P2PCDN系统通信消息格式定义

[一、前言 2](#_Toc462392381)

[二、Peer与Tracker消息格式定义 2](#_Toc462392382)

[2.1、Peer请求Peerlist消息格式定义 2](#_Toc462392383)

[2.1.1 Peer发送 2](#_Toc462392384)

[2.1.2 Tracker响应 2](#_Toc462392385)

[2.2、Peer上报信息消息格式定义 3](#_Toc462392386)

[2.3、Peer接收热点预推消息格式定义 7](#_Toc462392387)

[2.4、Peer NAT穿透消息格式定义 7](#_Toc462392388)

[2.4.1 Peer UDP心跳 7](#_Toc462392389)

[2.4.2 Tracker UDP心跳响应 7](#_Toc462392390)

[2.4.3 Peer UDP请求协助穿透 8](#_Toc462392391)

[2.4.4 Tracker要求Peer穿透 8](#_Toc462392392)

[三、Peer与Peer间消息格式定义 9](#_Toc462392393)

[3.1、Peer握手请求报文 9](#_Toc462392394)

[3.2、Peer握手响应报文 10](#_Toc462392395)

[3.3、Peer拥有报文 10](#_Toc462392396)

[3.4、Peer请求分片报文 11](#_Toc462392397)

[3.5、Peer分片响应报文 11](#_Toc462392398)

[四、Tracker与CC消息格式定义 12](#_Toc462392399)

[4.1、TS向CC获取URLHash-Infohash对应全量表 12](#_Toc462392400)

[4.2、TS向CC获取预推任务列表 13](#_Toc462392401)

[4.3、TS向CC请求某contentID 的urlhash、infohash列表 14](#_Toc462392402)

[4.4、TS向CC上报Peer内容获取连接数 14](#_Toc462392403)

[4.5、TS向CC上报Peer NAT穿越成功率 15](#_Toc462392404)

[4.6、TS向CC上报Peer 调度服务时延 16](#_Toc462392405)

[4.7、TS向CC上报Peer 切片请求时延 17](#_Toc462392406)

[4.8、TS向CC上报Peer上行、下行流量 18](#_Toc462392407)

[4.9、TS向CC上报Peer上行、下行带宽峰值 18](#_Toc462392408)

[4.10、TS向CC上报Peer存储总量、可用存储容量 19](#_Toc462392409)

[4.11、TS向CC上报设备CPU、内存占用率 20](#_Toc462392410)

[4.12、TS向CC上报TS节点管理数 21](#_Toc462392411)

[4.13、TS向CC上报TS调度成功率 21](#_Toc462392412)

[五、IP与CC消息格式定义 22](#_Toc462392413)

[5.1、IP向CC发布内容元数据信息 22](#_Toc462392414)

[5.2、IP从CC删除内容元数据信息 23](#_Toc462392415)

[六、CDN 与CC消息格式定义 24](#_Toc462392416)

[6.1、内容注入 24](#_Toc462392417)

[6.2、内容删除 24](#_Toc462392418)

[6.1、内容预推送 24](#_Toc462392419)

# 一、前言

本文档描述P2PCDN系统各模块通信消息格式定义。

# 二、Peer与Tracker消息格式定义

Peer和Tracker之间的通信通过两条Socket完成。一条长连接TCP Socket用于完成Peerlist请求、Peer信息上报，以及接收Tracker的热点内容预推指令；一条UDP Socket用于进行NAT穿透。

## 2.1、Peer请求Peerlist消息格式定义

### 2.1.1 Peer发送

Peer给Tracker发送的请求指令格式：

POST /get\_peerlist HTTP/1.1

Host: xxx

Content-Length: xxxx

Connection: Keep-Alive

{ "PeerID": "Peer’s PeerID",

"URLHash": "URL hash value"

}

说明：1.采用POST协议发送请求消息，请求路径为 /get\_peerlist

2. Host，填写Tracker的IP地址，或者其域名（若有的话）

3. Content-Length，为JSON负载的长度

4. PeerID，Peer的特征串，每个Peer节点唯一，可以在初始随机生成，之后一直沿用（一般用SHA1哈希算法获取20字节值，即40字节可打印字符串）

5. URLHash，Peer所请求的下载任务URL哈希值

### 2.1.2 Tracker响应

Tracker接收到Peer请求后，给Peer发出Peerlist响应。

HTTP/1.1 200 OK

Server: xxx

Content-Length: xxxx

Connection: Keep-Alive

{

"URLHash": "Task URL Hash Value",

"ContentHash": "Task Content Hash Value",

"PeerList": [

{"PeerID": "Peer’s PeerID", "UDPIP": "Peer’s UDP Listen IP", "UDPPort": "Peer’s UDP Listen Port"},

…,

{"PeerID": "Peer’s PeerID", "UDPIP": "Peer’s UDP Listen IP", "UDPPort": "Peer’s UDP Listen Port"}

]

}

说明：1.采用HTTP/1.1 200 OK发送响应消息

2. Server，填写Tracker的IP地址，或者其域名（若有的话）

3. Content-Length，为JSON负载的长度

4. URLHash，为本次响应Peerlist对应的URL哈希值

5. ContentHash，为本次请求所对应的内容哈希值

6. PeerID，Peer的特征串，每个Peer节点的唯一标识

7. UDPIP，Peer所监听的UDP IP地址

8. UDPPort，Peer所监听的UDP Port端口

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

## 2.2、Peer上报信息消息格式定义

Peer给Tracker发送的上报消息采用HTTP POST协议。依据上报消息的不同，上报路径也不一样。

POST报文格式如下：

POST /xxxx HTTP/1.1

Host: xxx

Content-Length: xxxx

Connection: Keep-Alive

{ ……

}

说明：1.采用POST协议发送请求消息，请求路径依据消息类型决定，见下表定义

2. Host，填写Tracker的IP地址，或者其域名（若有的话）

3. Content-Length，为JSON负载的长度

4. 每次上报消息仅含一行JSON串，具体如下表定义。

\* 以上JSON格式仅为便于阅读，真实使用时，每行为一个独立的JSON串，串中不包含换行符。

具体上报的消息格式定义

|  |  |  |  |
| --- | --- | --- | --- |
| 相关指标 | POST路径 | JSON串 | 说明 |
| Peer节点获取内容时连接的Peer节点数 | /api/peer/taskpeercnt | {  "peerID": "Peer’s PeerID",  "infohash": "Content’s Hash",  "peerCnt": Total Peer’s Count,  "timeStart": "Time Start Counting",  "timeEnd": " Time End Counting"  } | peerID,上报Peer的PeerID  infohash, 上报的Infohash  peerCnt, 上报的Peer连接数  timeStart, 统计开始时间,格式如 2016-09-23 1l:49:20  timeEnd, 统计结束时间 |
| Peer节点媒体数据连接NAT穿越成功率 | /api/peer/natsuccrate | {  "peerID": "Peer’s PeerID",  "infohash ": "Content’s Hash",  "NATSuccessRate": SuccessRate,  "timeStart": "Time Start Counting",  "timeEnd": " Time End Counting"  } | peerID, 上报Peer的PeerID  infohash, 上报的Infohash  NATSuccessRate, 上报的Peer连接数  timeStart, 统计开始时间  timeEnd, 统计结束时间 |
| Peer节点调度系统服务时延 | /api/peer/schedulingdelay | {  "peerID": "Peer’s PeerID",  "infohash": "Content’s Hash",  "schedulingDelay": Scheduling Delay,  "logTime": "Log Time"  } | peerID, 上报Peer的PeerID  infohash, 上报的Infohash  schedulingDelay, 上报调度时延  logTime, 记录时间 |
| 切片请求时延 | /api/peer/slicedelay | {  "peerID": "Peer’s PeerID",  "infohash": "Content’s Hash",  "sliceDelay": SliceDelay Delay,  "logTime": "Log Time"  } | peerID, 上报Peer的PeerID  infohash, 上报的Infohash  sliceDelay, 上报调度时延  logTime, 记录时间 |
| Peer节点北向上行、下行流量 | /api/peer/volume | {  "peerID": "Peer’s PeerID",  "uploadVolume": "Total Upload Volume",  "downloadVolume": "Total Download Volume",  "timeStart": "Time Start Counting",  "timeEnd": " Time End Counting"  } | peerID, 上报Peer的PeerID  uploadVolume, 上报上行流量  downloadVolume, 上报下行流量  timeStart, 统计开始时间  timeEnd, 统计结束时间 |
| Peer节点北向上行、下行带宽峰值 | /api/peer/peakrate | {  "peerID": "Peer’s PeerID",  "uploadPeakRate": Upload Peak Rate,  "downloadPeakRate": Download Peak Rate,  "timeStart": "Time Start Counting",  "timeEnd": " Time End Counting"  } | peerID, 上报Peer的PeerID  uploadPeakRate, 上报上行带宽峰值  downloadPeakRate, 上报下行带宽峰值  timeStart, 统计开始时间  timeEnd, 统计结束时间 |
| Peer本地存储总量、可用存储总量 | /api/device/disk | {  "deviceID": "Peer’s PeerID",  "type": "Peer",  "dTotal": Total Storage,  "dUsage": Used Storage,  "logTime": "Log Time"  } | deviceID, 上报Peer的PeerID  type, type: Ip, Tracker, Peer  dTotal, 上报存储总量  dUsage, 上报已用存储容量  logTime, 记录生成时间 |
| Peer节点CPU、内存占用率 | /api/device/info | {  "type": “Ip”,  "deviceID": "IDValue",  "cpu": cpu use rate,  "mem": mem use rate,  "logTime": "Log Time"  } | Type, 节点类型: Ip, Tracker, Peer  deviceID, 上报DeviceID  cpu, 上报CPU使用率，如55  mem, 上报内存使用率, 如65  logTime, 记录生成时间 |

Tracker返回响应格式如下。

HTTP/1.1 200 OK

Server: xxx

Content-Length: xxx

Connection: Keep-Alive

{"msg": "upload success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 2.3、Peer接收热点预推消息格式定义

具体格式定义，待与CDN厂商讨论后决定。

## 2.4、Peer NAT穿透消息格式定义

Peer的NAT穿透，使用第三方协助的方式，即每个Peer与Tracker都维持有UDP通信。

例：当PeerA要连接PeerB时，PeerA将PeerB的PeerIDB发送给Tracker。Tracker收到后，向PeerA发送PeerB的公网地址IPB和PortB，指令PeerA连接PeerB的公网地址和端口；同时，Tracker向PeerB发送PeerA的公网地址IPA和PortA ，指令PeerB连接PeerA。从而，PeerA 和PeerB 都从各自所处于的NAT之后，向对方发送了UDP报文，打通各自由内向外的NAT隧道。

为了保证Tracker所持有的Peer地址信息正确性，Peer需要周期性的向Tracker发送心跳信息。

### 2.4.1 Peer UDP心跳

Peer周期性向Tracker发送UDP的心跳信息。Tracker依据此信息来更新Peer的UDP监听地址、端口信息。以下JSON格式为UDP负载。

{ "Action": "HeartBeat",

"PeerID": "Peer’s PeerID",

"LocalUDPIP": "Local UDP IP",

"LocalUDPPort": Local UDP Port

}

说明：1. Action，标识本次报文的动作，HeartBeat为上报心跳

2. PeerID，Peer的特征串，每个Peer节点的唯一标识

3. LocalUDPIP，Peer所监听的本地UDP IP地址

4. LocalUDPPort，Peer所监听的本地UDP Port端口

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

### 2.4.2 Tracker UDP心跳响应

Tracker返回Peer的UDP心跳响应，内容包含Peer的公网IP地址和端口，Peer可以依据此信息判断自己的网络类型（如是否为NAT，即PubIP不等于LocalIP，暂时不适用）。

{ "Action": "HeartBeatResponse",

"PeerID": "Peer’s PeerID",

"LocalUDPIP": "Local UDP IP",

"LocalUDPPort": Local UDP Port,

"PublicUDPIP": "Public UDP IP",

"PublicUDPPort": Public UDP Port

}

说明：1. Action，标识本次报文的动作，HeartBeatResponse为心跳响应

2. PeerID，Peer的特征串，每个Peer节点的唯一标识

3. LocalUDPIP，Peer所监听的本地UDP IP地址

4. LocalUDPPort，Peer所监听的本地UDP Port端口

5. PublicUDPIP，Tracker所获取Peer的公网UDP IP地址

6. PublicUDPPort，Tracker所获取Peer的公网UDP Port端口

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

### 2.4.3 Peer UDP请求协助穿透

Peer需要连接NAT之后的另外一个节点时，需要Tracker进行协助穿透。穿透原理如本节开始所描述。

{ "Action": "NATTraversalAssist",

"PeerID": "Peer’s PeerID",

"TargetPeerID": "Target Peer’s PeerID",

"ContentHash": "Peer’s Task Content Hash"

}

说明：1. Action，标识本次报文的动作，NATTraversalAssist为请求Tracker协助穿透

2. PeerID，Peer的特征串，每个Peer节点的唯一标识

3. TargetPeerID，Peer所期望连接的目标Peer节点的PeerID

4. ContentHash，Peer正在进行下载任务的内容哈希值。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

### 2.4.4 Tracker要求Peer穿透

不论是对于请求协助穿透的Peer节点，还是被通知穿透的待连接Peer节点，Tracker都会向它们发送NAT穿透指令，指示其连接对端Peer节点。

{ "Action": "NATTraversalOrder",

"TargetPeerID": "Target Peer’s PeerID",

"TargetPeerIP": "Target Peer’s UDP IP",

"TargetPeerPort": "Target Peer’s UDP Port"

}

说明：1. Action，标识本次报文的动作，NATTraversalOrder 为Tracker要求Peer穿透

2. TargetPeerID，Peer的特征串，指示即将连接对端Peer的PeerID

3. TargetPeerIP，对端Peer节点的公网UDP地址

4. TargetPeerPort，对端Peer节点的公网Port端口

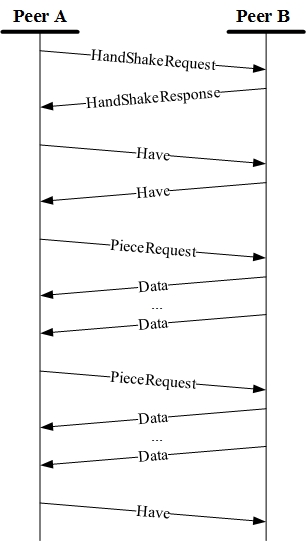
Peer收到指令后，使用与Tracker通信的同一个UDP Socket，向对端Peer发起P2P的握手报文。P2P通信格式见下文。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

# 三、Peer与Peer间消息格式定义

Peer通过Tracker实现NAT协助穿透后，Peer之间即可以发送P2P报文，进行P2P的交互。P2P的交互分为三个过程：1. 握手HandShake；2. 拥有Have；3. 数据交互Data。具体格式定义如下。

通信过程采用UDP协议实现。



## 3.1、Peer握手请求报文

Peer发送握手报文，报文中包含本机PeerID，待下载任务的Content哈希值等。

{ "Action": "P2PHandShakeRequest",

"PeerID": "Target Peer’s PeerID",

"ContentHash": "Task Content Hash Value"

}

说明：1. Action，标识本次报文的动作，P2PHandShakeRequest为Peer发送的握手请求

2. PeerID，Peer的特征串，指示Peer的PeerID

3. ContentHash，指示本次下载任务的内容哈希值

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

## 3.2、Peer握手响应报文

Peer收到握手请求报文后，判断本地是否有其请求的资源，并进行响应。

{ "Action": "P2PHandShakeResponse",

"Code": 0/1,

"PeerID": "Target Peer’s PeerID",

"ContentHash": "Task Content Hash Value"

}

说明：1. Action，标识本次报文的动作，P2PHandShakeResponse为Peer发送的握手响应

2. Code，Peer握手响应的特征值，1：指示成功，有资源；0：指示失败，无资源

3. PeerID，Peer的特征串，指示Peer的PeerID

4. ContentHash，指示本次交互任务的内容哈希值

当code的值为0时，标识本Peer无此资源，握手失败，无需进行后续交互。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

## 3.3、Peer拥有报文

当Peer握手成功后，Peer将向对端发出Have报文，表示本机所拥有的资源情况。如果Peer正在下载资源，那么当Peer的所具有的分片信息更新时，可以通过该报文通知与其连接的其它Peer（即该报文在握手成功后，随时可能发送）。

{ "Action": "P2PHave",

"PeerID": "Target Peer’s PeerID",

"ContentHash": "Task Content Hash Value",

"Pieces": [

{"Offset": xxxx, "Length": xxxx},

{"Offset": xxxx, "Length": xxxx},

….

{"Offset": xxxx, "Length": xxxx}

]

}

说明：1. Action，标识本次报文的动作，P2PHave为Peer发送的资源信息报文

2. PeerID，Peer的特征串，指示Peer的PeerID

3. ContentHash，指示本次交互任务的内容哈希值

4. Offset，标识所具有分片资源的起始偏移

5. Length，标识所具有分片资源的长度

当具有所有分片资源时，Offset = 0，Length = FileSize。

收到Have报文的Peer节点，只需更新本机Peerlist中该节点的分片状态信息，无需回复该报文。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

## 3.4、Peer请求分片报文

Peer选择某个具有其所需分片的节点，向其发送分片请求报文。

{ "Action": "P2PPieceRequest",

"PeerID": "Target Peer’s PeerID",

"ContentHash": "Task Content Hash Value",

"RequestOffset": xxxx,

"RequestLength": xxxx

}

说明：1. Action，标识本次报文的动作，P2PPieceRequest为Peer发送的分片请求报文

2. PeerID，Peer的特征串，指示Peer的PeerID

3. ContentHash，指示本次交互任务的内容哈希值

4. RequestOffset，标识所请求分片资源的起始偏移

5. RequestLength，标识所请求分片资源的长度

RequestLength的值可以大于1024。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

## 3.5、Peer分片响应报文

Peer接收到分片请求报文后，向请求节点发送分片响应。每个响应报文最多含1024字节数据；对于一个请求报文，Peer可以通过多个报文进行响应。

{ "Action": "P2PPieceResponse",

"PeerID": "Target Peer’s PeerID",

"ContentHash": "Task Content Hash Value",

"DataOffset": xxxx,

"DataLength": 1~1024

}

+ Data

说明：1. Action，标识本次报文的动作，P2PPieceResponse为Peer发送的分片响应报文

2. PeerID，Peer的特征串，指示Peer的PeerID

3. ContentHash，指示本次交互任务的内容哈希值

4. DataOffset，标识本次所发送分片资源的起始偏移

5. DataLength，标识本次所发送分片资源的起始偏移，取值1~1024

6. Data，即紧跟在JSON格式字符串之后，为分片数据的二进制

DataLength的取值为1~1024，代表每个UDP报文发送的数据量不超过1K字节，防止UDP报文被分片。

\* 以上JSON格式仅为便于阅读，真实使用时不包含换行符

# 四、Tracker与CC消息格式定义

## 4.1、TS向CC获取URLHash-Infohash对应全量表

Tracker周期性的向CC发送获取URLHash-Infohash对应权量表的请求，CC返回全量表的压缩文件。

全量表中，每一行存储一条URLHash与Infohash的对应值，采用“,”分隔。

Tracker向CC发送的请求消息如下：

GET /info/cc2ts.tar.gz HTTP/1.1

Host: www.p2pcdn-cc.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

Upgrade-Insecure-Requests: 1

CC收到请求后，回复压缩文件。

HTTP/1.1 200 OK

Accept-Ranges: bytes

Connection: Keep-Alive

Content-Length: 121

Content-Type: application/x-gzip

Date: Thu, 22 Sep 2016 11:49:45 GMT

Etag: "79-53d16903532af"

Keep-Alive: timeout=5, max=100

Last-Modified: Thu, 22 Sep 2016 11:00:01 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

………………

URLHash-InfoHash文件格式如下<urlhash,infohash>：

203c32cda6b3732d504d459b3553a58bf7369a39,2ba5e8ff9686c563528883591920b0f4e5edcc58

9062ff4fb860c9c664ac7380b471f2a44c038238,8b7f5436734dfb7e7ebc1cc8783971311b827164

……..

ecb384e79c78ed18ee3ccee4cb202c114d56d64d,aec9f16387ac9d51b47ed67505ac5133e9801d4f

## 4.2、TS向CC获取预推任务列表

Tracker周期性的向CC发送获取热点预推列表的请求，CC返回热点列表内容。

Tracker向CC发送的请求消息如下：

GET /api/prepush/contentID HTTP/1.1

Host: www.p2pcdn-cc.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

Upgrade-Insecure-Requests: 1

CC收到请求后，回复热点预推内容列表。

Connection: Keep-Alive

Content-Length: 58

Content-Type: application/json; charset=UTF-8

Date: Thu, 22 Sep 2016 11:56:31 GMT

Keep-Alive: timeout=5, max=100

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

[{"contentID":"2222222","lastTime":"1899-12-30 05:00:00"}]

## 4.3、TS向CC请求某contentID 的urlhash、infohash列表

Tracker依据contentID向CC请求该热点预推的urlhash、infohash列表。

Tracker向CC发送的请求消息如下：

GET /api/info/contentID?contentID=xxx HTTP/1.1

Host: www.p2pcdn-cc.com

User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:48.0) Gecko/20100101 Firefox/48.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3

Accept-Encoding: gzip, deflate

Connection: keep-alive

Upgrade-Insecure-Requests: 1

CC收到请求后，回复热点预推内容列表。

Connection: Keep-Alive

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

Date: Thu, 22 Sep 2016 11:56:31 GMT

Keep-Alive: timeout=5, max=100

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

{"contentID":"2222222","info":[{"url":"http:\/\/1","urlhash":"12222111","infohash":"2323233"}]}

## 4.4、TS向CC上报Peer内容获取连接数

Tracker采用POST接口方式，向CC上报Peer在获取内容时连接Peer的节点数。

POST /api/peer/taskpeercnt HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"infohash": "Content’s Hash", // 上报的Infohash

"peerCnt": Total Peer’s Count, // 上报的Peer连接数

"timeStart": "Time Start Counting", // 统计开始时间 2016-09-23 1l:49:20

"timeEnd": " Time End Counting" // 统计结束时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

………….

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.5、TS向CC上报Peer NAT穿越成功率

Tracker采用POST接口方式，向CC上报Peer在获取内容时NAT穿透成功率。

POST /api/peer/natsuccrate HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"infohash ": "Content’s Hash", // 上报的Infohash

"NATSuccessRate": SuccessRate, // 上报的Peer连接数

"timeStart": "Time Start Counting", // 统计开始时间

"timeEnd": " Time End Counting" // 统计结束时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.6、TS向CC上报Peer 调度服务时延

Tracker采用POST接口方式，向CC上报Peer在获取内容时的Peer调度时延。

POST /api/peer/schedulingdelay HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"infohash": "Content’s Hash", // 上报的Infohash

"schedulingDelay": Scheduling Delay, // 上报调度时延

"logTime": "Log Time" // 记录时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.7、TS向CC上报Peer 切片请求时延

Tracker采用POST接口方式，向CC上报Peer在获取每个切片内容时的时延。

POST /api/peer/slicedelay HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"infohash": "Content’s Hash", // 上报的Infohash

"sliceDelay": SliceDelay Delay, // 上报调度时延

"logTime": "Log Time" // 记录时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.8、TS向CC上报Peer上行、下行流量

Tracker采用POST接口方式，向CC上报Peer在采样时间内的上行、下行流量。

POST /api/peer/volume HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"uploadVolume": "Total Upload Volume", // 上报上行流量

"downloadVolume": "Total Download Volume", // 上报下行流量

"timeStart": "Time Start Counting", // 统计开始时间

"timeEnd": " Time End Counting" // 统计结束时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.9、TS向CC上报Peer上行、下行带宽峰值

Tracker采用POST接口方式，向CC上报Peer在采样时间内的上行、下行带宽峰值。

POST /api/peer/peakrate HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"peerID": "Peer’s PeerID", // 上报Peer的PeerID

"uploadPeakRate": Upload Peak Rate, // 上报上行带宽峰值

"downloadPeakRate": Download Peak Rate, // 上报下行带宽峰值

"timeStart": "Time Start Counting", // 统计开始时间

"timeEnd": " Time End Counting" // 统计结束时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.10、TS向CC上报Peer存储总量、可用存储容量

Tracker采用POST接口方式，向CC上报Peer存储总量和可用存储容量。

POST /api/device/disk HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: 266

[

{

"deviceID": "Peer’s PeerID", // 上报Peer的PeerID

"type": "Peer", // type: Ip, Tracker, Peer

"dTotal": Total Storage, // 上报存储总量

"dUsage": Used Storage, // 上报已用存储容量

"logTime": "Log Time" // 记录生成时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.11、TS向CC上报设备CPU、内存占用率

Tracker采用POST接口方式，向CC上报Tracker/Peer的CPU、内存占用率。

POST /api/device/info HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: 266

[

{

"type": “Ip”, // 节点类型: Ip, Tracker, Peer

"DeviceID": "IDValue", // 上报DeviceID

"cpu": 55, // 上报CPU使用率

"mem": 66, // 上报内存使用率

"logTime": "Log Time" // 记录生成时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.12、TS向CC上报TS节点管理数

Tracker采用POST接口方式，向CC上报Tracker管理节点的数量。

POST /api/tracker/peercnt HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"TSID": "TS Device ID", // 上报Tracker DeviceID

"peerCnt": Peer Counts, // 上报所管理的Peer节点数

"logTime": "Log Time" // 记录生成时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

………….

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

## 4.13、TS向CC上报TS调度成功率

Tracker采用POST接口方式，向CC上报Tracker的调度成功率。

POST /api/tracker/schedulingsuccrate HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxxx

[

{

"TSID": "TS Device ID", // 上报Tracker DeviceID

"schedulingSuccRate": Success Rate, // 上报Tracker调度成功率

"logTime": "Log Time" // 记录生成时间

},

…

]

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

………….

{"msg": "put success", "res": "0"} //res, 状态值，0表示成功，其他值为错误

# 五、IP与CC消息格式定义

## 5.1、IP向CC发布内容元数据信息

IP采用POST接口方式，向CC上报内容元数据，包括url、urlhash、infohash等。

POST /api/iptask/contentID HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: 266

{

"contentID": “ID value”,

"contentDes": "content description",

"contentName": “content name”,

"info":[

{"url": “content url”,

"urlhash": " content url hash",

"infohash": " content hash"},

……

{}

]

}

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

………….

{“msg”:”put success”,”res”:”0”} //res, 状态值，0表示成功，其他值为错误

## 5.2、IP从CC删除内容元数据信息

IP采用POST接口方式，从CC上删除内容元数据。

POST /api/iptask/contentID/delete HTTP/1.1

Host: www.p2pcdn-cc.com

Accept: \*/\*

Content-Type: application/json; charset=utf-8

Content-Length: xxx

{

"contentID": “ID value”

}

CC 返回响应。

HTTP/1.1 200 OK

Date: Thu, 22 Sep 2016 12:04:33 GMT

Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.1e-fips mod\_fcgid/2.3.9 PHP/5.4.16 mod\_wsgi/3.4 Python/2.7.5

X-Powered-By: PHP/5.4.16

Content-Length: xxx

Content-Type: application/json; charset=UTF-8

………….

{“msg”:”delete success”,”res”:”0”} //res, 状态值，0表示成功，其他值为错误

# 六、CDN 与CC消息格式定义

**按照中兴技术方案设计**

## 6.1、内容注入

## 6.2、内容删除

## 6.1、内容预推送