`

Get file contents as a list

Parameters `filename` (*str*) – File path

Returns Output list of the content

Return type *list*

`owtf.utils.file.get_log_path(process_name)`

Get the log file path based on the process name :param process_name: Process name :type process_name: *str*
:return: Path to the specific log file :rtype: *str*

`owtf.utils.file.get_logs_dir()`

Get log directory by checking if abs or relative path is provided in config file

`owtf.utils.file.get_output_dir()`

Gets the output directory for the session

Returns The path to the output directory

Return type *str*

`owtf.utils.file.get_output_dir_target()`

Returns the output directory for the targets

Returns Path to output directory

Return type *str*

`owtf.utils.file.get_target_dir(target_url)`

Gets the specific directory for a target in the target output directory

Parameters `target_url` (*str*) – Target URL for which directory path is needed

Returns Path to the target URL specific directory

Return type *str*

owtf.utils.formatters module

owtf.utils.formatters

CLI string formatting

class `owtf.utils.formatters.ConsoleFormatter` (*fmt=None, datefmt=None*)

Bases: `logging.Formatter`

Custom formatter to show logging messages differently on Console

debug_fmt = '\x1b[92m[*] {} \x1b[0m'

error_fmt = '\x1b[91m[-] {} \x1b[0m'

format (*record*)

Choose format according to record level

Parameters `record` (*str*) – Record to format

Returns Formatted string

Return type *str*

info_fmt = '\x1b[94m[+] {} \x1b[0m'

warn_fmt = '\x1b[93m[!] {} \x1b[0m'

class `owtf.utils.formatters.FileFormatter` (**args, **kwargs*)

Bases: `logging.Formatter`

Custom formatter for log files

owtf.utils.http module

owtf.utils.http

`owtf.utils.http.deep_update(source, overrides)`

Update a nested dictionary or similar mapping.

Modify `source` in place.

Return type `collections.Mapping`

`owtf.utils.http.derive_http_method(method, data)`

Derives the HTTP method from Data, etc

Parameters

- **method** (*str*) – Method to check

- **data** (*str*) – Data to check

Returns Method found

Return type *str*

`owtf.utils.http.extract_method(wrapped_method)`

Gets original method if wrapped_method was decorated

Return type any([types.FunctionType, types.MethodType])

`owtf.utils.http.is_method(method)`

owtf.utils.ip module

owtf.utils.ip

`owtf.utils.ip.get_ip_from_hostname(hostname)`

Get IP from the hostname

Parameters **hostname** (*str*) – Target hostname

Returns IP address of the target hostname

Return type *str*

`owtf.utils.ip.get_ips_from_hostname(hostname)`

Get IPs from the hostname

Parameters **hostname** (*str*) – Target hostname

Returns IP addresses of the target hostname as a list

Return type *list*

`owtf.utils.ip.hostname_is_ip(hostname, ip)`

Test if the hostname is an IP.

Parameters

- **hostname** (*str*) – the hostname of the target.
- **ip** (*str*) – the IP (v4 or v6) of the target.

Returns True if the hostname is an IP, False otherwise.

Return type *bool*

`owtf.utils.ip.is_internal_ip(ip)`

Parses the input IP and checks if it is a private IP

Parameters **ip** (*str*) – IP address

Returns True if it is a private IP, otherwise False

Return type *bool*

owtf.utils.logger module

owtf.utils.logger

class owtf.utils.logger.OWTFLogger

Bases: object

disable_console_logging (**kwargs)

Disables console logging

Note: Must be called from inside the process because we should remove handler for that root logger. Since we add console handler in the last, we can remove the last handler to disable console logging

Parameters **kwargs** (*dict*) – Additional arguments to the logger

Returns

Return type None

enable_logging (**kwargs)

Enables both file and console logging

Note:

- process_name <– can be specified in kwargs
 - Must be called from inside the process because we are kind of overriding the root logger
-

Parameters **kwargs** (*dict*) – Additional arguments to the logger

Returns

Return type None

owtf.utils.process module

owtf.utils.process

owtf.utils.process.**check_pid** (*pid*)

Check whether pid exists in the current process table. UNIX only.

Parameters **pid** (*int*) – Pid to check

Returns True if pid exists, else false

Return type *bool*

owtf.utils.pycompat module

owtf.utils.pycompat

Helpers for compatibility between Python 2.x and 3.x.

owtf.utils.pycompat.**iteritems** (*d*, **kw)

owtf.utils.pycompat.**iterkeys** (*d*, **kw)

```
owtf.utils.pycompat.iterlists(d, **kw)
owtf.utils.pycompat.tervalues(d, **kw)
owtf.utils.pycompat.u(s)
```

owtf.utils.signals module

owtf.utils.signals

Most of it taken from the Flask code.

owtf.utils.strings module

owtf.utils.strings

```
owtf.utils.strings.add_to_dict(from_dict, to_dict)
Add the items from dict a with copy attribute to dict b
```

Parameters

- **from_dict** (*dict*) – Dict to copy from
- **to_dict** (*dict*) – Dict to copy to

Returns None

Return type None

```
owtf.utils.strings.gen_secure_random_str()
```

```
owtf.utils.strings.get_as_list(key_list)
Get values for keys in a list
```

Parameters **key_list** (*list*) – List of keys

Returns List of corresponding values

Return type *list*

```
owtf.utils.strings.get_header_list(key)
Get list from a string of values for a key
```

Parameters **key** (*str*) – Key

Returns List of values

Return type *list*

```
owtf.utils.strings.get_random_str(len)
Function returns random strings of length len
```

Parameters **len** (*int*) – Length

Returns Random generated string

Return type *str*

```
owtf.utils.strings.is_convertable(value, conv)
Convert a value
```

Parameters

- **value** –
- **conv** –

Returns

Return type

`owtf.utils.strings.list_to_dict_keys(list)`

Convert a list to dict with keys from list items

Parameters **list** (*list*) – list to convert

Returns The newly formed dictionary

Return type *dict*

`owtf.utils.strings.merge_dicts(a, b)`

Returns a by-value copy contained the merged content of the 2 passed dictionaries

Parameters

- **a** (*dict*) – Dict a
- **b** (*dict*) – Dict b

Returns New merge dict

Return type *dict*

`owtf.utils.strings.multi_replace(text, replace_dict)`

Recursive multiple replacement function :param text: Text to replace :type text: *str* :param replace_dict: The parameter dict to be replaced with :type replace_dict: *dict* :return: The modified text after replacement :rtype: *str*

`owtf.utils.strings.multi_replace_dict(text, replace_dict)`

Perform multiple replacements in one go using the replace dictionary in format: { 'search' : 'replace' }

Parameters

- **text** (*str*) – Text to replace
- **replace_dict** (*dict*) – The replacement strings in a dict

Returns *str*

Return type

`owtf.utils.strings.pad_key(key)`

Add delimiters.

Parameters **key** (*str*) – Key to pad

Returns Padded key string

Return type *str*

`owtf.utils.strings.paths_exist(path_list)`

Check if paths in the list exist

Parameters **path_list** (*list*) – The list of paths to check

Returns True if valid paths, else False

Return type *bool*

`owtf.utils.strings.remove_blanks_list(src)`

Removes empty elements from the list

Parameters **src** (*list*) – List

Returns New list without blanks

Return type *list*

`owtf.utils.strings.scrub_output(output)`

Remove all ANSI control sequences from the output

Parameters **output** (*str*) – Output to scrub

Returns Scrubbed output

Return type *str*

`owtf.utils.strings.str2bool(string)`

Converts a string to a boolean

Parameters **string** (*str*) – String to convert

Returns Boolean equivalent

Return type *bool*

`owtf.utils.strings.str_to_dict(string)`

Convert a string to a dict

Parameters **string** (*str*) – String to convert

Returns Resultant dict

Return type *dict*

`owtf.utils.strings.strip_key(key)`

Replaces key with empty space

Parameters **key** (*str*) – Key to clear

Returns Empty key

Return type *str*

`owtf.utils.strings.to_str(byte)`

`owtf.utils.strings.truncate_lines(str, num_lines, eol='\n')`

Truncate and remove EOL characters

Parameters

- **str** (*str*) – String to truncate
- **num_lines** (*int*) – Number of lines to process
- **EOL** (*char*) – EOL char

Returns Joined string after truncation

Return type *str*

`owtf.utils.strings.utf8(string)`

`owtf.utils.strings.wipe_bad_chars(filename)`

The function wipes bad characters from name of output file

Parameters **filename** (*str*) – The file name to scrub

Returns New replaced file filename

Return type *str*

owtf.utils.timer module

owtf.utils.timer

The time module allows the rest of the framework to time how long it takes for certain actions to execute and present this information in both seconds and human-readable form.

class owtf.utils.timer.**Timer** (*datetime_format*='%d/%m/%Y-%H:%M')

Bases: object

end_timer (*offset*='0')

Sets the end of the timer

Parameters *offset* (*str*) – Timer index

Returns

Return type None

static **get_current_date_time** ()

Current timestamp

Returns The current time as a timestamp

Return type *datetime*

get_current_date_time_as_str ()

Returns a datetime object as a string in a particular format

Returns Datetime object in string form

Return type *str*

get_elapsed_time (*offset*='0')

Gets the time elapsed between now and start of the timer in Unix epoch

Parameters *offset* (*str*) – Timer index

Returns Time difference

Return type *datetime*

get_elapsed_time_as_str (*offset*='0')

Returns the time elapsed a nice readable string

Parameters *offset* (*str*) – Timer index

Returns Time elapsed as a string

Return type *str*

get_end_date_time (*offset*='0')

Get the end time for the timer

Parameters *offset* (*str*) – Timer index

Returns End time for the timer as a timestamp

Return type *datetime*

get_end_date_time_as_str (*offset*='0')

Get the end time for the timer as a string

Parameters *offset* (*str*) – Timer index

Returns End time for the timer as a string

Return type *str*

get_start_date_time (*offset*='0')

Get the start time for the timer

Parameters **offset** (*str*) – Timer index

Returns Start time for the timer as a timestamp

Return type *datetime*

get_start_date_time_as_str (*offset*='0')

Get the start time for the timer as a string

Parameters **offset** (*str*) – Timer index

Returns Start time for the timer as a string

Return type *str*

get_time_as_str (*timedelta*)

Get the time difference as a human readable string

Parameters **timedelta** (*datetime.timedelta*) – Time difference

Returns Human readable form for the timedelta

Return type *str*

get_time_human (*seconds_str*)

Generates the human readable string for the timestamp

Parameters **seconds_str** (*str*) – Unix style timestamp

Returns Timestamp in a human readable string

Return type *str*

start_timer (*offset*='0')

Adds a start time to the timer

Parameters **offset** (*str*) – Timer index

Returns The start time for the timer

Return type *datetime*

timers = {}

Module contents

2.1.2 Submodules

2.1.3 owtf.config module

owtf.config

The Configuration object parses all configuration files, loads them into memory, derives some settings and provides framework modules with a central repository to get info.

2.1.4 owtf.constants module

owtf.constants

Ranking constants used across the framework.

2.1.5 owtf.core module

2.1.6 owtf.settings module

owtf.settings

It contains all the owtf global configs.

2.1.7 Module contents

3.1 Database Configuration

3.1.1 Basic Setup

The connection settings for postgres database are present in `~/.owtf/db.yaml` or `owtf/settings.py`.

```
DATABASE_IP: 127.0.0.1
DATABASE_PORT: 5432
DATABASE_NAME: owtfdb
DATABASE_USER: owtf_db_user
DATABASE_PASS: random_password
```

3.2 Framework Configuration (Optional)

Some basic settings like, where should the interface server listen etc.. can be controlled from a config file present at `~/.owtf/conf/framework.yaml`. All the default values are ready by default.

4.1 Starting OWTF

Warning: Before starting OWTF, make sure you have the postgres database server running. This can be easily ensured by using `scripts/db_run.sh`

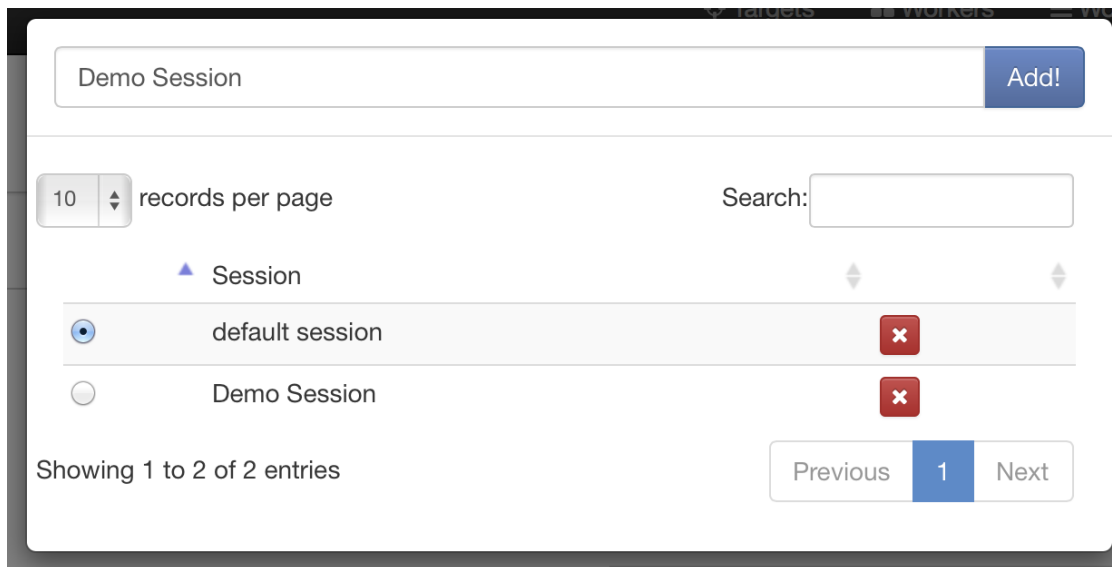
OWTF is controlled and used from a web interface, so you have to launch OWTF from command line and then move on to your favourite browser. OWTF can be launched by

```
./owtf.py
```

The interface url is printed onto the console, so that you can directly click on it

4.2 Using Sessions

In order to keep things simple and separate, OWTF provides support for sessions. A session is your classification of targets. You can have the same target in multiple sessions.



4.3 Managing Targets

The targets page also known as the target manager presents a ton of information. It has three important features

- A textarea to add new targets
- A targets table to go search through targets
- A session manager to manage sessions
- A button to launch plugins against targets
- A button to export targets to a text file - helpful when you have a large number of targets in scope

4.3.1 New Targets

Just add the urls seperated by a new line & press the button to add targets

4.3.2 Remove Targets

To present the information in an orderly fashion, all targets are shown in the form of a table. The labels beside the target name shows the severity of any vulnerability discovered either by OWTF or by user (yes, user can have his own rankings)

4.4 Understanding Plugins

4.4.1 Types of Plugins

There are loads of plugins available in OWTF, but what is interesting is their categorization. All the plugins are categorized into multiple groups and types

- **WEB**

Add Targets



```
http://testhtml5.vulnweb.com
http://vicnum.ciphertechs.com
http://www.webscantest.com
http://blasze.com/xsstestsuite/
http://zero.webappsecurity.com
http://testphp.vulnweb.com
http://testaspnet.vulnweb.com
http://testasp.vulnweb.com
http://demo.testfire.net|
http://hackademic1.teilar.gr
```

Add Targets!

Fig. 1: Multiple targets can be added at once

10

records per page

Search: vulnweb

<input type="checkbox"/>	Target		Actions
<input type="checkbox"/>	http://testphp.vulnweb.com/ (176.28.50.165)	Low	<div>— ×</div>
<input type="checkbox"/>	http://testasp.vulnweb.com/ (87.230.29.167)	Medium	<div>— ×</div>
<input type="checkbox"/>	http://testhtml5.vulnweb.com/#/latest (176.28.50.165)	Info	<div>— ×</div>
<input type="checkbox"/>	http://testaspnet.vulnweb.com/ (87.230.29.167)	Info	<div>— ×</div>

Showing 1 to 4 of 4 entries (filtered from 11 total entries)

Previous

1

Next

Fig. 2: All the targets in the present session are shown in the targets table. A search box can be used to search among the targets

- active
- external
- grep
- passive
- semi-passive
- **NET**
 - active
 - bruteforce
- **AUX**
 - se
 - exploit etc. . .

4.4.2 Launching Plugins

Plugins can be launched from the targets table or from the individual target report. In order to launch plugins against multiple targets, select the targets from the target manager and launch plugins

<input type="checkbox"/>	OWTF-IG-005	Application Discovery	active	web	Active probing for app discovery
<input type="checkbox"/>	OWTF-WVS-001	Arachni Unauthenticated	active	web	Active Vulnerability Scanning without credentials via Arachni
<input type="checkbox"/>	OWTF-CM-008	HTTP Methods and XST	active	web	Active probing for HTTP methods
<input type="checkbox"/>	OWTF-CM-003	Infrastructure Configuration Management	active	web	Active Probing for fingerprint analysis
<input type="checkbox"/>	OWTF-WVS-002	Nikto Unauthenticated	active	web	Active Vulnerability Scanning without credentials via nikto
<input type="checkbox"/>	OWTF-CM-006	Old Backup and Unreferenced Files	active	web	Active probing for juicy files (DirBuster)
<input type="checkbox"/>	OWTF-WVS-006	Skipfish Unauthenticated	active	web	Active Vulnerability Scanning without credentials via Skipfish
<input type="checkbox"/>	OWTF-CM-001	Testing for SSL-TLS	active	web	Active probing for SSL configuration
<input type="checkbox"/>	OWTF-WSP-001	Visit URLs	active	web	Visit URLs found by other tools, some could be sensitive: need permission
<input type="checkbox"/>	OWTF-WVS-004	W3AF Unauthenticated	active	web	Active Vulnerability Scanning without credentials via w3af
<input type="checkbox"/>	OWTF-WVS-003	Wapiti Unauthenticated	active	web	Active Vulnerability Scanning without credentials via Wapiti
<input type="checkbox"/>	OWTF-IG-004	Web Application Fingerprint	active	web	Active probing for fingerprint analysis
<input type="checkbox"/>	OWTF-WVS-005	Websecurify Unauthenticated	active	web	Active Vulnerability Scanning without credentials via Websecurify

Fig. 3: To know more about any plugin, read the help text present in the last column of plugin launcher

The screenshot shows the MacinOWTF interface. At the top, there is a 'default session' button and a 'Run Plugins' button. Below these, there is a 'records per page' dropdown set to '10' and a search bar containing 'vulnweb'. The main part of the interface is a table with two columns: 'Target' and 'Actions'. The table contains four rows of targets, each with a checkbox, a URL, a severity level, and action buttons. Below the table, it says 'Showing 1 to 4 of 4 entries (filtered from 11 total entries)' and a pagination bar with 'Previous', '1', and 'Next' buttons.

<input checked="" type="checkbox"/>	Target	Actions
<input checked="" type="checkbox"/>	http://testphp.vulnweb.com/ (176.28.50.165) Low	<input type="button" value="-"/> <input type="button" value="x"/>
<input checked="" type="checkbox"/>	http://testasp.vulnweb.com/ (87.230.29.167) Medium	<input type="button" value="-"/> <input type="button" value="x"/>
<input checked="" type="checkbox"/>	http://testhtml5.vulnweb.com/#/latest (176.28.50.165) Info	<input type="button" value="-"/> <input type="button" value="x"/>
<input checked="" type="checkbox"/>	http://testaspnet.vulnweb.com/ (87.230.29.167) Info	<input type="button" value="-"/> <input type="button" value="x"/>

Showing 1 to 4 of 4 entries (filtered from 11 total entries)

Previous 1 Next

Fig. 4: Multi select targets to launch plugins against them

4.5 Analyzing results

After the execution of plugins, you can navigate to the individual target report to go through the results of the plugins executed for that target. The report looks like this

Individual aspects for going through the report

4.5.1 Understanding plugin report

For better organization, all plugins of the same test code are grouped together. When you open a plugin report and click on a test code, you get to see the related plugins that are run for that target

Each test group has an expandable report. The text of the link consists of there parts

- Code of the test group as per the mapping (Eg: **OWTF-CM-008**)
- Name of the test group as per the mapping (Eg: **HTTP Methods and XST**)
- Pentester translations for the code (Eg: **PUT,TRACE, WebDAV etc..**)

Now if you proceed to select a plugin type, you can see the corresponding report

The details presented in a plugin report are:

- Run time of the plugin
- Time interval during which it was running

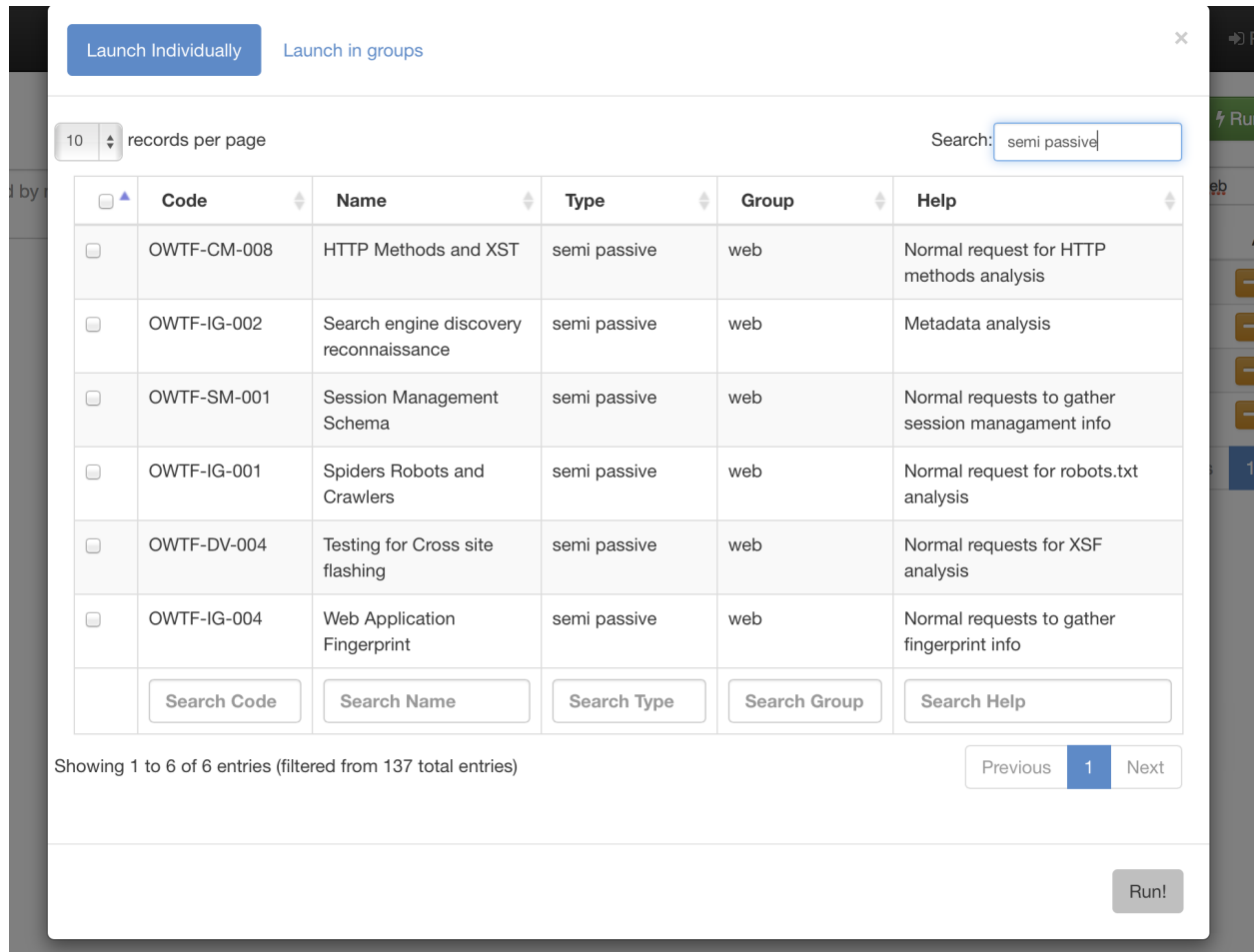


Fig. 5: Search and select plugins individually when needed

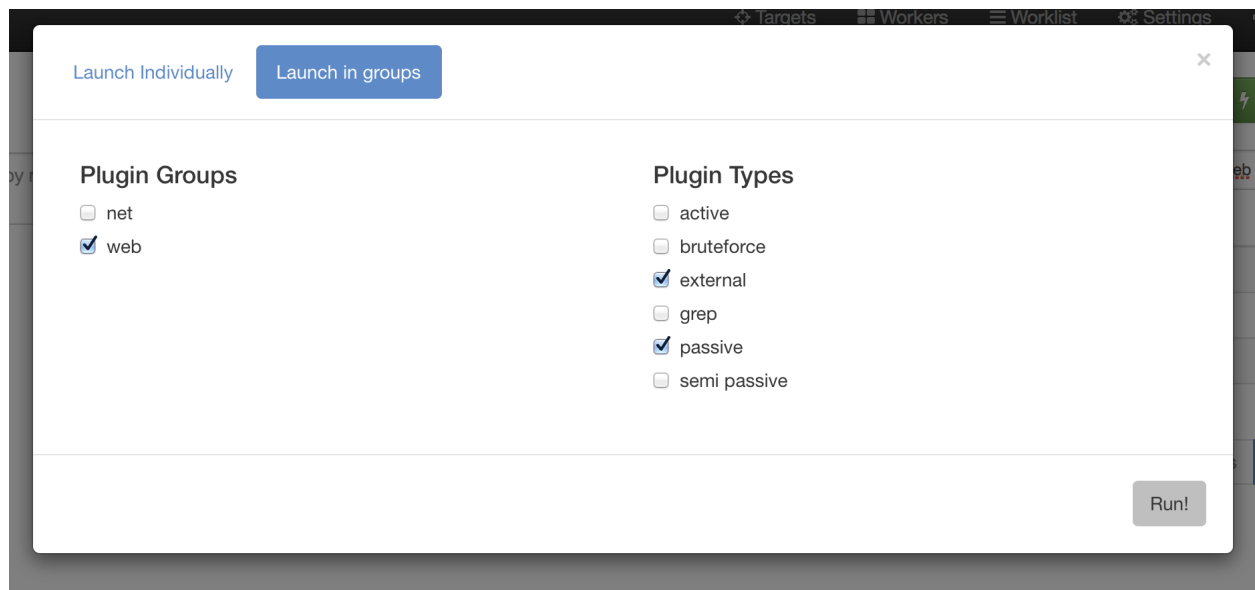


Fig. 6: Select plugins in groups when needed

<http://crackme.cenzic.com/Kelev/view/home.php>

(68.233.193.133)

Critical

Filter
Refresh
Run Plugins
User Sessions
Logs

OWTF-AJ-001 Testing for AJAX Vulnerabilities

OWTF-AJ-002 Testing for AJAX

OWTF-AT-001 Testing for Credentials Transport Passwords in clear-text

OWTF-AT-002 Testing for User Enumeration User Enumeration

OWTF-AT-003 Default or Guessable User Account Default accounts

Fig. 7: Target report

OWTF-CM-008 HTTP Methods and XST PUT, TRACE, WebDAV, etc

Type: external passive semi passive

Semi passive
— OWTF-CM-008

RUNTIME	TIME INTERVAL	STATUS	OUTPUT FILES	ACTIONS
5s, 592ms	19/09/2014-21:06 19/09/2014-21:06	Successful	Browse	

Notes

MORE DETAILS

HTTP Transactions

Request

Response

```

OPTIONS http://demo.testfire.net HTTP/1.1
Host: demo.testfire.net
Accept-Encoding: identity
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:6.0) Gecko/20100101 Firefox/15.0

```

```

200 OK
Content-Length: 0
X-Powered-By: ASP.NET
X-Http-Reason: OK
Server: Microsoft-IIS/6.0
Allow: OPTIONS, TRACE, GET, HEAD
Date: Fri, 19 Sep 2014 18:08:10 GMT
Public: OPTIONS, TRACE, GET, HEAD, POST

```

- Status of the plugin (i.e if it was aborted by user etc..)
- A button to rerun the plugin
- A button to delete the plugin output
- A button to add notes
- Actual plugin output

If you click on the **Browse** button, then any file saved by the plugin can be seen

Index of

- [../](#)
- [curl OPTIONS Check 1.txt](#)
- [curl OPTIONS Check 2.txt](#)

Index of

- [../](#)
- [Arachni.txt](#)
- [arachni_report.txt](#)
- [arachni_report2014-09-11 01 14 22.afr](#)
- [arachni_report2014-09-11 01 14 22.html](#)
- [arachni_report2014-09-11 01 14 22.txt](#)
- [arachni_report2014-09-11 01 14 22.xml](#)

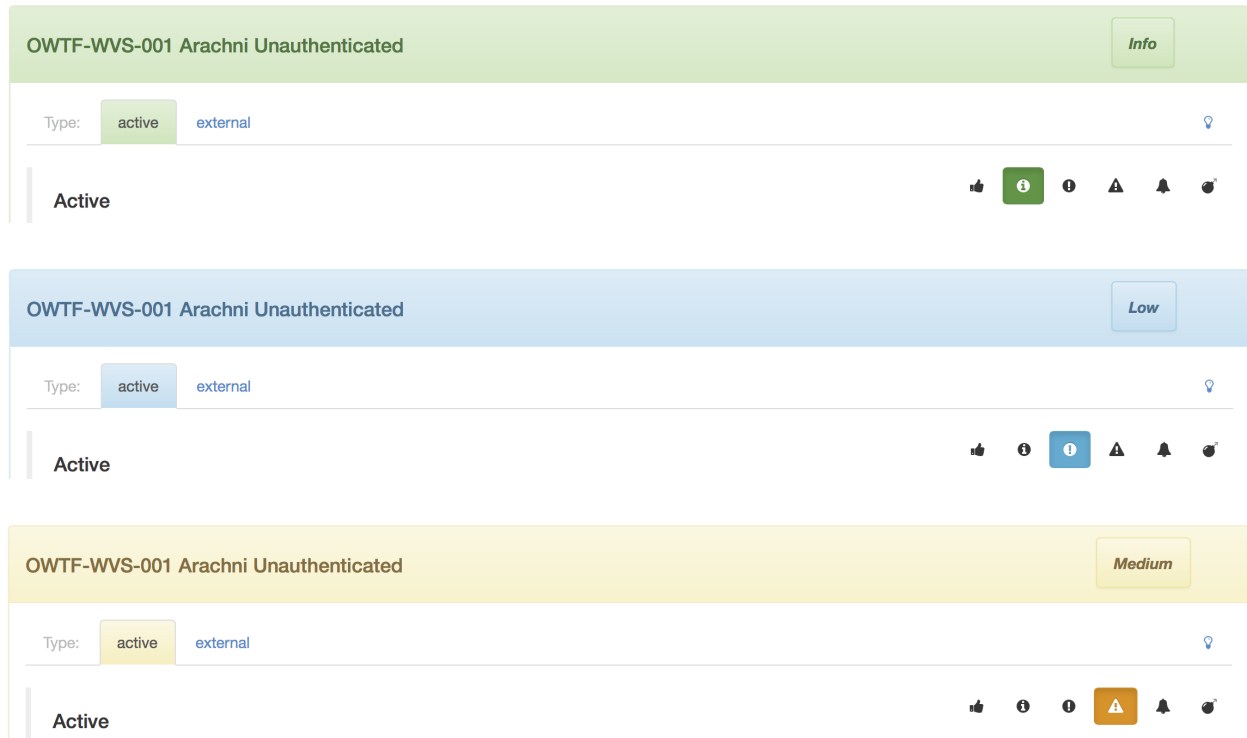
Fig. 8: Files of Arachni active plugin

4.5.2 Saving your analysis

Once you start analyzing the plugin results, there is a need for ranking those findings along with saving some necessary information if needed. OWTF has both these features

Manual Ranking

In order to rank a plugin output, you can use the ranking buttons based on severity



Notes

Ranking is not the only thing, you can also write and save notes as well. Click on the **NOTES** button to open an editor and once you are done, click on the same button to save and close the editor

4.5.3 Advanced Filter

Advanced filter is used to filter the plugin results. Click on the **FILTER** button in the target report and you are good to go

As it can be seen from above image, you can filter the plugin outputs based on multiple criteria. You can even change the mapping of the results. Let us try the latest OWASP v4

4.5.4 Transaction Log

All the transactions that ever happened through the OWTF proxy can be searched through transaction log. You can search in multiple fields. A sample look of the transaction log is in

OWTF-WVS-001 Arachni Unauthenticated **Critical**

Type: active external 💡

Active 👍 ⓘ ⚠️ 🔔 📌

RUNTIME	TIME INTERVAL	STATUS	DELETE
0s, 20ms	11/09/2014-16:45 11/09/2014-16:45	Successful	

A screenshot of a web browser window. The address bar shows the URL 'http://sometarget.com/workarea/login.aspx'. The page content displays the 'ektron' logo at the top, followed by two input fields labeled 'User:' and 'Pwd:'. Below these fields is a prominent 'LOGIN' button. The background of the login page is decorated with a repeating pattern of various icons. The browser's interface, including the toolbar and status bar, is visible around the page content.

Advanced Filter

Status

☐ Aborted
 ☐ Aborted (by user)
 ☐ Successful

Plugin group

☐ web

Mapping

☐ NIST
 ☐ OWASP_V3
 ☐ OWASP_V4

Owtf rank

☐ -1

User rank

☐ -1
 ☐ 1
 ☐ 2
 ☐ 3
 ☐ 4
 ☐ 5

Plugin type

☐ active
 ☐ external
 ☐ grep
 ☐ passive
 ☐ semi_passive

Clear Filters!

OTG-INPVAL-017 Testing for HTTP Splitting/Smuggling

OTF-INFO-003 Review Webserver Metafiles for Information Leakage robots.txt Analysis

OTG-INFO-001 Conduct Search Engine Discovery and Reconnaissance Google Hacking, Metadata

Info

OTG-INFO-006 Identify application entry points Crawling

High

OTG-INFO-002 Fingerprint Web Server What is that site running?









Medium

OTG-INFO-004 Enumerate Applications on Webserver Port Scanning, Whois

OTG-ERR-001 Analysis of Error Codes Error Messages

the image below.

10 records per page

Link	Time	OPTIONS	Status	URL
 	0s, 831ms	OPTIONS	200 OK	http://crackme.cenzic.com/Kelev/view/
 	0s, 412ms	OPTIONS	200 OK	http://crackme.cenzic.com/Kelev/view/home.php
 	0s, 417ms	OPTIONS	200 OK	http://crackme.cenzic.com/Kelev/
 	0s, 609ms	OPTIONS	200 OK	http://crackme.cenzic.com/Kelev/php/

Showing 1 to 4 of 4 entries (filtered from 2,055 total entries)

There are two ways in which individual transactions can be viewed

- Each transaction in new tab
- Transaction in a modal window

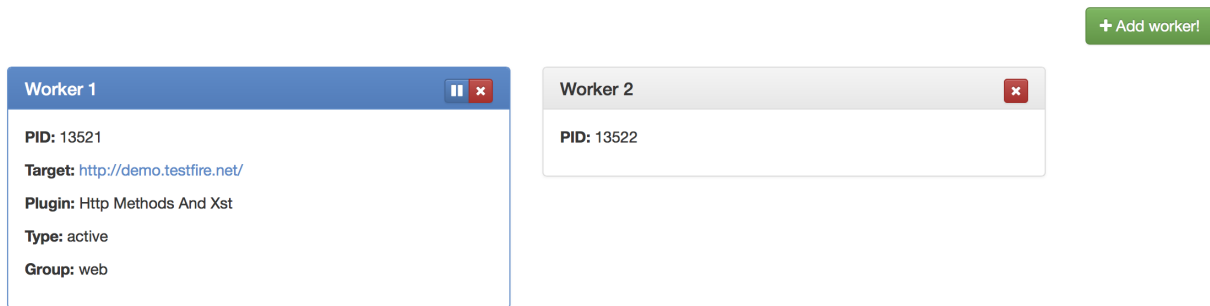
Clicking on the info button will open a model window which allows you to navigate back & forth between the filtered transactions. The search words are highlighted as well.



4.6 Managing Workers

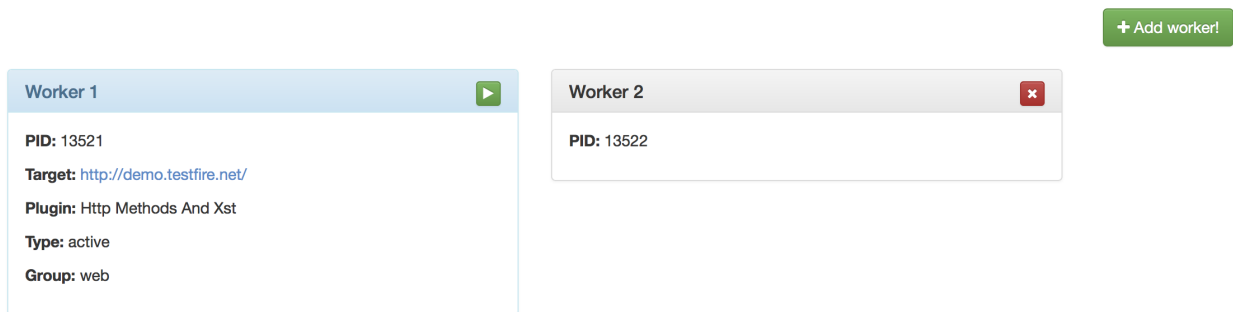
Workers are the actual processes that run the plugins. Control over these worker processes is provided from the worker manager page.

There are three main controls in the worker manager:



4.6.1 Pausing/Resuming Workers

You can **pause/resume** all the workers at the same time or pause them individually through the workers page. We care a lot about your time. If your Internet connection down or if any target is not responding and your web vulnerability scanner plugin is halfway through? Don't worry, we got your back. All you have to do is pause the worker and resume it when the target is back up. Isn't this l33t?



4.6.2 Abort Workers

You can **abort** any worker. If you wish to abort any plugin during execution, just click on the red cross. Do the same if you wish to remove an extra idle worker.

4.6.3 Add Workers

You can **add new workers** on the fly if you have many targets and are running many plugins simultaneously.

Warning: Maximum of one plugin per target will be running at any moment in time

The screenshot shows the OWASP OWTF interface with a top navigation bar containing links for Targets, Workers, Worklist, Settings, PlugnHack, and Help. A green button labeled '+ Add worker!' is in the top right. Below the navigation bar, four worker panels are displayed, each with a title bar (Worker 1 to Worker 4) and a close button. Each panel contains the following information:

- Worker 1:** PID: 3540, Target: <http://zero.webappsecurity.com/>, Plugin: Testing For Dos Locking Customer Accounts, Type: external, Group: web.
- Worker 2:** PID: 3542, Target: <http://www.webscantest.com/>, Plugin: Testing For Captcha, Type: external, Group: web.
- Worker 3:** PID: 5108, Target: <http://biasze.com/xsstestsuite/>, Plugin: Storing Too Much Data In Session, Type: external, Group: web.
- Worker 4:** PID: 5688, Target: <http://testaspnet.vulnweb.com/>, Plugin: Bypassing Authorization Schema, Type: external, Group: web.

4.7 Controlling Worklist

work When any plugin is launched against a target, it adds a (plugin, target) combination to the worklist. This combination is known as work.

worklist The list consisting of all work which are yet to be assigned to a worker.

Worklist can be managed from the worklist manager which looks like this

The screenshot shows the OWASP OWTF Worklist Manager interface. At the top right, there are buttons for 'Pause All' and 'Resume All'. Below these, there is a dropdown menu for 'records per page' set to '10'. The main part of the interface is a table with the following columns: Est. Time (min), Actions, Target, Plugin Group, Plugin Type, and Plugin Name. The table contains three entries, all with the target <http://demo.testfire.net/> and the plugin group 'web'.

Est. Time (min)	Actions	Target	Plugin Group	Plugin Type	Plugin Name
0s, 36ms		http://demo.testfire.net/	web	passive	HTTP Methods and XST
7s, 354ms		http://demo.testfire.net/	web	semi passive	HTTP Methods and XST
0s, 8ms		http://demo.testfire.net/	web	external	HTTP Methods and XST

Below the table, it says 'Showing 1 to 3 of 3 entries'. At the bottom right, there are buttons for 'Previous', '1' (selected), and 'Next'.

Worklist table provides interesting information like:

- Estimated time for which the plugin will run
- All details about the plugin and the target against which it is launched

4.7.1 Pausing Work





















Individual works or whole worklist can be paused. This will stop the work from getting assigned to any worker. The interesting part is `worklist` is persistent, i.e. if you pause the whole worklist and exit OWTF, the works will still be there in paused state when you start OWTF again.

OWASP OWTF

TargetsWorkersWorklistSettingsPlugnHackHelp

10 records per page

Pause AllResume All

Est. Time (min)	Actions	Target	Plugin Group	Plugin Type	Plugin Name
0s, 23ms	 	http://zero.webappsecurity.com/	web	external	Cookies attributes
0s, 26ms	 	http://zero.webappsecurity.com/	web	external	DOM based Cross Site Scripting
	 	http://zero.webappsecurity.com/	web	external	Exposed Session Variables
	 	http://zero.webappsecurity.com/	web	external	HTTP GET parameters REST Testing
	 	http://zero.webappsecurity.com/	web	external	HTTP Methods and XST
	 	http://zero.webappsecurity.com/	web	external	How to test AJAX
	 	http://zero.webappsecurity.com/	web	external	IMAP SMTP Injection
	 	http://zero.webappsecurity.com/	web	external	Identify application entry points
	 	http://zero.webappsecurity.com/	web	external	Infrastructure Configuration Management
	 	http://zero.webappsecurity.com/	web	external	Logout and Browser Cache Management

Showing 1 to 10 of 696 entries

Previous1234570Next

4.7.2 Deleting Work

Any work can be deleted from the worklist. The search boxes will help in filtering of the works when there are many entries.

- Unable to install **pycurl** library, getting **main.ConfigurationError: Could not run curl-config?**

Luckily, we have faced this issue. If you ran the install script and still got this error, you can let us know. If not, check [this issue](#) on how to fix it.

- Unable to run OWTF because of **ImportError: No module named cryptography.hazmat.bindings.openssl.binding?**

This actually means you do not have cryptography python module installed. It is recommended to rerun the install script (or) to just install the missing python libraries using the following command.

```
pip2 install --upgrade -r install/owtf.pip
```

- Unable to run OWTF because of **TypeError: parse_requirements() missing 1 required keyword argument: 'session'**

This is because of an older version of pip installed in your System. To resolve this run the following commands

```
pip install --upgrade pip (run as root if required)
python install/install.py
```


CHAPTER 6

Want Help or Request a feature?

There are many ways in which you can reach the OWTF Team

- [IRC channel \(irc.freenode.net\)](#)
- [Github Issue Tracker](#)
- [User mailing list](#)
- [Developers mailing list](#)

O

- [owtf](#), 27
 - [owtf.api](#), 6
 - [owtf.api.handlers](#), 6
 - [owtf.api.utils](#), 6
 - [owtf.config](#), 26
 - [owtf.constants](#), 27
 - [owtf.db](#), 7
 - [owtf.lib](#), 10
 - [owtf.lib.cli_options](#), 7
 - [owtf.lib.exceptions](#), 8
 - [owtf.lib.filelock](#), 9
 - [owtf.managers](#), 11
 - [owtf.plugin](#), 11
 - [owtf.protocols](#), 12
 - [owtf.protocols.smtp](#), 12
 - [owtf.proxy](#), 15
 - [owtf.proxy.cache_handler](#), 12
 - [owtf.proxy.gen_cert](#), 13
 - [owtf.proxy.socket_wrapper](#), 14
 - [owtf.proxy.tor_manager](#), 14
 - [owtf.settings](#), 27
 - [owtf.shell](#), 15
 - [owtf.utils](#), 26
 - [owtf.utils.app](#), 15
 - [owtf.utils.commands](#), 16
 - [owtf.utils.error](#), 16
 - [owtf.utils.file](#), 17
 - [owtf.utils.formatters](#), 19
 - [owtf.utils.http](#), 19
 - [owtf.utils.ip](#), 20
 - [owtf.utils.logger](#), 20
 - [owtf.utils.process](#), 21
 - [owtf.utils.pycompat](#), 21
 - [owtf.utils.signals](#), 22
 - [owtf.utils.strings](#), 22
 - [owtf.utils.timer](#), 25

A

`abort_framework()` (in module `owtf.utils.error`), 16
`acquire()` (`owtf.lib.filelock.FileLock` method), 10
`add_to_dict()` (in module `owtf.utils.strings`), 22
`api_assert()` (in module `owtf.lib.exceptions`), 9
`APIError`, 8
`Application` (class in `owtf.utils.app`), 16
`authenticate()` (`owtf.proxy.tor_manager.TOR_manager` method), 14
`available()` (`owtf.lib.filelock.FileLock` method), 10

C

`CacheHandler` (class in `owtf.proxy.cache_handler`), 12
`calculate_hash()` (`owtf.proxy.cache_handler.CacheHandler` method), 12
`capture_exception()` (`owtf.utils.error.SentryProxy` method), 17
`catch_io_errors()` (in module `owtf.utils.file`), 17
`check_pid()` (in module `owtf.utils.process`), 21
`clean_temp_storage_dirs()` (in module `owtf.utils.file`), 17
`cleanup_target_dirs()` (in module `owtf.utils.file`), 17
`codecs_open()` (`owtf.utils.file.FileOperations` static method), 17
`ConsoleFormatter` (class in `owtf.utils.formatters`), 19
`create_missing_dirs()` (`owtf.utils.file.FileOperations` static method), 17
`create_output_dir_target()` (in module `owtf.utils.file`), 18
`create_response_object()` (`owtf.proxy.cache_handler.CacheHandler` method), 12
`create_temp_storage_dirs()` (in module `owtf.utils.file`), 18

D

`DatabaseNotRunningException`, 8
`DBIntegrityException`, 8
`debug_fmt` (`owtf.utils.formatters.ConsoleFormatter` attribute), 19
`deep_update()` (in module `owtf.utils.http`), 19
`derive_http_method()` (in module `owtf.utils.http`), 19
`directory_access()` (in module `owtf.utils.file`), 18
`disable_console_logging()` (`owtf.utils.logger.OWTFLogger` method), 21
`DummyObject` (class in `owtf.proxy.cache_handler`), 13
`dump()` (`owtf.proxy.cache_handler.CacheHandler` method), 12
`dump_file()` (`owtf.utils.file.FileOperations` static method), 17

E

`enable_logging()` (`owtf.utils.logger.OWTFLogger` method), 21
`end_timer()` (`owtf.utils.timer.Timer` method), 25
`error_fmt` (`owtf.utils.formatters.ConsoleFormatter` attribute), 19
`extract_method()` (in module `owtf.utils.http`), 20

F

`FileFormatter` (class in `owtf.utils.formatters`), 19
`FileLock` (class in `owtf.lib.filelock`), 10
`FileLock.FileLockException`, 10
`FileOperations` (class in `owtf.utils.file`), 17
`format()` (`owtf.utils.formatters.ConsoleFormatter` method), 19
`FrameworkAbortException`, 8
`FrameworkException`, 8

G

`gen_secure_random_str()` (in module `owtf.utils.strings`), 22

[gen_signed_cert\(\)](#) (in module [owtf.proxy.gen_cert](#)), 13
[get_as_list\(\)](#) (in module [owtf.utils.strings](#)), 22
[get_command\(\)](#) (in module [owtf.utils.commands](#)), 16
[get_current_date_time\(\)](#) ([owtf.utils.timer.Timer](#) static method), 25
[get_current_date_time_as_str\(\)](#) ([owtf.utils.timer.Timer](#) method), 25
[get_dir_worker_logs\(\)](#) (in module [owtf.utils.file](#)), 18
[get_elapsed_time\(\)](#) ([owtf.utils.timer.Timer](#) method), 25
[get_elapsed_time_as_str\(\)](#) ([owtf.utils.timer.Timer](#) method), 25
[get_end_date_time\(\)](#) ([owtf.utils.timer.Timer](#) method), 25
[get_end_date_time_as_str\(\)](#) ([owtf.utils.timer.Timer](#) method), 25
[get_file_as_list\(\)](#) (in module [owtf.utils.file](#)), 18
[get_header_list\(\)](#) (in module [owtf.utils.strings](#)), 22
[get_ip_from_hostname\(\)](#) (in module [owtf.utils.ip](#)), 20
[get_ips_from_hostname\(\)](#) (in module [owtf.utils.ip](#)), 20
[get_log_path\(\)](#) (in module [owtf.utils.file](#)), 18
[get_logs_dir\(\)](#) (in module [owtf.utils.file](#)), 18
[get_option_from_user\(\)](#) (in module [owtf.utils.error](#)), 16
[get_output_dir\(\)](#) (in module [owtf.utils.file](#)), 18
[get_output_dir_target\(\)](#) (in module [owtf.utils.file](#)), 18
[get_random_str\(\)](#) (in module [owtf.utils.strings](#)), 22
[get_sentry_client\(\)](#) (in module [owtf.utils.error](#)), 17
[get_start_date_time\(\)](#) ([owtf.utils.timer.Timer](#) method), 26
[get_start_date_time_as_str\(\)](#) ([owtf.utils.timer.Timer](#) method), 26
[get_target_dir\(\)](#) (in module [owtf.utils.file](#)), 19
[get_time_as_str\(\)](#) ([owtf.utils.timer.Timer](#) method), 26
[get_time_human\(\)](#) ([owtf.utils.timer.Timer](#) method), 26

H

[hostname_is_ip\(\)](#) (in module [owtf.utils.ip](#)), 20

I

[info_fmt](#) ([owtf.utils.formatters.ConsoleFormatter](#) attribute), 19
[InvalidActionReference](#), 8
[InvalidConfigurationReference](#), 8
[InvalidErrorReference](#), 8

[InvalidMappingReference](#), 8
[InvalidMessageReference](#), 8
[InvalidParameterType](#), 8
[InvalidSessionReference](#), 8
[InvalidTargetReference](#), 8
[InvalidTransactionReference](#), 8
[InvalidUrlReference](#), 8
[InvalidWorkerReference](#), 9
[InvalidWorkReference](#), 9
[is_convertable\(\)](#) (in module [owtf.utils.strings](#)), 22
[is_internal_ip\(\)](#) (in module [owtf.utils.ip](#)), 20
[is_method\(\)](#) (in module [owtf.utils.http](#)), 20
[is_tor_running\(\)](#) ([owtf.proxy.tor_manager.TOR_manager](#) static method), 14
[iteritems\(\)](#) (in module [owtf.utils.pycompat](#)), 21
[iterkeys\(\)](#) (in module [owtf.utils.pycompat](#)), 21
[iterlists\(\)](#) (in module [owtf.utils.pycompat](#)), 21
[intervalues\(\)](#) (in module [owtf.utils.pycompat](#)), 22

L

[list_to_dict_keys\(\)](#) (in module [owtf.utils.strings](#)), 23
[load\(\)](#) ([owtf.proxy.cache_handler.CacheHandler](#) method), 13
[locked\(\)](#) ([owtf.lib.filelock.FileLock](#) method), 10
[log_and_exit_handler\(\)](#) (in module [owtf.utils.error](#)), 17

M

[make_dirs\(\)](#) ([owtf.utils.file.FileOperations](#) static method), 17
[match\(\)](#) ([owtf.api.utils.VersionMatches](#) method), 6
[merge_dicts\(\)](#) (in module [owtf.utils.strings](#)), 23
[mkdir\(\)](#) ([owtf.utils.file.FileOperations](#) static method), 17
[msg_configure_tor\(\)](#) ([owtf.proxy.tor_manager.TOR_manager](#) static method), 14
[msg_start_tor\(\)](#) ([owtf.proxy.tor_manager.TOR_manager](#) static method), 15
[multi_replace\(\)](#) (in module [owtf.utils.strings](#)), 23
[multi_replace_dict\(\)](#) (in module [owtf.utils.strings](#)), 23

O

[open\(\)](#) ([owtf.utils.file.FileOperations](#) static method), 17
[open_connection\(\)](#) ([owtf.proxy.tor_manager.TOR_manager](#) method), 15
[owtf](#) (module), 27
[owtf.api](#) (module), 6
[owtf.api.handlers](#) (module), 6
[owtf.api.utils](#) (module), 6
[owtf.config](#) (module), 26

owtf.constants (module), 27
 owtf.db (module), 7
 owtf.lib (module), 10
 owtf.lib.cli_options (module), 7
 owtf.lib.exceptions (module), 8
 owtf.lib.filelock (module), 9
 owtf.managers (module), 11
 owtf.plugin (module), 11
 owtf.protocols (module), 12
 owtf.protocols.smtp (module), 12
 owtf.proxy (module), 15
 owtf.proxy.cache_handler (module), 12
 owtf.proxy.gen_cert (module), 13
 owtf.proxy.socket_wrapper (module), 14
 owtf.proxy.tor_manager (module), 14
 owtf.settings (module), 27
 owtf.shell (module), 15
 owtf.utils (module), 26
 owtf.utils.app (module), 15
 owtf.utils.commands (module), 16
 owtf.utils.error (module), 16
 owtf.utils.file (module), 17
 owtf.utils.formatters (module), 19
 owtf.utils.http (module), 19
 owtf.utils.ip (module), 20
 owtf.utils.logger (module), 20
 owtf.utils.process (module), 21
 owtf.utils.pycompat (module), 21
 owtf.utils.signals (module), 22
 owtf.utils.strings (module), 22
 owtf.utils.timer (module), 25
 OWTFLogger (class in owtf.utils.logger), 21

P

pad_key () (in module owtf.utils.strings), 23
 parse_options () (in module owtf.lib.cli_options), 7
 paths_exist () (in module owtf.utils.strings), 23
 PluginAbortException, 9
 PluginException, 9
 PluginsAlreadyLoaded, 9
 PluginsDirectoryDoesNotExist, 9
 purge () (owtf.lib.filelock.FileLock method), 10

R

release () (owtf.lib.filelock.FileLock method), 10
 remove_blanks_list () (in module owtf.utils.strings), 23
 renew_ip () (owtf.proxy.tor_manager.TOR_manager method), 15
 request_from_cache () (in module owtf.proxy.cache_handler), 13
 response_from_cache () (in module owtf.proxy.cache_handler), 13

rm_tree () (owtf.utils.file.FileOperations static method), 17
 run () (owtf.proxy.tor_manager.TOR_manager method), 15

S

scrub_output () (in module owtf.utils.strings), 24
 SentryProxy (class in owtf.utils.error), 16
 setup_signal_handlers () (in module owtf.utils.error), 17
 start_timer () (owtf.utils.timer.Timer method), 26
 starttls () (in module owtf.proxy.socket_wrapper), 14
 str2bool () (in module owtf.utils.strings), 24
 str_to_dict () (in module owtf.utils.strings), 24
 strip_key () (in module owtf.utils.strings), 24

T

Timer (class in owtf.utils.timer), 25
 timers (owtf.utils.timer.Timer attribute), 26
 to_str () (in module owtf.utils.strings), 24
 tor_control_process () (owtf.proxy.tor_manager.TOR_manager method), 15
 TOR_manager (class in owtf.proxy.tor_manager), 14
 truncate_lines () (in module owtf.utils.strings), 24

U

u () (in module owtf.utils.pycompat), 22
 UnreachableTargetException, 9
 UnresolvableTargetException, 9
 usage () (in module owtf.lib.cli_options), 8
 user_abort () (in module owtf.utils.error), 16
 utf8 () (in module owtf.utils.strings), 24

V

VersionMatches (class in owtf.api.utils), 6

W

warn_fmt (owtf.utils.formatters.ConsoleFormatter attribute), 19
 wipe_bad_chars () (in module owtf.utils.strings), 24