Unreal Tournament 99 Query Protocol

Unreal Tournament, like many other games of the time frame around year 2000 allow for LAN and online play. The former is done by looking for game servers on the local network while the latter game servers are looked up on master servers. In case of UT, the actual information query is therefore a bit different for LAN and online servers.

The game uses up to three ports for connectivity, which can be set rather freely. The LAN query port is a single UDP port while the actual (online) play ports are an adjacent pair of UDP ports that you can select by configuring the first one of the two. The default for online servers are UDP ports 7777, resp. 7778, and also 7779 and 7780. The first one is used for the actual game connectivity (playing) while the second one is always used for querying server details.

The LAN port on the other hand is solely used to find out about the set game port. The default port is 8777 (UDP). However it will work just fine, if you choose a port within range 8777-8786. Those ports are queried automatically via broadcast queries. The server then answers by sending the game port to the client, which in turn now knows where to connect to the game server.

LAN Server Queries

REPORTQUERY

When trying to figure out where servers on the local network can be found, the game client sends the string REPORTQUERY to address 255.255.255.255 (local broadcast) on the ten UDP ports 8777 to 8786.

If a server is configured accordingly (you can set a port outside of the mentioned port range or turn off LAN support altogether), it will then respond with the following string pattern:

`ut` *`7780`*

The string part `ut ` (ut, followed by a space, 0x20) is always the same. The second part is the port to be used for actual query connections. In this example the result would be port 7780 for query connections, implying port 7779 for game connections. This port is then used with the info query, to obtain further information about the server.

REPORT

There is another query request sent to the LAN query port, that gives a very limited information: REPORT. The server will respond with a string similar to this one: `ut 7779 utserver Turbine DM 0/10`. All elements are separated by a space (0x20). Again, the `ut ` header is static. It is followed by the game port, host name of the server, name of the map and game mode and the number of players and number of allowed players. This reply is probably meant for displaying purposes. It can be parsed automatically (first 3 complete words are separate entities, last two are also, leaving the map name as something that can contain spaces). However it is a lot more complete to use the above `\status\` query on the query port.

General Game Server Queries

These queries are all directed at the query port, which is the second one of the two necessary ports of any UT server.

info

Sending the string `\info\` will result in the server answering with a string similar to this one:

```
\hostname\UT99 @ Mobile Infanterie\hostport\7779\maptitle\Phobos
Moon\mapname\DM-
Phobos\gametype\DeathMatchPlus\numplayers\0\maxplayers\10\gamemode\openplayin
g\gamever\451\minnetver\432\worldlog\true\wantworldlog\true\queryid\9.1\final
\
```

Those are the key-value pairs describing the properties of the game server, including the hostport (actual game port), played map, game type and other technical information:

- `hostname`: the name configured for this server,
- `hostport`: the actual game port, where the client will connect to for playing,
- `maptitle`: name or the map currently being played,
- `mapname`: technical name of the map currently being played,
- `gametype`: the game type currently being played, e. g. Deathmatch,
- `numplayers`: currently playing humans,
- `maxplayers`: maximum number of players allowed to be connected simultaneously,
- `gamemode`: ?,
- `gamever`: version of the game,
- `minnetver`: minimum required version of the game client to successfully connect to this server,
- `worldlog`: does the server use the game's score log functionalitiy?,
- `wantworldlog`: is the server configured to try and use the stats server functionality and
- `queryid`: an up-counting query number

The list is also concluded by `\final\`, marking the end of the key-value pair list.

status

Once this information was obtained, a second query request is sent to the server's query port, using the string `\status\`. This gives some additional information about what is going on on the server:

```
\gamename\ut\gamever\451\minnetver\432\location\0\hostname\UT99 @ Mobile
Infanterie\hostport\7779\maptitle\Phobos Moon\mapname\DM-
Phobos\gametype\DeathMatchPlus\numplayers\0\maxplayers\10\gamemode\openplayin
g\gamever\451\minnetver\432\worldlog\true\wantworldlog\true\listenserver\Fals
e\password\False\timelimit\15\fraglimit\20\minplayers\4\changelevels\True\tou
rnament\False\gamestyle\Hardcore\botskill\Average\AdminName\7Saturn\AdminEMai
l\7saturn@gmx.de\queryid\10.1\final\
```

This returns basically the same information, but adding the following information on top:

- `listenserver`: Is this a dedicated server or a listen server?,
- `password`: does the server require the client to authenticate with a password before joining the match?,
- `timelimit`: how long in minutes is a map being played at most?,
- `fraglimit`: what is the highest number of frags after which the map ends?,
- `minplayers`: to what number of players will the server be filled, if the number of human players is less than this value?,
- `changelevels`: do the maps change?,
- `tournament`: is it 1on1?,
- `gamestyle`: ?,
- `botskill`: how strong are the bots set to play?,
- `AdminName`: who is the admin on this server?,
- `AdminEMail`: where can you reach the admin? and
- `queryid`: ?.

Again, the end of the key-value-pairs is marked with the sub string `\final\`.

basic

Another variation of this can be done on the query port of the server, sending `\basic\`. This will result in the server answering with something similar to this:
`\gamename\ut\gamever\451\minnetver\432\location\0\queryid\21.1\final\`

This will also give a key-value pair list, with the following information:

- `gamename`: what game is the server playing? For UT99 that is always `ut`,
- `gamever`: what version of the game is the server running?,
- `minnetver`: what is the minimum version of the game client in order to be allowed to play on this server?,
- `location`: where is the server located? and
- `queryid`: an up-counting query number.

Combinations

The queries *basic*, *info* and *rules* can be combined in one query by sending `\basic\\info\\rules\`. This way the server will answer with all the data sent back, that the single queries would return by themselves.

Master Server Queries

In principle, the queries of server details mentioned in section LAN Server Queries can be applied as well for internet servers. But in order to get a list of available game servers a master server has to be queried. This is done via a TCP connection. The original master server for UT99 hosted by Epic Games is still available via the domain name *unreal.epicgames.com:28900* (IP: 199.255.40.174 on 2021-12-12).

When connecting to the master server the server will begin by sending the following string: `\basic\\secure\wookie`. The client then sends the string: `\gamename\ut\location\0\validate\2/TYFMRc\final\` and in a subsequent message: `\list\\gamename\ut\final\`. The 2/TYFMRc is actually an identifier for UT, as many games use very similar protocols, that differ only minor. This ensures, that the client identifies itself as a UT99 client.

The server then answers with a string of the following form:

`\ip\`*`IP-Address:Port`*`\ip\`*`Another-IP-Address:Port`*`\final\`

So the server sends a list of IP addresses and ports of known servers, each one preceded by the string `\ip\`. After the last server of the list, the string `\final\` finishes up the list. The IPs and ports are noted in the same fashion as usual, e.g. *192.168.0.1:7777*. With this information the client will then query the servers, as is shown above.

Master Server Announcement

In order to show up on a master server, a game server must announce itself to the master server. This is done by sending a so-called heartbeat signal. The UDP datagram for that signal looks like this: `\heartbeat\`*`7781`*`\gamename\ut\gamever\`*`436`*.

The values for *7781* and *436* may vary. The latter names the server's version, the former the game port (which in turn implies the query port as game port + 1).

The master server itself does not seem to make any sanity checks (e.g., check, whether the announced server is actually there any longer and an actual UT99 game server). However, as the clients do the detail queries themselves, any not existing server is not displayed (but delivered to the requesting client anyways).

Game Types

Here is a list of used game type identifiers in server replies:

Identifier	Game Type
DeathMatchPlus	Deathmatch
TeamGamePlus	Team Deathmatch
CTFGame	Capture the Flag
Domination	Domination
Assault	Assault
LastManStanding	Last Man Standing

[Games Database] [Unreal Tournament]