Burp-UI Documentation

Release 0.0.7.1

Ziirish

January 03, 2017

Contents

Burp-UI is a web-ui for Burp backup written in python with Flask and jQuery/Bootstrap. You may have a look a the README file first.

CHAPTER 1

Documentation

1.1 Requirements

Please note that currently, Burp-UI must be running on the same server that runs the burp-server.

For LDAP authentication (optional), we need the `ldap3` module.

```
pip install ldap3
```

If you would like to use SSL, you will need the `python-openssl` package. On Debian:

```
aptitude install python-openssl
```

1.2 Installation

Burp-UI is written in Python with the Flask micro-framework. The easiest way to install Burp-UI is to use `pip`.

On Debian, you can install `pip` with the following command:

```
aptitude install python-pip
```

Once `pip` is installed, you can install `Burp-UI` this way:

```
pip install burp-ui
```

You can setup various parameters in the burpui.cfg file. This file can be specified with the `-c` flag or should be present in `/etc/burp/burpui.cfg`. By default Burp-UI ships with a sample file located in `$INSTALLDIR/share/burpui/etc/burpui.sample.cfg`. (*$INSTALLDIR* defaults to */usr/local* when using pip **outside** a virtualenv)

Then you can run `burp-ui`: `burp-ui`

By default, `burp-ui` listens on all interfaces (including IPv6) on port 5000.

You can then point your browser to http://127.0.0.1:5000/

1.2.1 Instructions

In order to make the *on the fly* restoration/download functionality work, you need to check a few things:

1. Provide the full path of the burp (client) binary file

2. Provide the full path of an empty directory where a temporary restoration will be made. This involves you have enough space left on that location on the server that runs `Burp-UI`
3. Launch Burp-UI with a user that can proceed restorations and that can write in the directory above
4. Make sure to configure a client on the server that runs Burp-UI that can restore files of other clients (option *restore_client* in burp-server configuration)

1.2.2 Options

```
Usage: burp-ui [options]

Options:
  -h, --help            show this help message and exit
  -v, --verbose         verbose output
  -d, --debug           verbose output (alias)
  -V, --version         print version and exit
  -c CONFIG, --config=CONFIG
                        configuration file
  -l FILE, --logfile=FILE
                        output logs in defined file
```

1.3 Usage

Burp-UI has been written with modularity in mind. The aim is to support Burp from the stable to the latest versions. Burp exists in two major versions: 1.x.x and 2.x.x. The version 2.x.x is currently in heavy development and should bring a lot of improvements, but also a lot of rework especially regarding the `status port` which is the main communication system between Burp and Burp-UI.

Both *Versions* are supported by Burp-UI thanks to its modular design. The consequence is you have various options in the configuration file to suite every bodies needs.

There are also different modules to support *Authentication* and *ACL* within the web-interface.

> **Warning:** Burp-UI tries to be the less intrusive as possible, nevertheless it ships with the ability to manage Burp's configuration files. This feature **requires** Burp-UI to be launched on the **same** server that hosts your Burp instance. You also have to make sure the user that runs Burp-UI has **enough** privileges to edit those files.

1.3.1 Configuration

The burpui.cfg configuration file contains a `[Global]` section as follow:

```
[Global]
# On which port is the application listening
port: 5000
# On which address is the application listening
# '::' is the default for all IPv6
bind: ::
# enable SSL
ssl: false
# ssl cert
sslcert: /etc/burp/ssl_cert-server.pem
# ssl key
```

```
sslkey: /etc/burp/ssl_cert-server.key
# burp server version 1 or 2
version: 1
# Handle multiple bui-servers or not
# If set to 'false', you will need to declare at least one 'Agent' section (see
# bellow)
standalone: true
# authentication plugin (mandatory)
# list the misc/auth directory to see the available backends
# to disable authentication you can set "auth: none"
auth: basic
# acl plugin
# list misc/acl directory to see the available backends
# default is no ACL
acl: basic
```

Each option is commented, but here is a more detailed documentation:

- *port*: On which port is Burp-UI listening. This option is ignored when using Gunicorn.
- *bind*: On which address is Burp-UI listening. This option is ignored when using Gunicorn.
- *ssl*: Whether to enable SSL or not. This option is ignored when using Gunicorn.
- *sslcert*: SSL certificate to use when SSL support is enabled.
- *sslkey*: SSL key to use when SSL support is enabled.
- *version*: What version of Burp this Burp-UI instance manages. Can either be *1* or *2*. This parameter determines which backend is loaded at runtime.

  (see *Versions* for more details)
- *standalone*: Burp-UI can run in two different modes. If it runs in standalone mode (meaning you set this parameter to *true*), you can only address **one** Burp server of the version specified by the previous parameter.

  If this option is set to *false*, Burp-UI will run as a *proxy* allowing you to address multiple Burp servers. In this mode, you need to configure **at least one** *Agent* section in your configuration file. You also need to run one `bui-agent` per server.

  (see *Modes* for more details)
- *auth*: What *Authentication* backend to use.
- *acl*: What *ACL* module to use.

There is also a `[UI]` section in which you can configure some *UI* parameters:

```
[UI]
# refresh interval of the pages in seconds
refresh: 180
# refresh interval of the live-monitoring page in seconds
liverefresh: 5
```

Each option is commented, but here is a more detailed documentation:

- *refresh*: Time in seconds between two refresh of the interface.
- *liverefresh*: Time in seconds between two refresh of the *live-monitor* page.

1.3.2 Modes

Burp-UI provides two modes:

- *Standalone*
- *Multi-Agent*

These modes allow you to either access a single Burp server or multiple Burp servers hosted on separated hosts.

Standalone

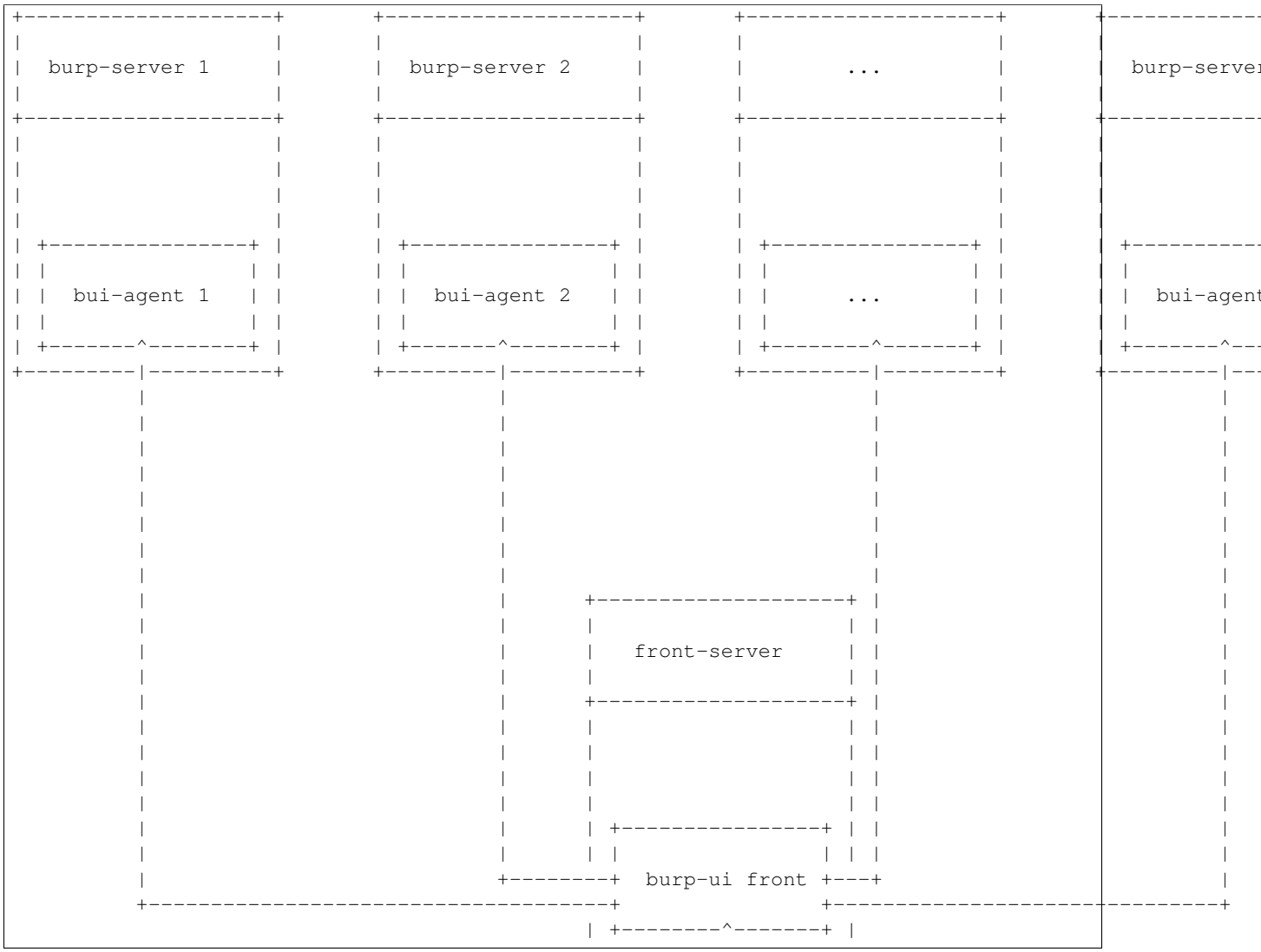
This mode is the **default** and the easiest one. It can be activated by setting the *standalone* parameter in the `[Global]` section of your burpui.cfg file to *true*:

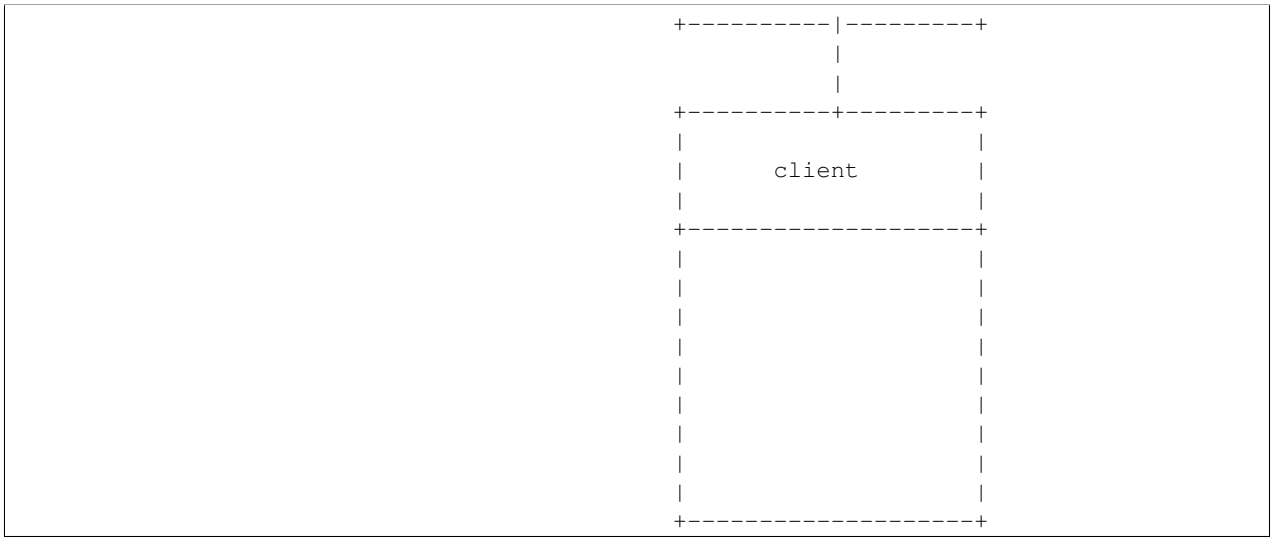
```
[Global]
standalone: true
```

That's all you need to do for this mode to work.

Multi-Agent

This mode allows you access multiple Burp servers through the bui-agent. Here is a schema to illustrate the architecture:





To enable this mode, you need to set the *standalone* parameter of the `[Global]` section of your burpui.cfg file to *false*:

```
[Global]
standalone: false
```

Once this mode is enabled, you have to create **one** `[Agent]` section **per** agent you want to connect to in your burpui.cfg file:

```
# If you set standalone to 'false', add at least one section like this per
# bui-agent
[Agent:agent1]
# bui-agent address
host: 192.168.1.1
# bui-agent port
port: 10000
# bui-agent password
password: azerty
# enable SSL
ssl: true
# socket timeout (in seconds)
timeout: 5

[Agent:agent2]
# bui-agent address
host: 192.168.2.1
# bui-agent port
port: 10000
# bui-agent password
password: ytreza
# enable SSL
ssl: true
# socket timeout (in seconds)
timeout: 5
```

Note: The sections must be called `[Agent:<label>]` (case sensitive)

To configure your agents, please refer to the bui-agent page.

1.3.3 Versions

Burp-UI ships with two different backends:

- *Burp1*
- *Burp2*

These backends allow you to either connect to a Burp server version 1.x.x or 2.x.x.

Burp1

The *burp-1* backend can be enabled by setting the *version* option to *1* in the `[Global]` section of your burpui.cfg file:

```
[Global]
version: 1
```

Now you can add *burp-1* backend specific options:

```
# burp1 backend specific options
[Burp1]
# burp status address (can only be '127.0.0.1' or '::1')
bhost: ::1
# burp status port
bport: 4972
# burp binary
burpbin: /usr/sbin/burp
# vss_strip binary
stripbin: /usr/sbin/vss_strip
# burp client configuration file used for the restoration (Default: None)
bconfcli: /etc/burp/burp.conf
# burp server configuration file used for the setting page
bconfsrv: /etc/burp/burp-server.conf
# temporary directory to use for restoration
tmpdir: /tmp
```

Each option is commented, but here is a more detailed documentation:

- *bhost*: The address of the Burp server. In burp-1.x.x, it can only be *127.0.0.1* or *::1*
- *bport*: The port of Burp's status port.
- *burpbin*: Path to the Burp binary (used for restorations).
- *stripbin*: Path to the Burp *vss_strip* binary (used for restorations).
- *bconfcli*: Path to the Burp client configuration file.
- *bconfsrv*: Path to the Burp server configuration file.
- *tmpdir*: Path to a temporary directory where to perform restorations.

Burp2

The *burp-2* backend can be enabled by setting the *version* option to *2* in the `[Global]` section of your burpui.cfg file:

```
[Global]
version: 2
```

Now you can add *burp-2* backend specific options:

```
# burp2 backend specific options
[Burp2]
# burp binary
burpbin: /usr/sbin/burp
# vss_strip binary
stripbin: /usr/sbin/vss_strip
# burp client configuration file used for the restoration (Default: None)
bconfcli: /etc/burp/burp.conf
# burp server configuration file used for the setting page
bconfsrv: /etc/burp/burp-server.conf
# temporary directory to use for restoration
tmpdir: /tmp
```

Each option is commented, but here is a more detailed documentation:

- *burpbin*: Path to the Burp binary (used for restorations).
- *stripbin*: Path to the Burp *vss_strip* binary (used for restorations).
- *bconfcli*: Path to the Burp client configuration file.
- *bconfsrv*: Path to the Burp server configuration file.
- *tmpdir*: Path to a temporary directory where to perform restorations.

1.3.4 Authentication

Burp-UI provides some authentication backends in order to restrict access only to granted users. There are currently two different backends:

- *LDAP*
- *Basic*

To disable the *authentication* backend, set the *auth* option of the `[Global]` section of your burpui.cfg file to *none*:

```
[Global]
auth: none
```

LDAP

The *ldap* authentication backend has some dependencies, please refer to the requirements page. To enable this backend, you need to set the *auth* option of the `[Global]` section of your burpui.cfg file to *ldap*:

```
[Global]
auth: ldap
```

Now you can add *ldap* specific options:

```
# ldapauth specific options
[LDAP]
# LDAP host
host: 127.0.0.1
# LDAP port
port: 389
# Encryption type to LDAP server (none, ssl or tls)
# - try tls if unsure, otherwise ssl on port 636
encryption: ssl
```

```
# Attribute to use when searching the LDAP repository
#searchattr: sAMAccountName
searchattr: uid
# LDAP filter to find users in the LDAP repository
# - {0} will be replaced by the search attribute
# - {1} will be replaced by the login name
filter: (&({0}={1})(burpui=1))
#filter: (&({0}={1})(|(userAccountControl=512)(userAccountControl=66048)))
# LDAP base
base: ou=users,dc=example,dc=com
# Binddn to list existing users
binddn: cn=admin,dc=example,dc=com
# Bindpw to list existing users
bindpw: Sup3rS3cr3tPa$$w0rd
```

Note: The *host* options accepts URI style (ex: ldap://127.0.0.1:389)

Basic

In order for the *basic* authentication backend to be enabled, you need to set the *auth* option of the `[Global]` section of your burpui.cfg file to *basic*:

```
[Global]
auth: basic
```

Now you can add *basic* specific options:

```
# basicauth specific options
# Note: in case you leave this section commented, the default login/password
# is admin/admin
[BASIC]
admin: password
user1: otherpassword
```

Note: Each line defines a new user with the *key* as the username and the *value* as the password

1.3.5 ACL

Burp-UI implements some mechanisms to restrict access on some resources only for some users. There is currently only one backend:

- *Basic ACL*

To disable the *acl* backend, set the *acl* option of the `[Global]` section of your burpui.cfg file to *none*:

```
[Global]
acl: none
```

Basic ACL

The *basic* acl backend can be enabled by setting the *acl* option of the `[Global]` section of your burpui.cfg file to *basic*:

```
[Global]
acl: basic
```

Now you can add *basic acl* specific options:

```
# basicacl specific options
# Note: in case you leave this section commented, the user 'admin' will have
# access to all clients whereas other users will only see the client that have
# the same name
[BASIC:ACL]
# Please note the double-quote around the username on the admin line are
# mandatory!
admin: ["user1","user2"]
# You can also overwrite the default behavior by specifying which clients a
# user can access
user3: ["client4", "client5"]
# In case you are not in a standalone mode, you can also specify which clients
# a user can access on a specific Agent
user4: {"agent1": ["client6", "client7"], "agent2": ["client8"]}
```

Warning: The double-quotes are **mendatory**

1.4 Gunicorn

Starting from v0.0.6, Burp-UI supports Gunicorn in order to handle multiple users simultaneously because some operations (like the online restoration) may take some time and thus may block any further requests. With Gunicorn, you have several workers that can proceed the requests so you can handle more users.

You need to install `gunicorn` and `eventlet`:

```
pip install eventlet
pip install gunicorn
```

You will then be able to launch Burp-UI this way:

```
gunicorn -k eventlet -w 4 'burpui:init(conf="/path/to/burpui.cfg")'
```

When using `gunicorn`, the command line options are not available. Instead, run the Burp-UI `init` method directly. Here are the parameters you can play with:

- conf: Path to the Burp-UI configuration file
- debug: Whether to run Burp-UI in debug mode or not to get some extra logging
- logfile: Path to a logfile in order to log Burp-UI internal messages

There is a sample configuration file available here.

1.4.1 Reverse Proxy

You may want to add a reverse proxy so Burp-UI can be accessed on port 80 (or 443) along with other applications.

Here is a sample configuration for nginx:

```
server {
    listen 80;
    server_name burpui.example.com;

    access_log  /var/log/nginx/burpui.access.log;
    error_log   /var/log/nginx/burpui.error.log;

    location / {

        # you need to change this to "https", if you set "ssl" directive to "on"
        proxy_set_header   X-FORWARDED_PROTO http;
        proxy_set_header   Host              $http_host;
        proxy_set_header   X-Forwarded-For   $remote_addr;

        proxy_read_timeout 300;
        proxy_connect_timeout 300;

        proxy_pass http://localhost:5000;
    }
}
```

1.5 bui-agent

The bui-agent is a kind of proxy between a Burp server and your Burp-UI server. These agents must be launched on every server hosting a Burp instance you'd like to monitor.

They have a specific buiagent.cfg configuration file with a `[Global]` section as below:

```
[Global]
# On which port is the application listening
port: 10000
# On which address is the application listening
# '0.0.0.0' is the default for all IPv4
bind: 0.0.0.0
# enable SSL
ssl: true
# ssl cert
sslcert: /etc/burp/ssl_cert-server.pem
# ssl key
sslkey: /etc/burp/ssl_cert-server.key
# burp server version (currently only burp 1.x is implemented)
version: 1
# agent password
password: password
# socket timeout (in seconds)
timeout: 5
```

Each option is commented, but here is a more detailed documentation:

- *port*: On which port is bui-agent listening.
- *bind*: On which address is bui-agent listening.
- *ssl*: Whether to communicate with the Burp-UI server over SSL or not.
- *sslcert*: What SSL certificate to use when SSL is enabled.
- *sslkey*: What SSL key to use when SSL is enabled.

- *version*: What version of Burp this bui-agent instance manages. (see Burp-UI versions for more details)
- *password*: The shared secret between the Burp-UI server and bui-agent.

As with Burp-UI, you need a specific section depending on the *version* value. Please refer to the Burp-UI versions section for more details.

1.5.1 Example

Here is a full usage example:

```
# On the server called 'agent1'
agent1:~$ python path/to/bui-agent.py -c path/to/buiagent.cfg

# On the server called 'agent2'
agent2:~$ python path/to/bui-agent.py -c path/to/buiagent.cfg

# On the server called 'front'
front:~$ python path/to/burp-ui.py -c path/to/burpui.cfg
```

This example uses three servers. You then only need to point your browser to http://front:5000/ for instance, and the Burp-UI instance will *proxify* the requests to the two agents for you.

1.6 Development

If you wish to use the latest and yet unstable version (eg. master), you can install it using `pip` too, but I would recommend you to use a `virtualenv`.

To do so, run the following commands:

```
mkdir /opt/bui-venv
pip install virtualenv
virtualenv /opt/bui-venv
source /opt/bui-venv/bin/activate
pip install git+https://git.ziirish.me/ziirish/burp-ui.git
```

You can uninstall/disable this *Burp-UI* setup by typing `deactivate` and removing the `/opt/bui-venv` directory.

1.7 Contributing

Contributions are welcome. You can help in any way you want, for instance by opening issues on the bug tracker, sending patches, etc.

There is also a dedicated website. Currently it only hosts a Discourse instance where you ca discuss with each other. No need to create another account, the one you use in the bug tracker can be imported automatically!

Feel free to use it and post your tips and remarks.

The address is: http://burpui.ziirish.me/

1.7.1 Troubleshooting

In case you encounter troubles with `Burp-UI`, you should run it with the `-d` flag and paste the relevant output within your bug-report. Please also give the version of `burp` AND `Burp-UI`. Since v0.0.6 you can use the `-V` or `--version` flag in order to get your version number.

1.7.2 Known Issues

1. SSL issue

My new SSL certificate seem to be unknown on older systems like debian wheezy. Thus, you may have some SSL failure while trying to clone my repository. In order to fix this error, you can run the following command as root that will add my certificate in your trust list:

```
echo -n | \
openssl s_client -showcerts -connect git.ziirish.me:443 \
-servername git.ziirish.me 2>/dev/null | \
sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >>/etc/ssl/certs/ca-certificates.crt
```

2. SSH issue

People that would like to clone the repository over SSH will face an authentication failure even if they added a valid SSH key in their user settings. The reason is I only have *one* public IP address so I must use port redirections to have multiple SSH instances running. To fix the issue, you should configure your SSH client by adding the following lines in your `~/.ssh/config` file:

```
Host git.ziirish.me
   Port 2222
```

1.8 Changelog

1.8.1 Current

- Full changelog

1.8.2 0.0.7

- Add Burp-2 backend
- Add sortable tables
- Add ACL support
- Add support client-side encrypted backups while performing an online restoration
- Add multiple archive format
- Add better Active Directory support
- Improvement: better config file parser
- Improvement: better logging with Gunicorn
- Improvement: full support of server configuration file + clientconfdir
- Fix issue #35

- Fix issue #37
- Fix issue #41
- Fix issue #42
- Fix issue #46
- Fix issue #49
- Fix issue #53
- Fix issue #54
- Fix issue #59
- Fix issue #62
- Fix issue #68
- Fix issue #69
- Fix issue #70
- Fix issue #71
- Fix issue #72
- doc on readthedocs
- Two merge requests from Wade Fitzpatrick (!1 and !2)
- API refactoring
- Security fixes
- Bufixes
- Full changelog

1.8.3 0.0.6

- Add gunicorn support
- Add init script for CentOS
- Add init script for Debian
- Add autofocus login field on login page
- Add burp-server configuration panel
- Fix issue #25
- Fix issue #26
- Fix issue #30
- Fix issue #32
- Fix issue #33
- Fix issue #34
- Fix issue #35
- Fix issue #39
- Code cleanup

- Improve unit tests
- Bugfixes
- Full changelog

1.8.4 0.0.5

- Add multi-server support
- Fix bugs
- Full changelog

1.8.5 0.0.4

- Add the ability to download files directly from the web interface
- Full changelog

1.8.6 0.0.3

- Add authentication
- Full changelog

1.8.7 0.0.2

- Fix bugs
- Full changelog

1.8.8 0.0.1

- Initial release

1.9 FAQ

1.9.1 How to configure my *firewall*?

When running Burp-UI in standalone mode, the embedded webserver listens on port **5000** on all interfaces.

The Burp-UI agents listens on port **10000** by default.

1.9.2 What are the default credentials?

The default login / password is *admin* / *admin* with the basic authentication backend.

1.9.3 How can I start Burp-UI as a daemon?

There are several *init scripts* provided by some users available here.

The recommanded way to run Burp-UI in production is to use Gunicorn. You can refer to the gunicorn section of this documentation for more details.

1.9.4 Are there any known issues?

There is a known issue section in this documentation.

1.9.5 How can I contribute?

You can refer to the contributing section of this documentation.

1.10 API doc (for dev)

GET /api/render-live-template
API: render_live_tpl :param name: the client name if any. You can also use the GET parameter 'name' to achieve the same thing :returns: HTML that should be included directly into the page

GET /api/running-clients.json
GET method provided by the webservice.

The *JSON* returned is:

```
{
    "results": [ ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **client** (*str*) – Ask a specific client in order to know if it is running a backup

Returns The *JSON* described above.

GET /api/clients-report.json
GET method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "backups": [
        {
          "name": "client1",
          "number": 15
        },
        {
          "name": "client2",
          "number": 1
```

```
            }
          ],
          "clients": [
            {
              "name": "client1",
              "stats": {
                "total": 296377,
                "totsize": 57055793698,
                "windows": "false"
              }
            },
            {
              "name": "client2",
              "stats": {
                "total": 3117,
                "totsize": 5345361,
                "windows": "true"
              }
            }
          ]
        }
      ]
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above

GET /api/server-config
GET method provided by the webservice.

The *JSON* returned is:


```
    {
      "boolean": [
        "daemon",
        "fork",
        "..."
      ],
      "defaults": {
        "address": "",
        "autoupgrade_dir": "",
        "ca_burp_ca": "",
        "ca_conf": "",
        "ca_name": "",
        "ca_server_name": "",
        "client_can_delete": true,
        "...": "..."
      },
      "integer": [
        "port",
        "status_port",
        "..."
      ],
      "multi": [
```

```
        "keep",
        "restore_client",
        "..."
      ],
      "placeholders": {
        "autoupgrade_dir": "path",
        "ca_burp_ca": "path",
        "ca_conf": "path",
        "ca_name": "name",
        "ca_server_name": "name",
        "client_can_delete": "0|1",
        "...": "..."
      },
      "results": {
        "boolean": [
          {
            "name": "hardlinked_archive",
            "value": false
          },
          {
            "name": "syslog",
            "value": true
          },
          { "...": "..." }
        ],
        "clients": [
          {
            "name": "testclient",
            "value": "/etc/burp/clientconfdir/testclient"
          }
        ],
        "common": [
          {
            "name": "mode",
            "value": "server"
          },
          {
            "name": "directory",
            "value": "/var/spool/burp"
          },
          { "...": "..." }
        ],
        "includes": [],
        "includes_ext": [],
        "integer": [
          {
            "name": "port",
            "value": 4971
          },
          {
            "name": "status_port",
            "value": 4972
          },
          { "...": "..." }
        ],
        "multi": [
          {
            "name": "keep",
```

```
            "value": [
              "7",
              "4"
            ]
          },
          { "...": "..." }
        ]
      },
      "server_doc": {
        "address": "Defines the main TCP address that the server listens on. The default is either '
        "...": "..."
      },
      "string": [
        "mode",
        "address",
        "..."
      ],
      "suggest": {
        "compression": [
          "gzip1",
          "gzip2",
          "gzip3",
          "gzip4",
          "gzip5",
          "gzip6",
          "gzip7",
          "gzip8",
          "gzip9"
        ],
        "mode": [
          "client",
          "server"
        ],
        "...": []
      }
    }
```


Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/servers.json

The `burpui.api.servers.ServersStats` resource allows you to retrieve statistics about servers/agents.

This resource is part of the `burpui.api.servers` module.

GET /api/running.json

GET method provided by the webservice.

The *JSON* returned is:


```
    {
        "results": false
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/clients.json
GET method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "last": "2015-05-17 11:40:02",
      "name": "client1",
      "state": "idle"
    },
    {
      "last": "never",
      "name": "client2",
      "state": "idle"
    }
  ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above

GET /api/live.json
API: live :returns: the live status of the server

GET /clients-report
Global report

GET /live-monitor
Live status monitor view

GET /client
Specific client overview

GET /
Home page

GET /api/ (*server*) **/client-tree.json/**
*name***/int:** *backup* **GET** method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "date": "2015-05-21 14:54:49",
      "gid": "0",
      "inodes": "173",
      "mode": "drwxr-xr-x",
      "name": "/",
```

```
          "parent": "",
          "size": "12.0KiB",
          "type": "d",
          "uid": "0"
        }
      ]
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns The *JSON* described above.

GET /api/ (*server*) **/client-stat.json/**
*name***/int:** *backup* **GET** method provided by the webservice.

The *JSON* returned is:


```
    {
      "results": {
        "dir": {
          "changed": 0,
          "deleted": 0,
          "new": 394,
          "scanned": 394,
          "total": 394,
          "unchanged": 0
        },
        "duration": 5,
        "efs": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "encrypted": true,
        "end": 1422189124,
        "files": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "files_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 1421,
          "scanned": 1421,
          "total": 1421,
```

```
      "unchanged": 0
    },
    "hardlink": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "meta": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "meta_enc": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "number": 1,
    "received": 1679304,
    "softlink": {
      "changed": 0,
      "deleted": 0,
      "new": 1302,
      "scanned": 1302,
      "total": 1302,
      "unchanged": 0
    },
    "special": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "start": 1422189119,
    "total": {
      "changed": 0,
      "deleted": 0,
      "new": 3117,
      "scanned": 3117,
      "total": 3117,
      "unchanged": 0
    },
    "totsize": 5345361,
    "vssfooter": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
```

```
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssfooter_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "windows": "false"
      }
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **`server`** (*str*) – Which server to collect data from when in multi-agent mode
- **`name`** (*str*) – The client we are working on
- **`backup`** (*int*) – The backup we are working on

Returns The *JSON* described above.

`POST /api/` (*server*) **`/restore/`**
name/**int:** *backup* **POST** method provided by the webservice. This method returns a `flask.Response` object.

Parameters

- **`server`** (*str*) – Which server to collect data from when in multi-agent mode
- **`name`** (*str*) – The client we are working on
- **`backup`** (*int*) – The backup we are working on

Returns A `flask.Response` object representing an archive of the restored files

`GET /` (*server*) **`/client-browse/`**
name/**int:** *backup*/**int:** *encrypted* Browse a specific backup of a specific client

GET **/client-browse/**(*name*)**/**
int: *backup***/int:** *encrypted* Browse a specific backup of a specific client

GET **/api/client-tree.json/**(*name*)**/**
int: *backup* **GET** method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "date": "2015-05-21 14:54:49",
      "gid": "0",
      "inodes": "173",
      "mode": "drwxr-xr-x",
      "name": "/",
      "parent": "",
      "size": "12.0KiB",
      "type": "d",
      "uid": "0"
    }
  ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns The *JSON* described above.

GET **/api/client-stat.json/**(*name*)**/**
int: *backup* **GET** method provided by the webservice.

The *JSON* returned is:

```
{
  "results": {
    "dir": {
      "changed": 0,
      "deleted": 0,
      "new": 394,
      "scanned": 394,
      "total": 394,
      "unchanged": 0
    },
    "duration": 5,
    "efs": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "encrypted": true,
```

```
        "end": 1422189124,
        "files": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "files_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 1421,
          "scanned": 1421,
          "total": 1421,
          "unchanged": 0
        },
        "hardlink": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "meta": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "meta_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "number": 1,
        "received": 1679304,
        "softlink": {
          "changed": 0,
          "deleted": 0,
          "new": 1302,
          "scanned": 1302,
          "total": 1302,
          "unchanged": 0
        },
        "special": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
```

```
        },
        "start": 1422189119,
        "total": {
          "changed": 0,
          "deleted": 0,
          "new": 3117,
          "scanned": 3117,
          "total": 3117,
          "unchanged": 0
        },
        "totsize": 5345361,
        "vssfooter": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssfooter_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "windows": "false"
      }
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns The *JSON* described above.

POST /api/restore/ (*name*) **/**
int: *backup* **POST** method provided by the webservice. This method returns a `flask.Response` object.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns A `flask.Response` object representing an archive of the restored files

GET /api/ (*server*) **/render-live-template/**
name API: render_live_tpl :param name: the client name if any. You can also use the GET parameter 'name' to achieve the same thing :returns: HTML that should be included directly into the page

GET /api/ (*server*) **/running-clients.json/**
client **GET** method provided by the webservice.

The *JSON* returned is:

```
{
    "results": [ ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **client** (*str*) – Ask a specific client in order to know if it is running a backup

Returns The *JSON* described above.

GET /api/ (*server*) **/client-stat.json/**
name **GET** method provided by the webservice.

The *JSON* returned is:

```
{
  "results": {
    "dir": {
      "changed": 0,
      "deleted": 0,
      "new": 394,
      "scanned": 394,
      "total": 394,
      "unchanged": 0
    },
    "duration": 5,
    "efs": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "encrypted": true,
    "end": 1422189124,
    "files": {
```

```
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "files_enc": {
            "changed": 0,
            "deleted": 0,
            "new": 1421,
            "scanned": 1421,
            "total": 1421,
            "unchanged": 0
          },
          "hardlink": {
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "meta": {
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "meta_enc": {
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "number": 1,
          "received": 1679304,
          "softlink": {
            "changed": 0,
            "deleted": 0,
            "new": 1302,
            "scanned": 1302,
            "total": 1302,
            "unchanged": 0
          },
          "special": {
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "start": 1422189119,
```

```
        "total": {
          "changed": 0,
          "deleted": 0,
          "new": 3117,
          "scanned": 3117,
          "total": 3117,
          "unchanged": 0
        },
        "totsize": 5345361,
        "vssfooter": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssfooter_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "vssheader_enc": {
          "changed": 0,
          "deleted": 0,
          "new": 0,
          "scanned": 0,
          "total": 0,
          "unchanged": 0
        },
        "windows": "false"
      }
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns The *JSON* described above.

GET /api/ (*server*) **/server-config/**
path: *conf* **GET** method provided by the webservice.

The *JSON* returned is:


```
{
  "boolean": [
    "daemon",
    "fork",
    "..."
  ],
  "defaults": {
    "address": "",
    "autoupgrade_dir": "",
    "ca_burp_ca": "",
    "ca_conf": "",
    "ca_name": "",
    "ca_server_name": "",
    "client_can_delete": true,
    "...": "..."
  },
  "integer": [
    "port",
    "status_port",
    "..."
  ],
  "multi": [
    "keep",
    "restore_client",
    "..."
  ],
  "placeholders": {
    "autoupgrade_dir": "path",
    "ca_burp_ca": "path",
    "ca_conf": "path",
    "ca_name": "name",
    "ca_server_name": "name",
    "client_can_delete": "0|1",
    "...": "..."
  },
  "results": {
    "boolean": [
      {
        "name": "hardlinked_archive",
        "value": false
      },
      {
        "name": "syslog",
        "value": true
      },
      { "...": "..." }
    ],
    "clients": [
      {
        "name": "testclient",
        "value": "/etc/burp/clientconfdir/testclient"
      }
    ],
    "common": [
      {
        "name": "mode",
        "value": "server"
```

```
        },
        {
          "name": "directory",
          "value": "/var/spool/burp"
        },
        { "...": "..." }
      ],
      "includes": [],
      "includes_ext": [],
      "integer": [
        {
          "name": "port",
          "value": 4971
        },
        {
          "name": "status_port",
          "value": 4972
        },
        { "...": "..." }
      ],
      "multi": [
        {
          "name": "keep",
          "value": [
            "7",
            "4"
          ]
        },
        { "...": "..." }
      ]
    },
    "server_doc": {
      "address": "Defines the main TCP address that the server listens on. The default is either '
      "...": "..."
    },
    "string": [
      "mode",
      "address",
      "..."
    ],
    "suggest": {
      "compression": [
        "gzip1",
        "gzip2",
        "gzip3",
        "gzip4",
        "gzip5",
        "gzip6",
        "gzip7",
        "gzip8",
        "gzip9"
      ],
      "mode": [
        "client",
        "server"
      ],
      "...": []
    }
```

```
    }
```

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/ (*server*) **/client.json/**
name **GET** method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "date": "2015-01-25 13:32:00",
      "deletable": true,
      "encrypted": true,
      "number": "1"
    }
  ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on

Returns The *JSON* described above.

GET / (*server*) **/client-browse/**
*name***/int:** *backup* Browse a specific backup of a specific client

GET / (*server*) **/backup-report/**
*name***/int:** *backup* Backup specific report

GET /client-browse/ (*name*) **/**
int: *backup* Browse a specific backup of a specific client

GET /backup-report/ (*name*) **/**
int: *backup* Backup specific report

GET /api/render-live-template/ (*name*)
API: render_live_tpl :param name: the client name if any. You can also use the GET parameter 'name' to achieve the same thing :returns: HTML that should be included directly into the page

GET /api/running-clients.json/ (*client*)
GET method provided by the webservice.

The *JSON* returned is:

```
{
    "results": [ ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **client** (*str*) – Ask a specific client in order to know if it is running a backup

Returns The *JSON* described above.

GET /api/client-stat.json/ (*name*)
GET method provided by the webservice.

The *JSON* returned is:


```
{
  "results": {
    "dir": {
      "changed": 0,
      "deleted": 0,
      "new": 394,
      "scanned": 394,
      "total": 394,
      "unchanged": 0
    },
    "duration": 5,
    "efs": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "encrypted": true,
    "end": 1422189124,
    "files": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "files_enc": {
      "changed": 0,
      "deleted": 0,
      "new": 1421,
      "scanned": 1421,
      "total": 1421,
      "unchanged": 0
    },
    "hardlink": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "meta": {
      "changed": 0,
      "deleted": 0,
```

```
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "meta_enc": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "number": 1,
    "received": 1679304,
    "softlink": {
      "changed": 0,
      "deleted": 0,
      "new": 1302,
      "scanned": 1302,
      "total": 1302,
      "unchanged": 0
    },
    "special": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "start": 1422189119,
    "total": {
      "changed": 0,
      "deleted": 0,
      "new": 3117,
      "scanned": 3117,
      "total": 3117,
      "unchanged": 0
    },
    "totsize": 5345361,
    "vssfooter": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "vssfooter_enc": {
      "changed": 0,
      "deleted": 0,
      "new": 0,
      "scanned": 0,
      "total": 0,
      "unchanged": 0
    },
    "vssheader": {
```

```
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "vssheader_enc": {
            "changed": 0,
            "deleted": 0,
            "new": 0,
            "scanned": 0,
            "total": 0,
            "unchanged": 0
          },
          "windows": "false"
        }
      }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on
- **backup** (*int*) – The backup we are working on

Returns The *JSON* described above.

GET **/api/server-config/** (**path:** *conf*)
GET method provided by the webservice.

The *JSON* returned is:


```
      {
        "boolean": [
          "daemon",
          "fork",
          "..."
        ],
        "defaults": {
          "address": "",
          "autoupgrade_dir": "",
          "ca_burp_ca": "",
          "ca_conf": "",
          "ca_name": "",
          "ca_server_name": "",
          "client_can_delete": true,
          "...": "..."
        },
        "integer": [
          "port",
          "status_port",
          "..."
        ],
        "multi": [
          "keep",
          "restore_client",
```

```
      "..."
    ],
    "placeholders": {
      "autoupgrade_dir": "path",
      "ca_burp_ca": "path",
      "ca_conf": "path",
      "ca_name": "name",
      "ca_server_name": "name",
      "client_can_delete": "0|1",
      "...": "..."
    },
    "results": {
      "boolean": [
        {
          "name": "hardlinked_archive",
          "value": false
        },
        {
          "name": "syslog",
          "value": true
        },
        { "...": "..." }
      ],
      "clients": [
        {
          "name": "testclient",
          "value": "/etc/burp/clientconfdir/testclient"
        }
      ],
      "common": [
        {
          "name": "mode",
          "value": "server"
        },
        {
          "name": "directory",
          "value": "/var/spool/burp"
        },
        { "...": "..." }
      ],
      "includes": [],
      "includes_ext": [],
      "integer": [
        {
          "name": "port",
          "value": 4971
        },
        {
          "name": "status_port",
          "value": 4972
        },
        { "...": "..." }
      ],
      "multi": [
        {
          "name": "keep",
          "value": [
            "7",
```

```
                "4"
              ]
            },
            { "...": "..." }
          ]
        },
        "server_doc": {
          "address": "Defines the main TCP address that the server listens on. The default is either '
          "...": "..."
        },
        "string": [
          "mode",
          "address",
          "..."
        ],
        "suggest": {
          "compression": [
            "gzip1",
            "gzip2",
            "gzip3",
            "gzip4",
            "gzip5",
            "gzip6",
            "gzip7",
            "gzip8",
            "gzip9"
          ],
          "mode": [
            "client",
            "server"
          ],
          "...": []
        }
      }
```


Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/client.json/ (*name*)
GET method provided by the webservice.

The *JSON* returned is:


```
      {
        "results": [
          {
            "date": "2015-01-25 13:32:00",
            "deletable": true,
            "encrypted": true,
            "number": "1"
          }
        ]
      }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **name** (*str*) – The client we are working on

Returns The *JSON* described above.

GET /api/ (*server*) **/render-live-template**
API: render_live_tpl :param name: the client name if any. You can also use the GET parameter 'name' to achieve the same thing :returns: HTML that should be included directly into the page

GET /api/ (*server*) **/running-clients.json**
GET method provided by the webservice.

The *JSON* returned is:

```
{
    "results": [ ]
}
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode
- **client** (*str*) – Ask a specific client in order to know if it is running a backup

Returns The *JSON* described above.

GET /api/ (*server*) **/clients-report.json**
GET method provided by the webservice.

The *JSON* returned is:

```
{
  "results": [
    {
      "backups": [
        {
          "name": "client1",
          "number": 15
        },
        {
          "name": "client2",
          "number": 1
        }
      ],
      "clients": [
        {
          "name": "client1",
          "stats": {
            "total": 296377,
            "totsize": 57055793698,
            "windows": "false"
          }
        },
        {
          "name": "client2",
          "stats": {
            "total": 3117,
```

```
                "totsize": 5345361,
                "windows": "true"
              }
            }
          ]
        }
      ]
    }
```


The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above

GET /api/ (*server*) **/server-config**
GET method provided by the webservice.

The *JSON* returned is:


```
    {
      "boolean": [
        "daemon",
        "fork",
        "..."
      ],
      "defaults": {
        "address": "",
        "autoupgrade_dir": "",
        "ca_burp_ca": "",
        "ca_conf": "",
        "ca_name": "",
        "ca_server_name": "",
        "client_can_delete": true,
        "...": "..."
      },
      "integer": [
        "port",
        "status_port",
        "..."
      ],
      "multi": [
        "keep",
        "restore_client",
        "..."
      ],
      "placeholders": {
        "autoupgrade_dir": "path",
        "ca_burp_ca": "path",
        "ca_conf": "path",
        "ca_name": "name",
        "ca_server_name": "name",
        "client_can_delete": "0|1",
        "...": "..."
      },
      "results": {
        "boolean": [
```

```
            {
              "name": "hardlinked_archive",
              "value": false
            },
            {
              "name": "syslog",
              "value": true
            },
            { "...": "..." }
          ],
          "clients": [
            {
              "name": "testclient",
              "value": "/etc/burp/clientconfdir/testclient"
            }
          ],
          "common": [
            {
              "name": "mode",
              "value": "server"
            },
            {
              "name": "directory",
              "value": "/var/spool/burp"
            },
            { "...": "..." }
          ],
          "includes": [],
          "includes_ext": [],
          "integer": [
            {
              "name": "port",
              "value": 4971
            },
            {
              "name": "status_port",
              "value": 4972
            },
            { "...": "..." }
          ],
          "multi": [
            {
              "name": "keep",
              "value": [
                "7",
                "4"
              ]
            },
            { "...": "..." }
          ]
        },
        "server_doc": {
          "address": "Defines the main TCP address that the server listens on. The default is either '
          "...": "..."
        },
        "string": [
          "mode",
          "address",
```

```
        "..."
      ],
      "suggest": {
        "compression": [
          "gzip1",
          "gzip2",
          "gzip3",
          "gzip4",
          "gzip5",
          "gzip6",
          "gzip7",
          "gzip8",
          "gzip9"
        ],
        "mode": [
          "client",
          "server"
        ],
        "...": []
      }
    }
```

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/ (*server*) **/running.json**
GET method provided by the webservice.

The *JSON* returned is:

```
    {
        "results": false
    }
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above.

GET /api/ (*server*) **/clients.json**
GET method provided by the webservice.

The *JSON* returned is:

```
    {
      "results": [
        {
          "last": "2015-05-17 11:40:02",
          "name": "client1",
          "state": "idle"
        },
        {
          "last": "never",
          "name": "client2",
```

```
            "state": "idle"
          }
        ]
      }
```

The output is filtered by the `burpui.misc.acl` module so that you only see stats about the clients you are authorized to.

Parameters

- **server** (*str*) – Which server to collect data from when in multi-agent mode

Returns The *JSON* described above

GET /api/ (*server*) **/live.json**
API: live :returns: the live status of the server

GET / (*server*) **/client-browse/**
name Browse a specific backup of a specific client

GET / (*server*) **/client-report/**
name Specific client report

GET / (*server*) **/backup-report/**
name Backup specific report

GET / (*server*) **/live-monitor/**
name Live status monitor view

GET / (*server*) **/client/**
name Specific client overview

GET /client-browse/ (*name*)
Browse a specific backup of a specific client

GET /client-report/ (*name*)
Specific client report

GET /backup-report/ (*name*)
Backup specific report

GET /live-monitor/ (*name*)
Live status monitor view

GET /client/ (*name*)
Specific client overview

GET /static/ (**path:** *filename*)
Function used internally to send static files from the static folder to the browser.

New in version 0.5.

GET / (*server*) **/clients-report**
Global report

GET / (*server*) **/live-monitor**
Live status monitor view

GET / (*server*) **/client**
Specific client overview

Indices and tables

- search

/

/(server)

/api

/backup-report

/client

/client-browse

GET /client-browse/(name)/(int:backup)/(int:encrypted), 24

/client-report

GET /client-report/(name), 43

/clients-report

GET /clients-report, 21

/live-monitor

GET /live-monitor, 21
GET /live-monitor/(name), 43

/static

GET /static/(path:filename), 43