



easyshare

Command line application for transfer files between network hosts.
Like FTP, but (hopefully) easier.

What's easyshare

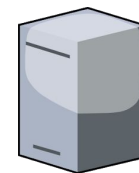
- A command line client-server application
 - Client: **es**
 - Server: **esd**
- **Handles** local and remote files
(e.g. create directories, move files, remove files)
- **Transfers** files to and from remote hosts
- Automatically **discover** available *sharings* through **broadcast**

Technologies

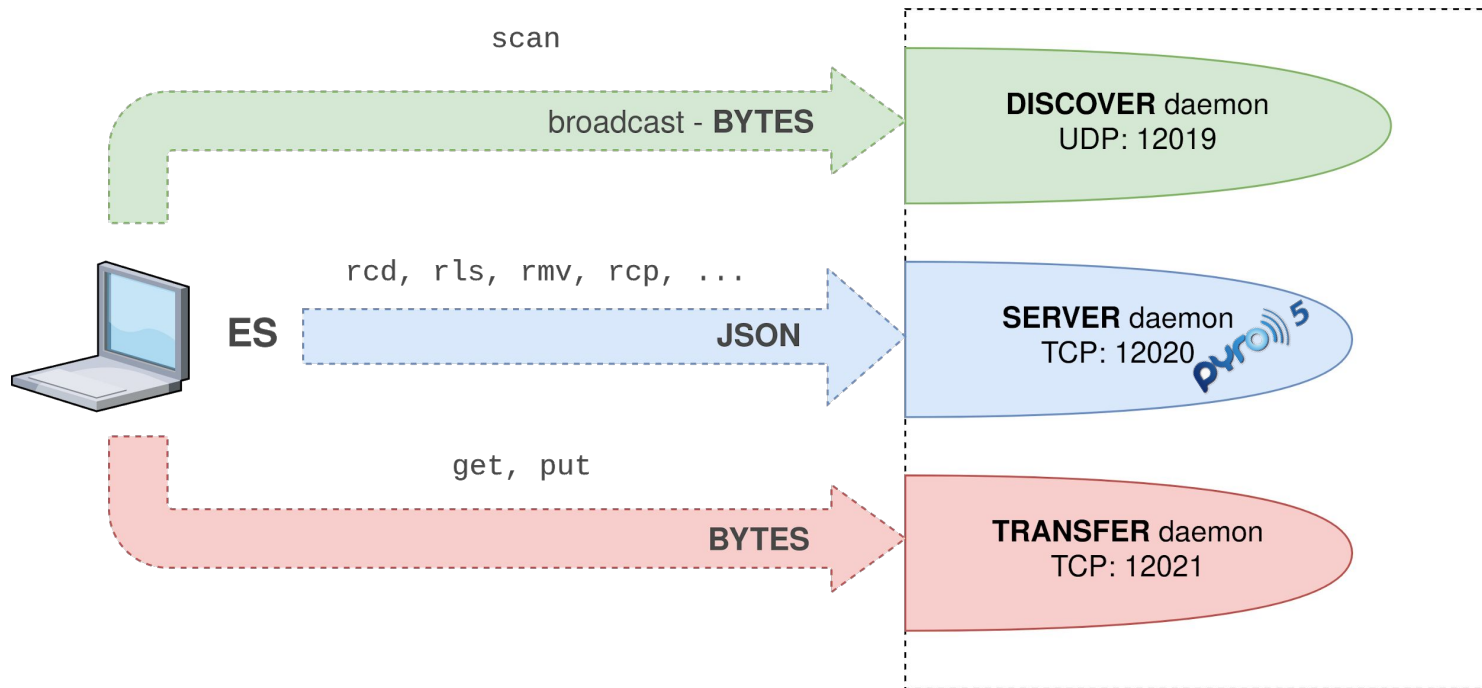
- Written in *Python 3*
- Uses *Pyro 5* for high level operation
*A library that enables you to build applications in which objects can talk to each other over the network, with minimal programming effort. **You can just use normal Python method calls to call objects running on other machines.** Pyro is a pure Python library and runs on many different platforms and Python versions.*
- Uses Python's socket for low level operation that need efficiency, such as file transfer



Network architecture



ESD



An overview: the commands

Local commands

pwd
ls
tree
cd
mkdir
cp
mv
rm
exec

Misc commands

help
exit
trace
verbose

Remote commands

rpwd
rls
rtree
rcd
rmkdir
rcp
rmv
rrm
rexec

Server info commands

info
list
ping

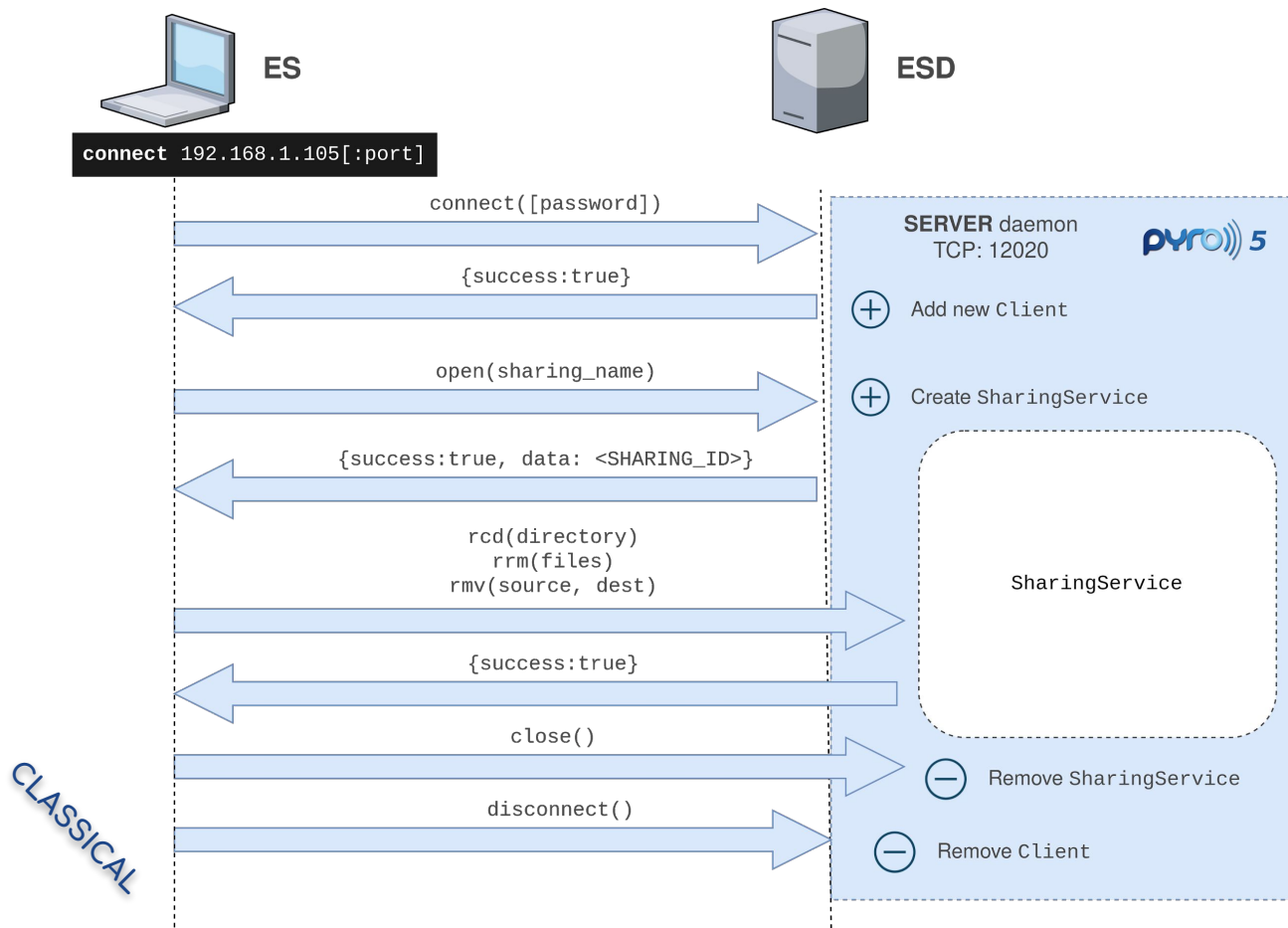
Connection establishment commands

scan
connect
disconnect
open
close

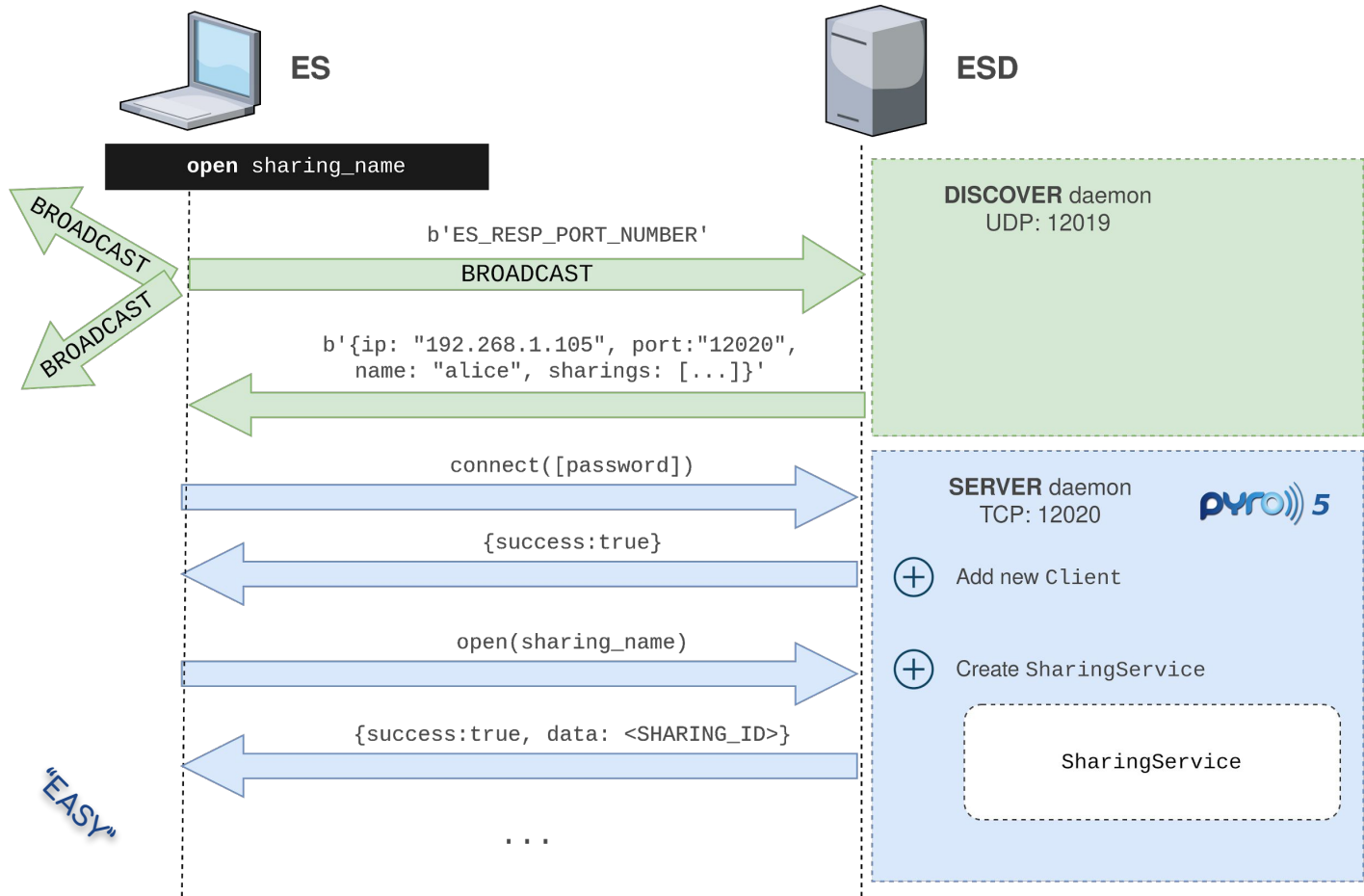
Transfer commands

get
put

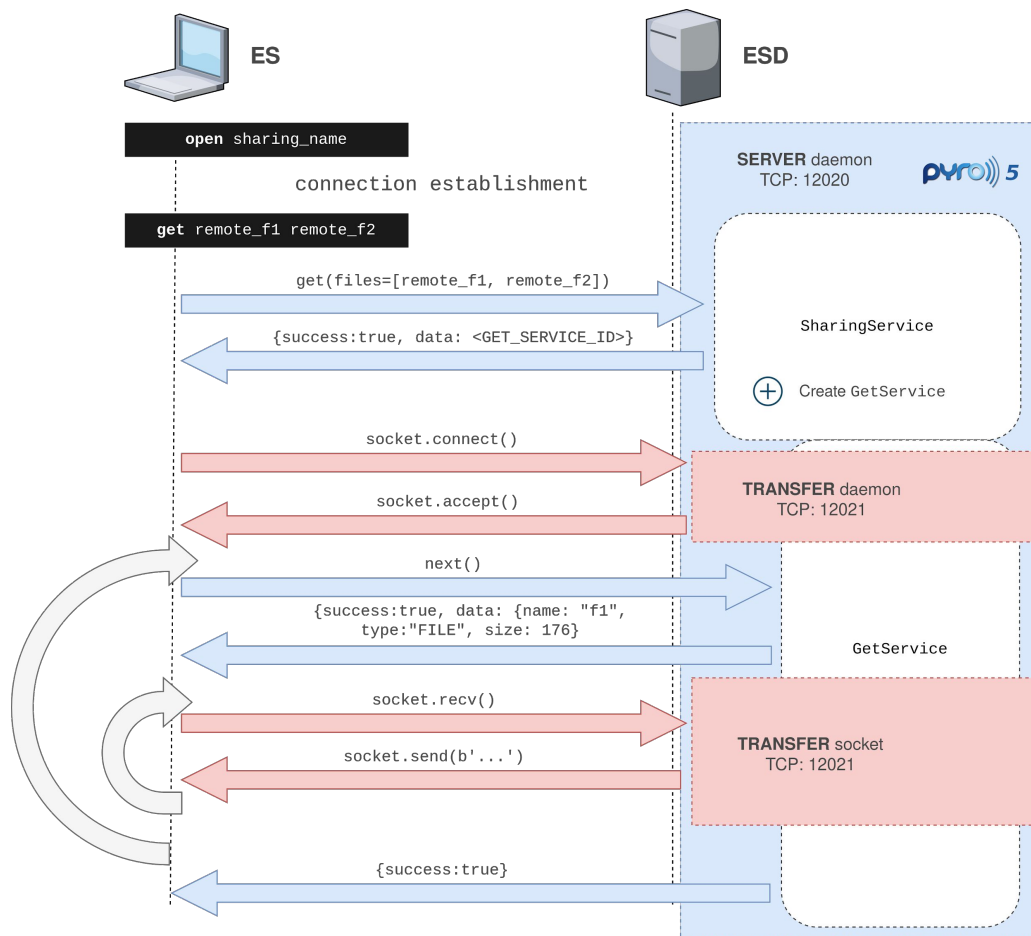
Connection establishment: DIRECTLY



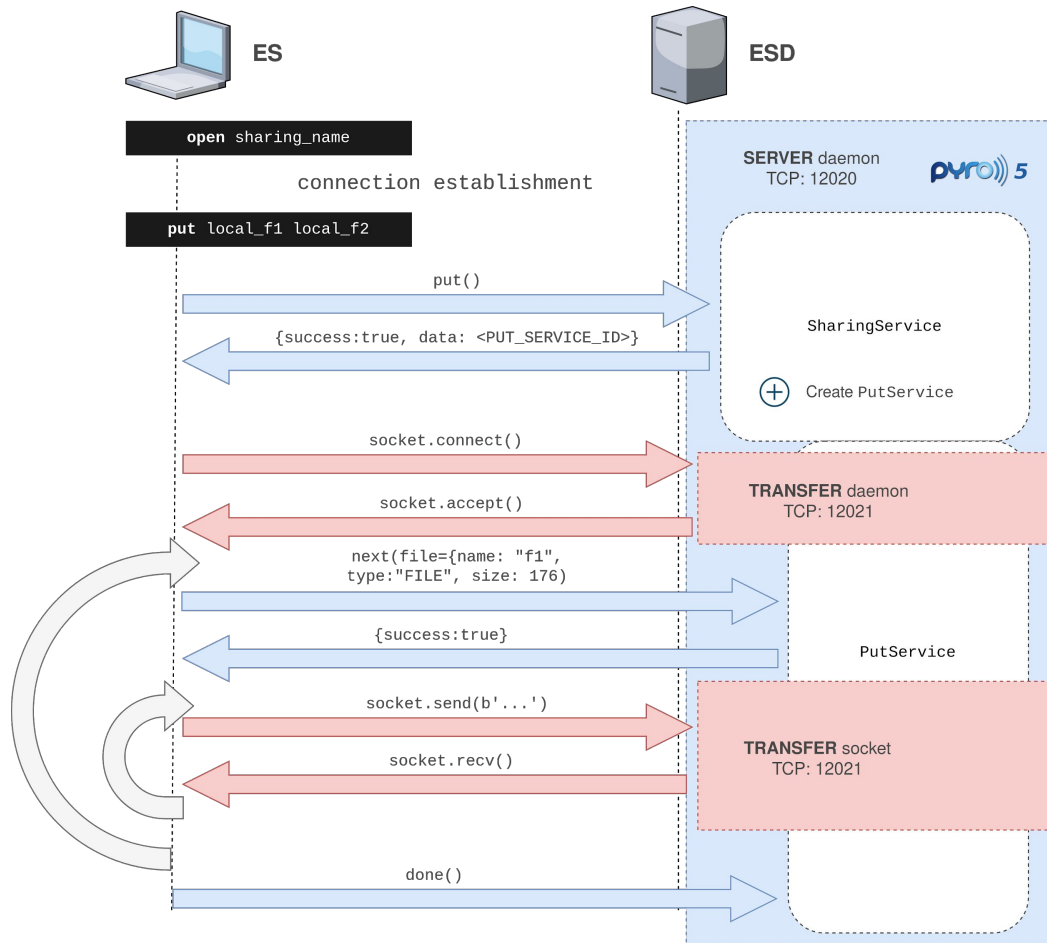
Connection establishment: DISCOVERING



File transfer: get



File transfer: put



A step inside: es

Connection
establishment
(DISCOVER)

Remote files
management

Files transfer
get

```
+ easyshare git:(master) X python -m easyshare.es
/home/stefano/Develop/Python/easyshare> cd /tmp
/tmp> scan
1. stefano-arch (192.168.1.105:12020)
   DIRECTORIES
   • shared
/tmp> open shared
stefano-arch.shared:/ — /tmp> rtree
├── dir
│   ├── f3
│   └── f4
├── f1
└── f2
stefano-arch.shared:/ — /tmp> rmv f1 dir
stefano-arch.shared:/ — /tmp> rtree
├── dir
│   ├── f1
│   ├── f3
│   └── f4
├── f2
stefano-arch.shared:/ — /tmp> rrm dir/f4
stefano-arch.shared:/ — /tmp> rtree
├── dir
│   ├── f1
│   └── f3
└── f2
stefano-arch.shared:/ — /tmp> rcd dir
stefano-arch.shared:/dir — /tmp> get f1 f3
GET f1                                ██████████ 100%    51KB/51KB    0s
GET f3                                ██████████ 100%   128KB/128KB  0s
GET outcome: OK
Files      2 (179KB)
Time       0s
Avg. speed 20.4MB/s
stefano-arch.shared:/dir — /tmp> █
```

A step inside: esd

Scenario 1.

When

Easy and fast sharing
of one file or directory

How

Specifying sharing
location as argument
of the command

```
→ easyshare git:(master) X python -m easyshare.esd /tmp/shared
=====

SERVER INFO

Name:          stefano-arch
Address:       192.168.1.105
Server port:   12020
Transfer port: 12021
Discover port: 12019
Auth:         none
SSL:          False
Remote execution: False

=====

SHARINGS

* shared --> /tmp/shared

=====

RUNNING...

Client connected: 192.168.1.105:55196
Client connected: 192.168.1.105:55198
Client disconnected: 192.168.1.105:55196
```

A step inside: esd

Scenario 2.

When

Sharing of multiple
files or directories

How

Specifying the path to
the config file via **-c**

esd.conf

```
# ===== SERVER INFO =====  
name=stefano-arch  
  
# ===== SHARINGS =====  
[download]  
  path="/home/stefano/Downloads"  
  
[shared]  
  path="/tmp/shared"  
  readonly=true
```

```
→ easyshare git:(master) ✗ python -m easyshare.esd -c /tmp/esd.conf  
=====
```

SERVER INFO	
Name:	stefano-arch
Address:	192.168.1.105
Server port:	12020
Transfer port:	12021
Discover port:	12019
Auth:	none
SSL:	False
Remote execution:	False

```
=====
```

SHARINGS

- * download --> /home/stefano/Downloads
- * shared --> /tmp/shared

```
=====
```

RUNNING...

A step inside: esd

Scenario 2+.

When

Sharing of multiple files
or directories with more
security

How

Specifying the path to
the (well configured)
config file via **-c**

Server password
(plain or hashed)

SSL

Read-only flag for
specific sharings

esd.conf

```
# ===== SERVER INFO =====  
  
name=stefano-arch  
  
# Password (plain or hashed)  
password=asecurepassword  
#password=1$u6MaRLZpr1e5HWcksBmDsQ==  
  
# SSL  
ssl=true  
ssl_cert="/tmp/cert.pem"  
ssl_privkey="/tmp/privkey.pem"  
  
# ===== SHARINGS =====  
  
[download]  
    path="/home/stefano/Downloads"  
  
[shared]  
    path="/tmp/shared"  
    readonly=true
```

Need help?

help <command>

```
COMMAND
open - open a remote sharing (eventually discovering it)

SYNOPSIS
open SHARING_LOCATION
o SHARING_LOCATION

DESCRIPTION
Open a sharing whose location is specified by SHARING_LOCATION.

SHARING_LOCATION has the following syntax:
<sharing_name>[@<server_name>|<address>[:<port>]]

See the section EXAMPLES for more examples about SHARING_LOCATION.

The following rules are applied for establish a connection:

1. If SHARING_LOCATION is a valid <sharing_name> (e.g. shared), a discover is performed for figure out to which server the sharing belongs to.
2. If SHARING_LOCATION has the form <sharing_name>@<server_name>[:port] (e.g. shared@alice-arch) a discover is performed as well as in case 1. and the <server_name> and the <port> act only as a filter (i.e. the connection won't be established if they don't match).
3. If SHARING_LOCATION has the form <sharing_name>@<address> (e.g. shared@192.168.1.105) the connection will be tried to be established directly to the server at the default port. If the attempt fails, a discover is performed for figure out which port the server is really bound to and another attempt is done.
4. If SHARING_LOCATION has the form <sharing_name>@<address>:<port> (e.g. shared@192.168.1.105:12020) the connection
```

:

```
COMMAND
rls - list remote directory content

SYNOPSIS
rls [OPTION]... [DIR]

rls [OPTION]... [SHARING_LOCATION] [DIR]

SHARING_LOCATION must be specified if and only if not already connected to a remote sharing. In that case the connection will be established before execute the command, as "open SHARING_LOCATION" would do.

Type "help open" for more information about SHARING_LOCATION format.

DESCRIPTION
List content of the remote DIR or the current remote directory if no DIR is specified.

OPTIONS
-a, --all          show hidden files too
-g, --group        group by file type
-l               show more details
-r, --reverse      reverse sort order
-s, --sort-size    sort by size
-S               show files size

SEE ALSO
Type "help ls" for the local analogous.
```

(END)

Not enough (help)?

verbose, trace

```
/home/stefano/Develop/Python/easyshare> verbose 4
Verbosity = 4 (debug)
[DEBUG] {easyshare.es.shell} Command execution: OK
[DEBUG] {easyshare.es.shell} Connected to esd : False
[DEBUG] {easyshare.es.shell} Connected to sharing: False
/home/stefano/Develop/Python/easyshare> connect stefano-arch
[DEBUG] {easyshare.es.shell} Detected command 'connect'
[INFO] {easyshare.es.client} Executing connect(['stefano-arch'])
[DEBUG] {easyshare.es.client} serverconnection_parser_provider -> 'not connect' parser
[DEBUG] {easyshare.args} Inspecting argument stefano-arch
[DEBUG] {easyshare.args} Considering 'stefano-arch' as positional parameter
[DEBUG] {easyshare.args} 0 unparsed args w/o positional: []
[DEBUG] {easyshare.args} 1 positional arguments: ['stefano-arch']
[DEBUG] {easyshare.args} Parsing and append to bucket...
  NoopParams : 1 mandatory, 0 optional
  params = ['stefano-arch']
  offset = 0
[DEBUG] {easyshare.args} [0] Mandatory: checking validity: param_ok_hook('stefano-arch')
[DEBUG] {easyshare.args} Parsing and appending to bucket (taking 0:1)
[INFO] {easyshare.args} > ['stefano-arch']
[DEBUG] {easyshare.args} 0 unparsed args: []
[INFO] {easyshare.es.client} Parsed command arguments
{
  "parsed": {
    "null": [
      "stefano-arch"
    ]
  },
  "unparsed": []
}
[INFO] {easyshare.es.client} >> CONNECT
[DEBUG] {easyshare.es.common} ServerLocation.parse() -> stefano-arch
```

```
/tmp> trace
Tracing = 1 (enabled)
/tmp> open download
>> <broadcast>:12019
>> DISCOVER b'\x00' (53359)
<< 192.168.1.105:53615
<< DISCOVER
{
  "success": true,
  "data": {
    "name": "stefano-arch",
    "sharings": [
      {
        "name": "download",
        "ftype": "dir",
        "read_only": true
      },
      {
        "name": "shared",
        "ftype": "dir",
        "read_only": true
      }
    ],
    "ssl": false,
    "auth": false,
    "ip": "192.168.1.105",
    "port": 12020,
    "discoverable": true,
    "discover_port": 12019
  }
}
>> 192.168.1.105:12020 (stefano-arch)
>> connect (password=None)
<< 192.168.1.105:12020 (stefano-arch)
<< connect
{
  "success": true
}
>> 192.168.1.105:12020 (stefano-arch)
>> open ("download")
<< 192.168.1.105:12020 (stefano-arch)
<< open
{
  "success": true,
  "data": "25154f0ed37a4c46b802157238c77253"
}
stefano-arch.download:/ — /tmp>
```


Other features

- **Command** and command's **options** completion with TAB
- Local and remote **files** completion with TAB
- Designed for work behind **NAT** too
(apart from the broadcast discovery)
- Man pages available
man es
man esd
- Helper for generate passwords hashes and configuration file
es-tools

Easy to install, as well

```
pip install easyshare
```

Links

- <https://github.com/Docheinstein/easyshare>
- <https://pypi.org/project/easyshare/>



```
#!/bin/sh  
exit 0
```

