|  |
| --- |
| **算法1:** flvTag切片发送 |
| **输入:** 字节数组flv\_tag |
| **while** next flv\_tag **do**  **if** len(flv\_tag) <= MAX\_RTP\_PAYLOAD\_LEN **then**  rp ← Rtp包(initialization)  rp.marker ← true  rp.payload ← flv\_tag  发送rp  **else**  slice\_num ← len(flv\_tag)%MAX\_RTP\_PAYLOAD\_LEN + 1  **for** i=0 to slice\_num **do**  rp ← Rtp包(initialization)  **if** !last slice **then**  rp.marker ← false  **else**  rp.marker ← true  **end if**  rp.payload ← flv\_tag[i\*MAX\_RTP\_PAYLOAD\_LEN,  (i+1)\*MAX\_RTP\_PAYLOAD\_LEN]  发送rp  **end for**  **end if**  **end while** |

|  |
| --- |
| **算法2:** Rtp排序与重传 |
| **输入:** 新的Rtp数据包rp  **输出:** 有序的Rtp数据包pkt |
| **while** next rp **do**  firstSeq ← rtpQueue[0].Seq;  rtpQueue[rp.Seq - firstSeq] ← rp;  **for** len(rtpQueue) > 等待区大小 **do**  p ← rtpQueue[0];  **if** p为空 **then**  重传p;  **end if**  输出 p  **for** rtpQueue[0]不为空 **do**  p ← rtpQueue[0];  输出p  **end for**  **end while** |

|  |
| --- |
| **算法3:** Rtp重组flvTag |
| **输入:** 有序的Rtp数据包rp  **输出:** flv Tag字节数组flvTag |
| **while** next rp **do**  marker ← rp.marker;  pos ← 0  **if** marker为0 **then**  **if** rp为初始分片 **then**  payload ← rp.payload;  TagSize ← 解析playload;  flvTag ← 长度TagSize的字节数组(initialize);  flvTag[0:len(payload)] ← payload;  pos ← len(payload)  **else**  flvTag[pos:len(payload)] ← payload;  pos ← pos + len(payload)  **end if**  **else**  flvTag[pos:len(payload)] ← payload;  pos ← 0  **return** flvTag  **end if**  **end while** |