P2P FILE SHARING

‘The Octopus’ Group  
  
Team Members:

AHMAD, FARHAN

ALIPOURSIMAKANI, KAMRAN

ANDERSSON EKSTRÖM, MAX

BERNTSSON, FREDRIK

CHADALAPAKA, GAYATRI

GHASEMI REZAEI, AMIN

IQBAL, NAYYAR

KUKKAPALLI, NAGA VYSHNAVI

NYHLÉN, JESPER

ROUTHU, VENKATA SAI KALYAN

SHAD MANFEAT, SEYEDEH MERSEDEH

ZAREI, KAMBIZ

Type of Document: Acceptance test plan

Version 1.0

Publication Date: May 8th ,2016

**1. PREFACE:**

***Section 2*: Glossary and abbreviations**

**Section 3: Acceptance Test Plan**

**Section 4: References’**

**Release v1.0 on 2016-05-08**

- Initial release

**2: Glossary and abbreviations**

Dark Peer - Is the peer which exceeds the valid time interval assigned by the bootstrap server and thus removed from list of known peers, it will be marked and known as Dark Peer in the network.

Dark Content - Is the files located on any dark peers.

Swarm - The main file which each peer gets from the server at its first connection which includes list of shared files, list of peers and information about which node shares the file and the swarm metadata.

Swarm Metadata - Includes filenames and file message digest

File Metadata - Is the set of headers combined with the filename.

Bootstrap-Server - Server which will inform the other nodes about the presence of any node connected to network and will fetch the swarm metadata from them.

Service: simply is the outcome of chain of functions to reach it designed and defined purpose.

Server: the device which is going to provide the services by using its resources.

Interface: is the presentation of results which is done by the servers based on service, and it is the visible part of a system architecture.

**3: Acceptance Test Plan**

The requirements to be fulfilled by our system will be listed below with atleast one testcase to show that the system works as intended.

For full information;

-on the requirements, check the provided requirement document [1].

-on the test cases, check the provided test document [2].

|  |  |  |  |
| --- | --- | --- | --- |
| **Requirement ID** | **Description** | **Test Case ID** | **Description** |
| Req-Sys\_101 | All peers shall be installed with the static ip address of the default bootstrap server. | Test\_201 | Possible connect bootstrap upon start client |
| Req-Back\_102 | All peers shall connect to the bootstrap server upon start. | Test\_201 | Possible connect bootstrap upon start client |
| Req-Back\_103 | All peers shall recieve 3 randomly selected IP addresses of others peers upon start. | Test\_202 | Test if you get the 3 connecting peers IPs from Bootstrap when the peer is starting |
| Req-Back\_104 | All peers shall receive a list of published swarms upon start. | Test\_203 | Test if you get the published swarms when the client is started |
| Req-Back\_105 | All peers shall receive a blacklist of IP addresses that have been banned from the service upon start. | Test\_204 | Test if you get the blacklist from Bootstrap |
| Req-Back\_106 | All peers shall block communication with the IP adresses that are banned. | Test\_205 | Test communication with a banned IP-address |
| Req-Back\_107 | All IP addressess except the bootstrap servers should have a validity time of 3 minutes. | Test\_206 | Check validity time of ips on bootstrap server |
| Req-Back\_108 | All peers should refresh its validity time, this refresh should be done every 2 minutes and extend the validity time by a value of 3 minutes. | Test\_206 | Check validity time of ips on bootstrap server |
| Req-Back\_109 | All IP addresses that has exceeded their validity time shall be removed from the bootstrap. | Test\_206 | Check validity time of ips on bootstrap server |
| Req-Back\_110 | All swarms without IP addresses shall be removed from the bootstrap server. | Test\_303 | Test if a swarm without any peers is shown in the bootstrap server. |
| Req-Back\_111 | All peers and servers shall synchronize with a NTP-server and use the same timezone. | Test\_209, Test\_301 | -Check that the peer synced with NTP  -Test if the Bootstrap servers time is synced with NTP |
| Req-Back\_112 | All peers should update their information from the bootstrap server every 3 minutes. | Test\_210 | Test if a peer updates its information from Bootstrap every 3 minute |
| Req-Back\_113 | All peers which are not able to connect to the default bootstrap server, should connect to another bootstrap server. If that also fails, they should use the current data they have. | Test\_211 | Test if a peer connects to the secondary bootstrap in case the default one fauls. |
| Req-Back\_114 | All peers which can not connect to a bootstrap server and does not have the necessary data should shut down. | Test\_212 | Test if a peer shuts down if it doesn´t have the proper data. |
| Req-Back\_115 | When a peer receives the 3 IP addresses of other peers it should establish a HTTPS connection with them. | Test\_213 | Test if a peer communicates with other peers over HTTPS. |
| Req-Back\_116 | If a peer can´t connect to any of the IP addresses it receives it should recieve 3 new ones from the bootstrap server at the next peer update interval. | Test\_214 | Test if a peer gets new IPs to connect to in case connection to the first ones fails. |
| Req-Back\_117 | A peer should at most have 3 connections to other peers at the same time. | Test\_215 | Test if a peer can connect to 4 different peers. |
| Req-Back\_118 | The bootstrap servers should synchronize their data with eachother every 3 minutes. | Test\_304 | Test if the data between bootstrap servers is synchronizing every 3 minutes. |
| Req-Back\_119 | All swarms shall be divided into blocks with a sequence of 1024-bytes. | Test\_401 | Test if all swarms are divided into the correct block sizes |
| Req-Back\_120 | All swarms shall have two message digest checksum of the file contents and the metadata which uses SHA-1 | Test\_402 | Test if the messages have a two message digest checksum |
| Req-Back\_121 | All peers that connects to an IP address should exchange a maximum of 3 IP addresses with each other. | Test\_216 | Test if connecting peers exchange at most 3 ip addresses with eachother |
| Req-Back\_122 | When a file has finished downloading the system shall recompute the message digest and verify the file is intact. | Test\_217 | Test if the message digest works when a file has finished downloading |
| Req-Back\_123 | When a peer is uploading a swarm to another peer it should add their IP address to the metadata of the swarm. | Test\_218 | Test if a peer who is downloading a file, gets added to the list of peers who has that file. |
| Req-API\_124 | The system shall communicate between user-to-user, user-to-server and server-to-server with a RESTFUL API with JSON data encoding | Test\_406 | Test if the communication between users and servers are done with a RESTFUL API with JSON data encoding |
| Req-Sys\_125 | The system shall have an option to restart with encryption disabled. | Test\_403 | Test if it´s possible to start with encryption disabled |
| Req-Sys\_126 | The system shall have encryption on the user-to-user, user-to-server and server-to-server communication. | Test\_404 | Test if the system have encryption between user-to-user, user-to-server, server-to-server |
| Req-Sys\_127 | The system shall support more than 2 peers. | Test\_405 | Test if the system supports 3 peers |
| Req-Front\_130 | The peers shall provide an option to download a swarm in the GUI. | Test\_102 | To test whether it´s possible to download a swarm |
| Req-Front\_131 | The peers shall provide an option to create swarms in the GUI | Test\_101 | To test whether it´s possible to create swarms in the GUI. |
| Req-Front\_132 | The peers shall provide an option to see a list of files in the GUI | Test\_101 | To test whether it´s possible to create swarms in the GUI. |
| Req-Front\_133 | The peers shall provide an option to search for files in the GUI | Test\_103 | To test if you can search for files in the GUI |
| Req-Front\_135 | The peers shall provide an option to delete a swarm in the GUI. | Test\_105 | Testing to delete a swarm from the GUI |
| Req-Front\_136 | The peers shall provide an option to choose whether a swarm is dark or visible in the GUI upon creation. | Test\_106 | Testing if you can set a swarm as dark in the GUI upon creation |
| Req-Front\_137 | The peers shall show the progress of uploads and downloads. | Test\_107, Test\_108 | -Testing if you can see progress of uploads  -Testing if you can see progress of downloads |
| Req-Front\_138 | The peers shall show the estimated time to complete downloads | Test\_109 | Test if you can see an estimate of how long is left of downloads |
| Req-Front\_139 | The peers shall show the list of the IP addresses associated with a swarm | Test\_110 | Test if you can see all IP addresses which the swarm can be downloaded from |
| Req-Front\_140 | The peers shall show all IP addresses of connected peers. | Test\_111 | Test if you can see IP addresses of the connected peers |
| Req-Front\_141 | The peers shall show an error message in case a file digest fails and provide an option to redownload the file. | Test\_112 | Test if you can redownload a file in case a file digest fails |

**4. REFERENCES:**

[1] Tests Document v1.1

[2] Requirements Document v1.2