实时消息协议---流的分块

摘要：

本备忘录描述实时消息协议块流。块流是一种应用层协议，主要用于通过一种合适的传输层协议（例如 TCP）复用、打包多媒体数据流（音频，视频和交互数据）。

目录：

1. 简介

本文档规定实时消息协议块流（RTMP 块流）。分块为更高层的多媒体流协议提供复用和分组服务。

虽然 RTMP 块流是为协同 RTMP 协议工作而设计的，但是它任然可以处理任何发送消息流的协议。每个消息包含时间戳和负载类型标志。RTMP 块流和 RTMP 共同适用于各种形式的音视频应用，从点到点和点到多的实时直播，到 vod 服务，到交互式视频会议。

当配合像[TCP]这样的可靠传输协议使用时，RTMP 块流保证跨流的所有消息按时间戳序列一个接一个地传输。RTMP 块流不提供优先级或类似的控制，但是可以通过更高层的协议提供类似的服务。例如，视频直播服务可以基于每个消息的发送时间和答复时间选择丢弃视频消息，使慢的客户端能及时接受到音频消息。

RTM 块流包含自己的带内协议控制消息，并且提供了让更高层的协议嵌入用户控制消息的机制。

1.1 术语

本文档中的关键词”必须”、”一定不”、”要求”、”可以”、”不可以”、”应该”、”不应该”、”建议”、”可能”和”可选”的解释参考文档[BCP14][RFC2119]。

2. 定义

负载：

分组中所包含的数据。例如音频样本和压缩视频数据。负载格式和解释不在本文档的描述范围之内。

分组:

一个数据分组由固定的头和负载数据组成。一些底层协议可能需要定义分组的封装。

端口:

在一个给定计算机中区分不同目标的抽象。在 TCP/IP 协议中用一个小的整数来表示端口。OSI 中的传输选择器等同于端口的概念。

传输地址:

用于表示一个传输层终端的网络地址和端口的组合。例如 IP 地址和 TCP 端口。分组从源地址传输到目标地址。

消息流:

允许消息流动的逻辑上的通讯通道。

消息流 ID:

每个消息所关联的 ID，用于区分其所在的消息流。

块:

一个消息片段。消息通常在被放到网络上传输之前被分成小的部分并且被交错存取。分块确保跨流的所有消息按时间戳顺序被不断的传输。

块流:

允许块按照某一方向流动的逻辑上的通讯通道。块流可以从客户端流向服务端，也可以从服务端流向客户端。

块流 ID:

每个块所关联的用于区分其所在块流的 ID。

复用:

把分开的音视频数据整合到一个数据流，让多个音视频流可以同步地传输的过程。

解复用:

复用的反向过程。交互的音视频数据被收集成原始的音频数据和视频数据。

3. 字节序、对齐和时间格式

所有完整的字段都是按网络字节序被承载的，即，零字节是第一个字节，零位是一个字或字段中最显著的位。这种字节序就是所谓的”big-endian”。这种传输顺序的详细描述见[STD5]。除非另行说明，本文档中的数字都是十进制数。

在没有特别说明的情况下 RTMP 块流中的所有数据都是按字节对齐的。例如，一个 16 位字段可能在奇数字节偏移的位置。在标有延拓的地方，延拓字节应赋予零值。

RTMP 块流中的时间戳是用整数表示的，以毫秒为单位的相对时间，相对于一个未规定的起始时间。一般，每个块流的时间戳都从 0 开始，但是只要通讯的双方用统一的起始时间，可以不使用这种方法。要注意的是，这样跨多个块流（特别是不同主机之间）的同步需要用另外的机制来实现。

时间戳必须是单调递增的，并且是线性增长的。这样可以使应用程序处理同步，测量带宽，注入检测和进行流控制。

因为时间戳只有 32 位长，所以只能在 50 天以内循环。但是，因为流是可以不断的运行的，潜在地可以多年才结束？，所以 RTMP 块流应用程序必须对减法和比较使用模运算，并且能够处理这种回绕。只要通讯双方一致，任何合理的方法都可使用。例如，一个应用程序可以假设，相邻的时间戳是 2^31 毫秒，那么 1000 在 4000000000 之后，3000000000 在 4000000000 之前。

时间戳增量也是以毫秒为单位的无符号整数。时间戳增量可以是 24 位或 32 位长。

4. 消息格式(可以参考消息格式文档的第 4 节)

消息格式依赖于上层协议，可以被分成多个块以支持复用。但是消息格式必须包含下面这些创建块所必须的字段。

时间戳：

消息时间戳，占 4 个字节。

长度：

消息负载的长度，如果消息头无法被消减的话，应该包含在长度里，这个字段在块头中占 3 个字节。

类型 ID：

一些类型 ID 是为消息控制协议保留的。这些 ID 繁衍供 RTMP 块流协议和高层协议使用的信息。所有其他的 ID 都用于更高层协议。RTMP 块流协议对这些 ID 做不透明处理。事实上，RTMP 块流协议不需要用本字段的值来区分类型；所有消息可以是同类型的，或者应用可以使用本字段来区分同步轨道而不是区分类型。本字段占一个字节。

消息流 ID：

消息流 ID 可以是任何的值。被复用到相同的块流的消息流依靠其消息 ID 来解复用。除此之外，对于 RTMP 块流协议来说，这个值是不透明的。这个值在块头中占 4 个字节，并且使用小字节序。

5.握手

一个 RTMP 连接以握手开始。这里的握手和其他协议的握手不一样。这里的握手由三个固定大小的块组成，而不是可变大小的块加上头。

客户端（发起连接的一方）和服务端各自发送三个相同的块。这些块如果是客户端发送的话记为 C0，C1 和 C2，如果是服务端发送的话记为 S0，S1 和 S2。

5.1 握手队列

握手开始于客户端发送 C0，C1 块。

在发送 C2 之前客户端必须等待接收 S1 。

在发送任何数据之前客户端必须等待接收 S2。

服务端在发送 S0 和 S1 之前必须等待接收 C0，也可以等待接收 C1。

服务端在发送 S2 之前必须等待接收 C1。

服务端在发送任何数据之前必须等待接收 C2。

5.2 C0 和 S0 消息格式

C0 和 S0 是单独的一个字节 。

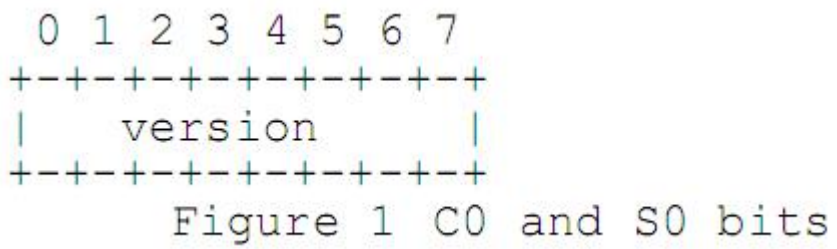

Figure 1 C0 and S0 bits

下面是 C0 和 S0 包的字段说明。

版本：8 位

在 C0 中这个字段表示客户端要求的 RTMP 版本 。在 S0 中这个字段表示服务器选择的 RTMP 版本。本规范所定义的版本是 3；0-2 是早期产品所用的，已被丢弃；4-31 保留在未来使用 ；32-255 不允许使用 （为了区分其他以某一字符开始的文本协议）。如果服务无法识别客户端请求的版本，应该返回 3 。客户端可以选择减到版本 3 或选择取消握手。

5.3 C1 和 S1 消息格式

C1 和 S1 消息有 1536 字节长，由下列字段组成。

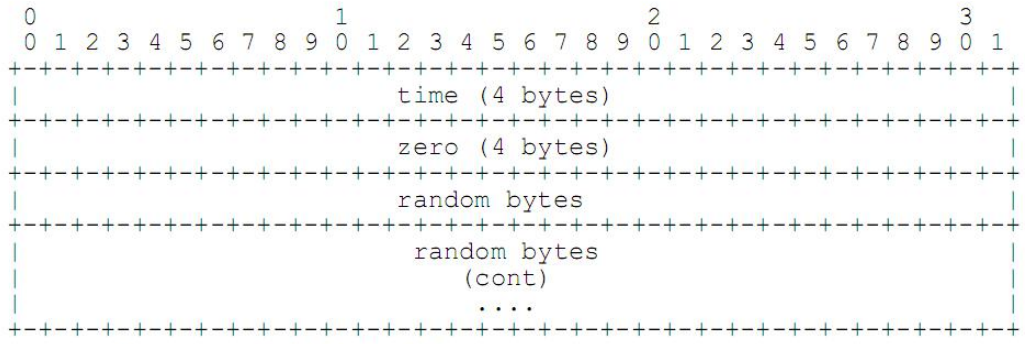

Figure 2 C1 and S1 bits

时间：4 字节

本字段包含时间戳。该时间戳应该是发送这个数据块的端点的后续块的时间起始点。可以是 0，或其他的任何值。为了同步多个流，端点可能发送其块流的当前值。

零：4 字节

本字段必须是全零。

随机数据：1528 字节。

本字段可以包含任何值。因为每个端点必须用自己初始化的握手和对端初始化的握手来区分身份，所以这个数据应有充分的随机性。但是并不需要加密安全的随机值，或者动态值。

5.4 C2 和 S2 消息格式

C2 和 S2 消息有 1536 字节长。只是 S1 和 C1 的回复。本消息由下列字段组成。

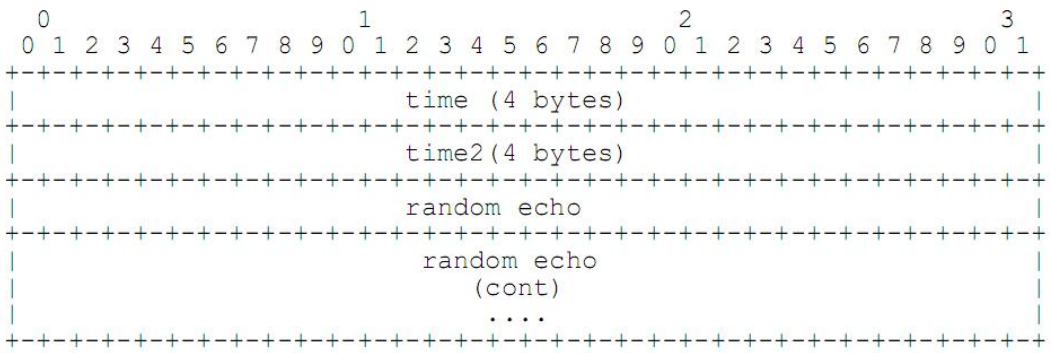

Figure 3 C2 and S2 bits

时间：4 字节

本字段必须包含对等段发送的时间（对 C2 来说是 S1，对 S2 来说是 C1）。

时间 2：4 字节

本字段必须包含先前发送的并被对端读取的包的时间戳。

随机回复：1528 字节

本字段必须包含对端发送的随机数据字段（对 C2 来说是 S1，对 S2 来说是 C1）。每个对等端可以用时间和时间 2 字段中的时间戳来快速地估计带宽和延迟。但这样做可能并不实用。

5.5 握手示意图

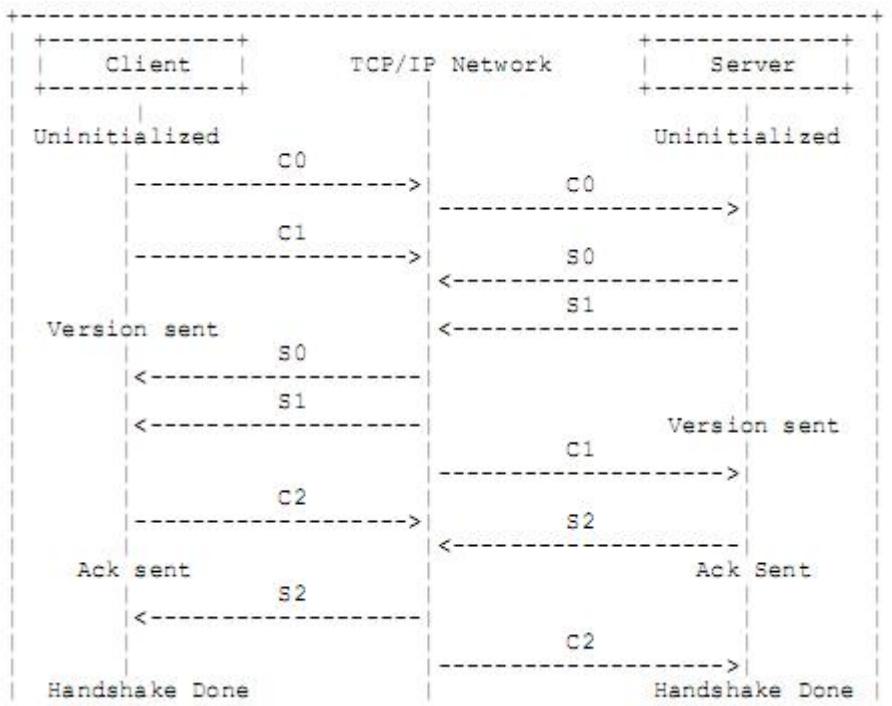

Figure 4 Pictorial Representation of Handshake

下表描述握手中的状态机。

状态	描述
未初始化	在这个状态中发送双方的版本 。此时客户端和服务端都未初始化。客户端在 C0 包中发送

	版本号。如果服务端支持那个版本，则发送 S0 和 S1 作为响应，否则，服务端采用适当的行为作为响应。在 RTMP 规范中应终止连接 。
版本已发送	在未初始化状态之后客户端和服务端都进入版本已发送状态。客户端等待 S1 包，服务端等待 C1 包。在接收到所等待的包后客户端发送 C2 包，服务端发送 S2 包。进入发送确认状态。
确认发送	客户端和服务端依次等待 S2 和 C2。
握手完成	客户端和服务端发送消息。

6. 分块

握手之后，连接开始复用一个或多个块流。每个块流承载来自一个消息流的一类消息。每个被创建的块都关联到一个唯一的块流 ID。所有的块都通过网络传输。在传输过程中，必须一个块发送完之后再发送下一个块。在接收端，每个块都根据块 ID 被收集成消息。

分块使高层协议的大消息分割成小的消息，保证大的低优先级消息不阻塞小的高优先级消息。

分块把原本应该消息中包含的信息压缩在块头中减少了小块消息发送的开销。

块大小是可配置的。这个可以在 7.1 节中描述的块消息中完成。最大块是 65535 字节，最小块是 128 字节。块越大 CPU 使用率越低，但是也导致大的写入，在低带宽下产生其他内容的延迟。块大小对每个方向都保持独立。

6.1 块格式

块由头和数据组成。块头由三部分组成：

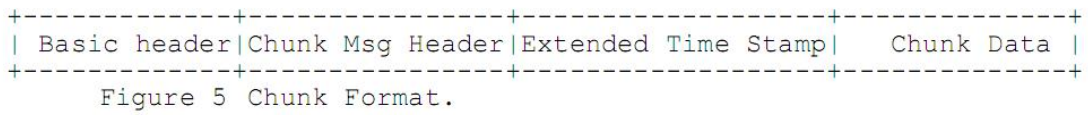

Figure 5 Chunk Format.

块基本头：1 到 3 字节

本字段包含块流 ID 和块类型。块类型决定编码的消息头的格式。长度取决于块流 ID。块流 ID 是可变长字段。

块消息头：0，3，7 或 11 字节。

本字段编码要发送的消息的信息。本字段的长度，取决于块头中指定的块类型。

扩展时间戳：0 个或 4 字节

本字段必须在发送普通时间戳（普通时间戳是指块消息头中的时间戳）设置为 0xffffff 时发送，正常时间戳为其他值时都不应发送本值。当普通时间戳的值小于 0xffffff 时，本字段不用出现，而应当使用正常时间戳字段。

6.1.1 块基本头

块基本头编码块流 ID 和块类型（在下图中用 fmt 表示）。块类型决定编码消息头的格式。块基本头字段可能是 1，2 或 3 个字节。这取决于块流 ID。

一个实现应该用最少的数据来表示 ID？。

本协议支持 65597 种流，ID 从 3-65599。ID 0、1、2 作为保留。0，表示 ID 的范围是 64-319（第二个字节+64）；1，表示 ID 范围是 64-65599（第三个字节*256+第二个字

节+64）；2 表示低层协议消息。没有其他的字节来表示流 ID。3-63 表示完整的流 ID。3-63 之间的值表示完整的流 ID。没有其他的字节表示流 ID。

0-5（不显著的）位表示块流 ID。

块流 ID 2-63 可以用 1 字节的字段表示

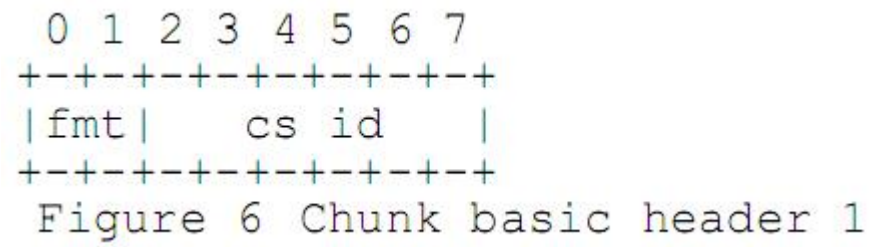

Figure 6 Chunk basic header 1

块流 ID 64-319 可以用 2-字节表示。ID 计算是（第二个字节+64）

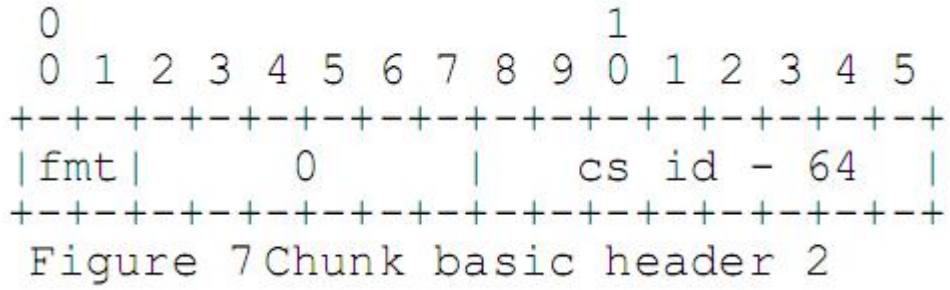

Figure 7 Chunk basic header 2

块流 ID64-65599 可以表示为 3 个字节 。ID 计算为第三个字节*255+第二个字节+64

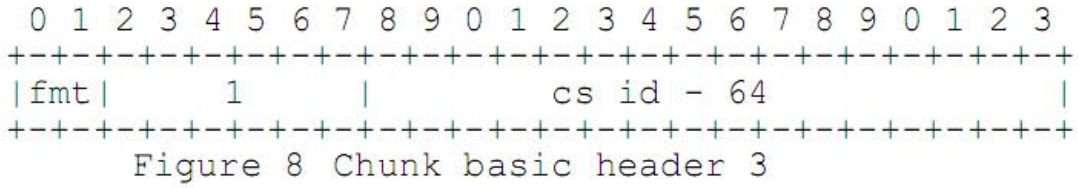

Figure 8 Chunk basic header 3

Cs id：6 位

本字段表示范围在 2-63 的块流 ID。值 0 和 1 表示本字段的 2 或 3 字节版本

Fmt：2 位

本字段标识块消息头的 4 种格式。每种流类型的块消息头在下一节中表示。

Cs id-64：8-16 位

本字段包含块流 ID 减去 64 的值。例如 365，应 在 cs id 中表示 1，而用这里的 1 6 位表示 301。

块流 ID 在 64-319 范围之内，可以用 2 个字节版本表示，也可以用 3 字节版本表示。

6.1.2 块消息头

有四种格式的块消息 ID，供块流基本头中的 fmt 字段选择。

一个实现应该使用最紧致的方式来表示块消息头。

6.1.2.1 类型 0

0 类型的块长度为 11 字节。在一个块流的开始和时间戳返回的时候必须有这种块。

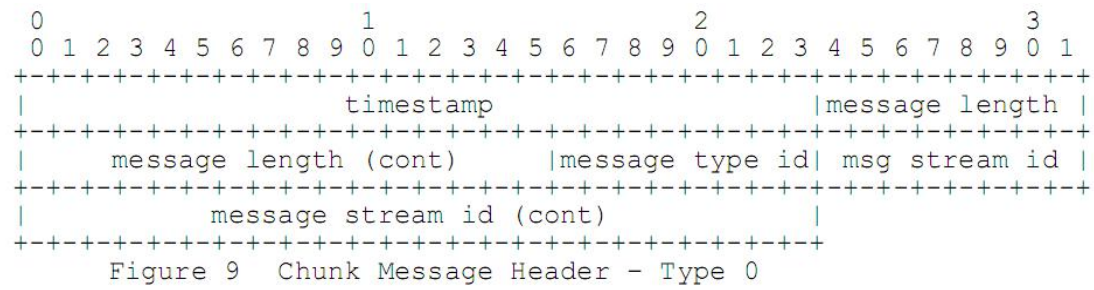

Figure 9 Chunk Message Header - Type 0

时间戳：3 字节

对于 0 类型的块。消息的绝对时间戳在这里发送。如果时间戳大于或等于 16777215（16 进制 0x00ffffff），该值必须为 16777215，并且扩展时间戳必须出现。否则该值就是整个的时间戳。

6.1.2.2. 类型 1

类型 1 的块占 7 个字节长。消息流 ID 不包含在本块中。块的消息流 ID 与先前的块相同。具有可变大小消息的流，在第一个消息之后的每个消息的第一个块应该使用这个格式。

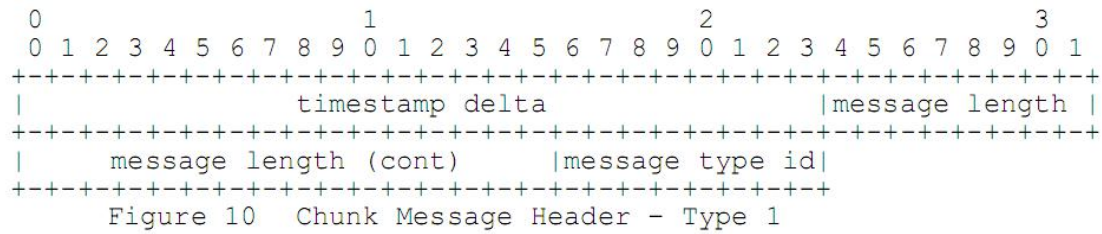

Figure 10 Chunk Message Header - Type 1

6.1.2.3. 类型 2

类型 2 的块占 3 个字节。既不包含流 ID 也不包含消息长度。本块使用的流 ID 和消息长度与先前的块相同。具有固定大小消息的流，在第一个消息之后的每个消息的第一个块应该使用这个格式。

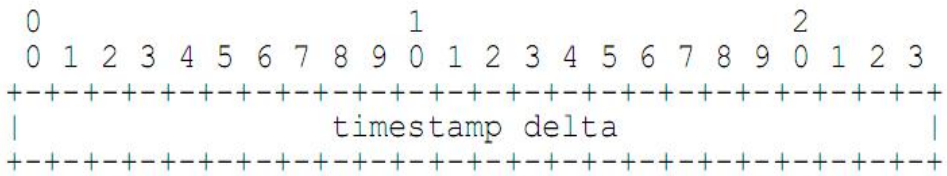

Figure 11 Chunk Message Header - Type 2

6.1.2.4 类型 3

类型 3 的块没有头。流 ID，消息长度，时间戳都不出现。这种类型的块使用与先前块相同的数据。当一个消息被分成多个块，除了第一块以外，所有的块都应使用这种类型。示例可参考 6.2.2 节中的例 2 。由相同大小，流 ID，和时间间隔的流在类型 2 的块之后应使用这种块。示例可参考 6.2.1 节中的例 1 。

如果第一个消息和第二个消息的时间增量与第一个消息的时间戳相同，那么 0 类型的块之后必须是 3 类型的块而，不需要类型 2 的块来注册时间增量。如果类型 3 的块在类型 0 的块之后，那么类型 3 的时间戳增量与 0 类型的块的时间戳相同。

时间戳增量：3 字节

对于类型 1 的块和类型 2 的块，本字段表示先前块的时间戳与当前块的时间

戳的差值。如果增量大于等于 1677215（16 进制 0x00ffffff），这个值必须是 16777215 ，并且扩展时间戳必须出现。否则这个值就是整个的增量。

消息长度：3 字节

对于类型 0 或类型 1 的块本字段表示消息的长度。

注意，这个值通常与负载长度是不相同的。The chunk payload length is the maximum chunk size for all but the last chunk， and the remainder (which may be the entire length， for small messages) for the last chunk.

消息类型 ID：1 字节

对于 0 类型和 1 类型的块，本字段发送消息类型。

消息流 ID：4 字节

对于 0 类型的块，本字段存储消息流 ID。通常，在一个块流中的消息来自于同一个消息流。虽然，由于不同的消息可能复用到一个块流中而使头压缩无法有效实施。但是，如果一个消息流关闭而另一个消息流才打开，那么通过发送一个新的 0 类型的块重复使用一个存在的块流也不是不可以。

6.1.3. 扩展时间戳

只有当块消息头中的普通时间戳设置为 0x00ffffff 时，本字段才被传送。如果普通时间戳的值小于 0x00ffffff，那么本字段一定不能出现。如果时间戳字段不出现本字段也一定不能出现。类型 3 的块一定不能含有本字段。本字段在块消息头之后，块时间之前。

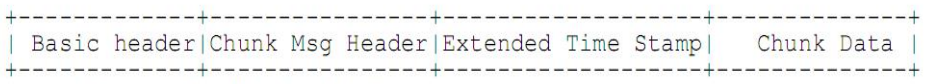

Figure 12 Chunk Format.

6.2 示例

6.2.1. 例 1

例 1 展示一个简单的音频消息流。这个例子显示了信息的冗余。

	Message Stream ID	Message Type ID	Time	Length
Msg # 1	12345	8	1000	32
Msg # 2	12345	8	1020	32
Msg # 3	12345	8	1040	32
Msg # 4	12345	8	1060	32

Figure 13 Sample Audio messages to be made into chunks

下表显示了这个流产生的块。从消息 3 开始，数据传输开始优化。在消息 3 之后，每个消息只有一个字节的开销。

	Chunk Stream ID	Chunk Type	Header Data	No.of Bytes After Header	Total No.of Bytes in the Chunk
Chunk#1	3	0	delta: 1000 length: 32 type: 8 stream	32	44

			ID:1234 (11bytes)		
Chunk#2	3	2	20 (3 bytes)	32	36
Chunk#3	3	3	none(0 bytes)	32	33
Chunk#4	3	3	none(0 bytes)	32	33

Figure 14 Format of each of the chunks of audio messages above

6.2.2 例 2

例 2 演示一个消息由于太长，而被分割成 128 字节的块。

	Message Stream ID	Message TYpe ID	Time	Length
Msg # 1	12346	9 (video)	1000	307

Figure 15 Sample Message to be broken to chunks

下面是产生的块。

	Chunk Stream ID	Chunk Type	Header Data	No. of Bytes after Header	Total No. of bytes in the chunk
Chunk#1	4	0	delta: 1000 length: 307 type: 9 streamID: 12346 (11 bytes)	128	140
Chunk#2	4	3	none (0 bytes)	128	129
Chunk#3	4	3	none (0 bytes)	51	52

Figure 16 Format of each of the broken chunk.

7.协议控制消息

RTMP 块流支持一些协议控制消息。这些消息包含 rtmp 块流协议需要的信息，并且不能传播到更高层的协议。当前有两种消息用于 RTMP 块流。一个协议消息用于设置块大小，另一个用于由于余下的块不可得而取消一个消息。

协议控制消息应该含有消息流 ID 0（称为控制流）和块流 ID 2，并且应有最高的发送优先级。

每个协议控制消息有一个固定大小的负载，并且作为一个独立的块发送。

7.1 设置块大小

协议控制消息 1，设置块大小，用于通知对端新的最大块大小。

块大小可以设置默认值，但是客户端或服务端可以改变和更新这个值。例如，假设一个客户端想发送 131 字节的音频数据，而块大小是 128。那么，客户端可以向服务端发送一个协议消息通知对方块大小设置为 131 字节。然后，客户端就可以在一个块中发送音频数据。

最大块大小是 65535 字节。块大小在每个方向上保持独立。

7.2 取消消息

本协议控制消息用于通知对等端，不用再等待接收块来完成消息，而可以丢弃以前通过块流接收到的消息了。 这个协议消息的负载是对等端接收到的块流 ID。一个应用程序在关闭的时候发送这个消息，以告诉对方不需要继续处理消息了。

8. 参考文献

8.1. 标准参考

[1] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[2] Crocker， D. and Overell， P.(Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

[RFC2119] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[RFC2234] Crocker， D. and Overell， P.(Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

8.2.资料参考

[3] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in TCP and Its Effect on Busy Servers"， Proc. Infocom 1999 pp. 1573-1583.

[Fab1999] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in TCP and Its Effect on Busy Servers"， Proc. Infocom 1999 pp. 1573-1583.

9.感谢

地址：

Adobe Systems Incorporated
345 Park Avenue
San Jose， CA 95110-2704

RTMP 消息格式

----------实时消息协议 消息格式

版权声明

摘要

本备忘录描述实时消息协议。实时消息协议是一种应用层协议，主要用于通过一种合适的传输层协议复用、打包多媒体数据流（音频，视频和交互数据）。

目录

1. 简介

本文档规定实时消息协议。实时消息协议规定通过传输层传输的消息的格式。

虽然 RTMP 被设计与实时消息块流协议一起工作，但是仍然可以用任何其他的传输协议来发送消息。RTMP 块流和 RTMP 一起适用于广泛的音视频应用。从点到点和点到多的实时直播，到 vod 服务，到交互式视频会议。

1.1 术语

本文档中的关键词”必须”、”一定不”、”要求”、”可以”、”不可以”、”应该”、”不应该”、”建议”、”可能”和”可选”的解释参考文档[BCP14][RFC2119]。

2. 定义

消息流:

允许消息流动的逻辑上的通讯通道。

消息流 ID:

每个消息所关联的 ID，用于区分其所在的消息流。

负载：

包中所包含的数据。例如音频样本和压缩视频数据。负载格式和解释不在本文档的描述范围之内。

分组:

一个数据分组包含固定的头和负载数据。一些底层协议可能需要定义分组的封装？。

3. 字节序、对齐和时间格式

所有完整的字段都是按网络字节序被承载的，即，在一个字的字段中零字节是第一个出现的字节，零位是最显著位。这种字节序就是所谓的”big-endian”。这种传输顺序的详细描述见[STD5]。除非另行说明，本文档中的数字都是十进制数。

在没有特别说明的情况下 RTMP 块流中的所有数据都是按字节对齐的。例如，一个 16 位字段可能在奇数字节位置。在标有延拓的地方，延拓字节应赋予零值。

RTMP 块流中的时间戳是用整数表示的以毫秒为单位的相对时间。相对于一个未指定的起始时间。一般，每个块流的时间戳都从 0 开始。但是，只要通讯的双方用统一的起始时间，可以不使用这种方法。要注意的是，这样跨多个块流（特别是不同主机之间）的同步需要用另外的机制来实现。

时间戳必须是单调递增的，并且应该是线性增长的。这样可以使应用程序处理同步，

测量带宽，注入检测和进行流控制。

因为时间戳只有 32 位长，所以只能在 50 天以内循环。但是因为流是可以不断的运行的，潜在地可以多年才结束？，所以 RTMP 块流应用程序必须对减法和比较使用模运算，并且能够处理这种回绕。只要通讯双方一致，任何合理的方法都可使用。例如，一个应用程序中，相邻的时间戳是 2^31 毫秒，那么 1000 在 4000000000 之后，3000000000 在 4000000000 之前。

时间戳增量也是以毫秒为单位的无符号整数。时间戳增量可以是 24 位或 32 位长。

4. 消息格式

服务端和客户端通过在网络上发送 RTMP 消息进行通讯。消息可能包含音频，视频，数据，或其他的消息。

RTMP 消息分头和负载两部分。

4.1 消息头

消息头包含下面的内容：

消息类型：

一个字节字段用于表示消息类型。范围在 1-7 内的消息 ID 用于协议控制消息。

负载长度：

三个字节字段用于表示负载的字节数。设置为 big-endian 格式。

时间戳：

四字节字段包含消息的时间戳。4 个字节用 big-endian 方式打包。

消息流 ID：

三字节字段标识消息流。这些字节设置为 big-endian 格式。

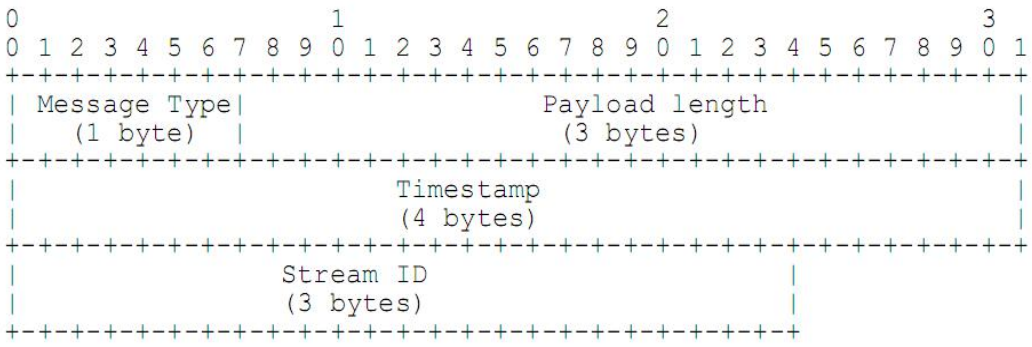

Figure 1 Message header

4.2 消息负载

负载时消息中包含的真实数据。例如，它可以是音频样本或压缩的视频数据。负载格式和解释不在本文档的范围之内。

5 协议控制消息

RTMP 保留消息类型 ID 在 1-7 之内的消息为协议控制消息。这些消息包含 RTMP 块流协议和 RTMP 协议本身要使用的信息。ID 为 1 和 2 用于 RTMP 块流协议。ID 在 3-6 之内用于 RTMP 本身。ID 7 的消息用于边缘服务与源服务器。

协议控制消息必须有消息流 ID 0 和块流 ID 2，并且有最高的发送优先级。

每个协议控制消息都有固定大小的负载。

5.1 设置块大小

协议控制消息 1，设置块大小，用于通知对等端使用新的最大块大小。

块大小的值被承载为 4 字节大小的负载。块大小有默认的值，但是如果发送者希望改变这个值，则用本消息通知对等端。例如，一个客户端想要发送 131 字节的数据，而块大小是默认的 128 字节。那么，客户端发送的消息要分成两个块发送。客户端可以选择改变块大小为 131 字节，这样消息就不用被分割为两个块。客户端必须向服务端发送本消息来通知对方块大小改为 131 字节。

最大块大小为 65536 字节。服务端向客户端通讯的块大小与客户端向服务端通讯的块大小互相独立。

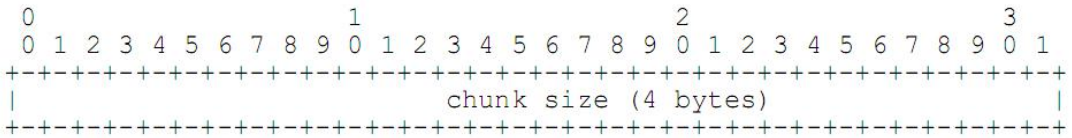

Figure 2 Pay load for the protocol message 'Set Chunk Size'

块大小：32 位

本字段承载新的块大小。新的块大小用于之后的这个块流得所有块。

5.2 取消消息

协议控制消息 2 是取消消息，用于通知正在等待接收块以完成消息的对等端，丢弃一个块流中已经接收的部分并且取消对该消息的处理。对等端把这个消息的负载当作要丢弃的消息的块流 ID。当发送者已经发送了一个消息的一部分，但是希望告诉接收者消息的余下部分不再发送时，发送本消息。

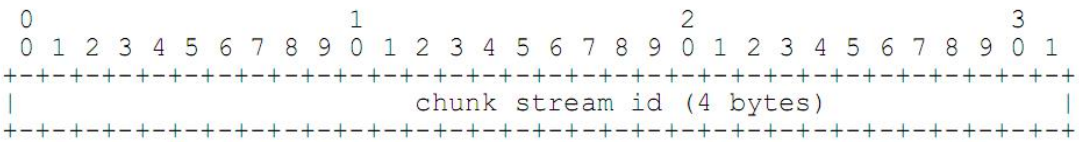

Figure 3 Pay load for the protocol message 'Abort Message'.

块流 ID：32 位

本字段承载要丢弃的消息所在的块流 ID。

5.3 确认（致谢）

客户端或服务端在接收到数量与窗口大小相等的字节后发送确认（致谢）到对方。窗口大小是在没有接收到接收者发送的确认（致谢）消息之前发送的字节数的最大值。服务端在建立连接之后发送窗口大小。本消息指定序列号。序列号，是到当前时间为止已经接收到的字节数。

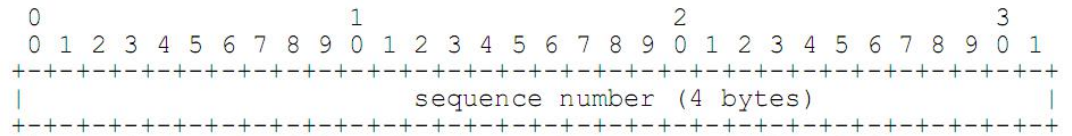

Figure 4 Pay load for the protocol message 'Acknowledgement'.

序列号：32 位

本字段含有到目前为止接收到的字节数。

5.4 用户控制消息

客户端或服务端发送本消息通知对方用户的控制事件。本消息承载事件类型和事件数据。

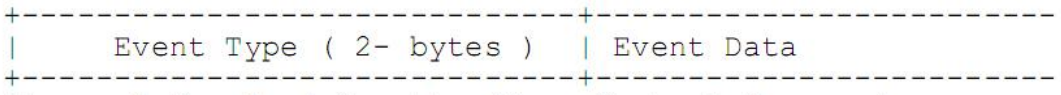

Figure 5 Pay load for the 'User Control Message'.

消息数据的头两个字节用于标识事件类型。事件类型之后是事件数据。事件数据字段是可变长的。

5.5 确认（致谢）窗口大小

客户端或服务端发送本消息来通知对方发送确认（致谢）消息的窗口大小。例如，服务端希望每当发送的字节数等于窗口大小时从客户端收到确认（致谢）。服务端在成功处理了客户端的连接请求后向客户端更新窗口大小。

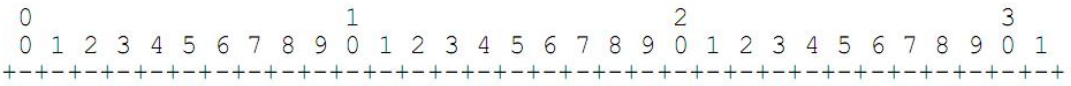

Figure 6 Pay load for 'Window Acknowledgement Size'.

5.6 设置对等端带宽

客户端或服务端发送本消息更新对等端的输出带宽。输出带宽值与窗口大小值相同。如果对等端在本消息中收到的值与窗口大小不相同，则发回确认（致谢）窗口大小消息。

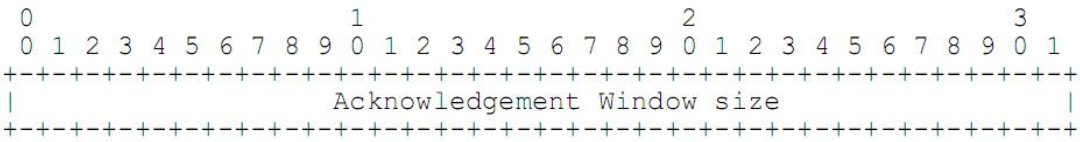

Figure 7 Pay load for 'Set Peer Bandwidth'

发送者可以在限制类型字段把消息标记为硬（0），软（1），或者动态（2）。如果是硬限制对等端必须按提供的带宽发送数据。如果是软限制，对等端可以灵活决定带宽，发送端可以限制带宽？。如果是动态限制，带宽既可以是硬限制也可以是软限制。

6. 参考

6.1. 标准参考

[1] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[2] Crocker， D. and Overell， P. (Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

[RFC2119] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[RFC2234] Crocker， D. and Overell， P. (Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

6.2. 参考资料

[3] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in TCP

and Its Effect on Busy Servers"， Proc. Infocom 1999 pp. 1573-
1583.
[Fab1999] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in
TCP and Its Effect on Busy Servers"， Proc. Infocom 1999 pp.
1573-1583.

地址:
Adobe Systems Incorporated
345 Park Avenue
San Jose， CA 95110-2704

RTMP 命令消息

摘要
本文档描述服务端和客户端之间通讯的各种类型的消息和命令。

目录

1. 简介

服务端和客户端之间交换的消息包括发送音频数据的音频消息，发送视频数据的视频消息，发送用户数据的数据消息，共享对象消息和命令消息。共享对象消息，在多个客户端和服务端之间管理分布数据。

命令消息在客户端和服务端之间承载 AMF 编码命令。客户端和服务端可以通过使用命令消息向对方请求远程过程调用？。

2. 定义

消息流:

允许消息流动的逻辑上的通讯通道。

消息流 ID:

每个消息所关联的 ID，用于区分其所在的消息流。

远程过程调用:

客户端或服务端调用一个远端的子例程或进程的请求。

元数据：

数据的描述。一部电影的元数据包括电影标题、时长、创建日期等。

应用实例：

客户端请求连接的服务端应用实例。

行为消息格式：

用于串行化 ActionScript 对象图形的二进制紧致格式。格式规范：

AMF0（http://opensource.adobe.com/wiki/download/attachments/1114283/amf0_spec_121207.pdf?version=1）

AMF3（http://opensource.adobe.com/wiki/download/attachments/1114283/amf3_spec_05_05_08.pdf?version=1）

3. 消息类型

服务端和客户端通过在互连网发送消息来通讯。消息包括音频消息，视频消息，命令消息，共享对象消息，数据消息，用户控制消息。

3.1 命令消息

命令消息承载用 AMF 编码的客户端与服务端之间的命令。消息类型为 20 的用 AMF0 编码，消息类型为 17 的用 AMF3 编码。

这些消息用于在远端实现连接，创建流，发布，播放和暂停等操作。状态，结果等命令消息用于通知发送者请求命令的状态。命令消息由命令名，传输 ID，和包含相关参数的命令对象组成。客户端或服务端可以使用命令消息向远端请求远程过程调用。

3.2 数据消息

客户端或服务端通过本消息向对方发送元数据和用户数据。元数据包括数据的创建时间、时长、主题等细节。消息类型为 18 的用 AMF0 编码，消息类型为 15 的用 AMF3 编码。

3.3 共享对象消息

共享对象是跨多个客户端，实例同步的 FLASH 对象（名值对的集合）。消息类型 kMsgContainer=19 用 AMF0 编码，kMsgContainerEx=16 用 AMF3 编码，这两个消息用于共享对象事件。每个消息可包含多个事件。

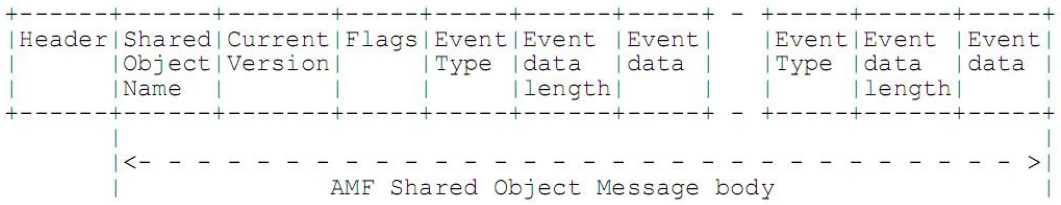

Figure 1 The shared object message format

支持下列的事件类型：

事件	描述
Use (=1)	客户端发送这个事件通知服务端创建一个命名的共享对象。
Release (=2)	当客户端删除共享对象时，发送这个事件通知服务端。
Requeset change (=3)	当请求改变一个共享对象的某个命名参数的关联的值时，客户端发送本消息。
Change (=4)	当某个命名参数的关联值被改变时，服务端发送本事件给所有的客户端。
Success (=5)	当接受请求改变事件后，服务端向发请求的客户端响应本消息。
SendMessage (=6)	客户端向服务端发送本事件广播一个消息。接收到本事件后服务端向包括发送本事件在内的所有客户端广播一个消息。
Status (=7)	针对一个错误状态，服务端向客户端发送本事件。
Clear (=8)	服务端向客户端发送本事件，清除一个共享对象。本事件也作为客户端在连接时发送 use 事件的响应。
Remove (=9)	服务端发送本事件使客户端删除一个插槽。
Request Remove (=10)	客户端删除一个插槽时向服务端发送本事件。
Use Success(=11)	当连接成功时服务端向客户端发送本事件。

3.4 音频消息

客户端或服务端发送本消息用于发送音频数据。消息类型 8 ，保留为音频消息。

3.5 视频消息

客户端或服务端使用本消息向对方发送视频数据。消息类型值 9，保留为视频消息。视频消息比较大，会造成其他消息的延迟。为了避免这种情况，这种消息被关联为最低优先级。

3.6 聚合消息

聚合消息是含有一个消息列表的一种消息。消息类型值 22，保留用于聚合消息。

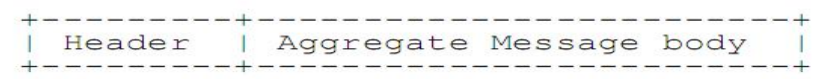

Figure 2 The aggregate message format

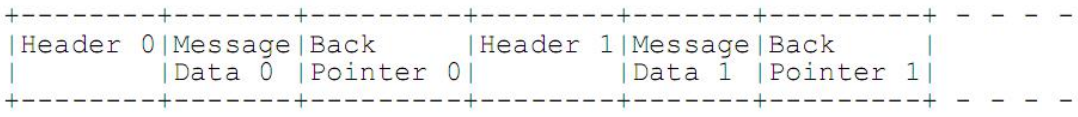

Figure 3 The aggregate message body format

后端包含前面消息的包含头在内的大小。这个设置匹配 flv 文件格式，用于后向搜索。

使用聚合消息的好处：

块流在每个块中最多发送一个块。因此增加块的大小并且使用聚合消息可以减少块数。

子消息可以在内存上连续存储，这样可以在调用系统通过网络发送数据时更有效。

3.7 用户控制消息

客户端或服务端发送本消息通知对方用户控制事件。关于本消息的格式，可以参考 RTMP 消息文档中的用户控制消息一节 。（译者注：用户控制消息是协议控制消息的一类，消息类型 ID 位 4）

支持下列类型的用户控制事件：

事件	描述
Stream Begin(=0)	服务端向客户端发送本事件通知对方一个流开始起作用可以用于通讯。在默认情况下，服务端在成功地从客户端接收连接命令之后发送本事件，事件 ID 为 0。事件数据是表示开始起作用的流的 ID。
Stream EOF(=1)	服务端向客户端发送本事件通知客户端，数据回放完成。如果没有发行额外的命令，就不再发送数据。客户端丢弃从流中接收的消息。4 字节的事件数据表示，回放结束的流的 ID。
StreamDry(=2)	服务端向客户端发送本事件通知客户端，流中没有更多的数据。如果服务端在一定周期内没有探测到更多的数据，就可以通知客户端流枯竭。4 字节的事件数据表示枯竭流的 ID。
SetBuffer Length(=3)	客户端向服务端发送本事件，告知对方自己存储一个流的数据的缓存的长度（毫秒单位）。当服务端开始处理一个流得时候发送本事件。事件数据的头四个字节表示流 ID，后 4 个字节表示缓存长度（毫秒单位）。
StreamIsRecorded(=4)	服务端发送本事件通知客户端，该流是一个录制流。4 字节的事件数据表示录制流的

	ID。
PingRequest(=6)	服务端通过本事件测试客户端是否可达。事件数据是 4 个字节的事件戳。代表服务调用本命令的本地时间。客户端在接收到 kMsgPingRequest 之后返回 kMsgPingResponse 事件。
PingResponse(=7)	客户端向服务端发送本消息响应 ping 请求。事件数据是接收 kMsgPingRequest 请求的 时间。

4 命令类型

客户端和服务端 交换 AMF 编码的命令。发送端发送命令消息。命令消息由命令名，传输 ID，和命令对象组成。命令对象由一系列的相关参数组成。例如，连接命令包含”app”参数，这个参数告知服务端，客户端要连接的应用名。接收端处理命令并且返回含有相同传输 ID 的响应。响应字符串含有_result 或_error 或一个方法名，例如，一个 verifyclient，或一个 contactExternalServer。

一个含有_result 或_error 的命令字符串表示一个响应。传输 ID，指示出响应所参考的显著的命令？。这个相当于 IMAP 或其他协议中的标签。命令字符串中的方法名表示发送端想要在接收端运行一个方法。

下列的类对象用于发送各种命令：

NetConnection-代表服务端和客户端之间连接的更高层的对象。

NetStream-代表发送音频流，视频流和其他相关数据的通道的对象。我们也发送像播放，暂停等控制数据流动的命令。

4.1 NetConnection 命令

NetConnection 管理服务端和客户端之间的双路连接。另外，它还支持异步远程方法调用。

下列的方法可以通过 NetConnection 发送。

----------连接

----------调用

----------关闭

----------创建流

4.1.1 连接

客户端向服务端发送一个连接命令请求连接到一个服务应用实例。

客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名。设置为”connect”
传输 ID	数字	总是设为 1
命令对象	对象	含有名值对的命令信息对象
可选的用户变量	对象	任何可选信息

下面是在连接命令的命令对象中使用的名值对的描述：

属性	类型	描述	示例值
App	字符	客户端要连接到的服务应用名	Testapp

	串		
Flashver	字符串	Flash 播放器版本。和应用文档中 getversion（）函数返回的字符串相同。	FMSc/1.0
SwfUrl	字符串	发起连接的 swf 文件的 url	file://C:/ FlvPlayer.swf
TcUrl	字符串	服务 url。有下列的格式。protocol://servername:port/appName/appInstance	rtmp://localhost::1935/testapp/instance1
fpad	布尔值	是否使用代理	true or false
audioCodecs	数字	指示客户端支持的音频编解码器	SUPPORT_SND_MP3
videoCodecs	数字	指示支持的视频编解码器	SUPPORT_VID_SORENSON
pageUrl	字符串	SWF 文件被加载的页面的 Url	http:// somehost/sample.html
objectEncoding	数字	AMF 编码方法	kAMF3

音频编解码器属性的值：

源码常量	使用	值
SUPPORT_SND_NONE	原始音频数据，无压缩	0x0001
SUPPORT_SND_ADPCM	ADPCM 压缩	0x0002
SUPPORT_SND_MP3	mp3 压缩	0x0004
SUPPORT_SND_INTEL	没有使用	0x0008
SUPPORT_SND_UNUSED	没有使用	0x0010
SUPPORT_SND_NELLY8	NellyMoser 8KHZ 压缩	0x0020
SUPPORT_SND_NELLY	NellyMose 压缩（5，11，22 和 44KHZ）	0x0040
SUPPORT_SND_G711A	G711A 音频压缩（只用于 flash media server）	0x0080
SUPPORT_SND_G711U	G711U 音频压缩（只用于 flash media server）	0x0100
SUPPORT_SND_NELLY16	NellyMoser 16KHZ 压缩	0x0200
SUPPORT_SND_AAC	AAC 编解码	0x0400
SUPPORT_SND_SPEEX	Speex 音频	0x0800
SUPPORT_SND_ALL	所有 RTMP 支持的音频格式	0x0fff

视频编解码器属性值：

源码常量	使用	值
SUPPORT_VID_UNUSED	废弃的值	0x0001
SUPPORT_VID_JPEG	废弃的值	0x0002
SUPPORT_VID_SORENSON	Sorenson Flash video	0x0004
SUPPORT_VID_HOMEBREW	V1 screen sharing	0x0008
SUPPORT_VID_VP6 (On2)	On2 video (Flash 8+)	0x0010
SUPPORT_VID_VP6ALPHA (On2 with alpha channel)	On2 video with alpha channel	0x0020
SUPPORT_VID_HOMEBREWV (screensharing v2)	Screen sharing version 2 (Flash 8+)	0x0040
SUPPORT_VID_H264	H264 视频	0x0080
SUPPORT_VID_ALL	RTMP 支持的所有视频编解码器	0x00ff

视频函数属性值：

源码常量	使用	值
SUPPORT_VID_CLIENT_SEEK	Indicates that the client can seek frame-accurate on the client	1

对象编码属性值：

源码常量	使用	值
kAMF0	Flash 6 和以后版本支持的 AMF0 对象编码	0
kAMF3	来自 Flash 9(AS3)的 AMF3 编码	3

从服务端到客户端的命令结构：

字段名	类型	描述
命令名	字符串	含有_result 或_error，表示响应时结果还是错误。
传输 ID	数字	对于调用连接响应，传输 ID 是 1
属性	对象	描述连接属性（例如 fmsver）的名-值对
信息	对象	描述来自服务端的响应的名值对。’code’，‘level’，‘description’是这些名字中的一小部分。.

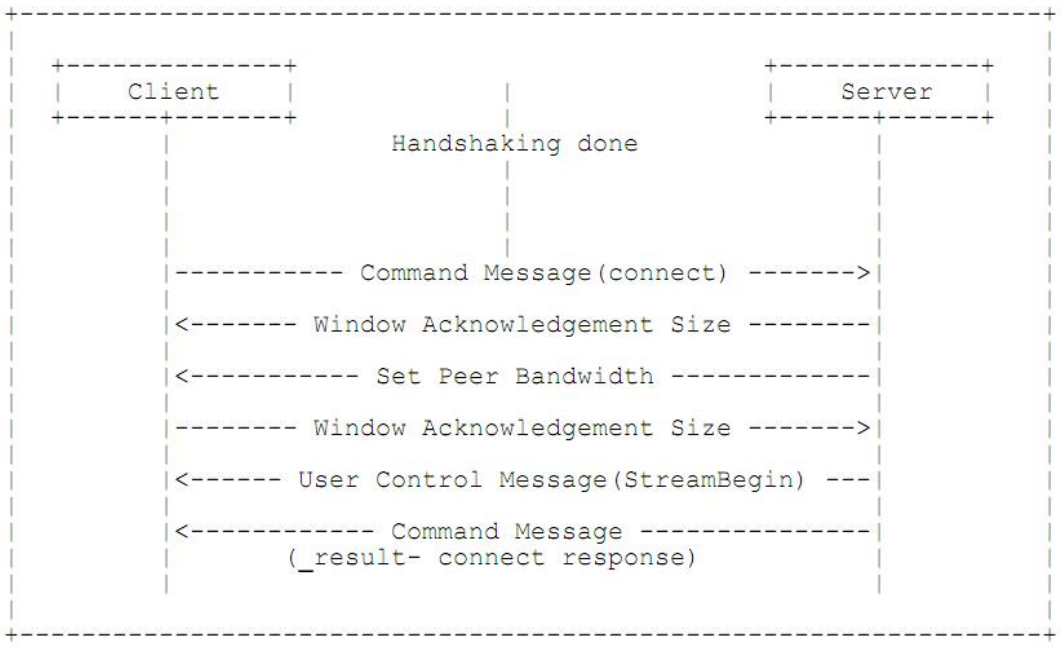

Figure 4 Message flow in the connect command

命令执行过程中的消息流是：

1.客户端发送连接命令到服务端，请求与一个服务应用实例建立连接。

2.接收到连接命令后，服务端发送”窗口确认（致谢）消息”到客户端。服务端同时连接到连接命令中提到的应用。

3.服务端发送”设置带宽”协议消息到客户端。

4.在处理完”设置带宽”消息后，客户端发送”窗口确认（致谢）大小”消息到服务端。

5.服务端发送用户控制消息中的流开始消息到客户端。

6.服务端发送结果命令消息通知客户端连接状态。该命令指定传输 ID（对于连接命令总是 1）。同时还指定一些属性，例如，Flash media server 版本（字符串），能力（数字），以及其他的连接信息，例如，层（字符串），代码（字符串），描述（字符串），对象编码（数字）等。

4.1.2 调用

NetConnection 对象的调用方法在接收端运行远程过程调用。远程方法的名作为调用命令的参数。

从发送端到接收端的命令结构如下：

字段名	类型	描述
过程名	字符串	所调用的远程过程的名字
传输 ID	数字	如果希望有响应则给出传输 ID，否则，传递一个 0 值
命令对象	对象	如果存在任何命令信息，则设置此对象。否则，设置为空类型。
可选变量	对象	所提供的任何可选变量

响应的命令结构：

字段名	类型	描述
命令名	字符串	命令名
传输 ID	数字	响应所属的命令的 ID
命令结构	对象	如果存在任何命令信息，则设置此对象。否则，设置为空类型。
响应	对象	来自所调用的方法的响应

4.1.3. 创建流

客户端发送本命令到服务端创建一个消息通讯的逻辑通道。音频，视频和元数据的发布是由创建流命令建立的流通道承载的。

NetConnection 本身是默认的流通道，具有流 ID 0 。协议和一少部分命令消息，包括创建流，就使用默认的通讯通道。

客户端到命令端的命令结构：

字段	类型	描述
命令名	字符串	命令名，设置为”createstream”
传输 ID	数字	命令的传输 ID
命令对象	对象	如果存在任何命令信息，则设置此对象。否则，设置为空类型

从服务端到客户端的命令结构：

字段	类型	描述
命令名	字符串	含有_result 或_error;表示响应是结果或是错误
传输 ID	数字	响应所属的命令的 ID
命令对象	对象	如果存在任何命令信息，则设置此对象。否则，设置为空类型
流 ID	数字	返回值或者是一个流 ID，或者是一个错误信息对象

4.3 NetStream 命令

NetStream 定义，基于连接客户端和服务端的 NetConnection 对象的，可以使音频流，视频流和数据消息传输的通道。对于多数据流，一个 NetConnection 对象可以支持多个 NetStreams。

下列的命令可以在 netstream 上发送。

1. 播放
2. 播放 2
3. 删除流
4. 关闭流

5. 接收音频
6. 接收视频
7. 发布
8. 搜索
9. 暂停

4.2.1 播放

客户端发送本命令到服务端播放一个流。使用本命令多次也可以创建一个播放列表。如果想创建一个可以在不同的直播流或录制流间切换的动态播放列表，可以多次使用播放命令，并且将重设设为假。相反，如果想立即播放一个流。清楚队列中正在等待的其它流，将重设设为真。

从客户端到服务端的命令结构入下：

字段名	类型	描述
命令名	字符串	命令名，设为”play”
传输 ID	数字	设置为 0
命令对象	NULL	命令信息不存在，设为 NULL 类型
流名	字符串	要播放的流名。对于播放一个 FLV 文件，则流名不带文件后缀（例如，"sample”）。对于回放 MP3 或 ID3 标签，必须在流名前加 MP3（例如：”MP3:Sample”）。对于播放 H264/AAC 文件，必须在流名前加 MP4 前缀，并且指定扩展名，例如播放 sample.m4v，则指定”mp4:sample.m4v”。
开始	数字	一个指定开始时间的可选参数。默认值是-2，意味着用户首先尝试播放流名中指定的直播流。如果流名字段中指定的直播流不存在，则播放同名的录制流。如果本字段设置为-1，则只播放流名字段中指定的直播流。如果，本字段为 0，或正值，则在本字段指定的时间，播放流名字段中指定的录制流。如果指定的录制流不存在，则播放播放列表中的下一项。
时长	数字	指定播放时长的可选字段。默认值是-1。-1 意味着播放一个直播流，直到没有数据可以活到，或者播放一个录

		制流知道结束。如果本字段位 0，则播放一个录制流中从 start 字段中指定的时间开始的单个帧。假设，start 字段中指定的是大于或等于 0 的值。如果本字段为正值，则以本字段中的值为时间周期播放直播流。之后，则以时长字段中指定的时间播放录制流？。（如果一个流在时长字段中指定的时间之前结束，则回放结束。）。如果传递一个非-1 的负值，则把值当作-1。
Reset	布尔值	指定是否刷新先前的播放列表的 BOOL 值或数值。

从服务端到客户端的命令结构：

字段	类型	描述
命令名	字符串	命令名。如果播放成功，则命令名设为响应状态。
描述	字符串	如果播放命令成功，则客户端从服务端接收 NetStream.Play.Start 状态响应消息。如果指定的流没有找到，则接收到 NetStream.Play.StreamNotFound 响应状态消息。

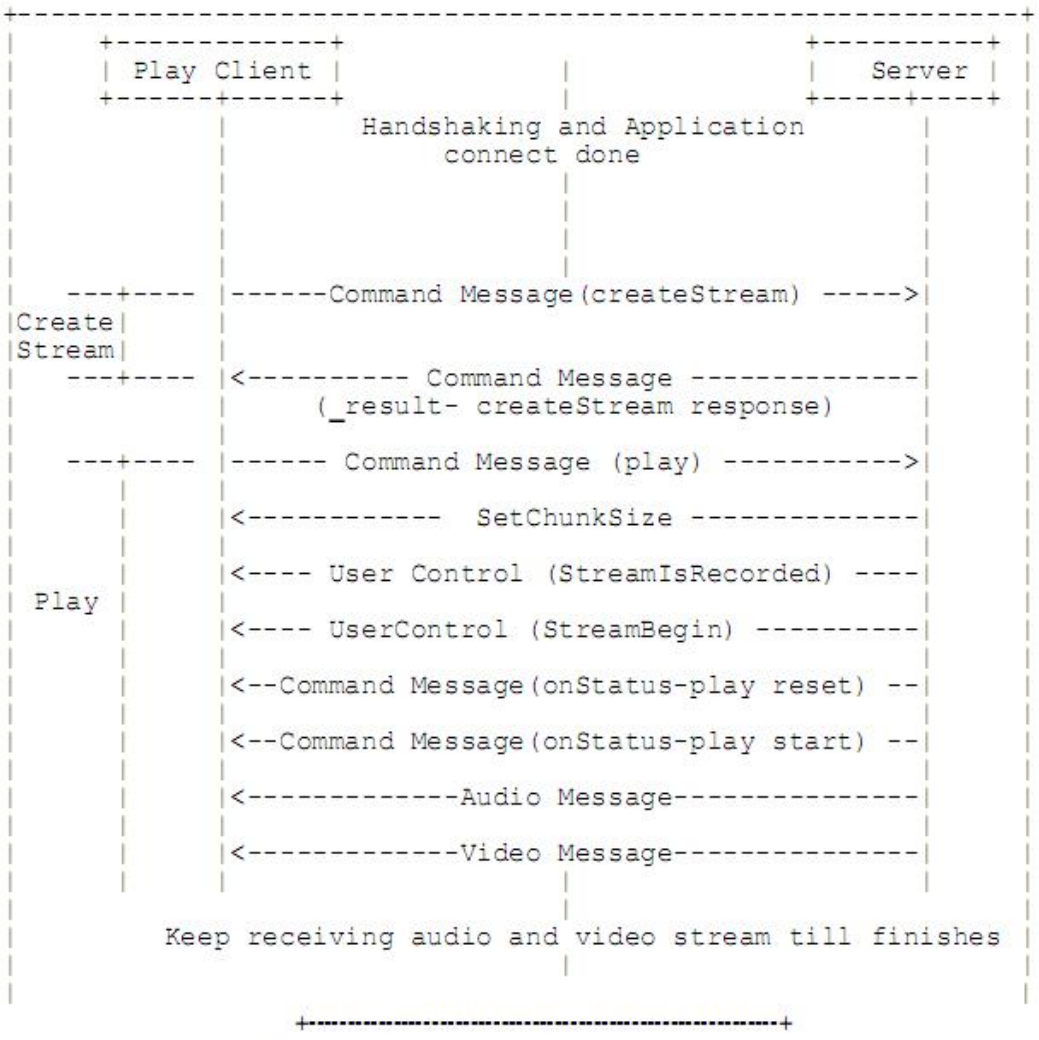

Figure 5 Message flow in the play command

执行命令的消息流：

1. 客户端从服务端接收到流创建成功消息，发送播放命令到服务端。
2. 接收到播放命令后，服务端发送协议消息设置块大小。
3. 服务端发送另一个协议消息（用户控制消息），并且在消息中指定事件”streamisrecorded”和流 ID。消息承载的头 2 个字，为事件类型，后 4 个字节为流 ID。
4. 服务端发送事件”streambegin”的协议消息（用户控制），告知客户端流 ID。
5. 服务端发送响应状态命令消息 NetStream.Play.Start&NetStream.Play.reset，如果客户端发送的播放命令成功的话。只有当客户端发送的播放命令设置了 reset 命令的条件下，服务端才发送 NetStream.Play.reset 消息。如果要发送的流没有找的话，服务端发送 NetStream.Play.StreamNotFound 消息。

在此之后服务端发送客户端要播放的音频和视频数据。

4.2.2 播放

和播放命令不同，播放 2 命令可以切换到不同的码率，而不用改变已经播放的内容的时间线。服务端对播放 2 命令可以请求的多个码率维护多个文件。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设置为“play2”
传输 ID	数字	传输 ID 设置为 2
开始时间	对象	存储数字值的 AMF 编码对象。本字段的值，指定以秒位单位的流的开始位置。如果本字段传递 0，则流从当前时间开始播放。
旧流名	对象	存储字符串值的 AMF 编码对象。该值是一个含有流值参数和旧流名的字符串
流名	对象	存储字符串值的 AMF 编码对象。存储要播放的流的名字。
时长	对象	存储数字值的 AMF 编码对象。该值指定要播放的流的总时长。
转换	对象	存储 AMF 编码对象的字符串值。该值定义播放列表转换模式(switch 或 swap)。Switch：通过切换流的单码率版本实现多码率流。SWAP：用 streamname 字段的值替换 oldstreamname 中的值，并且存储剩余的播放列表队列。然而在这种情况下，服务端并不对流的内容作任何假设，而只是把它们当作不同的流内容。因此，只在流的两段切换，或者从不切换。

命令的消息流如下所示：

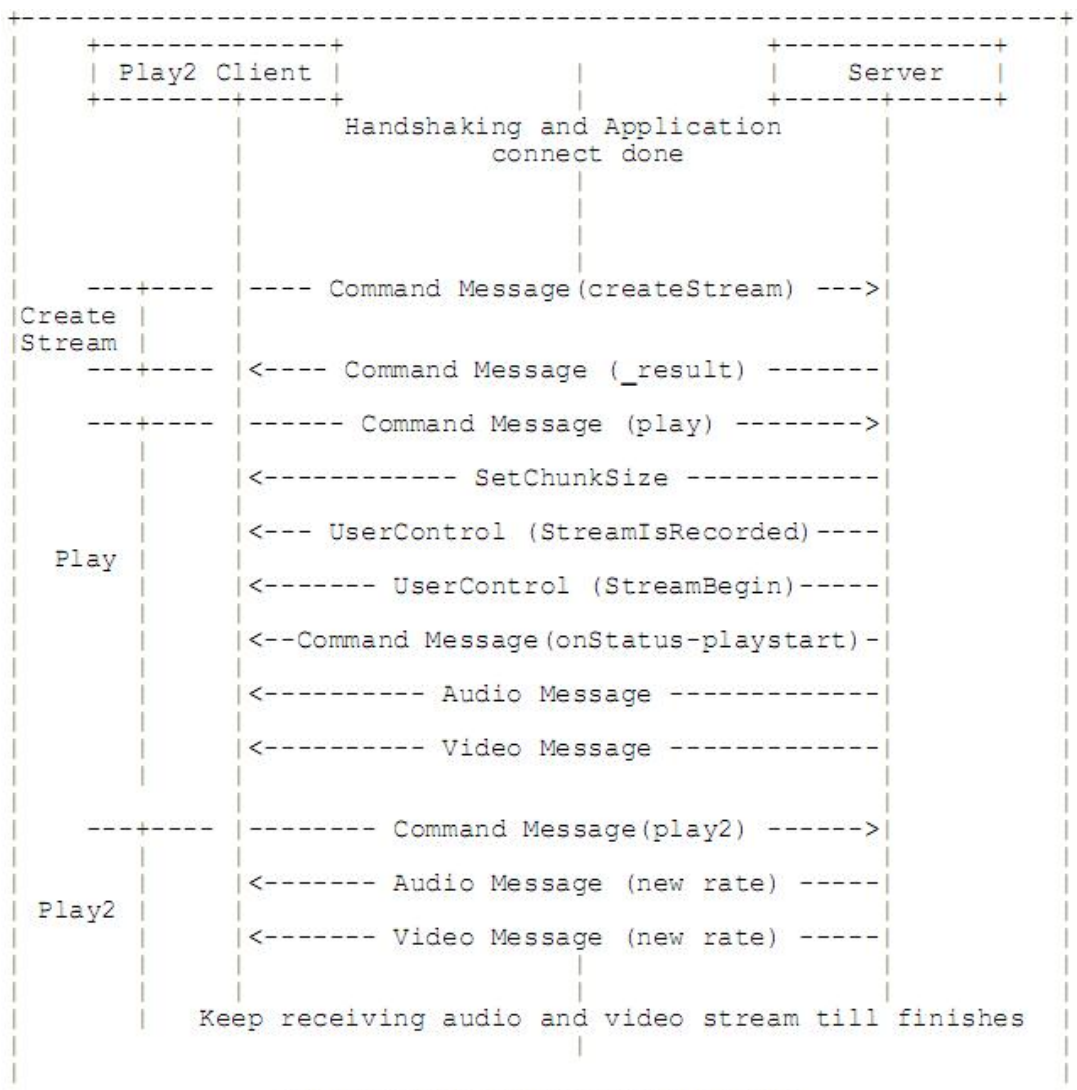

Figure 1 Message flow in the play2 command

4.2.3 删除流

当 NetStream 对象销毁的时候发送删除流命令。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设置为”deleteStream”
传输 ID	数字	传输 ID 设置为 0
命令对象	空	命令信息对象不存在。设为空类型。
流 ID	数字	要销毁的流的 ID

服务端不返回任何响应。

4.2.4 接收音频

NetStream 对象发送接收音频消息通知服务端发送还是不发送音频到客户端。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设为”接收音频”
传输 ID	数字	传输 ID 设置为 0
命令对象	空	命令信息对象不存在。设为空类型。
布尔标志	布尔值	真假表示是否接收音频流

服务端不返回响应.

4.2.5 接收视频

NetStream 对象发送 receiveVideo 消息通知服务端是否发送视频到客户端。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设为”receiveVideo”
传输 ID	数字	传输 ID 设置为 0
命令对象	空	命令信息对象不存在。设为空类型。
布尔标志	布尔值	真假表示是否接收视频流

服务端不返回任何响应。

4.2.6 发布

客户端发送一个发布命令，发布一个命名流到服务端。使用这个名字，任何客户端可以播放该流并且接收音频，视频，和数据消息。

从客户端到服务端的命令如下：

字段名	类型	描述
命令名	字符串	命令名，设置为”seek”
传输 ID	数字	传输 ID 为 0
命令对象	空	命令信息对象不存在。设为空类型。
发布名	字符串	流发布的名字
发布类型	字符串	发布类型。 设置为”live”，”record”或”append”。 “record”:流被发布，并且数据被录制到一个新的文件。文件被存储到服务端的服务应用的目录的一个子目录下。如果文件已经存在则重写文件。 “append”:流被发布并且附加到一个文件之后。如果没有发现文件则创建一个文件。 “live”:发布直播数据而不录制到文件。

服务端用响应状态命令响应表示一个发布的开始

4.2.7 搜寻

客户端发送搜寻命令在一个媒体文件中或播放列表中搜寻偏移。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设置为”seek”
传输 ID	数字	传输 ID 设置为 0
命令对象	空	命令信息对象不存在。设为空类型
毫秒	数字	在播放列表中搜寻的毫秒数

如果搜寻成功服务端发送一个 NetStream.Seek.Notify 状态消息。如果失败，则返回含有_error 的消息。

4.2.8 暂停

客户端发送暂停命令告诉服务端暂停或开始一个命令。

从客户端到服务端的命令结构如下：

字段名	类型	描述
命令名	字符串	命令名，设置为”pause”
传输 ID	数字	本命令没有传输 ID。设为 0
命令对象	空	
暂停/取消暂停 标志	布尔	真或假，指示暂停还是恢复播放
毫秒	数字	流暂停或恢复播放的毫秒数。当回放恢复的时候，服务端将只发送含有大于该值得时间戳。

当流暂停的时候，服务端发送一个 NetStream.Pause.Notify 状态消息。当恢复播放的时候发送 NetStream.Unpause.Notify 消息。如果失败，则返回_error 消息。

5. 消息交换示例

下面有一些示例解释 RTMP 的消息交换

5.1 发布录制视频

下面的例子演示一个发布者如何发布一个流并且传输视频到服务端。其他的客户端可以使用这些发布的流并且播放视频。

5.2 广播一个共享对象消息

下面的例子演示共享对象创建和变化期间消息的交换。同时也演示了共享对象消息的广播。

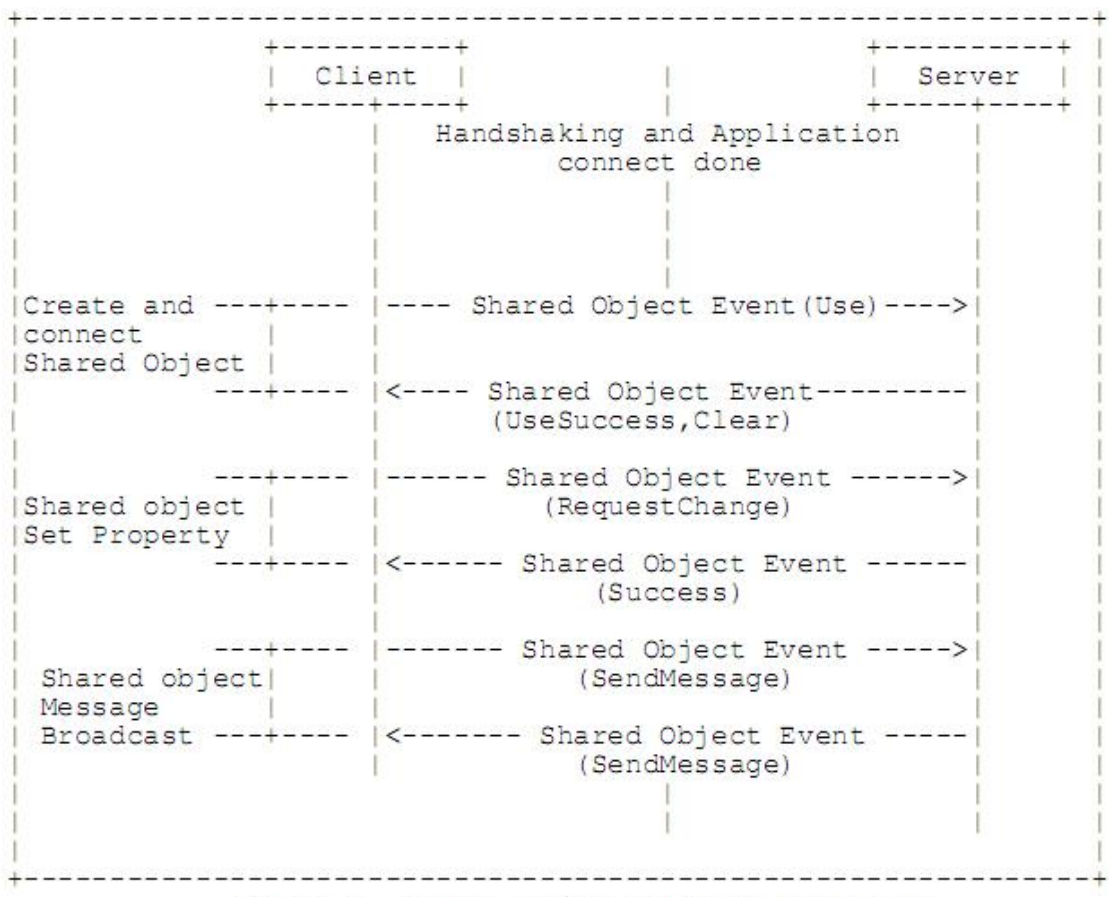

Figure 1 Shared object message broadcast

5.3 从录制流发布元数据

下面的例子演示发布元数据的消息交换。

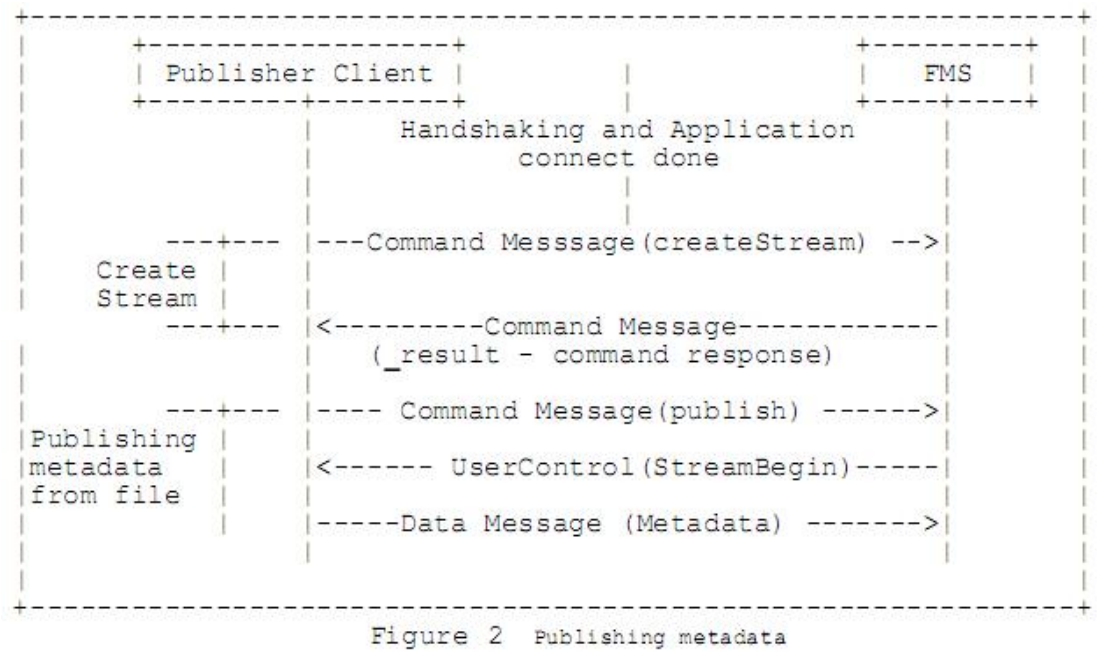

Figure 2 Publishing metadata

6. 参考

6.1. 标准参考

[1] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[2] Crocker， D. and Overell， P.(Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

[RFC2119] Bradner， S.， "Key words for use in RFCs to Indicate Requirement Levels"， BCP 14， RFC 2119， March 1997.

[RFC2234] Crocker， D. and Overell， P.(Editors)， "Augmented BNF for Syntax Specifications: ABNF"， RFC 2234， Internet Mail Consortium and Demon Internet Ltd.， November 1997.

6.2. 参考资料

[3] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in TCP and Its Effect on Busy Servers"， Proc. Infocom 1999 pp. 1573-1583.

[Fab1999] Faber， T.， Touch， J. and W. Yue， "The TIME-WAIT state in TCP and Its Effect on Busy Servers"， Proc. Infocom 1999 pp. 1573-1583.

7. 确认（致谢）

Address:
Adobe Systems Incorporated
345 Park Avenue
San Jose， CA 95110-2704