TaskServer

Abstract

This document describes TaskServer, the part of Lorem Ipsum project. The aim of server is to provide task suites for application. Below you can find information about server features, installation and maintenance instruction and protocol description.

Features

- serving single task suite per server instance
- listing currently available task suite
- authorization of users before allowing to download the suite
- setting limit for maximum number of downloads for particular users
- storing information about identifiers of user's devices

System requirements

The server requires following components to be available on target platform:

- Python 3.3 or higher (http://www.python.org/)
- Tornado Web Server 3.2 or higher (http://www.tornadoweb.org/)
- SQLite 3.8 or higher (http://sqlite.org/)
- OpenSSL 1.0.1e or higher (https://www.openssl.org/)

Server should work on any platform supporting features. One of such platform is Fedora 20 (Linux), which is the suggested one. Installation instruction below describes some details about installing required components.

Installation

Following steps describe installation process on target machine. Each step contains both general information and details about how to proceed on suggested operating system (Fedora 20).

1. **General**: Install Python 3.3 or newer (it is also called CPython). On most of operating systems you can install it using bundled package manager. Otherwise, you can always look for instructions and appropriate packages on http://www.python.org/.

   Fedora: Please run following command in the system terminal:

`sudo yum -y install python3`

If prompted, provide your root password.

2. **General**: Install Tornado Web Server 3.2 or newer. On most of operating systems you can install it using bundled package manager. Otherwise, you can always look for release on http://www.tornadoweb.org/.

   Note: Usual package name for Tornado is `python-tornado` or `python3-tornado`. If you have both options, please install version *for Python 3* (the latter one).

   Fedora: Please run following command in the system terminal:

   `sudo yum -y install python3-tornado`

   If prompted, provide your root password.

3. **General**: Install SQLite 3.8 or newer. On most of operating systems you can install it using bundled package manager. Otherwise, you can always look for instructions and appropriate packages on http://sqlite.org/.

   Fedora: Please run following command in the system terminal:

   `sudo yum -y install sqlite`

   If prompted, provide your root password.

4. **General**: Install OpenSSL 1.0.1e or newer. On most of operating systems you can install it using bundled package manager. Otherwise, you can always look for instructions and appropriate packages on https://www.openssl.org/.

   Fedora: Please run following command in the system terminal:

   `sudo yum -y install openssl`

   If prompted, provide your root password.

5. **General**: Copy TaskServer directory to appropriate place in your hard disk.

   Fedora: Please run following command in the system terminal:

   `cp -R <source> <target>`

   replacing `<source>` by the location of `TaskServer` directory.

6. *Optional*: Do it, if you prefer to use other port number than the default, *40555*.

   General: Open the configuration file `server.conf` and add following line:

   `port=<port number>`

   replacing `<port number` with the port number that you'd like to run server on.

   Fedora: Please run following command in the system terminal:

`echo "port=<port number>" >> server.conf`

replacing `<port number>` with the port number that you'd like to run server on.

7. **General**: Run `server.py` in detached mode. This can be done by running with `--start` argument *or* by using system `nohup` command. While being in TaskServer server directory, full command line may look like:

   `python3 server.py --start`

   or

   `nohup python3 server.py&`

   If your system support hashbangs, you can use:

   `./server.py --start`

   or

   `nohup ./server.py&`

   Note: Make sure the server is started *using Python 3*, not the older version that may be installed on your system.

   Fedora: Please navigate in the system terminal to the directory where the server has been copied and run the following command:

   `./server.py --start`

   or

   `nohup ./server.py&`

If everything went, the server should be available through HTTPS protocol. Please open web browser and enter the following address:

`https://<hostname>:40555/`

or, in case if you set different port number:

`https://<hostname>:<port number>/`

replacing `<hostname>` by IP-address or host name of your server, and `<port number>` by the number that has been set in `server.conf`.

If you see the message informing that *connection is untrusted* or similar, this means that server is working properly. The message comes from the fact that server uses custom, self-signed certificate which is properly recognised by Lorem Ipsum application, but not by the web browsers.

Maintenance

Starting and stopping server

To start server, please run following command:

`./server.py --start`

To stop server, please run following command:

`./server.py --stop`

If your operating system does not support hashbang, you may need to replace `./server.py` parts by `python3 server.py`.

Served task suite

The task suite that is provided by server instance is always stored in `TaskServer/suites` directory. Task suite package must meet several restrictions. All restrictions you may find in description of *PrepareSuite* tool, following one is the only that is required and checked by server:

Suite file name *must* use following scheme: `<suite_name>-<suite_version>.zip`

In this scheme:

- `<suite_name>` *must* be replaced by full name of the task suite. This name *may* contain spaces, and *may* contain `-` (*hypen/minus* character). This name is the one that will be visible for user in the application after suite installation is finished.
- `<suite_version>` *must* be replaced by full version of the task suite. The version *must not* contain `-` (*hypen/minus*) character. Suggested versioning scheme is *major.minor.build*, however choosing any particular scheme of versioning is up to suite maintainer.

User's account and management

Task server uses HTTP Basic Authorization scheme to protect access to all resources. Thus, to access task suite, user must provide valid username and password. Verification that is done on server is based on content of SQLite database.

Database of users is stored in file `database.sqlite`, that is stored in server's directory. File may be opened using SQLite's command line interface, or one of many available graphical interfaces (for example Sqliteman, SQLite Manager for Firefox or phpLiteAdmin). Also, a few administrative scripts are available - they will be described later

Database schema is as follows (in *SQL language*):

```
CREATE TABLE Users (
    ID INTEGER PRIMARY KEY,
    UserName TEXT NOT NULL UNIQUE,
```

```
    Password TEXT NOT NULL,
    InstallLimit INTEGER
);

CREATE TABLE Devices (
    ID INTEGER PRIMARY KEY,
    UserID INTEGER NOT NULL REFERENCES Users(ID),
    DeviceID TEXT NOT NULL
);
```

Table `Users` contains information about user's accounts. `UserName` is the name of the user, while `Password` is her password. The password is the Unicode character string, encoded in UTF-8, hashed using SHA-512 digest. The digest itself is stored as hexadecimal string. `InstallLimit` is the integer number that specifies count of devices where user is allowed to install task suite. In case when there is no such limit, this value should be `NULL`.

Table `Devices` contains information about all devices where task suite has been downloaded and installed. `UserID` is the identifier of particular user, and `DeviceID` is single device identifier that has been submitted during installation.

Adding new users To add new user, please execute following command:

```
./utils-add-user.sh <UserName> <Password> <InstallLimit>
```

replacing `<UserName>` by the name of the new user, `<Password>` by her password and `<InstallLimit>` by the number of maximum installations allowed for this user (or NULL if no such limit should be set).

Example:

```
./utils-add-user.sh Humphrey somepassword 3
```

Removing users To remove user, please execute following command:

```
./utils-delete-user.sh <UserName>
```

replacing `<UserName>` by the name of the user that shall be deleted.

Example:

```
./utils-delete-user.sh George
```

Listing all installations perfomed by particular user To list identifiers for all devices where particular user has installed the task suite, please execute following command:

```
./utils-list-devices.sh <UserName>
```

replacing `<UserName>` by the name of the user that's devices you'd like to list.

Example:

```
./utils-list-devices.sh George
```

Protocol

Communication between Task Server and the application is conducted using HTTPS protocol (protocols HTTP and TLS as defined in RFC 2616 and RFC 6520). Server's interface has defined two endpoints, that are used for communication.

Authentication

To authenticate, application **must** provide user name and password, that has been previously registered in server's database. Those data should be provided using **HTTP Basic Authentication** scheme.

Endpoints

/,

Method: **GET**

Arguments: *none*

Result: **JSON**-encoded object of following scheme:

```
"suites": [
  {
    "name": "<suite name>",
    "version": "<suite version>",
    "url": "<download URL>""
  }
]
```

Following fields are replaced by proper values:

`<suite name>` - the name of the task suite, as presented for the user

`<suite version>` - the version of the task suite

`<download URL>` - the URL that can be used to download task suite. In current implementation, it is always `/suite/<suite name>/<suite version>` (see below).

Note: despite that protocol is itself designed to support more than one task suite per server (by using array as value for `"suites"` node), application nor task server are not currently support it.

`/suite/<suite name>/<suite version>`

Method: **GET**

Arguments:

`device_id` - identifier of device that is downloading task suite. The application uses randomly generated UUID, as it practically guarantees uniqueness.

Result: task suite file, as `application/octet-stream`

In case if user has exceeded her installation limit, 403 status code is returned, with message `Installation limit exceeded` in response content.