Protocol for Cross Application File Streaming v1.0 (PFCAFS)

Table of Contents

1 Introduction

1.1 Purpose

The PFCAFS protocol is an application layer protocol to allow for the streaming of files from a server host to a rendering box and finally to a controlling system.

1.2 Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 .

An implementation is not compliant if it fails to satisfy one or more of the MUST or REQUIRED level requirements for the protocols it implements. An implementation that satisfies all the MUST or REQUIRED level and all the SHOULD level requirements for its protocols is said to be "unconditionally compliant"; one that satisfies all the MUST level requirements but not all the SHOULD level requirements for its protocols is said to be "conditionally compliant."

1.3 Terminology

Renderer

This application processess and minipulates the file chunks before being set off to the rest of the applications. In this verision of the protocol the renderer MUST be built without an internal buffer and MUST ask for file chunks when needed.

Controller

This application act as the user interface to the rest of the protocol.

Server

Ths application acts as the storage for the protocol. This is usually done as disk storage but other storage mediums MAY be used.

Control operations

These are the set of user actions that they may take on the controller. These are passed to the renderer so it may take actions on those control operations.

The control operations defined for this version of PFCAFS is:

- Start
- Play
- Resume
- Restart

File chunk

This is a small section of the file sent to the renderer from the controller. In this verion of PFCAFS the file chunks MUST be 100 bytes. The last chunk in a file, if it doesn't meet the chunk size MUST be padded with NULL bytes

1.4 Overall Operation

1. Controller requests file listing from server
2. User selects wanted file
3. Controller requests renderer to render file
4. Renderer requests file chunk from server
5. Once file chunk is processed by the renderer it is passed to the controller
6. steps 4 - 5 are repeated for each chunk in the file

PFCAFS communication usually takes place over TCP/IP connections. The default port is TCP 9842, but other ports can be used. This does not preclude PFCAFS from being implemented on top of any other protocol on

the Internet, or on other networks. PFCAFS only presumes a reliable transport; any protocol that provides such guarantees can be used; the mapping of the PFCAFS/1.0 request and response structures onto the transport data units of the protocol in question is outside the scope of this specification.

2 Headers

2.1 Base headers

All messages MUST be prepended by

`PFCAFS 1.0 \r\n`
`TARGET RESOURCE <target> \r\n`
`<operation headers>`
`<other vendor headers>` .

The header block SHALL end with a `\r\n\r\n` sequence

2.2 Request headers

All request messages MUST contain the operation header `REQUEST <request type>`

2.3 Response headers

All request messages MUST contain the operation header `RESPONSE TO <request type>`

2.4 vendor headers

These are headers defined by the implimenter if PFCAFS these MUST be ignored by applications that don't expect these.

3 Controller to Server Communications

3.1 File list request

- The target SHALL be `SERVER`
- The request type SHALL be `LIST`
- The body is OPTIONAL in this version of PFCAFS
    - The body MAY be used by implementations to act as a filter
    - End servers that do not support filtering by the body MUST ignore this field

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE SERVER \r\n
REQUEST LIST \r\n\r\n
```

4 Controller to Renderer Communications

4.1 Start rendering request

- The target SHALL be `RENDERER`
- The request type SHALL be `START`
- The body SHALL be the file name to be rendered

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE RENDERER \r\n
REQUEST START \r\n\r\n
TEST_FILE\r\n
```

4.2 Pause rendering request

- The target SHALL be `RENDERER`
- The request type SHALL be `PAUSE`
- The body is OPTIONAL in this version of PFCAFS
    - Venders my define fields in the body to further specify the pause details
    - Renderers that do not support these fields MUST ignore the fields

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE RENDERER \r\n
REQUEST PAUSE \r\n\r\n
```

4.3 Resume rendering request

- The target SHALL be `RENDERER`
- The request type SHALL be `RESUME`
- The body is OPTIONAL in this version of PFCAFS

    - Venders my define fields in the body to further specify the resume details
    - Renderers that do not support these fields MUST ignore the fields

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE RENDERER \r\n
REQUEST RESUME \r\n\r\n
```

4.4 Restart rendering request

- The target SHALL be `RENDERER`
- The request type SHALL be `RESTART`
- The body is OPTIONAL in this version of PFCAFS
    - Venders my define fields in the body to further specify the resume details
    - Renderers that do not support these fields MUST ignore the fields

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE RENDERER \r\n
REQUEST RESTART \r\n\r\n
```

5 Renderer to Server Communications

5.1 File chunk request

- The target SHALL be `SERVER`
- The request type SHALL be `CHUNK`
- The body SHALL be `<filename>,<chunkid>`
- The filename MUST be restricted to match the regex
    - `^\w*(\.\w{1,3})?$`

Full request:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE SERVER \r\n
REQUEST CHUNK \r\n\r\n
file_name.txt,156\r\n
```

6 Server to Renderer Communications

6.1 File chunk response

- The target SHALL be `RENDERER`
- The request type SHALL be `CHUNK`
- The body SHALL be the chunk of the file

```
PFCAFS 1.0 \r\n
TARGET RESOURCE RENDERER \r\n
RESPONSE TO CHUNK \r\n\r\n
this is an example of a chunk \r\n
```

7 Renderer to controller communications

7.1 rended chunk response

- The target SHALL be `CONTROLLER`
- The request type SHALL be `START`
- The body SHALL be the chunk of the file
- This response will be repeated multiple times for one request

```
PFCAFS 1.0 \r\n
TARGET RESOURCE CONTROLLER \r\n
RESPONSE TO START \r\n\r\n
this is an example of a RENDERED chunk \r\n
```

8 Server to controller communications

8.1 File list response

- The target SHALL be `CONTROLLER`
- The request type SHALL be `LIST`
- The response body SALL be the list available files and size
  - format `<FILE NAME>,<File Size (bytes)>`
- The filename MUST be restricted to match the regex
  - `^\w*(\.\w{1,3})?$`

Example response:

```
PFCAFS 1.0 \r\n
TARGET RESOURCE CONTROLLER \r\n
RESPONSE TO LIST \r\n\r\n
TEST_FILE,100 \r\n
TEST_FILE2.TXT,10000 \r\n
```

9 Media Information

9.1 Text

There are no special requirements to transact text across PFCAFS.

9.2 Multimedia

9.2.1 MP4 Video

If you are not planning on streaming the data live and waiting for the full file before playing no special requirements are needed.

If you are streaming the data live specail requirements MUST be taken. The render MUST put the file header and trailer for each data chunk.

9.2.2 MP3 Audio

The same requirements in 9.2.1 apply.