[BT（带中心Tracker）通信协议的分析](http://blog.chinaunix.net/uid-26548237-id-3056731.html) 2012-01-10 21:53:27

分类： 网络与安全

**BT通信协议举例分析**

现在的很多BT下载都采用了DHT网络，这样进行BT下载就不需要中心服务器了。本文针对的是需要中心服务器的BT下载。

  小弟我最近正在研究BT通信协议，网上的资料很全，但是不是那事详细和完整，因此，整理下来，一方面他日用到拿来看看，另一方面，希望对正在研究BT通信协议的有点帮助。若有不正之处，请指正。

**1.        BT协议的工作过程**

BT协议主要包括3个部分：.torrent文件的格式、tracker HTTP/HTTPS协议和peer wire协议(使用TCP)。其中tracker HTTP/HTTPS协议是BT客户机与tracker服务器之间的通信协议，peer wire是BT客户机之间的通信协议。

**1.1 torrent文件的结构**

       .torrent文件的内容，采用了B编码。B编码是一种简洁的数据组织方式，支持4种数据类型：bytestrings、integers、lists和dictionaries。integers、lists和dictionaries类型分别以字母i、l、d作为首定界符，以字母e作为尾定界符。bytestrings类型不使用首/尾定界符，其格式为<十进制表示的字符串长度>：<字符串>，如4:spam表示字符串“spam”。这4种数据类型嵌套使用构成了.torrent文件的内容。

       我们先看下所举例子的种子文件内容，种子文件中的服务器内容如下：

**d8:announce39:http://tracker.publicbt.com:80/announce13:announce-listll39:http://tracker.publicbt.com:80/announceel32:http://9.rarbg.com:2710/announceel44:udp://tracker.openbittorrent.com:80/announceel42:http://tracker.torrentbay.to:6969/announceel39:http://tracker.torrent.to:2710/announceel27:http://pow7.com:80/announceel28:http://10.rarbg.com/announceel38:http://exodus.desync.com:6969/announceel42:http://tracker.novalayer.org:6969/announceel35:udp://tracker.1337x.org:80/announceee**

其中的一些主要成份如下：

    ●announce：tracker服务器的URL，本例中为http：//tracker.cnxp.com:8080/announce。

    ●announce-list：可选。备用tracker服务器的URL列表，本例中为http://tracker.cnxp.com:8080/announce,http://btfans.3322.org:6969/announce等。

    ●creationdate：可选。.torrent文件的创建日期，使用标准的UNIX时间，本例中为1152105243。

    ●comment：可选。.torrent文件制作者添加的任意格式的说明。

    ●createdby：可选。制作.torrent文件的工具，本例中使用的制作工具是BitComet/0.67。

    ●encoding：可选。发布的资源使用的编码方式，在本例中使用的是GBK。

 ●info：发布的文件的信息。有两种格式，单文件格式和多文件格式。单文件格式包括length、md5sum（可选）、name、piecelength、pieces；多文件格式包括files、name、piecelength、pieces，其中files包括length、path、md5sum（可选），每一个文件都有单独的length、path、md5sum（可选）。

 另外，还有一些可选项，这里就不在累述。

**1.2 tracker HTTP/HTTPS协议**

BT客户端依次向.torrent中的trackerr服务器发送连接请求，以获得正在下载该文件的对等方列表。如果连接成功获得列表，就关闭连接，尝试与列表中的对等方建立连接；如果不成功，尝试下一个tracker服务器。

**第一、客户端获取Tracker服务器的IP地址**

[http://blog.chinaunix.net/attachment/201201/10/26548237_132620522428m5.png](http://blog.chinaunix.net/attachment/201201/10/26548237_132620522428m5.png)

[http://blog.chinaunix.net/attachment/201201/10/26548237_1326205275gVhH.png](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205275gVhH.png)[http://blog.chinaunix.net/attachment/201201/10/26548237_1326205319XXku.png](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205319XXku.png)[http://blog.chinaunix.net/attachment/201201/10/26548237_1326205387X4SY.png](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205387X4SY.png)[http://blog.chinaunix.net/attachment/201201/10/26548237_1326205391mcTc.png](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205391mcTc.png)

**图一 客户端发送DNS请求包**

从DNS服务器的响应来看，我们可以获取Tracker服务器的IP地址：

10.rarbg.com：是一个别名服务器

       主名称：tracker.publicbt.com：95.211.88.54、95.211.88.49、95.211.88.51、

       9.rarbg.com：                    83.149.126.97

               exodus.desync.com：              208.83.20.164

               pow7.com：                       同10.rarbg.com

               tracker.novalayer.org：          194.54.80.150

               tracker.publicbt.com：           95.211.88.54、95.211.88.49、95.211.88.51

       tracker.torrent.to：             127.0.0.1

               tracker.torrentbay.to：          109.235.55.11

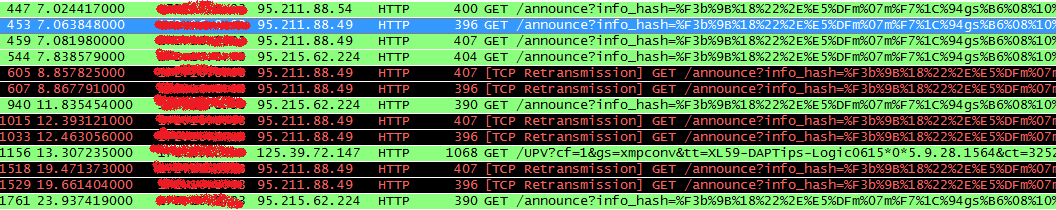
               tracker.1337x.org:               95.215.62.224

               tracker.openbittorrent.com: 95.215.62.26

       以上为各Tracker服务器的IP地址。

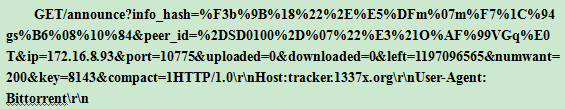
**第二、TrackerHTTP/HTTPS协议**

BT客户端依次向.torrent文件中的tracker服务器发送连接请求，以获取peers列表（主要是IP地址和监听端口）。如果连接成功获取列表，就关闭连接，尝试与列表中的对等方建立连接；如果不成功，尝试下一个tracker服务器。

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205588slRP.png)

**图二 BT客户端向Tracker服务器发送HTTP请求**

447号分组中的HTTP部分内容如图6所示。

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205642mo6W.png)

**图三 HTTP部分内容**

其中一些成分的含有如下：

●     info\_hash：.torrent文件中的info部分的sha1效验码，共20字节。Tracker服务器通过它在发布列表中找到对应的记录。

●     peer\_id：BT客户端的唯一性标识，在客户端启动时产生，共20bit。貌似没有具体的产生peer-id的算法，只要求能够保证唯一性即可。

●     ip：可选，IP地址，没有的话服务器会自己找到。

●     port：监听端口，这里为10775。

●     uploaded/downloaded：上传/下载的字节数（从客户机向Tracker服务器发送”started”开始计算）。

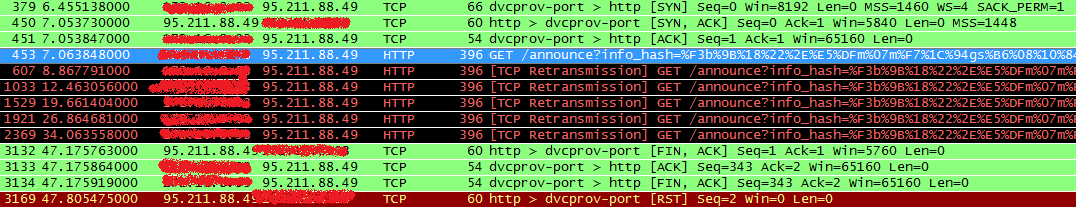
●     left：还需下载的字节数。

●     numwant：可选。客户端希望从Tracker服务器得到的对等方的数目。

●     key:可选。一个扩展的唯一性标识，即使改变了IP地址，也可以使用该字段标识该BT客户机。

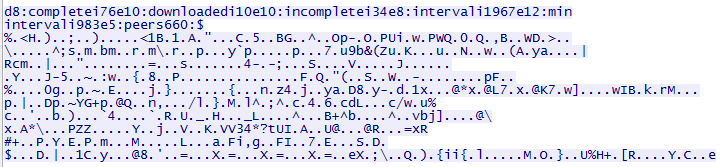
●     compact：压缩标志。如果值为1表示接受压缩格式的对等方列表，即用6byte表示一个对等方      （前4byte表示IP地址，后2byte表示端口号）；值为0表示不接受。

       另外，在与Tracker服务器交互的过程中，有很多服务器的已经被封杀禁止，所以，会有很多这样的结果。

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205739yreP.png)

**图四 被封后的Tracker服务器交互过程**

Tracker服务器有个tracker程序来管理这些请求，得到这一串代码后就会用info\_hash来查找列表，若找到就可以下载。接着就会反连（NatCheck）客户端的IP地址和端口，判断它是内网用户还是公网用户。接下来服务器返回现在正在下载这个文件的所有公网用户的IP和端口。返回的部分数据如图五所示。

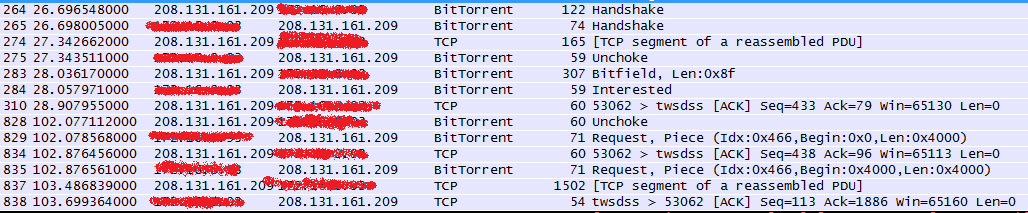
[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205769ep0R.png)

**图五 HTTP之上部分数据**

其中“660：”以及之前的部分使用的是ASCII字符集，“660：”之后的部分是用16进制表示的二进制数。从分组内容可以看出：完成整个文件下载的peers数为76；正在下载的peers数目为10个；还没有完成该文件下载的peers数目为34；interval的值为1967，也就是说BT客户端最多每隔1967个时间单位就与tracker服务器重新联系一次；最少时间间隔为983；peer部分共有660个字节，由于BT客户端支持对等方列表的压缩，即6个字节表示一个对等方，也就说返回的对等方个数为110个。

**1.3 peer wire协议**

BT客户机会尝试与返回的对等方列表中的部分对等方建立连接，下面是对等放列表中的208.131.161.209:53062为例，分析一下对等方之间的交互过程。如图六所示，只分析TCP之上的部分。约定对等方A指的是208.131.161.209:53062，对等方B指的是172.16.8.93:3012.

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205862gU77.png)

**图六 对等方间通信过程**

建立TCP连接之后，对等方之间的交互过程包括以下几步：

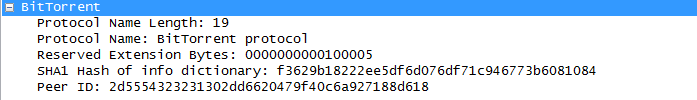
（1）       握手，通过Handshake分组实现。

（2）       互换所拥有的资源的情况。通过Bitfield分组实现。该例中，对等方B尚未下载任何资   源，故公布资源拥有情况的只有对等方A。对等方A通过分组283公布了自己资源拥有情况。

（3）       互通对资源的意愿情况，包括interested、nointerested、choke、unchoke等4种。

（4）       互相请求资源，通过request piece、piece分组实现。

（5）       断开连接。因为peer wire协议使用了TCP方式，对等方A与对等方B断开连接时，只需要断开他们之间的TCP连接即可。

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205885Dxod.png)

**图七 Handshake数据包的上层内容**

握手：Handshake：

       pstrlen: 的字符串长度，单个字节。

           pstr: 协议的标识符，字符串类型。

           reserved: 8 个保留字节。当前的所有实现都使用全0.这些字节里面的每一个字节都可以用来                      改变协议的行为。来自Bram 的邮件建议应该首先使用后面的位，以便可以使用前面                      的位来改变后面位的意义。

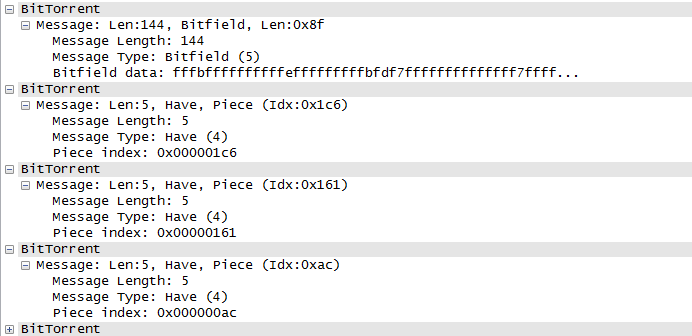
          info\_hash: 元信息文件中info 键(key)对应值的20 字节SHA1 哈希。这个nfo\_hash和在tracker                      请求中info\_hash 是同一个。

          peer\_id: 用于唯一标识客户端的20 字节字符串。这个peer\_id 通常跟在tracker 请求中传送                      的peer\_id相同(但也不尽然，例如在Azureus，就有一个匿名选项)。

       在BitTorrent 协议1.0 版本，pstrlen = 19, pstr = “BitTorrent protocol”。

       连接的发起者应该立即发送握手报文。如果接收方能够同时地服务多个torrent，它会等待发起者的握手报文(torrent 由infohash 唯一标识)。尽管如此，一旦接收方看到握手报文中的info\_hash 部分，接收方必须尽快响应。tracker 的NAT-checking 特性不会发送握手报文的peer\_id 字段。

       如果一个客户端接收到一个握手报文，并且该客户端没有服务这个报文的info\_hash，那么该客户端必须丢弃该连接。如果一个连接发起者接收到一个握手报文，并且该报文中peer\_id 与期望的peer\_id 不匹配，那么连接发起者应该丢弃该连接。注意发起者可能接收来自tracker 的peer 信息，该信息包含peer 注册的peer\_id。来自于tracker 的peer\_id 需要匹配握手报文中的peer\_id。

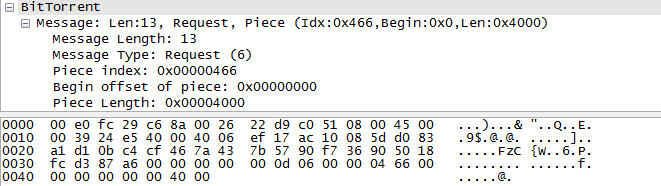
[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205917mX8w.png)[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205924pKSr.png)

**图八 Bitfiled数据包上层数据**

bitfield:

       itfield 报文可能仅在握手序列发送之后，其他消息发送之前立即发送。它是可选的，如果一个客户端没有piece(片)，就不需要发送该报文。bitfield 报文长度可变，其中x 是bitfield 的长度。payload 是一个bitfield，该bitfield 表示已经成功下载的piece(片)。第一个字节的高位相当于piece 索引0。设置为0 的位表示一个没有的piece，设置为1 的位表示有效的和可用的piece。末尾的冗余位设置为0。

       长度不对的bitfield 将被认为是一个错误。如果客户端接收到长度不对的bitfield 或者bitfield 有任一冗余位集，它应该丢弃这个连接。

[](http://blog.chinaunix.net/attachment/201201/10/26548237_1326205998EER4.png)

**图九 request piece数据包上层部分**

request piece分组结构如上图所示。

       该报文长度固定，用于请求一个块。payload包含如下信息：

       index：整数，指定从零开始的piece索引。

       begin：整数，指定piece中从零开始的字节偏移。

       length：整数，指定请求的长度。