**[MS-RDPECLIP]:**

**Remote Desktop Protocol: Clipboard Virtual Channel Extension**

**Intellectual Property Rights Notice for Open Specifications Documentation**

* **Technical Documentation.** Microsoft publishes Open Specifications documentation for protocols, file formats, languages, standards as well as overviews of the interaction among each of these technologies.
* **Copyrights**. This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you may make copies of it in order to develop implementations of the technologies described in the Open Specifications and may distribute portions of it in your implementations using these technologies or your documentation as necessary to properly document the implementation. You may also distribute in your implementation, with or without modification, any schema, IDL's, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications.
* **No Trade Secrets**. Microsoft does not claim any trade secret rights in this documentation.
* **Patents**. Microsoft has patents that may cover your implementations of the technologies described in the Open Specifications. Neither this notice nor Microsoft's delivery of the documentation grants any licenses under those or any other Microsoft patents. However, a given Open Specification may be covered by Microsoft [Open Specification Promise](http://go.microsoft.com/fwlink/?LinkId=214445) or the [Community Promise](http://go.microsoft.com/fwlink/?LinkId=214448). If you would prefer a written license, or if the technologies described in the Open Specifications are not covered by the Open Specifications Promise or Community Promise, as applicable, patent licenses are available by contacting [iplg@microsoft.com](mailto:iplg@microsoft.com).
* **Trademarks**. The names of companies and products contained in this documentation may be covered by trademarks or similar intellectual property rights. This notice does not grant any licenses under those rights. For a list of Microsoft trademarks, visit [www.microsoft.com/trademarks](http://www.microsoft.com/trademarks).
* **Fictitious Names**. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in this documentation are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

**Reservation of Rights**. All other rights are reserved, and this notice does not grant any rights other than specifically described above, whether by implication, estoppel, or otherwise.

**Tools**. The Open Specifications do not require the use of Microsoft programming tools or programming environments in order for you to develop an implementation. If you have access to Microsoft programming tools and environments you are free to take advantage of them. Certain Open Specifications are intended for use in conjunction with publicly available standard specifications and network programming art, and assumes that the reader either is familiar with the aforementioned material or has immediate access to it.

**Revision Summary**

| Date | Revision History | Revision Class | Comments |
| --- | --- | --- | --- |
| 4/3/2007 | 0.01 |  | Version 0.01 release |
| 7/3/2007 | 1.0 | Major | MLonghorn+90 |
| 7/20/2007 | 1.0.1 | Editorial | Changed language and formatting in the technical content. |
| 8/10/2007 | 1.0.2 | Editorial | Changed language and formatting in the technical content. |
| 9/28/2007 | 1.0.3 | Editorial | Changed language and formatting in the technical content. |
| 10/23/2007 | 1.0.4 | Editorial | Changed language and formatting in the technical content. |
| 11/30/2007 | 2.0 | Major | Updated and revised the technical content. |
| 1/25/2008 | 3.0 | Major | Updated and revised the technical content. |
| 3/14/2008 | 4.0 | Major | Updated and revised the technical content. |
| 5/16/2008 | 4.0.1 | Editorial | Changed language and formatting in the technical content. |
| 6/20/2008 | 4.0.2 | Editorial | Changed language and formatting in the technical content. |
| 7/25/2008 | 4.0.3 | Editorial | Changed language and formatting in the technical content. |
| 8/29/2008 | 4.0.4 | Editorial | Changed language and formatting in the technical content. |
| 10/24/2008 | 4.0.5 | Editorial | Changed language and formatting in the technical content. |
| 12/5/2008 | 4.0.6 | Editorial | Changed language and formatting in the technical content. |
| 1/16/2009 | 4.0.7 | Editorial | Changed language and formatting in the technical content. |
| 2/27/2009 | 4.0.8 | Editorial | Changed language and formatting in the technical content. |
| 4/10/2009 | 4.0.9 | Editorial | Changed language and formatting in the technical content. |
| 5/22/2009 | 4.0.10 | Editorial | Changed language and formatting in the technical content. |
| 7/2/2009 | 5.0 | Major | Updated and revised the technical content. |
| 8/14/2009 | 5.0.1 | Editorial | Changed language and formatting in the technical content. |
| 9/25/2009 | 5.1 | Minor | Clarified the meaning of the technical content. |
| 11/6/2009 | 6.0 | Major | Updated and revised the technical content. |
| 12/18/2009 | 7.0 | Major | Updated and revised the technical content. |
| 1/29/2010 | 8.0 | Major | Updated and revised the technical content. |
| 3/12/2010 | 8.0.1 | Editorial | Changed language and formatting in the technical content. |
| 4/23/2010 | 8.0.2 | Editorial | Changed language and formatting in the technical content. |
| 6/4/2010 | 8.1 | Minor | Clarified the meaning of the technical content. |
| 7/16/2010 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 8/27/2010 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 10/8/2010 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 11/19/2010 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 1/7/2011 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 2/11/2011 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 3/25/2011 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 5/6/2011 | 8.1 | None | No changes to the meaning, language, or formatting of the technical content. |
| 6/17/2011 | 8.2 | Minor | Clarified the meaning of the technical content. |
| 9/23/2011 | 8.2 | None | No changes to the meaning, language, or formatting of the technical content. |
| 12/16/2011 | 9.0 | Major | Updated and revised the technical content. |
| 3/30/2012 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 7/12/2012 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 10/25/2012 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 1/31/2013 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 8/8/2013 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 11/14/2013 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 2/13/2014 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 5/15/2014 | 9.0 | None | No changes to the meaning, language, or formatting of the technical content. |
| 6/30/2015 | 10.0 | Major | Significantly changed the technical content. |

Table of Contents

[1 Introduction 7](#_Toc423367443)

[1.1 Glossary 7](#_Toc423367444)

[1.2 References 7](#_Toc423367445)

[1.2.1 Normative References 8](#_Toc423367446)

[1.2.2 Informative References 8](#_Toc423367447)

[1.3 Overview 8](#_Toc423367448)

[1.3.1 Clipboard Basics 8](#_Toc423367449)

[1.3.1.1 Data Types 9](#_Toc423367450)

[1.3.1.1.1 Generic 9](#_Toc423367451)

[1.3.1.1.2 Palette 9](#_Toc423367452)

[1.3.1.1.3 Metafile 9](#_Toc423367453)

[1.3.1.1.4 File List 9](#_Toc423367454)

[1.3.1.1.5 File Stream 9](#_Toc423367455)

[1.3.1.2 Clipboard Format 10](#_Toc423367456)

[1.3.1.3 Monitoring Clipboard Updates 10](#_Toc423367457)

[1.3.1.4 Delayed Rendering of Clipboard Data 11](#_Toc423367458)

[1.3.2 Clipboard Redirection Virtual Channel Protocol 11](#_Toc423367459)

[1.3.2.1 Initialization Sequence 11](#_Toc423367460)

[1.3.2.2 Data Transfer Sequences 12](#_Toc423367461)

[1.3.2.2.1 Copy Sequence 13](#_Toc423367462)

[1.3.2.2.2 Locking and Unlocking Clipboard Data 13](#_Toc423367463)

[1.3.2.2.3 Paste Sequence 13](#_Toc423367464)

[1.3.2.3 Interacting with Local Clipboard and Applications 13](#_Toc423367465)

[1.4 Relationship to Other Protocols 15](#_Toc423367466)

[1.5 Prerequisites/Preconditions 15](#_Toc423367467)

[1.6 Applicability Statement 15](#_Toc423367468)

[1.7 Versioning and Capability Negotiation 15](#_Toc423367469)

[1.8 Vendor-Extensible Fields 15](#_Toc423367470)

[1.9 Standards Assignments 16](#_Toc423367471)

[2 Messages 17](#_Toc423367472)

[2.1 Transport 17](#_Toc423367473)

[2.2 Message Syntax 17](#_Toc423367474)

[2.2.1 Clipboard PDU Header (CLIPRDR\_HEADER) 17](#_Toc423367475)

[2.2.2 Initialization Sequence 17](#_Toc423367476)

[2.2.2.1 Clipboard Capabilities PDU (CLIPRDR\_CAPS) 18](#_Toc423367477)

[2.2.2.1.1 Capability Set (CLIPRDR\_CAPS\_SET) 18](#_Toc423367478)

[2.2.2.1.1.1 General Capability Set (CLIPRDR\_GENERAL\_CAPABILITY) 19](#_Toc423367479)

[2.2.2.2 Server Monitor Ready PDU (CLIPRDR\_MONITOR\_READY) 20](#_Toc423367480)

[2.2.2.3 Client Temporary Directory PDU (CLIPRDR\_TEMP\_DIRECTORY) 20](#_Toc423367481)

[2.2.3 Copy Sequence 20](#_Toc423367482)

[2.2.3.1 Format List PDU (CLIPRDR\_FORMAT\_LIST) 21](#_Toc423367483)

[2.2.3.1.1 Short Format Names (CLIPRDR\_SHORT\_FORMAT\_NAMES) 21](#_Toc423367484)

[2.2.3.1.1.1 Short Format Name (CLIPRDR\_SHORT\_FORMAT\_NAME) 21](#_Toc423367485)

[2.2.3.1.2 Long Format Names (CLIPRDR\_LONG\_FORMAT\_NAMES) 22](#_Toc423367486)

[2.2.3.1.2.1 Long Format Name (CLIPRDR\_LONG\_FORMAT\_NAME) 22](#_Toc423367487)

[2.2.3.2 Format List Response PDU (FORMAT\_LIST\_RESPONSE) 22](#_Toc423367488)

[2.2.4 Locking and Unlocking Clipboard Data 23](#_Toc423367489)

[2.2.4.1 Lock Clipboard Data PDU (CLIPRDR\_LOCK\_CLIPDATA) 23](#_Toc423367490)

[2.2.4.2 Unlock Clipboard Data PDU (CLIPRDR\_UNLOCK\_CLIPDATA) 23](#_Toc423367491)

[2.2.5 Paste Sequence 24](#_Toc423367492)

[2.2.5.1 Format Data Request PDU (CLIPRDR\_FORMAT\_DATA\_REQUEST) 24](#_Toc423367493)

[2.2.5.2 Format Data Response PDU (CLIPRDR\_FORMAT\_DATA\_RESPONSE) 24](#_Toc423367494)

[2.2.5.2.1 Packed Metafile Payload (CLIPRDR\_MFPICT) 25](#_Toc423367495)

[2.2.5.2.2 Packed Palette Payload (CLIPRDR\_PALETTE) 26](#_Toc423367496)

[2.2.5.2.2.1 Palette Entry (PALETTEENTRY) 26](#_Toc423367497)

[2.2.5.2.3 Packed File List (CLIPRDR\_FILELIST) 26](#_Toc423367498)

[2.2.5.2.3.1 File Descriptor (CLIPRDR\_FILEDESCRIPTOR) 27](#_Toc423367499)

[2.2.5.3 File Contents Request PDU (CLIPRDR\_FILECONTENTS\_REQUEST) 29](#_Toc423367500)

[2.2.5.4 File Contents Response PDU (CLIPRDR\_FILECONTENTS\_RESPONSE) 30](#_Toc423367501)

[3 Protocol Details 31](#_Toc423367502)

[3.1 Common Details 31](#_Toc423367503)

[3.1.1 Abstract Data Model 31](#_Toc423367504)

[3.1.1.1 Clipboard Format ID Map 31](#_Toc423367505)

[3.1.1.2 File List 31](#_Toc423367506)

[3.1.1.3 Direct File Access 32](#_Toc423367507)

[3.1.2 Timers 32](#_Toc423367508)

[3.1.3 Initialization 32](#_Toc423367509)

[3.1.4 Higher-Layer Triggered Events 32](#_Toc423367510)

[3.1.4.1 Local Clipboard Update 32](#_Toc423367511)

[3.1.4.2 Local Paste Operation 32](#_Toc423367512)

[3.1.5 Processing Events and Sequencing Rules 33](#_Toc423367513)

[3.1.5.1 Processing a Clipboard PDU 33](#_Toc423367514)

[3.1.5.2 Copy Sequence 33](#_Toc423367515)

[3.1.5.2.1 Sending a Format List PDU 33](#_Toc423367516)

[3.1.5.2.2 Processing a Format List PDU 33](#_Toc423367517)

[3.1.5.2.3 Sending a Format List Response PDU 34](#_Toc423367518)

[3.1.5.2.4 Processing a Format List Response PDU 34](#_Toc423367519)

[3.1.5.3 Locking and Unlocking Clipboard Data 34](#_Toc423367520)

[3.1.5.3.1 Sending a Lock Clipboard Data PDU 34](#_Toc423367521)

[3.1.5.3.2 Processing a Lock Clipboard Data PDU 34](#_Toc423367522)

[3.1.5.3.3 Sending an Unlock Clipboard Data PDU 34](#_Toc423367523)

[3.1.5.3.4 Processing a Unlock Clipboard Data PDU 34](#_Toc423367524)

[3.1.5.4 Paste Sequence 35](#_Toc423367525)

[3.1.5.4.1 Sending a Format Data Request PDU 35](#_Toc423367526)

[3.1.5.4.2 Processing a Format Data Request PDU 35](#_Toc423367527)

[3.1.5.4.3 Sending a Format Data Response PDU 35](#_Toc423367528)

[3.1.5.4.4 Processing a Format Data Response PDU 36](#_Toc423367529)

[3.1.5.4.5 Sending a File Contents Request PDU 36](#_Toc423367530)

[3.1.5.4.6 Processing a File Contents Request PDU 36](#_Toc423367531)

[3.1.5.4.7 Sending a File Contents Response PDU 36](#_Toc423367532)

[3.1.5.4.8 Processing a File Contents Response PDU 37](#_Toc423367533)

[3.1.6 Timer Events 37](#_Toc423367534)

[3.1.7 Other Local Events 37](#_Toc423367535)

[3.2 Client Details 37](#_Toc423367536)

[3.2.1 Abstract Data Model 37](#_Toc423367537)

[3.2.1.1 Server Capabilities 37](#_Toc423367538)

[3.2.2 Timers 37](#_Toc423367539)

[3.2.3 Initialization 37](#_Toc423367540)

[3.2.4 Higher-Layer Triggered Events 37](#_Toc423367541)

[3.2.5 Processing Events and Sequencing Rules 38](#_Toc423367542)

[3.2.5.1 Initialization Sequence 38](#_Toc423367543)

[3.2.5.1.1 Processing a Server Clipboard Capabilities PDU 38](#_Toc423367544)

[3.2.5.1.2 Processing a Monitor Ready PDU 38](#_Toc423367545)

[3.2.5.1.3 Sending a Client Clipboard Capabilities PDU 38](#_Toc423367546)

[3.2.5.1.4 Sending a Temporary Directory PDU 38](#_Toc423367547)

[3.2.6 Timer Events 38](#_Toc423367548)

[3.2.7 Other Local Events 39](#_Toc423367549)

[3.3 Server Details 39](#_Toc423367550)

[3.3.1 Abstract Data Model 39](#_Toc423367551)

[3.3.1.1 Client Capabilities 39](#_Toc423367552)

[3.3.1.2 Client Temporary Directory 39](#_Toc423367553)

[3.3.2 Timers 39](#_Toc423367554)

[3.3.3 Initialization 39](#_Toc423367555)

[3.3.4 Higher-Layer Triggered Events 39](#_Toc423367556)

[3.3.5 Processing Events and Sequencing Rules 39](#_Toc423367557)

[3.3.5.1 Initialization Sequence 39](#_Toc423367558)

[3.3.5.1.1 Sending a Server Clipboard Capabilities PDU 39](#_Toc423367559)

[3.3.5.1.2 Sending a Monitor Ready PDU 40](#_Toc423367560)

[3.3.5.1.3 Processing a Client Clipboard Capabilities PDU 40](#_Toc423367561)

[3.3.5.1.4 Processing a Temporary Directory PDU 40](#_Toc423367562)

[3.3.6 Timer Events 40](#_Toc423367563)

[3.3.7 Other Local Events 40](#_Toc423367564)

[4 Protocol Examples 41](#_Toc423367565)

[4.1 Annotated Initialization Sequence 41](#_Toc423367566)

[4.1.1 Server Clipboard Capabilities PDU 41](#_Toc423367567)

[4.1.2 Server Monitor Ready PDU 41](#_Toc423367568)

[4.1.3 Client Clipboard Capabilities PDU 41](#_Toc423367569)

[4.1.4 Client Temporary Directory PDU 42](#_Toc423367570)

[4.1.5 Format List PDU 43](#_Toc423367571)

[4.1.6 Format List Response PDU 43](#_Toc423367572)

[4.2 Annotated Copy Sequence 43](#_Toc423367573)

[4.2.1 Format List PDU 44](#_Toc423367574)

[4.2.2 Format List Response PDU 44](#_Toc423367575)

[4.3 Locking and Unlocking Clipboard Data 45](#_Toc423367576)

[4.3.1 Lock Clipboard Data PDU 45](#_Toc423367577)

[4.3.2 Unlock Clipboard Data PDU 45](#_Toc423367578)

[4.4 Annotated Paste Sequence 45](#_Toc423367579)

[4.4.1 Format Data Request PDU 45](#_Toc423367580)

[4.4.2 Format Data Response PDU 45](#_Toc423367581)

[4.4.3 File Contents Request PDU 46](#_Toc423367582)

[4.4.3.1 Requesting the Size of a File 46](#_Toc423367583)

[4.4.3.2 Requesting the Contents of a File 46](#_Toc423367584)

[4.4.4 File Contents Response PDU 46](#_Toc423367585)

[4.4.4.1 Sending the Size of a File 46](#_Toc423367586)

[4.4.4.2 Sending the Contents of a File 47](#_Toc423367587)

[4.4.5 Metafile Data Contained in a Format Data Response PDU 47](#_Toc423367588)

[4.4.6 Palette Data Contained in a Format Data Response PDU 52](#_Toc423367589)

[4.5 Retrieving a File List 72](#_Toc423367590)

[4.5.1 Format List PDU 72](#_Toc423367591)

[4.5.2 Format List Response PDU 72](#_Toc423367592)

[4.5.3 Format Data Request PDU 72](#_Toc423367593)

[4.5.4 Format Data Response PDU 73](#_Toc423367594)

[5 Security 77](#_Toc423367595)

[5.1 Security Considerations for Implementers 77](#_Toc423367596)

[5.2 Index of Security Parameters 77](#_Toc423367597)

[6 Appendix A: Product Behavior 78](#_Toc423367598)

[7 Change Tracking 79](#_Toc423367599)

[8 Index 81](#_Toc423367600)

# Introduction

The goal of the Remote Desktop Protocol: Clipboard Virtual Channel Extension is to allow users to seamlessly transfer data, via the system clipboard, between applications that are running on different computers. To accomplish this objective, the Remote Desktop Protocol: Clipboard Virtual Channel Extension specifies how to keep two distinct system clipboards in sync so that at any given time, the data available to an application on one computer (via its local clipboard) is identical to the data available to another application on a remote computer (via its local clipboard).

Sections 1.8, 2, and 3 of this specification are normative and can contain the terms MAY, SHOULD, MUST, MUST NOT, and SHOULD NOT as defined in [[RFC2119]](http://go.microsoft.com/fwlink/?LinkId=90317). Sections 1.5 and 1.9 are also normative but do not contain those terms. All other sections and examples in this specification are informative.

## Glossary

The following terms are specific to this document:

**client**: A computer on which the remote procedure call (RPC) client is executing.

**endpoint**: A network-specific address of a remote procedure call (RPC) server process for remote procedure calls. The actual name and type of the endpoint depends on the RPC protocol sequence that is being used. For example, for RPC over TCP (RPC Protocol Sequence ncacn\_ip\_tcp), an endpoint might be TCP port 1025. For RPC over Server Message Block (RPC Protocol Sequence ncacn\_np), an endpoint might be the name of a named pipe. For more information, see [[C706]](http://go.microsoft.com/fwlink/?LinkId=89824).

**little-endian**: Multiple-byte values that are byte-ordered with the least significant byte stored in the memory location with the lowest address.

**peer**: The entity being authenticated by the authenticator.

**protocol data unit (PDU)**: Information that is delivered as a unit among peer entities of a network and that may contain control information, address information, or data. For more information on remote procedure call (RPC)-specific PDUs, see [C706] section 12.

**server**: A computer on which the remote procedure call (RPC) server is executing.

**Unicode**: A character encoding standard developed by the Unicode Consortium that represents almost all of the written languages of the world. The [**Unicode**](#gt_c305d0ab-8b94-461a-bd76-13b40cb8c4d8) standard [[UNICODE5.0.0/2007]](http://go.microsoft.com/fwlink/?LinkId=154659) provides three forms (UTF-8, UTF-16, and UTF-32) and seven schemes (UTF-8, UTF-16, UTF-16 BE, UTF-16 LE, UTF-32, UTF-32 LE, and UTF-32 BE).

**virtual channel**: A transport used for communication between a client and a server component over a main data connection, in 1600-byte chunks, as specified in Static Virtual Channels in [[MS-RDPBCGR]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf).

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as defined in [[RFC2119]](http://go.microsoft.com/fwlink/?LinkId=90317). All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

## References

Links to a document in the Microsoft Open Specifications library point to the correct section in the most recently published version of the referenced document. However, because individual documents in the library are not updated at the same time, the section numbers in the documents may not match. You can confirm the correct section numbering by checking the [Errata](http://msdn.microsoft.com/en-us/library/dn781092.aspx).

### Normative References

We conduct frequent surveys of the normative references to assure their continued availability. If you have any issue with finding a normative reference, please contact [dochelp@microsoft.com](mailto:dochelp@microsoft.com). We will assist you in finding the relevant information.

[MS-RDPBCGR] Microsoft Corporation, "[Remote Desktop Protocol: Basic Connectivity and Graphics Remoting](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf)".

[MS-WMF] Microsoft Corporation, "[Windows Metafile Format](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-WMF%5d.pdf)".

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, [http://www.rfc-editor.org/rfc/rfc2119.txt](http://go.microsoft.com/fwlink/?LinkId=90317)

### Informative References

[MSDN-SHELLCLIP] Microsoft Corporation, "Shell Clipboard Formats", [http://msdn.microsoft.com/en-us/library/bb776902.aspx](http://go.microsoft.com/fwlink/?LinkId=90131)

## Overview

This section describes the fundamentals of the system clipboard and gives a high-level overview of the operation of the Remote Desktop Protocol: Clipboard Virtual Channel Extension.

### Clipboard Basics

The system clipboard provided by modern operating systems allows users to transfer data of various formats between applications that are running on the same computer.

To copy data from one application to another, a user first places a selection of data onto the clipboard. This is called a "copy operation". The user then switches to another application and, after having navigated to an appropriate location within this application, the data is extracted from the clipboard and inserted into the target location. This is called a "paste" operation.

The copy and paste operations are two distinct actions and form the basis for clipboard manipulation. The copy operation always takes place before the paste operation to guarantee that the correct data is transferred. The passage of time between the copy and paste operations does not affect the outcome of the transfer of data.

System clipboards are not limited to holding one particular format of data at a given time. Instead, they provide the ability to store data of various formats simultaneously. For example, assume that a user has copied some text to the clipboard. The application that places the text onto the clipboard can place multiple formats of that text onto the clipboard, such as [**Unicode**](#gt_c305d0ab-8b94-461a-bd76-13b40cb8c4d8), ASCII 8, or a graphical image representation. This flexibility enables a wide spectrum of applications to extract data from the clipboard. For example, some applications may be able to manipulate only ASCII 8 text. By allowing both Unicode and ASCII 8 formats to coexist on the clipboard, ASCII 8-only applications can obtain text from applications that natively manipulate Unicode characters, but are flexible enough to place data onto the clipboard in a wider array of formats.

Basic programmatic access to the clipboard provided by an operating system usually ensures that any application has the ability to do the following:

* Place data onto the clipboard.
* Extract data from the clipboard.
* Enumerate the data formats available on the clipboard.
* Register to receive notifications when the system clipboard is updated.

Applications that leverage operating system-supplied clipboard functionality can share data seamlessly, provided that the data is of the appropriate format. Of course, if an application does not make use of system-supplied clipboard functionality, its ability to exchange data with other applications is constrained and limited to ad-hoc or proprietary mechanisms.

#### Data Types

Data placed onto the clipboard can conform to any format, and any application can use this data as long as it is able to correctly interpret the format. The type of data that can be transferred by using the Remote Desktop Protocol: Clipboard Virtual Channel Extension is divided into five categories:

* [Generic data (section 1.3.1.1.1)](#Section_21ba2e40b4eb4edfbd76519907af6bba)
* [Palette data (section 1.3.1.1.2)](#Section_83bb9339fdb24d408f394fbcd47abf17)
* [Metafile data (section 1.3.1.1.3)](#Section_184cbfe69ccb44778127d97563ccfdd4)
* [File List data (section 1.3.1.1.4)](#Section_d005815303b84267b1e9447762558ece)
* [File Stream data (section 1.3.1.1.5)](#Section_6119fce572b84ec38871f0b785ededd3)

These five classes of data are the only data formats manipulated by the Remote Desktop Protocol: Clipboard Virtual Channel Extension.

##### Generic

Generic data is not manipulated or re-encoded by the Remote Desktop Protocol: Clipboard Virtual Channel Extension. It is treated as opaque and passed from one [**virtual channel**](#gt_13d22ba5-5d8e-4815-9728-0629e8422ceb) [**endpoint**](#gt_b91c1e27-e8e0-499b-8c65-738006af72ee) to another without any modification.

##### Palette

Palette data contains a predefined set of mappings from a given index to a red, green, and blue (RGB) triplet. Each triplet represents a color in the additive RGB color space. Palette data to be transferred between virtual channel endpoints is specially encoded for transport on the wire by the Remote Desktop Protocol: Clipboard Virtual Channel Extension.

##### Metafile

A Windows metafile (as specified in [[MS-WMF]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-WMF%5d.pdf) section 2) is a collection of structures that can store an image in an application-independent format. The stored image can be recreated by processing the metafile structures. Also called a vector image, a metafile contains a sequence of drawing commands and settings. The commands and settings recorded in a metafile object can be rendered on a display, output by a printer or plotter, stored in memory, or saved to a file or stream. Metafile data to be transferred between virtual channel endpoints is specially encoded and decoded for transport on the wire by the Remote Desktop Protocol: Clipboard Virtual Channel Extension.

##### File List

A File List contains a list of files to be transferred.

##### File Stream

A File Stream encapsulates the contents of a file that resides on some form of long-term storage. The Remote Desktop Protocol: Clipboard Virtual Channel Extension provides the ability to transfer selected chunks of a file between virtual channel endpoints, as opposed to having to transfer the entire file image. A File Stream can also be part of a larger collection of streams, where each stream can be referenced independently (for example, when transferring a group of files).

#### Clipboard Format

All data copied to a system clipboard must conform to a format specification, known as a Clipboard Format. Each Clipboard Format is identified by a unique numeric format ID. This format ID is used to tag the data on the clipboard so that any application enumerating the contents of the clipboard is able to determine the format of the data without having to extract and analyze it.

A textual name can also be associated with each Clipboard Format. This format name is required when the Clipboard Format ID that identifies the format is not constant across computer systems. For example, the Clipboard Format name for "XYZ Drawing Image" may be identified by the Clipboard Format ID "0x1234" on one system, but "0x4321" on another.

Some formats may be implicit to an operating system and hence be represented by a hard-coded numeric Clipboard Format ID that is constant across all computers running that operating system. In this case, a Clipboard Format name is not required because the well-known Clipboard Format ID can always be used to uniquely identify the type.

It is the responsibility of the operating system to assign unique Clipboard Format IDs and to manage the associated Clipboard Format names of data that may be placed on the system clipboard.

Within the context of the Remote Desktop Protocol: Clipboard Virtual Channel Extension, the [Palette (section 1.3.1.1.2)](#Section_83bb9339fdb24d408f394fbcd47abf17) and [Metafile (section 1.3.1.1.3)](#Section_184cbfe69ccb44778127d97563ccfdd4) format types use the following hard-coded Clipboard Format IDs.

| Format name | Format ID |
| --- | --- |
| Palette (section 1.3.1.1.2) | 9 |
| Metafile (section 1.3.1.1.3) | 3 |

When referring to palette or metafile data within the context of the Remote Desktop Protocol: Clipboard Virtual Channel Extension [**PDUs**](#gt_34715e6f-1612-4b2d-a4bb-3305c56e96f5), these Clipboard Format IDs are always used.

Within the context of the Remote Desktop Protocol: Clipboard Virtual Channel Extension, the File List format type uses the following hard-coded Clipboard Format name.

| Format name | Format Name |
| --- | --- |
| [File List (section 1.3.1.1.4)](#Section_d005815303b84267b1e9447762558ece) | "FileGroupDescriptorW" |

When referring to File List data within the context of the Remote Desktop Protocol: Clipboard Virtual Channel Extension PDUs, this Clipboard Format name is always used.

#### Monitoring Clipboard Updates

To be able to keep two independent system clipboards in sync, it is necessary to monitor both in order to detect when one of them has been updated with new [Clipboard Formats (section 1.3.1.2)](#Section_f293d8e478584f5fb00e8e67323d0538). This monitoring can be carried out by polling the contents of both clipboards on a regular basis to determine if the contents have been updated. A more efficient mechanism than polling is to register for update notifications when the clipboard is changed; however, the ability to register for update notifications may not be available on all operating systems.

#### Delayed Rendering of Clipboard Data

Delayed rendering is a data transfer principle that makes it possible to keep two clipboards in sync while minimizing the network bandwidth required for communication. The underlying premise of delayed rendering is that data needs to be provided only when requested. When a copy operation takes place, the actual data associated with the [Clipboard Format (section 1.3.1.2)](#Section_f293d8e478584f5fb00e8e67323d0538) is not copied onto the remote clipboard. Only the format ID that represents the Clipboard Format is placed on the clipboard. The data associated with the Clipboard Format is sent only if a paste operation is executed. The Remote Desktop Protocol: Clipboard Virtual Channel Extension requires that the system clipboard support delayed rendering of data to ensure the efficiency of clipboard synchronization.

### Clipboard Redirection Virtual Channel Protocol

The Remote Desktop Protocol: Clipboard Virtual Channel Extension is divided into two distinct sequences:

* [Initialization Sequence (section 1.3.2.1)](#Section_a5cae3c9170c4154992d9ac8a149cc7e)
* [Data Transfer Sequence (section 1.3.2.2)](#Section_395bc830f2c240e5a3f323e41183b777)

During the Initialization Sequence, the connection is set up and capabilities and settings exchanged. The transfer of [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) IDs, names, and data takes place during the Data Transfer Sequence.

#### Initialization Sequence

The goal of the Initialization Sequence is to establish the [**client**](#gt_60e0e1fa-66fe-41e1-b5e3-ceab97e53506) and the [**server**](#gt_434b0234-e970-4e8c-bdfa-e16a30d96703) capabilities, exchange settings, and synchronize the initial state of the client and server clipboards.

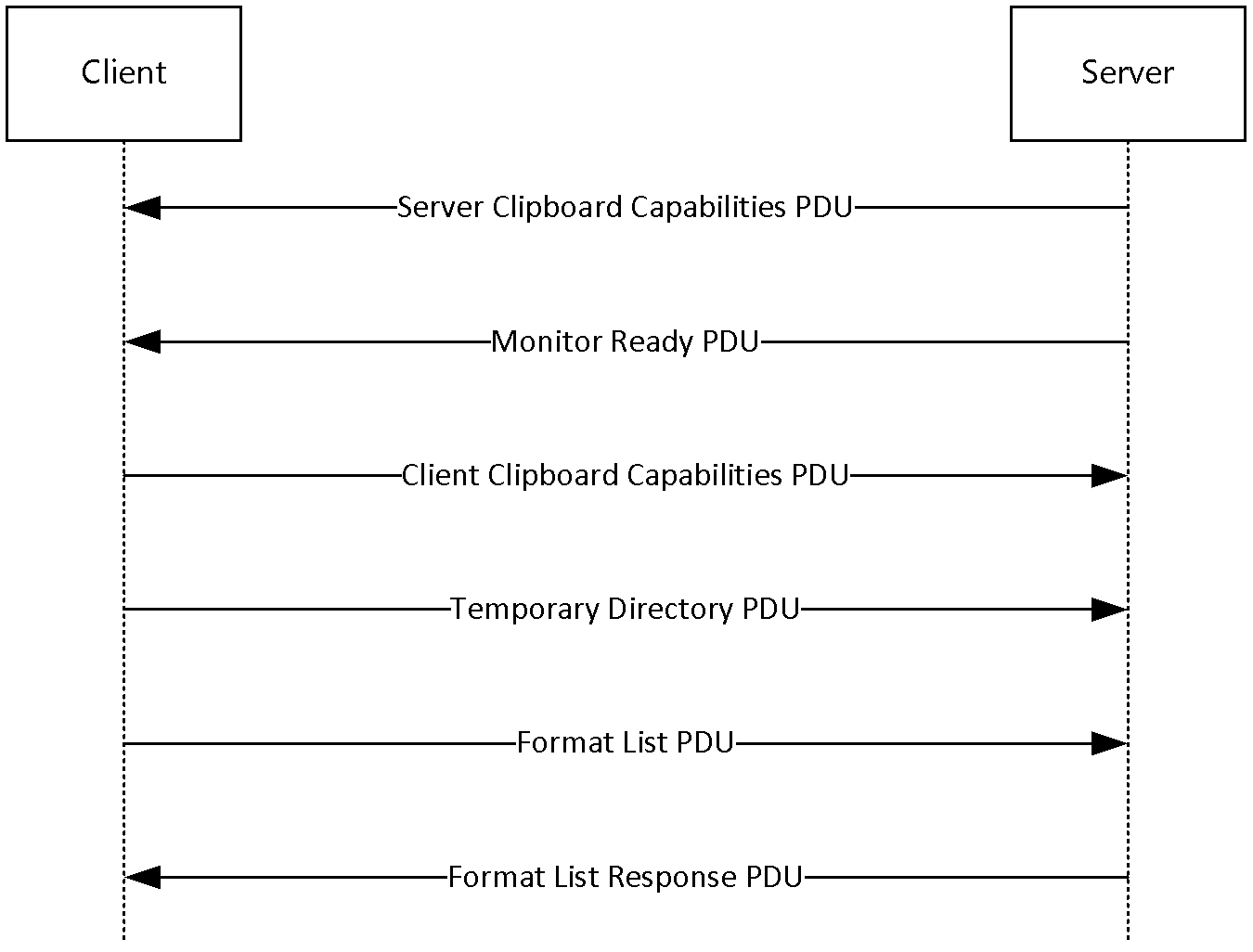


Figure : Clipboard Redirection Initialization Sequence

1. The server sends a [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) to the client to advertise the capabilities that it supports.
2. The server sends a [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69) to the client.
3. Upon receiving the Monitor Ready PDU, the client transmits its capabilities to the server by using a Clipboard Capabilities PDU.
4. The client sends the [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e) to inform the server of a location on the client file system that can be used to deposit files being copied to the client. To make use of this location, the server must be able to access it directly. At this point, the client and the server capability negotiation is complete.
5. The final stage of the Initialization Sequence involves synchronizing the [Clipboard Formats](#Section_f293d8e478584f5fb00e8e67323d0538) on the server clipboard with the client. This is accomplished by effectively mimicking a copy operation on the client by forcing it to send a [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673).
6. The server responds with a [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2).

#### Data Transfer Sequences

The goal of the Data Transfer Sequences is to perform a copy or paste operation. The diagram that follows presents a possible data transfer sequence.

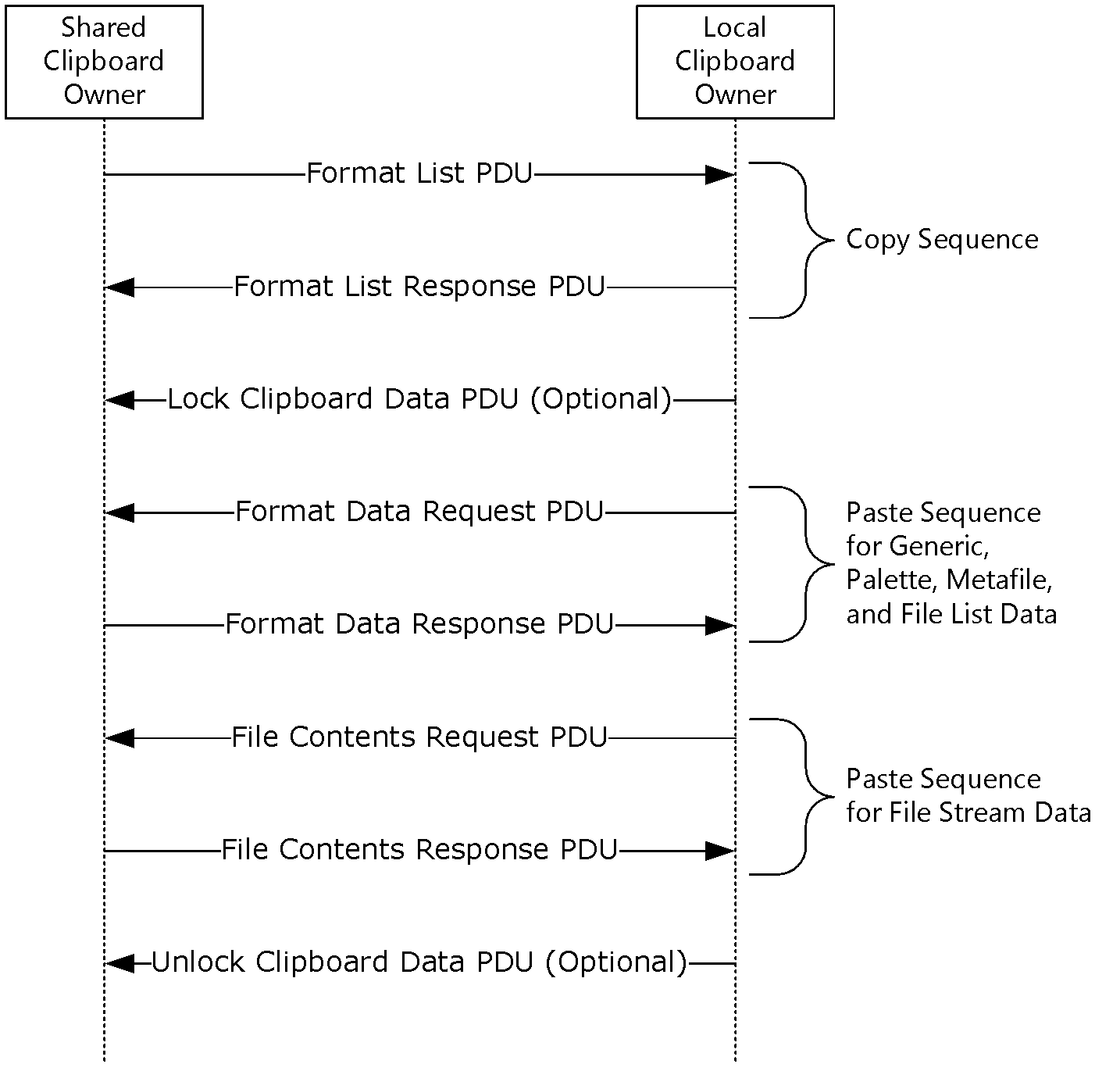


Figure : Data transfer using the shared clipboard

1. The sequence of messages for a copy operation is the same for all format types, as specified in section [1.3.2.2.1](#Section_887747ab9bad490f8ddb68c6365e58dd).
2. However, the messages exchanged to transfer File Stream data during a paste operation differs from those used to transfer other format data, as specified in section [1.3.2.2.3](#Section_30688d0996b646f8af18ea1998bb7987).

##### Copy Sequence

The goal of the Copy Sequence is to synchronize the list of available formats across the client and the server clipboards.

The [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) is sent by the Remote Desktop Protocol: Clipboard Virtual Channel Extension virtual channel endpoint on which the local clipboard has been updated. This endpoint is called the "Shared Clipboard Owner." The Format List PDU contains a list of the updated formats that are available on the clipboard of the Shared Owner. The recipient of the Format List PDU must update its local system clipboard with IDs of the [Clipboard Formats](#Section_f293d8e478584f5fb00e8e67323d0538) that are available on the Shared Owner, and then send the [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2) in response. The sender of the Format List Response PDU is called the "Local Clipboard Owner." The format data is delay rendered, as specified in section [1.3.1.4](#Section_fa309d1b803444bfb927adfc753e69c1).

##### Locking and Unlocking Clipboard Data

The [Lock Clipboard Data PDU](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc) can be sent at any point in time by a Local Clipboard Owner after the clipboard capabilities and temporary directory have been exchanged in the Clipboard Initialization Sequence. The purpose of this PDU is to request that the Shared Clipboard Owner retain all File Stream data on the clipboard until the [Unlock Clipboard Data PDU](#Section_2ae0ff2619b24eb8af0a4811a6a18906) is received.

##### Paste Sequence

The goal of the Paste Sequence is to transfer the data for a single format from the Shared Clipboard Owner to the Local Clipboard Owner.

The [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) is sent by the Local Clipboard Owner in response to a paste operation. This PDU contains the Clipboard Format ID, relative to the Shared Clipboard Owner's system, of the data that is required to complete the paste operation on the Local Clipboard Owner. The Shared Clipboard Owner retrieves the requested data from its local system clipboard and sends it to the Local Clipboard Owner in a [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b).

The [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2) and [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85) are used to implement the transfer of files. The Local Clipboard Owner must first request the list of files available from the clipboard. Upon receipt of this list, the Local Clipboard Owner can request the contents of a file listed therein by sending a File Contents Request PDU to the Shared Clipboard Owner. The resultant file contents data is transmitted by the Shared Clipboard Owner to the Local Clipboard Owner by using a File Contents Response PDU.

#### Interacting with Local Clipboard and Applications

The following diagram and accompanying explanation illustrate how an application, the system clipboards, and the Remote Desktop Protocol: Clipboard Virtual Channel Extension endpoints interact during a copy-and-paste operation.

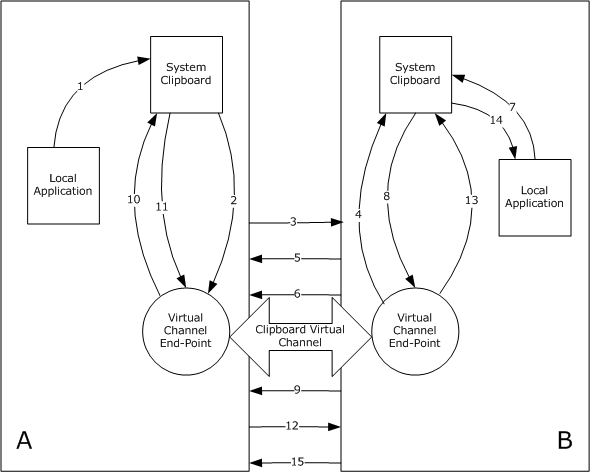


Figure : Interaction of applications, the system clipboards, and virtual channel endpoints

The copy phase is performed in steps 1 through 5 and the paste phase is performed in steps 6 through 15:

1. Local Application A copies data to System Clipboard A.
2. Virtual Channel Endpoint A is notified of the clipboard update.
3. A list of formats on System Clipboard A are sent to Virtual Channel Endpoint B in a [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673).
4. Virtual Channel Endpoint B updates System Clipboard B.
5. Virtual Channel Endpoint B confirms success of System Clipboard B update in a [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2).
6. Virtual Channel Endpoint B sends the optional [Lock Clipboard Data PDU](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc) to request that any File Stream data on System Clipboard A remain accessible until the [Unlock Clipboard Data PDU](#Section_2ae0ff2619b24eb8af0a4811a6a18906) is sent.
7. Local Application B requests data from System Clipboard B.
8. System Clipboard B requests delay-rendered data from Virtual Channel Endpoint B.
9. Virtual Channel Endpoint B sends a request for data of requested type using a [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) or [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2).
10. Virtual Channel Endpoint A requests data from System Clipboard A.
11. System Clipboard A returns data to Virtual Channel Endpoint A.
12. Data is sent to Virtual Channel Endpoint B using a [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) or [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85).
13. Virtual Channel Endpoint B supplies System Clipboard B with data.
14. System Clipboard B supplies Application B with data.
15. Virtual Channel Endpoint B sends the optional Unlock Clipboard Data PDU.

## Relationship to Other Protocols

The Remote Desktop Protocol: Clipboard Virtual Channel Extension is embedded in a static virtual channel transport, as specified in [[MS-RDPBCGR]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf) sections 1.3.3, 2.2.6 and 3.1.5.2.

## Prerequisites/Preconditions

The Remote Desktop Protocol: Clipboard Virtual Channel Extension operates only after the static virtual channel transport (as specified in [[MS-RDPBCGR]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf) sections 1.3.3, 2.2.6 and 3.1.5.2) is fully established and the Confirm Active PDU ([MS-RDPBCGR] section 2.2.1.13.2) has been transmitted from the client to the server. If the static virtual channel transport is terminated, no other communication over the Remote Desktop Protocol: Clipboard Virtual Channel Extension occurs.

## Applicability Statement

The Remote Desktop Protocol: Clipboard Virtual Channel Extension is designed to be run within the context of a Remote Desktop Protocol virtual channel established between a client and server. This protocol is applicable when bidirectional data transfer between the local client clipboard and the clipboard in the remote session (hosted on the server) is required.

## Versioning and Capability Negotiation

The Remote Desktop Protocol: Clipboard Virtual Channel Extension is capability-based. The client and the server exchange capabilities during the protocol [Initialization Sequence (section 1.3.2.1)](#Section_a5cae3c9170c4154992d9ac8a149cc7e) by using the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3). Capability sets are packaged in a combined capability set structure. This structure contains a count of the number of capability sets, followed by the contents of the individual capability sets.

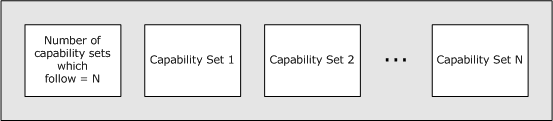


Figure : Combined capability set structure

After the capabilities have been received and stored, the client and the server do not send PDUs or data formats that cannot be processed by the [**peer**](#gt_e5d0d91c-9a39-493f-ab1b-f36ce840e6a2).

## Vendor-Extensible Fields

None.

## Standards Assignments

None.

# Messages

## Transport

The Remote Desktop Protocol: Clipboard Virtual Channel Extension is designed to operate over a static virtual channel, as specified in [[MS-RDPBCGR]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf) sections 1.3.3, 2.2.6 and 3.1.5.2. The virtual channel name is "CLIPRDR". The Remote Desktop Protocol layer manages the creation, setup, and transmission of data over the virtual channel.

## Message Syntax

The following sections contain Remote Desktop Protocol: Desktop Composition Virtual Channel Extension message syntax.

### Clipboard PDU Header (CLIPRDR\_HEADER)

The CLIPRDR\_HEADER structure is present in all clipboard PDUs. It is used to identify the PDU type, specify the length of the PDU, and convey message flags.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| | Value | Meaning | | --- | --- | | CB\_MONITOR\_READY  0x0001 | [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69) | | CB\_FORMAT\_LIST  0x0002 | [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) | | CB\_FORMAT\_LIST\_RESPONSE  0x0003 | [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2) | | CB\_FORMAT\_DATA\_REQUEST  0x0004 | [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) | | CB\_FORMAT\_DATA\_RESPONSE  0x0005 | [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) | | CB\_TEMP\_DIRECTORY  0x0006 | [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e) | | CB\_CLIP\_CAPS  0x0007 | [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) | | CB\_FILECONTENTS\_REQUEST  0x0008 | [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2) | | CB\_FILECONTENTS\_RESPONSE  0x0009 | [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85) | | CB\_LOCK\_CLIPDATA  0x000A | [Lock Clipboard Data PDU](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc) | | CB\_UNLOCK\_CLIPDATA  0x000B | [Unlock Clipboard Data PDU](#Section_2ae0ff2619b24eb8af0a4811a6a18906) |   msgType | | | | | | | | | | | | | | | | msgFlags | | | | | | | | | | | | | | | |
| dataLen | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**msgType (2 bytes):** An unsigned, 16-bit integer that specifies the type of the clipboard PDU that follows the **dataLen** field.

**msgFlags (2 bytes):** An unsigned, 16-bit integer that indicates message flags.

| Value | Meaning |
| --- | --- |
| CB\_RESPONSE\_OK  0x0001 | Used by the Format List Response PDU, Format Data Response PDU, and File Contents Response PDU to indicate that the associated request Format List PDU, Format Data Request PDU, and File Contents Request PDU were processed successfully. |
| CB\_RESPONSE\_FAIL  0x0002 | Used by the Format List Response PDU, Format Data Response PDU, and File Contents Response PDU to indicate that the associated Format List PDU, Format Data Request PDU, and File Contents Request PDU were not processed successfully. |
| CB\_ASCII\_NAMES  0x0004 | Used by the [Short Format Name](#Section_07765daa65da4b0087eada32b155e12d) variant of the Format List Response PDU to indicate that the format names are in ASCII 8. |

**dataLen (4 bytes):** An unsigned, 32-bit integer that specifies the size, in bytes, of the data which follows the Clipboard PDU Header.[<1>](#Appendix_A_1)

### Initialization Sequence

The following sections contain Remote Desktop Protocol: Desktop Composition Virtual Channel Extension message syntax for the [Initialization Sequence (section 1.3.2.1)](#Section_a5cae3c9170c4154992d9ac8a149cc7e).

#### Clipboard Capabilities PDU (CLIPRDR\_CAPS)

The Clipboard Capabilities PDU is an optional PDU used to exchange capability information. It is first sent from the server to the client. Upon receipt of the [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69), the client sends the Clipboard Capabilities PDU to the server.

If this PDU is not sent by a Remote Desktop Protocol: Clipboard Virtual Channel Extension endpoint, it is assumed that the endpoint is using the default values for each capability field.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| cCapabilitiesSets | | | | | | | | | | | | | | | | pad1 | | | | | | | | | | | | | | | |
| capabilitySets (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_CLIP\_CAPS (0x0007), while the **msgFlags** field MUST be set to 0x0000.

**cCapabilitiesSets (2 bytes):** An unsigned, 16-bit integer that specifies the number of [CLIPRDR\_CAPS\_SETs](#Section_c160dde9a3d845d79af75f5822b2e6a8), present in the **capabilitySets** field.

**pad1 (2 bytes):** An unsigned, 16-bit integer used for padding. Values in this field are ignored.

**capabilitySets (variable):** A variable-sized array of capability sets, each conforming in structure to the CLIPRDR\_CAPS\_SET.

##### Capability Set (CLIPRDR\_CAPS\_SET)

The CLIPRDR\_CAPS\_SET structure is used to wrap capability set data and to specify the type and size of this data exchanged between the client and the server. All capability sets conform to this basic structure.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| capabilitySetType | | | | | | | | | | | | | | | | lengthCapability | | | | | | | | | | | | | | | |
| capabilityData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**capabilitySetType (2 bytes):** An unsigned, 16-bit integer used as a type identifier of the capability set.

| Value | Meaning |
| --- | --- |
| CB\_CAPSTYPE\_GENERAL  0x0001 | [General Capability Set](#Section_7718c8c9798d4788bb7564afdc913869) |

**lengthCapability (2 bytes):** An unsigned, 16-bit integer that specifies the combined length, in bytes, of the **capabilitySetType**, **capabilityData** and **lengthCapability** fields.

**capabilityData (variable):** Capability set data specified by the type given in the **capabilitySetType** field. This field is a variable number of bytes.

###### General Capability Set (CLIPRDR\_GENERAL\_CAPABILITY)

The CLIPRDR\_GENERAL\_CAPABILITY structure is used to advertise general clipboard settings.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| capabilitySetType | | | | | | | | | | | | | | | | lengthCapability | | | | | | | | | | | | | | | |
| version | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| generalFlags | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**capabilitySetType (2 bytes):** An unsigned 16-bit integer that specifies the type of the capability set. This field MUST be set to CB\_CAPSTYPE\_GENERAL (0x0001).

**lengthCapability (2 bytes):** An unsigned, 16-bit integer that specifies the length, in bytes, of the **capabilitySetType**, **capability data** and **lengthCapability** fields.

**version (4 bytes):** An unsigned, 32-bit integer that specifies the Remote Desktop Protocol: Clipboard Virtual Channel Extension version number. This field is for informational purposes and MUST NOT be used to make protocol capability decisions. The actual features supported are specified in the **generalFlags** field.

| Value | Meaning |
| --- | --- |
| CB\_CAPS\_VERSION\_1  0x00000001 | Version 1 |
| CB\_CAPS\_VERSION\_2  0x00000002 | Version 2 |

**generalFlags (4 bytes):** An unsigned, 32-bit integer that specifies the general capability flags.

| Value | Meaning |
| --- | --- |
| CB\_USE\_LONG\_FORMAT\_NAMES  0x00000002 | The [Long Format Name](#Section_feee15ee787d4ca2bb41136cee50d3de) variant of the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) is supported for exchanging updated format names. If this flag is not set, the [Short Format Name](#Section_07765daa65da4b0087eada32b155e12d) variant MUST be used. If this flag is set by both protocol endpoints, then the Long Format Name variant MUST be used. |
| CB\_STREAM\_FILECLIP\_ENABLED  0x00000004 | File copy and paste using stream-based operations are supported using the [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2) and [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85). |
| CB\_FILECLIP\_NO\_FILE\_PATHS  0x00000008 | Indicates that any description of files to copy and paste MUST NOT include the source path of the files. |
| CB\_CAN\_LOCK\_CLIPDATA  0x00000010 | Locking and unlocking of File Stream data on the clipboard is supported using the [Lock Clipboard Data PDU](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc) and [Unlock Clipboard Data PDU](#Section_2ae0ff2619b24eb8af0a4811a6a18906). |

If the General Capability Set is not present in the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3), then the default set of general capabilities MUST be assumed. By definition the default set does not specify any flags in the **generalFlags** field, that is the **generalFlags** field is set to 0x00000000.

#### Server Monitor Ready PDU (CLIPRDR\_MONITOR\_READY)

The Monitor Ready PDU is sent from the server to the client to indicate that the server is initialized and ready. This PDU is transmitted by the server after it has sent the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) to the client.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):**  A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_MONITOR\_READY (0x0001), while the **msgFlags** field MUST be set to 0x0000.

#### Client Temporary Directory PDU (CLIPRDR\_TEMP\_DIRECTORY)

The Temporary Directory PDU is an optional PDU sent from the client to the server. This PDU informs the server of a location on the client file system that MUST be used to deposit files being copied to the client. The location MUST be accessible by the server to be useful. Section [3.1.1.3](#Section_d66378b9575b4239aa52c32ea67d97e5) specifies how direct file access impacts file copy and paste. This PDU is sent by the client after receiving the [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| wszTempDir (520 bytes) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_TEMP\_DIRECTORY (0x0006), while the **msgFlags** field MUST be set to 0x0000.

**wszTempDir (520 bytes):** A 520-byte block that contains a null-terminated string that represents the directory on the client that MUST be used to store temporary clipboard related information. The supplied path MUST be absolute and relative to the local client system, for example, "c:\temp\clipdata". Any space not used in this field SHOULD be filled with null characters.

### Copy Sequence

The following sections contain Remote Desktop Protocol: Desktop Composition Virtual Channel Extension message syntax for the [Copy Sequence (section 1.3.2.2.1)](#Section_887747ab9bad490f8ddb68c6365e58dd).

#### Format List PDU (CLIPRDR\_FORMAT\_LIST)

The Format List PDU is sent by either the client or the server when its local system clipboard is updated with new clipboard data. This PDU contains the [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID and name pairs of the new Clipboard Formats on the clipboard.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| formatListData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FORMAT\_LIST (0x0002), while the **msgFlags** field MUST be set to 0x0000 or CB\_ASCII\_NAMES (0x0004) depending on the type of data present in the **formatListData** field.

**formatListData (variable):** An array consisting solely of either [Short Format Names](#Section_6a20969b779d452eab7c5c8ad6dc90b8) or [Long Format Names](#Section_704b2025336f424da81c42395c85808d). The type of structure used in the array is determined by the presence of the CB\_USE\_LONG\_FORMAT\_NAMES (0x00000002) flag in the **generalFlags** field of the [General Capability Set (section 2.2.2.1.1.1)](#Section_7718c8c9798d4788bb7564afdc913869). Each array holds a list of the Clipboard Format ID and name pairs available on the local system clipboard of the sender. If Short Format Names are being used, and the embedded Clipboard Format names are in ASCII 8 format, then the **msgFlags** field of the **clipHeader** must contain the CB\_ASCII\_NAMES (0x0004) flag.

##### Short Format Names (CLIPRDR\_SHORT\_FORMAT\_NAMES)

The CLIPRDR\_SHORT\_FORMAT\_NAMES structure holds a collection of [CLIPRDR\_SHORT\_FORMAT\_NAME](#Section_07765daa65da4b0087eada32b155e12d) structures.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| shortFormatNames (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**shortFormatNames (variable):** An array of CLIPRDR\_SHORT\_FORMAT\_NAME structures.

###### Short Format Name (CLIPRDR\_SHORT\_FORMAT\_NAME)

The CLIPRDR\_SHORT\_FORMAT\_NAME structure holds a [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID and Clipboard Format name pair.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| formatId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| formatName (32 bytes) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**formatId (4 bytes):** An unsigned, 32-bit integer specifying the Clipboard Format ID.

**formatName (32 bytes):** A 32-byte block containing the null-terminated name assigned to the Clipboard Format (32 ASCII 8 characters or 16 Unicode characters). If the name does not fit, it MUST be truncated. Not all Clipboard Formats have a name, and in that case the **formatName** field MUST contain only zeros.

##### Long Format Names (CLIPRDR\_LONG\_FORMAT\_NAMES)

The CLIPRDR\_LONG\_FORMAT\_NAMES structure holds a collection of [CLIPRDR\_LONG\_FORMAT\_NAME](#Section_feee15ee787d4ca2bb41136cee50d3de) structures.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| longFormatNames (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**longFormatNames (variable):** An array of CLIPRDR\_LONG\_FORMAT\_NAME structures.

###### Long Format Name (CLIPRDR\_LONG\_FORMAT\_NAME)

The CLIPRDR\_LONG\_FORMAT\_NAME structure holds a [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID and a Clipboard Format name pair.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| formatId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| wszFormatName (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**formatId (4 bytes):** An unsigned, 32-bit integer that specifies the Clipboard Format ID.

**wszFormatName (variable):** A variable length null-terminated Unicode string name that contains the Clipboard Format name. Not all Clipboard Formats have a name; in such cases, the **formatName** field MUST consist of a single Unicode null character.

#### Format List Response PDU (FORMAT\_LIST\_RESPONSE)

The Format List Response PDU is sent as a reply to the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673). It is used to indicate whether processing of the Format List PDU was successful.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FORMAT\_LIST\_RESPONSE (0x0003). The CB\_RESPONSE\_OK (0x0001) or CB\_RESPONSE\_FAIL (0x0002) flag MUST be set in the **msgFlags** field of the Clipboard PDU Header.

### Locking and Unlocking Clipboard Data

#### Lock Clipboard Data PDU (CLIPRDR\_LOCK\_CLIPDATA)

The Lock Clipboard Data PDU can be sent at any point in time after the clipboard capabilities and temporary directory have been exchanged in the Clipboard Initialization Sequence (section [1.3.2.1](#Section_a5cae3c9170c4154992d9ac8a149cc7e)) by a Local Clipboard Owner (section [1.3.2.2.1](#Section_887747ab9bad490f8ddb68c6365e58dd)). The purpose of this PDU is to request that the Shared Clipboard Owner (section 1.3.2.2.1) retain all File Stream (section [1.3.1.1.5](#Section_6119fce572b84ec38871f0b785ededd3)) data on the clipboard until the Unlock Clipboard Data PDU (section [2.2.4.2](#Section_2ae0ff2619b24eb8af0a4811a6a18906)) is received. This ensures that File Stream data can be requested by the Local Owner in a subsequent File Contents Paste Sequence (section [1.3.2.2.3](#Section_30688d0996b646f8af18ea1998bb7987)) by using the File Contents Request PDU (section [2.2.5.3](#Section_cbc851d34e6845f4929226872a9209f2)) even when the Shared Owner clipboard has changed and the File Stream data is no longer available.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| clipDataId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_LOCK\_CLIPDATA (0x000A), while the **msgFlags** field MUST be set to 0x0000.

**clipDataId (4 bytes):** An unsigned, 32-bit integer that is used to tag File Stream data on the Shared Owner clipboard so that it can be requested in a subsequent File Contents Request PDU (section 2.2.5.3).

#### Unlock Clipboard Data PDU (CLIPRDR\_UNLOCK\_CLIPDATA)

The Unlock Clipboard Data PDU can be sent at any point in time after the Clipboard Initialization Sequence (section [1.3.2.1](#Section_a5cae3c9170c4154992d9ac8a149cc7e)) by a Local Clipboard Owner (section [1.3.2.2.1](#Section_887747ab9bad490f8ddb68c6365e58dd)). The purpose of this PDU is to notify the Shared Clipboard Owner (section 1.3.2.2.1) that File Stream data that was locked in response to the Lock Clipboard Data PDU (section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc)) can be released.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| clipDataId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_UNLOCK\_CLIPDATA (0x000B), while the **msgFlags** field MUST be set to 0x0000.

**clipDataId (4 bytes):** An unsigned, 32-bit integer that identifies the File Stream data that was locked by the Lock Clipboard Data PDU (section 2.2.4.1) and can now be released.

### Paste Sequence

The following sections contain Remote Desktop Protocol: Desktop Composition Virtual Channel Extension message syntax for the [Paste Sequence (section 1.3.2.2.3)](#Section_30688d0996b646f8af18ea1998bb7987).

#### Format Data Request PDU (CLIPRDR\_FORMAT\_DATA\_REQUEST)

The Format Data Request PDU is sent by the recipient of the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673). It is used to request the data for one of the formats that was listed in the Format List PDU.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| requestedFormatId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FORMAT\_DATA\_REQUEST (0x0004), while the **msgFlags** field MUST be set to 0x0000.

**requestedFormatId (4 bytes):** An unsigned, 32-bit integer that specifies the [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID of the clipboard data. The Clipboard Format ID MUST be one listed previously in the Format List PDU.

#### Format Data Response PDU (CLIPRDR\_FORMAT\_DATA\_RESPONSE)

The Format Data Response PDU is sent as a reply to the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870). It is used to indicate whether processing of the Format Data Request PDU was successful. If the processing was successful, the Format Data Response PDU includes the contents of the requested clipboard data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| requestedFormatData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FORMAT\_DATA\_RESPONSE (0x0005). The CB\_RESPONSE\_OK (0x0001) or CB\_RESPONSE\_FAIL (0x0002) flag MUST be set in the **msgFlags** field of the Clipboard PDU Header structure.

**requestedFormatData (variable):** Variable length clipboard format data. The contents of this field MUST be one of the following types: generic, [Packed Metafile Payload](#Section_051ca890f6f14a9aa86bb59827b698bc), or [Packed Palette Payload](#Section_45ff6ca1795c44d28936633775ea0a3c).

##### Packed Metafile Payload (CLIPRDR\_MFPICT)

The CLIPRDR\_MFPICT structure is used to transfer a Windows metafile. The Windows metafile format is specified in [[MS-WMF]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-WMF%5d.pdf) section 2.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| mappingMode | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| xExt | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| yExt | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| metaFileData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**mappingMode (4 bytes):** An unsigned, 32-bit integer specifying the mapping mode in which the picture is drawn.

| Value | Meaning |
| --- | --- |
| MM\_TEXT  0x00000001 | Each logical unit is mapped to one device pixel. Positive x is to the right; positive y is down. |
| MM\_LOMETRIC  0x00000002 | Each logical unit is mapped to 0.1 millimeter. Positive x is to the right; positive y is up. |
| MM\_HIMETRIC  0x00000003 | Each logical unit is mapped to 0.01 millimeter. Positive x is to the right; positive y is up. |
| MM\_LOENGLISH  0x00000004 | Each logical unit is mapped to 0.01 inch. Positive x is to the right; positive y is up. |
| MM\_HIENGLISH  0x00000005 | Each logical unit is mapped to 0.001 inch. Positive x is to the right; positive y is up. |
| MM\_TWIPS  0x00000006 | Each logical unit is mapped to 1/20 of a printer's point (1/1440 of an inch), also called a twip. Positive x is to the right; positive y is up. |
| MM\_ISOTROPIC  0x00000007 | Logical units are mapped to arbitrary units with equally scaled axes; one unit along the x-axis is equal to one unit along the y-axis. |
| MM\_ANISOTROPIC  0x00000008 | Logical units are mapped to arbitrary units with arbitrarily scaled axes. |

For MM\_ISOTROPIC and MM\_ANISOTROPIC modes, which can be scaled, the **xExt** and **yExt** fields contain an optional suggested size in MM\_HIMETRIC units. For MM\_ANISOTROPIC pictures, **xExt** and **yExt** SHOULD be zero when no suggested size is given. For MM\_ISOTROPIC pictures, an aspect ratio MUST be supplied even when no suggested size is given. If a suggested size is given, the aspect ratio is implied by the size. To give an aspect ratio without implying a suggested size, the **xExt** and **yExt** fields are set to negative values whose ratio is the appropriate aspect ratio. The magnitude of the negative **xExt** and **yExt** values is ignored; only the ratio is used.

**xExt (4 bytes):** An unsigned, 32-bit integer that specifies the width of the rectangle within which the picture is drawn, except in the MM\_ISOTROPIC and MM\_ANISOTROPIC modes. The coordinates are in units that correspond to the mapping mode.

**yExt (4 bytes):** An unsigned, 32-bit integer that specifies the height of the rectangle within which the picture is drawn, except in the MM\_ISOTROPIC and MM\_ANISOTROPIC modes. The coordinates are in units that correspond to the mapping mode.

**metaFileData (variable):** The variable sized contents of the metafile as specified in [MS-WMF] section 2.

##### Packed Palette Payload (CLIPRDR\_PALETTE)

The CLIPRDR\_PALETTE structure is used to transfer palette format data.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| paletteEntriesData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**paletteEntriesData (variable):** A variable sized array of [PALETTEENTRY](#Section_6e0dc592ee78488286bce050533f6675) structures.

###### Palette Entry (PALETTEENTRY)

The PALETTEENTRY structure contains a single palette entry that specifies the red, green, and blue components for a given color index, in addition to any application-specific information related to the entry.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| red | | | | | | | | green | | | | | | | | blue | | | | | | | | extra | | | | | | | |

**red (1 byte):** An unsigned, 8-bit red color component.

**green (1 byte):** An unsigned, 8-bit green color component.

**blue (1 byte):** An unsigned, 8-bit blue color component.

**extra (1 byte):** This field MAY be used to convey application-specific palette information. Some applications use this field to specify how the palette entry is used.

##### Packed File List (CLIPRDR\_FILELIST)

The CLIPRDR\_FILELIST structure is used to describe a list of files, each file in the list being represented by a File Descriptor (section [2.2.5.2.3.1](#Section_a765d7842b394b889faa88f8666f9c35)).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| cItems | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fileDescriptorArray (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**cItems (4 bytes):** An unsigned 32-bit integer that specifies the number of entries in the **fileDescriptorArray** field.

**fileDescriptorArray (variable):** An array of File Descriptors (section 2.2.5.2.3.1). The number of elements in the array is specified by the **cItems** field.

###### File Descriptor (CLIPRDR\_FILEDESCRIPTOR)

The CLIPRDR\_FILEDESCRIPTOR structure describes the properties of a file.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| flags | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| reserved1 (32 bytes) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fileAttributes | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| reserved2 (16 bytes) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| lastWriteTime | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fileSizeHigh | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fileSizeLow | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| fileName (520 bytes) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**flags (4 bytes):** An unsigned 32-bit integer that specifies which fields contain valid data and the usage of progress UI during a copy operation.

| Value | Meaning |
| --- | --- |
| FD\_ATTRIBUTES  0x00000004 | The **fileAttributes** field contains valid data. |
| FD\_FILESIZE  0x00000040 | The **fileSizeHigh** and **fileSizeLow** fields contain valid data. |
| FD\_WRITESTIME  0x00000020 | The **lastWriteTime** field contains valid data. |
| FD\_SHOWPROGRESSUI  0x00004000 | A progress indicator SHOULD be shown when copying the file. |

**reserved1 (32 bytes):** An array of 32 bytes. This field MUST be initialized with zeros when sent and MUST be ignored on receipt.

**fileAttributes (4 bytes):** An unsigned 32-bit integer that specifies file attribute flags.

| Value | Meaning |
| --- | --- |
| FILE\_ATTRIBUTE\_READONLY  0x00000001 | A file that is read-only. Applications can read the file, but cannot write to it or delete it. |
| FILE\_ATTRIBUTE\_HIDDEN  0x00000002 | The file or directory is hidden. It is not included in an ordinary directory listing. |
| FILE\_ATTRIBUTE\_SYSTEM  0x00000004 | A file or directory that the operating system uses a part of, or uses exclusively. |
| FILE\_ATTRIBUTE\_DIRECTORY  0x00000010 | Identifies a directory. |
| FILE\_ATTRIBUTE\_ARCHIVE  0x00000020 | A file or directory that is an archive file or directory. Applications typically use this attribute to mark files for backup or removal. |
| FILE\_ATTRIBUTE\_NORMAL  0x00000080 | A file that does not have other attributes set. This attribute is valid only when used alone. |

**reserved2 (16 bytes):** An array of 16 bytes. This field MUST be initialized with zeros when sent and MUST be ignored on receipt.

**lastWriteTime (8 bytes):** An unsigned 64-bit integer that specifies the number of 100-nanoseconds intervals that have elapsed since 1 January 1601 to the time of the last write operation on the file.

**fileSizeHigh (4 bytes):** An unsigned 32-bit integer that contains the most significant 4 bytes of the file size.

**fileSizeLow (4 bytes):** An unsigned 32-bit integer that contains the least significant 4 bytes of the file size.

**fileName (520 bytes):** A null-terminated 260 character Unicode string that contains the name of the file.

#### File Contents Request PDU (CLIPRDR\_FILECONTENTS\_REQUEST)

The File Contents Request PDU is sent by the recipient of the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673). It is used to request either the size of a remote file copied to the clipboard or a portion of the data in the file.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| streamId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| lindex | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| dwFlags | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| nPositionLow | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| nPositionHigh | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| cbRequested | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| clipDataId (optional) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FILECONTENTS\_REQUEST (0x0008), while the **msgFlags** field MUST be set to 0x0000.

**streamId (4 bytes):** An unsigned, 32-bit format ID used to associate the File Contents Request PDU with the corresponding [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85). The File Contents Response PDU is sent as a reply and contains an identical value in the **streamId** field.

**lindex (4 bytes):** A signed, 32-bit integer that specifies the numeric ID of the remote file that is the target of the File Contents Request PDU. This field is used as an index that identifies a particular file in a [File List](#Section_c40918d1ab3840b4ae3ab8758e148926). This File List SHOULD have been obtained as clipboard data in a prior [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) and [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) exchange.

**dwFlags (4 bytes):** An unsigned, 32-bit integer that specifies the type of operation to be performed by the recipient.

| Value | Meaning |
| --- | --- |
| FILECONTENTS\_SIZE  0x00000001 | A request for the size of the file identified by the **lindex** field. The size MUST be returned as a 64-bit, unsigned integer. The **cbRequested** field MUST be set to 0x00000008 and both the **nPositionLow** and **nPositionHigh** fields MUST be set to 0x00000000. |
| FILECONTENTS\_RANGE  0x00000002 | A request for the data present in the file identified by the **lindex** field. The data to be retrieved is extracted starting from the offset given by the **nPositionLow** and **nPositionHigh** fields. The maximum number of bytes to extract is specified by the **cbRequested** field. |

The FILECONTENTS\_SIZE and FILECONTENTS\_RANGE flags MUST NOT be set at the same time.

**nPositionLow (4 bytes):** An unsigned, 32-bit integer that specifies the low bytes of the offset into the remote file, identified by the **lindex** field, from where the data needs to be extracted to satisfy a FILECONTENTS\_RANGE operation.

**nPositionHigh (4 bytes):** An unsigned, 32-bit integer that specifies the high bytes of the offset into the remote file, identified by the **lindex** field, from where the data needs to be extracted to satisfy a FILECONTENTS\_RANGE operation. This field is currently not used because offsets greater than 4,294,967,295 bytes are not supported, and MUST be set to zero.

**cbRequested (4 bytes):** An unsigned, 32-bit integer that specifies the size, in bytes, of the data to retrieve. For a FILECONTENTS\_SIZE operation, this field MUST be set to 0x00000008. In the case of a FILECONTENTS\_RANGE operation, this field contains the maximum number of bytes to read from the remote file.

**clipDataId (4 bytes):** An optional unsigned, 32-bit integer that identifies File Stream data which was tagged in a prior Lock Clipboard Data PDU (section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc)).

#### File Contents Response PDU (CLIPRDR\_FILECONTENTS\_RESPONSE)

The File Contents Response PDU is sent as a reply to the [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2). It is used to indicate whether processing of the File Contents Request PDU was successful. If the processing was successful, the File Contents Response PDU includes either a file size or extracted file data, based on the operation requested in the corresponding File Contents Request PDU.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 1  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 2  0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 3  0 | 1 |
| clipHeader | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| streamId | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| requestedFileContentsData (variable) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| ... | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

**clipHeader (8 bytes):** A [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54). The **msgType** field of the Clipboard PDU Header MUST be set to CB\_FILECONTENTS\_RESPONSE (0x0009). The CB\_RESPONSE\_OK (0x0001) or CB\_RESPONSE\_FAIL (0x0002) flag MUST be set in the **msgFlags** field of the Clipboard PDU Header.

**streamId (4 bytes):** An unsigned, 32-bit numeric ID used to associate the File Contents Response PDU with the corresponding File Contents Request PDU. The File Contents Request PDU that triggered the response MUST contain an identical value in the **streamId** field.

**requestedFileContentsData (variable):** This field contains a variable number of bytes. If the response is to a FILECONTENTS\_SIZE (0x00000001) operation, the **requestedFileContentsData** field holds a 64-bit, unsigned integer containing the size of the file. In the case of a FILECONTENTS\_RANGE (0x00000002) operation, the **requestedFileContentsData** field contains a byte-stream of data extracted from the file.

# Protocol Details

## Common Details

### Abstract Data Model

#### Clipboard Format ID Map

The Clipboard Format ID Map is used to translate local [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) IDs to remote Clipboard Format IDs.

For example, assume that on System A, the Clipboard Format with a format name of "Format X" maps to the format ID 0x00001111; and on System B, the format ID corresponding to the format name "Format X" might be 0x00002222:

* On System A, the format map entry for "Format X" would appear as follows:

Format X: Local ID 0x00001111 maps to Remote ID 0x00002222

* On System B, the format map entry for "Format X" would appear as follows:

Format X: Local ID 0x00002222 maps to Remote ID 0x00001111

The Clipboard Format ID Map is cleared and initialized whenever a [Format List PDU](#Section_4e41486468804b0d8af71838ca000c2d) is processed (section 3.1.5.2.2).

#### File List

When a collection of files is copied to the system clipboard, accompanying metadata containing the list of files, called the "file list," is also placed onto the clipboard using a generic, operating system-defined format. This list contains information about each file on the clipboard, such as the file name, size, and access permissions. Applications can examine the file list to enumerate the list of files available on the system clipboard.

When a paste operation is initiated to obtain the contents of a file on the clipboard, the index of the file in the file list, along with a description of the file chunks required, is sent to the system clipboard. The system clipboard responds by returning the file contents data that was requested.

The usage of the file list is best illustrated with a practical example:

1. Assume that a user copies two files to the clipboard so that the associated file list on the clipboard appears as follows (notice that the exact location of the files is not specified in the file list):
   1. temp\file1.txt (20 bytes)
   2. temp\file2.txt (10 bytes)
2. Next, assume that the user decides to paste the first 15 bytes of file1.txt into a target application that can accept file data. In this case, the target application examines the file list on the clipboard and issues a request for the first 15 bytes of the file in the file list at Index 1 (the system clipboard MUST be contacted because the exact location of the file on the local file system is not necessarily advertised in the file list). The system clipboard responds with the appropriate data.[<2>](#Appendix_A_2)

#### Direct File Access

If the client or server has direct access to the local file system of the peer, a [File List](#Section_c40918d1ab3840b4ae3ab8758e148926) that uses absolute paths (as opposed to relative paths) MAY be used to point directly to the source files, thus bypassing having to contact the system clipboard for the actual file contents.

For example:

1. Assume that the server is able to view the client files via a network share such that the client file c:\temp\file1.txt is accessible as: \\client-files\c\temp\file1.txt.
2. Next, assume that a File List (which contains the files c:\temp\file1.txt and c:\temp\file2.txt) is copied to the client's local clipboard. Then, when transmitting the data in the File List to the server, the client MAY modify the File List contents as follows:
   1. \\client-files\c\temp\file1.txt (20 bytes)
   2. \\client-files\c\temp\file2.txt (10 bytes)

Thus the server merely needs to retrieve the File List and directly access any of the files therein via the mapped network share to perform a file paste operation.

If direct access to the local file system of the peer endpoint is not possible, then file copy and paste using direct access MUST NOT be attempted. To enforce this condition, any attempt to request a File List with a Format Data Request PDU (section [2.2.5.1](#Section_b4c6e5c827dd4d07b15fd7e40302a870)) MUST fail and the resulting Format Data Response PDU (section [2.2.5.2](#Section_28c193b84cec413ea07b9235e5e15f6b)) MUST contain the CB\_RESPONSE\_FAIL (0x0002) flag.

### Timers

None.

### Initialization

The static virtual channel MUST be established, using the parameters specified in section [2.1](#Section_a373b2b937374c5fa296bc91a2f53344), before protocol operation can commence.

### Higher-Layer Triggered Events

This section contains details about the higher-layer triggered events.

#### Local Clipboard Update

When the local system clipboard is updated, the client or the server associated with the clipboard MUST send the [Format List PDU (section 3.1.5.2.1)](#Section_84bc5151fe6544c2a696216c4fd25c9c) to ensure that the formats available on the remote clipboards are kept in sync.

#### Local Paste Operation

When a local application requests data from the clipboard, and that data resides on the clipboard of a remote computer, the local computer MUST send the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) or the [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2), depending on the type of data requested.

### Processing Events and Sequencing Rules

#### Processing a Clipboard PDU

All clipboard PDUs are prefaced by the [Clipboard PDU Header](#Section_9e97ce45059743ddb1169f62a5b34d54) structure.

When processing a clipboard PDU, the **msgType** field in the header MUST first be examined to determine if the PDU is within the subset of expected messages. If the PDU is not expected, it SHOULD be ignored.

After determining that the PDU is in the correct sequence, the **dataLen** field MUST be examined to make sure that it is consistent with the amount of data read from the "CLIPRDR" static virtual channel. If this is not the case, the connection SHOULD be dropped.

#### Copy Sequence

##### Sending a Format List PDU

The fields of the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) are specified in section 2.2.3.1.

To construct the Format List PDU, the sender MUST enumerate all of the formats that are currently available from the local system clipboard, and for each format:

* Obtain the format ID associated with the [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538).
* Determine if the Clipboard Format has a corresponding format name.

The Format List PDU MUST be populated with this data. The usage of the [Short Format Names](#Section_6a20969b779d452eab7c5c8ad6dc90b8) structure or [Long Format Names](#Section_704b2025336f424da81c42395c85808d) structure MUST be based on the capabilities specified by the [General Capability Set](#Section_7718c8c9798d4788bb7564afdc913869). If short format names in ASCII 8 format are being used, the CB\_ASCII\_NAMES flag MUST be set in the **msgFlags** field of the **clipHeader** field. Unicode names MUST always be used with the long format names.

##### Processing a Format List PDU

The fields of the [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) are specified in section 2.2.3.1.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the format types present in the PDU MUST be extracted, processed, and placed on the local system clipboard.

For each [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) listed in the Format List PDU, the recipient of the PDU MUST do the following:

* Store the mapping of the remote Clipboard Format ID to the local Clipboard Format ID in the [Clipboard Format ID Map](#Section_680788b32bd84e2b9806589aba7cf814). The remote format ID is specified in the **formatId** field of the [Short Format Name](#Section_07765daa65da4b0087eada32b155e12d) structure and the [Long Format Name](#Section_feee15ee787d4ca2bb41136cee50d3de) structure.
* Update the local system clipboard by registering the local Clipboard Format ID as an available format for transfer. The system clipboard MUST support delayed rendering (as specified in section [1.3.1.4](#Section_fa309d1b803444bfb927adfc753e69c1)) for this step to be possible.

If the PDU was processed successfully and the local system clipboard was updated with all the received Clipboard Formats, the recipient MUST send a [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2) indicating success, as specified in section [3.1.5.2.3](#Section_59c53bcde7324603b0a8e8694b444c16). If the PDU could not be processed, or the local clipboard could not be updated, a Format List Response PDU indicating failure MUST be sent, as specified in section 3.1.5.2.3.

##### Sending a Format List Response PDU

The fields of the [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2) are specified in section 2.2.3.2.

The Format List Response PDU is sent to indicate the success or failure of processing the [Format List PDU](#Section_ba1464da316b4869b1d29b9c6eb418f4), as specified in section [3.1.5.2.2](#Section_4e41486468804b0d8af71838ca000c2d). On success, the **msgFlags** field of the **clipHeader** field MUST contain the CB\_RESPONSE\_OK flag. On failure, it MUST contain the CB\_RESPONSE\_FAIL flag.

##### Processing a Format List Response PDU

The fields of the [Format List Response PDU](#Section_e3c52df2c770429e8cbb510ca55836b2) are specified in section 2.2.3.2.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the response code MUST be extracted from the **msgFlags** field of the **clipHeader** field.

If the response code indicates that the associated [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) was processed successfully, then the recipient MUST be prepared to receive and process a [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) or a [File Contents Request PDU](#Section_df87c178ab02471aacdebb921aa1af85) with a request for format or file data, respectively.

If the response code indicates that processing of the Format List PDU was unsuccessful, then the recipient MUST respond to any subsequent Format Data Request PDUs or File Contents Request PDUs by sending a Format Data Response or File Contents Response indicating failure (sections [3.1.5.4.3](#Section_dc95d66a9b5f4ec28878357c623fcf4c) and [3.1.5.4.7](#Section_5d3991e9440d4e4d8b8d391c324c4007)).

#### Locking and Unlocking Clipboard Data

##### Sending a Lock Clipboard Data PDU

The fields of the Lock Clipboard Data PDU are specified in section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc).

It is permissible to send the Lock Clipboard Data at any point in time after the clipboard capabilities and temporary directory have been exchanged in the Clipboard Initialization Sequence (section [1.3.2.1](#Section_a5cae3c9170c4154992d9ac8a149cc7e)) has completed. The sender MUST be the Local Clipboard Owner (section 1.3.2.1).

The **clipDataId** field MUST contain an unsigned integer value that will serve as an identifier to uniquely tag any File Stream data (section [1.3.1.1.5](#Section_6119fce572b84ec38871f0b785ededd3)) on the clipboard of the Shared Clipboard Owner (section 1.3.2.1).

##### Processing a Lock Clipboard Data PDU

The fields of the Lock Clipboard Data PDU are specified in section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc).

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid and there is File Stream data (section [1.3.1.1.5](#Section_6119fce572b84ec38871f0b785ededd3)) on the clipboard, then the File Stream data MUST be stored so that any subsequent File Contents Request PDU (section [2.2.5.3](#Section_cbc851d34e6845f4929226872a9209f2)) can be serviced, even if the data is no longer available on the clipboard. The File Stream data MUST be stored until an Unlock Clipboard Data PDU (section [2.2.4.2](#Section_2ae0ff2619b24eb8af0a4811a6a18906)) is received.

##### Sending an Unlock Clipboard Data PDU

The fields of the Unlock Clipboard Data PDU are specified in section [2.2.4.2](#Section_2ae0ff2619b24eb8af0a4811a6a18906).

It is permissible to send the Unlock Clipboard Data at any point in time after the Clipboard Initialization Sequence (section [1.3.2.1](#Section_a5cae3c9170c4154992d9ac8a149cc7e)) has completed. The **clipDataId** field MUST contain an ID that was previously sent in a Lock Clipboard Data PDU (section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc)), but has not been sent in an Unlock Clipboard Data PDU.

##### Processing a Unlock Clipboard Data PDU

The fields of the Unlock Clipboard Data PDU are specified in section [2.2.4.2](#Section_2ae0ff2619b24eb8af0a4811a6a18906).

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid then the File Stream data that was stored and associated with the ID in the **clipDataId** field (section [3.1.5.3.2](#Section_597dee8cea46474cb2de2a843932b10c)) MUST be released. If there is no File Stream data associated with the ID, then the PDU MUST be ignored.

#### Paste Sequence

##### Sending a Format Data Request PDU

The fields of the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) are specified in section 2.2.5.1.

The [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID of the clipboard data MUST be specified in the **requestedFormatId** field.

* If Palette data is being requested, the **requestedFormatId** field MUST be set to 9.
* If Metafile data is being requested, the **requestedFormatId** field MUST be set to 3.
* If a File List is being requested, the Clipboard Format ID Map MUST be used to map the local Clipboard Format ID for the "FileGroupDescriptorW" Clipboard Format name to the remote Clipboard Format ID. The **requestedFormatId** field MUST then be populated with that value.

For all other formats the Clipboard Format ID Map MUST be used to map the local Clipboard Format ID of the requested clipboard data to the equivalent value on the remote system and then the **requestedFormatId** field MUST be populated with that value.

##### Processing a Format Data Request PDU

The fields of the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) are specified in section 2.2.5.1.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the requested [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID MUST be extracted from the PDU and the clipboard data retrieved from the local clipboard. The retrieved clipboard data MUST then be encoded appropriately, depending on the type:

* Metafile data MUST be encoded using the [Packed Metafile Payload](#Section_051ca890f6f14a9aa86bb59827b698bc) structure.
* Palette data MUST be encoded using the [Packed Palette Payload](#Section_45ff6ca1795c44d28936633775ea0a3c) structure.
* File List data MUST be encoded using the [Packed File List](#Section_3570c2e4cdd744608a7e1a4595f5ebdc) structure.
* If the clipboard data is not a metafile, palette, or file list, it is left unchanged.

The clipboard data MUST then be sent to the remote computer by using a [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b), as specified in section [3.1.5.4.3](#Section_dc95d66a9b5f4ec28878357c623fcf4c).

##### Sending a Format Data Response PDU

The fields of the [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) are specified in section 2.2.5.2.

The Format Data Response PDU is sent in response to the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870). During the processing of the [Format Data Request PDU (section 3.1.5.4.2)](#Section_96840dc2bb534934b34dfaf893a39234), the requested format data is retrieved from the local clipboard.

If there is format data to send, it MUST be copied into the **requestedFormatData** field and the **clipHeader** field MUST contain the CB\_RESPONSE\_OK (0x0001) flag. If the requested format data could not be retrieved or the sender received an unsuccessful Format List Response PDU (section [3.1.5.2.4](#Section_1ac0265a779b481da95b809f28d957dd)), then the **clipHeader** field MUST contain the CB\_RESPONSE\_FAIL (0x0002) flag and the **requestedFormatData** field MUST contain no data (zero-length).

##### Processing a Format Data Response PDU

The fields of the [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) are specified in section 2.2.5.2.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the attached data MUST be extracted if the **msgFlags** indicate success:

* Metafile data MUST be decoded using the [Packed Metafile Payload](#Section_051ca890f6f14a9aa86bb59827b698bc) structure.
* Palette data MUST be decoded using the [Packed Palette Payload](#Section_45ff6ca1795c44d28936633775ea0a3c) structure.
* File List data MUST be decoded using the [Packed File List](#Section_3570c2e4cdd744608a7e1a4595f5ebdc) structure.
* If the data is not a metafile, palette, or file list, it does not need to be decoded.

The processed clipboard data MUST be returned to the system clipboard so that the paste operation can be completed.

##### Sending a File Contents Request PDU

The fields of the [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2) are specified in section 2.2.5.3.

Prior to requesting any file contents data, the sender of the File Contents Request PDU MUST determine the appropriate index (specified in the **lindex** field) to identify the file on the remote clipboard. This index can be obtained through a [File List](#Section_c40918d1ab3840b4ae3ab8758e148926), which is transferred via the [Format Data Request PDU](#Section_b4c6e5c827dd4d07b15fd7e40302a870) and the [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b).

Knowledge of the size of a file on the remote clipboard, identified by a particular index value, is a prerequisite to requesting the actual contents of the file by using the File Contents Request PDU. The size, in bytes, of a particular file can be obtained from the File List associated with the file, or the File Contents Request PDU can be used to request the size by setting the FILECONTENTS\_SIZE (0x00000001) flag on the **dwFlags** field and populating the PDU fields.

If the size of a file on the remote clipboard is known, the File Contents Request PDU MUST be used to obtain the file contents at a particular offset by setting the FILECONTENTS\_RANGE (0x00000002) flag on the **dwFlags** field and populating the PDU fields. The specified range MUST be within the bounds of the file size.

##### Processing a File Contents Request PDU

The fields of the [Format Data Response PDU](#Section_28c193b84cec413ea07b9235e5e15f6b) are specified in section 2.2.5.2.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the data requested for the file (specified by the **lindex** field) MUST be returned to the sender. If the **clipDataId** field is present, then the locked File Stream data associated with the ID MUST be used to service the request.

The recipient of the [File Contents Request PDU](#Section_cbc851d34e6845f4929226872a9209f2) MUST perform a lookup using the **lindex** field to find the file that is the target of the request. The lookup most likely involves accessing the [File List](#Section_c40918d1ab3840b4ae3ab8758e148926) with which the current transaction is associated and using it to obtain the file information and contents.

After the file information has been acquired, the size or contents MUST be sent to the Remote Desktop Protocol: Clipboard Virtual Channel Extension endpoint by using the [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85), and sent as specified in section [3.1.5.4.7](#Section_5d3991e9440d4e4d8b8d391c324c4007). If the request cannot be satisfied, a File Contents Response PDU that contains the CB\_RESPONSE\_FAIL (0x0002) flag MUST be sent; otherwise, the CB\_RESPONSE\_OK (0x0001) flag MUST be specified.

##### Sending a File Contents Response PDU

The fields of the [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85) are specified in section 2.2.5.4.

If there is response data to send, the data MUST be copied into the **requestedFileContentsData** field and the **clipHeader** field MUST contain the CB\_RESPONSE\_OK (0x0001) flag. If the requested file data could not be retrieved or the sender received an unsuccessful Format List Response PDU (section [3.1.5.2.4](#Section_1ac0265a779b481da95b809f28d957dd)), then the **clipHeader** field MUST contain the CB\_RESPONSE\_FAIL (0x0002) flag and the **requestedFileContentsData** field MUST contain no data (zero-length).

##### Processing a File Contents Response PDU

The fields of the [File Contents Response PDU](#Section_df87c178ab02471aacdebb921aa1af85) are specified in section 2.2.5.4.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid and contains file contents data, the bytes MUST be extracted from the PDU and returned to the clipboard to satisfy the paste operation. If the PDU contains the size of the file, it MUST be read from the **requestedFileContentsData** field as a 64-bit [**little-endian**](#gt_079478cb-f4c5-4ce5-b72b-2144da5d2ce7) unsigned integer.

### Timer Events

None.

### Other Local Events

None.

## Client Details

### Abstract Data Model

#### Server Capabilities

The Server Capabilities store contains capability data received from the server in the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3). The client MUST ensure that it does not violate any of the server capabilities when sending data.

If a Clipboard Capabilities PDU is not received from the server, it MUST be assumed that the server is using the default capability values as specified in section 2.2.2.1.

### Timers

None.

### Initialization

The static virtual channel MUST be established, using the parameters as specified in section [2.1](#Section_a373b2b937374c5fa296bc91a2f53344), before protocol operation can commence.

### Higher-Layer Triggered Events

None.

### Processing Events and Sequencing Rules

#### Initialization Sequence

##### Processing a Server Clipboard Capabilities PDU

The fields of the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) are as specified in section 2.2.2.1.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the capability data MUST be extracted and stored in the [Server Capabilities](#Section_ff0fbd69743b407a9959497bdafa3297) store.

##### Processing a Monitor Ready PDU

The fields of the [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69) are as specified in section 2.2.2.2.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the client SHOULD do the following:

1. Send a [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) (as specified in section [3.2.5.1.3](#Section_1f3e2433fed14ff8a757ad1196437ac9)) to the server if it received the capabilities from the server.
2. Send a [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e) (as specified in section [3.2.5.1.4](#Section_de92863f69e84f6681e8fcbddb20b6e0)) to the server if it is necessary to inform the server of a location on the local client file system that MUST be used to deposit files being copied to the client.

After possibly sending the Clipboard Capabilities PDU and Temporary Directory PDU, the client MUST send a [Format List PDU](#Section_14e60d52e0da4e199455e8643ff17673) to the server, as specified in section [3.1.5.2.1](#Section_84bc5151fe6544c2a696216c4fd25c9c). This ensures that the peer system clipboards are in sync.

##### Sending a Client Clipboard Capabilities PDU

The fields of the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) are as specified in section 2.2.2.1.

The client MUST initialize the **generalFlags** field of the General Capability Set (section [2.2.2.1.1.1](#Section_7718c8c9798d4788bb7564afdc913869)) and indicate whether it supports the following features:

* Long format names
* File copy and paste using stream-based operations
* The removal of path data from the description of files to copy and paste
* Locking and unlocking clipboard data

If the [Server Capabilities](#Section_ff0fbd69743b407a9959497bdafa3297) store indicates that the server does not support a particular feature, then the client SHOULD NOT indicate support for that feature.

##### Sending a Temporary Directory PDU

The fields of the [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e) are specified in section 2.2.2.3.

Prior to sending the Temporary Directory PDU, the client MUST ensure that the location specified is accessible to the server. If this location is inaccessible or becomes inaccessible at a later time, all server-to-client file copies using direct file (section [3.1.1.3](#Section_d66378b9575b4239aa52c32ea67d97e5)) access MUST fail.

### Timer Events

None.

### Other Local Events

None.

## Server Details

### Abstract Data Model

#### Client Capabilities

The Client Capabilities store contains capability data received from the client in the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3). The server MUST ensure that it does not violate any of the client capabilities when sending data.

If a Clipboard Capabilities PDU is not received from the client, it MUST be assumed that the client is using the default capability values as specified in section 2.2.2.1.

#### Client Temporary Directory

The Client Temporary Directory store holds the path to a location on the client file system that MUST be used to deposit files being copied to the client. This information is received when processing the [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e), as specified in section [3.3.5.1.4](#Section_60012e81366e4cf5af4d768de1a8bc7b). If the Temporary Directory PDU is not received from the client, the server MUST NOT copy files to the client using direct file access techniques (section [3.1.1.3](#Section_d66378b9575b4239aa52c32ea67d97e5)).

### Timers

None.

### Initialization

The static virtual channel MUST be established, using the parameters as specified in section [2.1](#Section_a373b2b937374c5fa296bc91a2f53344), before protocol operation can commence.

### Higher-Layer Triggered Events

None.

### Processing Events and Sequencing Rules

#### Initialization Sequence

##### Sending a Server Clipboard Capabilities PDU

The fields of the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) are as specified in section 2.2.2.1.

The server MUST initialize the **generalFlags** field of the General Capability Set (section [2.2.2.1.1.1](#Section_7718c8c9798d4788bb7564afdc913869)) and indicate whether it supports the following features:

* Long format names
* File copy and paste using stream-based operations
* The removal of path data from the description of files to copy and paste
* Locking and unlocking clipboard data

After sending the Clipboard Capabilities PDU, the server MUST send the [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69) to the client, as specified in section [3.3.5.1.2](#Section_a9089520044c4390a5ac8a7f8fa6d2a1).

##### Sending a Monitor Ready PDU

The fields of the [Monitor Ready PDU](#Section_04d53575ba9e482887c268d88e034b69) are specified in section 2.2.2.2.

After sending the Monitor Ready PDU, the server MUST be prepared to start processing clipboard updates contained in [Format List PDUs](#Section_14e60d52e0da4e199455e8643ff17673), which it receives from the client, as specified in section [3.1.5.2.2](#Section_4e41486468804b0d8af71838ca000c2d). The server MUST still be prepared to receive and process the client [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) (as specified in section [3.3.5.1.3](#Section_a75719fd796e476f9c80d3be6b7bfe0f)) and [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e), as specified in section [3.3.5.1.4](#Section_60012e81366e4cf5af4d768de1a8bc7b).

##### Processing a Client Clipboard Capabilities PDU

The fields of the [Clipboard Capabilities PDU](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) are specified in section 2.2.2.1.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the capability data MUST be extracted and stored in the [Client Capabilities](#Section_5a36a35177024fa3bd97c133e4fce0d5) store as specified in 3.3.1.1.

##### Processing a Temporary Directory PDU

The fields of the [Temporary Directory PDU](#Section_7a0a0433d65a4d39a5b3931ca889633e) are specified in section 2.2.2.3.

The **clipHeader** field MUST be processed as specified in section [3.1.5.1](#Section_ed6118444eb34b29ac8c2576151b4d0c). If the PDU is valid, the temporary directory path MUST be extracted and stored in the [Client Temporary Directory store (section 3.3.1.2)](#Section_b052b5835ea2471c9771ecfd7d192865).

### Timer Events

None.

### Other Local Events

None.

# Protocol Examples

## Annotated Initialization Sequence

The following is an annotated dump of an [Initialization Sequence (section 1.3.2.1)](#Section_a5cae3c9170c4154992d9ac8a149cc7e).

### Server Clipboard Capabilities PDU

The following is an annotated dump of a server-to-client [Clipboard Capabilities PDU (section 2.2.2.1)](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3).

1. 00000000 07 00 00 00 10 00 00 00 01 00 00 00 01 00 0c 00 ................
2. 00000010 02 00 00 00 0e 00 00 00 ........
3. 07 00 -> CLIPRDR\_HEADER::msgType = CB\_CLIP\_CAPS (7)
4. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
5. 10 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0x10 = 16 bytes
6. 01 00 -> CLIPRDR\_CAPS::cCapabilitiesSets = 1
7. 00 00 -> CLIPRDR\_CAPS::pad1
8. 01 00 -> CLIPRDR\_CAPS\_SET::capabilitySetType = CB\_CAPSTYPE\_GENERAL (1)
9. 0c 00 -> CLIPRDR\_CAPS\_SET::lengthCapability = 0x0c = 12 bytes
10. 02 00 00 00 -> CLIPRDR\_GENERAL\_CAPABILITY::version = CB\_CAPS\_VERSION\_2 (2)
11. 0e 00 00 00 -> CLIPRDR\_GENERAL\_CAPABILITY::capabilityFlags = 0x0000000e
12. 0x0e
13. = 0x02 |
14. 0x04 |
15. 0x08
16. = CB\_USE\_LONG\_FORMAT\_NAMES |
17. CB\_STREAM\_FILECLIP\_ENABLED |
18. CB\_FILECLIP\_NO\_FILE\_PATHS

### Server Monitor Ready PDU

The following is an annotated dump of a [Monitor Ready PDU (section 2.2.2.2)](#Section_04d53575ba9e482887c268d88e034b69).

1. 00000000 01 00 00 00 00 00 00 00 ........
2. 01 00 -> CLIPRDR\_HEADER::msgType = CB\_MONITOR\_READY (1)
3. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
4. 00 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0 bytes

### Client Clipboard Capabilities PDU

The following is an annotated dump of a client-to-server [Clipboard Capabilities PDU (section 2.2.2.1)](#Section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3).

1. 00000000 07 00 00 00 10 00 00 00 01 00 00 00 01 00 0c 00 ................
2. 00000010 02 00 00 00 0e 00 00 00 ........
3. 07 00 -> CLIPRDR\_HEADER::msgType = CB\_CLIP\_CAPS (7)
4. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
5. 10 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0x10 = 16 bytes
6. 01 00 -> CLIPRDR\_CAPS::cCapabilitiesSets = 1
7. 00 00 -> CLIPRDR\_CAPS::pad1
8. 01 00 -> CLIPRDR\_CAPS\_SET::capabilitySetType = CB\_CAPSTYPE\_GENERAL (1)
9. 0c 00 -> CLIPRDR\_CAPS\_SET::lengthCapability = 0x0c = 12 bytes
10. 02 00 00 00 -> CLIPRDR\_GENERAL\_CAPABILITY::version = CB\_CAPS\_VERSION\_2 (2)
11. 0e 00 00 00 -> CLIPRDR\_GENERAL\_CAPABILITY::capabilityFlags = 0x0000000e
12. 0x0e
13. = 0x02 |
14. 0x04 |
15. 0x08
16. = CB\_USE\_LONG\_FORMAT\_NAMES |
17. CB\_STREAM\_FILECLIP\_ENABLED |
18. CB\_FILECLIP\_NO\_FILE\_PATHS

### Client Temporary Directory PDU

The following is an annotated dump of a [Temporary Directory PDU (section 2.2.2.3)](#Section_7a0a0433d65a4d39a5b3931ca889633e).

1. 00000000 06 00 00 00 08 02 00 00 43 00 3a 00 5c 00 44 00 ........C.:.\.D.
2. 00000010 4f 00 43 00 55 00 4d 00 45 00 7e 00 31 00 5c 00 O.C.U.M.E.~.1.\.
3. 00000020 45 00 4c 00 54 00 4f 00 4e 00 53 00 7e 00 31 00 E.L.T.O.N.S.~.1.
4. 00000030 2e 00 4e 00 54 00 44 00 5c 00 4c 00 4f 00 43 00 ..N.T.D.\.L.O.C.
5. 00000040 41 00 4c 00 53 00 7e 00 31 00 5c 00 54 00 65 00 A.L.S.~.1.\.T.e.
6. 00000050 6d 00 70 00 5c 00 63 00 64 00 65 00 70 00 6f 00 m.p.\.c.d.e.p.o.
7. 00000060 74 00 73 00 6c 00 68 00 72 00 64 00 70 00 5f 00 t.s.l.h.r.d.p.\_.
8. 00000070 31 00 5c 00 5f 00 54 00 53 00 41 00 42 00 44 00 1.\.\_.T.S.A.B.D.
9. 00000080 2e 00 74 00 6d 00 70 00 00 00 00 00 00 00 00 00 ..t.m.p.........
10. 00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
11. 000000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
12. 000000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
13. 000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
14. 000000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
15. 000000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
16. 000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
17. 00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
18. 00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
19. 00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
20. 00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
21. 00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
22. 00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
23. 00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
24. 00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
25. 00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
26. 00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
27. 000001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
28. 000001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
29. 000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
30. 000001d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
31. 000001e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
32. 000001f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
33. 00000200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
34. 06 00 -> CLIPRDR\_HEADER::msgType = CB\_TEMP\_DIRECTORY (6)
35. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
36. 08 02 00 00 -> CLIPRDR\_HEADER::dataLen = 0x208 = 520 bytes
37. 43 00 3a 00 5c 00 44 00 4f 00 43 00 55 00 4d 00
38. 45 00 7e 00 31 00 5c 00 45 00 4c 00 54 00 4f 00
39. 4e 00 53 00 7e 00 31 00 2e 00 4e 00 54 00 44 00
40. 5c 00 4c 00 4f 00 43 00 41 00 4c 00 53 00 7e 00
41. 31 00 5c 00 54 00 65 00 6d 00 70 00 5c 00 63 00
42. 64 00 65 00 70 00 6f 00 74 00 73 00 6c 00 68 00
43. 72 00 64 00 70 00 5f 00 31 00 5c 00 5f 00 54 00
44. 53 00 41 00 42 00 44 00 2e 00 74 00 6d 00 70 00
45. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
46. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
47. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
48. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
49. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
50. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
51. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
52. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
53. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
54. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
55. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
56. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
57. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
58. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
59. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
60. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
61. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
62. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
63. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
64. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
65. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
66. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
67. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
68. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
69. 00 00 00 00 00 00 00 00 -> CLIPRDR\_TEMP\_DIRECTORY::wszTempDir =
70. "c:\docume~1\eltons~1.ntd\locals~1\Temp\cdepotslhrdp\_1\\_TSABD.tmp"

### Format List PDU

The following is an annotated dump of a [Format List PDU (section 2.2.3.1)](#Section_14e60d52e0da4e199455e8643ff17673).

1. 00000000 02 00 00 00 24 00 00 00 04 c0 00 00 4e 00 61 00 ....$.......N.a.
2. 00000010 74 00 69 00 76 00 65 00 00 00 03 00 00 00 00 00 t.i.v.e.........
3. 00000020 08 00 00 00 00 00 11 00 00 00 00 00 ............
4. 02 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST (2)
5. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
6. 24 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0x24 = 36 bytes
7. 04 c0 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc004
8. 4e 00 61 00 74 00 69 00 76 00 65 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "Native"
9. 03 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0x03 = 3
10. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
11. 08 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0x08 = 8
12. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
13. 11 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0x11 = 17
14. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""

### Format List Response PDU

The following is an annotated dump of a [Format List Response PDU (section 2.2.3.2)](#Section_e3c52df2c770429e8cbb510ca55836b2).

1. 00000000 03 00 01 00 00 00 00 00 ........
2. 03 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST\_RESPONSE (3)
3. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
4. 00 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0 bytes

## Annotated Copy Sequence

The following is an annotated dump of a [Copy Sequence (section 1.3.2.2.1)](#Section_887747ab9bad490f8ddb68c6365e58dd).

### Format List PDU

The following is an annotated dump of a [Format List PDU (section 2.2.3.1)](#Section_14e60d52e0da4e199455e8643ff17673).

1. 00000000 02 00 00 00 e0 00 00 00 8a c0 00 00 52 00 69 00 ............R.i.
2. 00000010 63 00 68 00 20 00 54 00 65 00 78 00 74 00 20 00 c.h. .T.e.x.t. .
3. 00000020 46 00 6f 00 72 00 6d 00 61 00 74 00 00 00 45 c1 F.o.r.m.a.t...E.
4. 00000030 00 00 52 00 69 00 63 00 68 00 20 00 54 00 65 00 ..R.i.c.h. .T.e.
5. 00000040 78 00 74 00 20 00 46 00 6f 00 72 00 6d 00 61 00 x.t. .F.o.r.m.a.
6. 00000050 74 00 20 00 57 00 69 00 74 00 68 00 6f 00 75 00 t. .W.i.t.h.o.u.
7. 00000060 74 00 20 00 4f 00 62 00 6a 00 65 00 63 00 74 00 t. .O.b.j.e.c.t.
8. 00000070 73 00 00 00 43 c1 00 00 52 00 54 00 46 00 20 00 s...C...R.T.F. .
9. 00000080 41 00 73 00 20 00 54 00 65 00 78 00 74 00 00 00 A.s. .T.e.x.t...
10. 00000090 01 00 00 00 00 00 0d 00 00 00 00 00 04 c0 00 00 ................
11. 000000a0 4e 00 61 00 74 00 69 00 76 00 65 00 00 00 0e c0 N.a.t.i.v.e.....
12. 000000b0 00 00 4f 00 62 00 6a 00 65 00 63 00 74 00 20 00 ..O.b.j.e.c.t. .
13. 000000c0 44 00 65 00 73 00 63 00 72 00 69 00 70 00 74 00 D.e.s.c.r.i.p.t.
14. 000000d0 6f 00 72 00 00 00 03 00 00 00 00 00 10 00 00 00 o.r.............
15. 000000e0 00 00 07 00 00 00 00 00 ........
16. 02 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST (2)
17. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
18. e0 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0xe0 = 224 bytes
19. 8a c0 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc08a = 49290
20. 52 00 69 00 63 00 68 00 20 00 54 00 65 00 78 00
21. 74 00 20 00 46 00 6f 00 72 00 6d 00 61 00 74 00
22. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "Rich Text Format"
23. 45 c1 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc145 = 49477
24. 52 00 69 00 63 00 68 00 20 00 54 00 65 00 78 00
25. 74 00 20 00 46 00 6f 00 72 00 6d 00 61 00 74 00
26. 20 00 57 00 69 00 74 00 68 00 6f 00 75 00 74 00
27. 20 00 4f 00 62 00 6a 00 65 00 63 00 74 00 73 00
28. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "Rich Text Format Without Objects"
29. 43 c1 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc143 = 49475
30. 52 00 54 00 46 00 20 00 41 00 73 00 20 00 54 00
31. 65 00 78 00 74 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "RTF As Text"
32. 01 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 1
33. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
34. 0d 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0x0d = 13
35. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
36. 04 c0 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc004 = 49156
37. 4e 00 61 00 74 00 69 00 76 00 65 00 00 00 -> "Native"
38. 0e c0 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc00e = 49166
39. 4f 00 62 00 6a 00 65 00 63 00 74 00 20 00 44 00
40. 65 00 73 00 63 00 72 00 69 00 70 00 74 00 6f 00
41. 72 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "Object Descriptor"
42. 03 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 3
43. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
44. 10 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 16
45. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""
46. 07 00 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 7
47. 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatName = ""

### Format List Response PDU

The following is an annotated dump of a [Format List Response PDU (section 2.2.3.2)](#Section_e3c52df2c770429e8cbb510ca55836b2).

1. 00000000 03 00 01 00 00 00 00 00 ........
2. 03 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST\_RESPONSE (3)
3. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
4. 00 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0 bytes

## Locking and Unlocking Clipboard Data

### Lock Clipboard Data PDU

The following is an annotated dump of a Lock Clipboard Data Request PDU (section [2.2.4.1](#Section_150bac72bc7f42e59e8ecb5a0ddc7dbc)).

1. 00000000 0a 00 00 00 04 00 00 00 08 00 00 00 ............
2. 0a 00 -> CLIPRDR\_HEADER::msgType = CB\_LOCK\_CLIPDATA (10)
3. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
4. 04 00 00 00 -> CLIPRDR\_HEADER::dataLen = 4 bytes
5. 08 00 00 00 -> CLIPRDR\_LOCK\_CLIPDATA::clipDataId = 0x08

### Unlock Clipboard Data PDU

The following is an annotated dump of an Unlock Clipboard Data Request PDU (section [2.2.4.2](#Section_2ae0ff2619b24eb8af0a4811a6a18906)).

1. 00000000 0b 00 00 00 04 00 00 00 08 00 00 00 ............
2. 0b 00 -> CLIPRDR\_HEADER::msgType = CB\_UNLOCK\_CLIPDATA (11)
3. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
4. 04 00 00 00 -> CLIPRDR\_HEADER::dataLen = 4 bytes
5. 08 00 00 00 -> CLIPRDR\_UNLOCK\_CLIPDATA::clipDataId = 0x08

## Annotated Paste Sequence

The following is an annotated dump of a [Paste Sequence (section 1.3.2.2.3)](#Section_30688d0996b646f8af18ea1998bb7987).

### Format Data Request PDU

The following is an annotated dump of a [Format Data Request PDU (section 2.2.5.1)](#Section_b4c6e5c827dd4d07b15fd7e40302a870).

1. 00000000 04 00 00 00 04 00 00 00 0d 00 00 00 ............
2. 04 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_REQUEST (4)
3. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
4. 04 00 00 00 -> CLIPRDR\_HEADER::dataLen = 4 bytes
5. 0d 00 00 00 -> CLIPRDR\_FORMAT\_DATA\_REQUEST::requestedFormatId = 0x0d

### Format Data Response PDU

The following is an annotated dump of a [Format Data Response PDU (section 2.2.5.2)](#Section_28c193b84cec413ea07b9235e5e15f6b).

1. 00000010 05 00 01 00 18 00 00 00 68 00 65 00 6c 00 6c 00 ........h.e.l.l.
2. 00000020 6f 00 20 00 77 00 6f 00 72 00 6c 00 64 00 00 00 o. .w.o.r.l.d...
3. 05 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_RESPONSE (5)
4. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
5. 18 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0x18 = 24 bytes
6. 68 00 65 00 6c 00 6c 00 6f 00 20 00 77 00 6f 00
7. 72 00 6c 00 64 00 00 00 -> CLIPRDR\_FORMAT\_DATA\_RESPONSE::requestedFormatData

### File Contents Request PDU

#### Requesting the Size of a File

The following is an annotated dump of a [File Contents Request PDU (section 2.2.5.3)](#Section_cbc851d34e6845f4929226872a9209f2).

1. 00000000 08 00 00 00 18 00 00 00 02 00 00 00 01 00 00 00 ................
2. 00000010 01 00 00 00 00 00 00 00 00 00 00 00 08 00 00 00 ................
3. 00000020 00 00 00 00 00 00 00 00 ........
4. 08 00 -> -> CLIPRDR\_HEADER::msgType = CB\_FILECONTENTS\_REQUEST (8)
5. 00 00 -> -> CLIPRDR\_HEADER::msgFlags = 0
6. 18 00 00 00 -> CLIPRDR\_HEADER::dataLen = 24 bytes
7. 02 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::streamId = 2
8. 01 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::lindex = 1
9. 01 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::dwFlags = 0x000000001 = FILECONTENTS\_SIZE
10. 00 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::nPositionLow = 0
11. 00 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::nPositionHigh = 0
12. 08 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::cbRequested = 8

#### Requesting the Contents of a File

The following is an annotated dump of a [File Contents Request PDU (section 2.2.5.3)](#Section_cbc851d34e6845f4929226872a9209f2).

1. 00000000 08 00 00 00 18 00 00 00 02 00 00 00 01 00 00 00 ................
2. 00000010 02 00 00 00 00 00 00 00 00 00 00 00 08 00 00 00 ................
3. 00000020 00 00 00 00 00 00 00 00 ........
4. 08 00 -> CLIPRDR\_HEADER::msgType = CB\_FILECONTENTS\_REQUEST (8)
5. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
6. 18 00 00 00 -> CLIPRDR\_HEADER::dataLen = 24 bytes
7. 02 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::streamId = 2
8. 01 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::lindex = 1
9. 02 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::dwFlags = 0x000000002 = FILECONTENTS\_RANGE
10. 00 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::nPositionLow = 0
11. 00 00 00 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::nPositionHigh = 0
12. 00 00 01 00 -> CLIPRDR\_FILECONTENTS\_REQUEST::cbRequested = 0x00010000 = 65536 bytes

### File Contents Response PDU

#### Sending the Size of a File

The following is an annotated dump of a [File Contents Response PDU (section 2.2.5.4)](#Section_df87c178ab02471aacdebb921aa1af85).

1. 00000000 09 00 01 00 0c 00 00 00 02 00 00 00 2c 00 00 00 ............,...
2. 00000010 00 00 00 00 ....
3. 09 00 -> CLIPRDR\_HEADER::msgType = CB\_FILECONTENTS\_RESPONSE (9)
4. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
5. 0c 00 00 00 -> CLIPRDR\_HEADER::dataLen = 12 bytes
6. 02 00 00 00 -> CLIPRDR\_FILECONTENTS\_RESPONSE::streamId = 2
7. 2c 00 00 00 00 00 00 00 -> CLIPRDR\_FILECONTENTS\_RESPONSE::requestedFileContentsData = 44 bytes

#### Sending the Contents of a File

The following is an annotated dump of a [File Contents Response PDU (section 2.2.5.4)](#Section_df87c178ab02471aacdebb921aa1af85).

1. 00000000 09 00 01 00 30 00 00 00 02 00 00 00 54 68 65 20 ....0.......The
2. 00000010 71 75 69 63 6b 20 62 72 6f 77 6e 20 66 6f 78 20 quick brown fox
3. 00000020 6a 75 6d 70 73 20 6f 76 65 72 20 74 68 65 20 6c jumps over the l
4. 00000030 61 7a 79 20 64 6f 67 2e azy dog.
5. 09 00 -> CLIPRDR\_HEADER::msgType = CB\_FILECONTENTS\_RESPONSE (9)
6. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
7. 30 00 00 00 -> CLIPRDR\_HEADER::dataLen = 48 bytes
8. 02 00 00 00 -> CLIPRDR\_FILECONTENTS\_RESPONSE::streamId = 2
9. 54 68 65 20 71 75 69 63 6b 20 62 72 6f 77 6e 20
10. 66 6f 78 20 6a 75 6d 70 73 20 6f 76 65 72 20 74
11. 68 65 20 6c 61 7a 79 20 64 6f 67 2e -> CLIPRDR\_FILECONTENTS\_RESPONSE::requestedFileContentsData

### Metafile Data Contained in a Format Data Response PDU

The following is an annotated dump of a [Format Data Response PDU (section 2.2.5.2)](#Section_28c193b84cec413ea07b9235e5e15f6b) that contains a Windows metafile ([[MS-WMF]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-WMF%5d.pdf) section 2) wrapped in a [Packed Metafile Payload (section 2.2.5.2.1)](#Section_051ca890f6f14a9aa86bb59827b698bc) structure.

1. 00000000 05 00 01 00 1a 0a 00 00 08 00 00 00 2c 02 00 00 ............,...
2. 00000010 a7 01 00 00 01 00 09 00 00 03 07 05 00 00 01 00 ................
3. 00000020 e1 02 00 00 00 00 04 00 00 00 03 01 08 00 05 00 ................
4. 00000030 00 00 0c 02 f0 ff 15 00 05 00 00 00 0b 02 00 00 ................
5. 00000040 00 00 05 02 00 00 f7 00 00 03 00 01 00 00 00 00 ................
6. 00000050 80 80 80 00 80 00 00 00 80 80 00 00 00 80 00 00 ................
7. 00000060 00 80 80 00 00 00 80 00 80 00 80 00 80 80 40 00 ..............@.
8. 00000070 00 40 40 00 00 80 ff 00 00 40 80 00 80 00 ff 00 .@@......@......
9. 00000080 80 40 00 00 ff ff ff 00 c0 c0 c0 00 ff 00 00 00 .@..............
10. 00000090 ff ff 00 00 00 ff 00 00 00 ff ff 00 00 00 ff 00 ................
11. 000000a0 ff 00 ff 00 ff ff 80 00 00 ff 80 00 80 ff ff 00 ................
12. 000000b0 80 80 ff 00 ff 00 80 00 ff 80 40 00 00 00 00 00 ..........@.....
13. 000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
14. 000000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
15. 000000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
16. 000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
17. 00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
18. 00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
19. 00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
20. 00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
21. 00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
22. 00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
23. 00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
24. 00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
25. 00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
26. 00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
27. 000001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
28. 000001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
29. 000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
30. 000001d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
31. 000001e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
32. 000001f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
33. 00000200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
34. 00000210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
35. 00000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
36. 00000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
37. 00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
38. 00000250 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
39. 00000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
40. 00000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
41. 00000280 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
42. 00000290 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
43. 000002a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
44. 000002b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
45. 000002c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
46. 000002d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
47. 000002e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
48. 000002f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
49. 00000300 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
50. 00000310 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
51. 00000320 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
52. 00000330 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
53. 00000340 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
54. 00000350 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
55. 00000360 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
56. 00000370 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
57. 00000380 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
58. 00000390 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
59. 000003a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
60. 000003b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
61. 000003c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62. 000003d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
63. 000003e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
64. 000003f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
65. 00000400 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
66. 00000410 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
67. 00000420 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
68. 00000430 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
69. 00000440 00 00 00 00 00 00 00 00 00 00 00 00 04 00 00 00 ................
70. 00000450 34 02 00 00 03 00 00 00 35 00 e1 02 00 00 41 0b 4.......5.....A.
71. 00000460 20 00 cc 00 10 00 15 00 00 00 00 00 f0 ff 15 00 ...............
72. 00000470 00 00 00 00 28 00 00 00 15 00 00 00 10 00 00 00 ....(...........
73. 00000480 01 00 08 00 00 00 00 00 80 01 00 00 00 00 00 00 ................
74. 00000490 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
75. 000004a0 80 80 80 00 00 00 80 00 00 80 80 00 00 80 00 00 ................
76. 000004b0 80 80 00 00 80 00 00 00 80 00 80 00 40 80 80 00 ............@...
77. 000004c0 40 40 00 00 ff 80 00 00 80 40 00 00 ff 00 80 00 @@.......@......
78. 000004d0 00 40 80 00 ff ff ff 00 c0 c0 c0 00 00 00 ff 00 .@..............
79. 000004e0 00 ff ff 00 00 ff 00 00 ff ff 00 00 ff 00 00 00 ................
80. 000004f0 ff 00 ff 00 80 ff ff 00 80 ff 00 00 ff ff 80 00 ................
81. 00000500 ff 80 80 00 80 00 ff 00 40 80 ff 00 00 00 00 00 ........@.......
82. 00000510 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
83. 00000520 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
84. 00000530 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
85. 00000540 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
86. 00000550 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
87. 00000560 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
88. 00000570 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
89. 00000580 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
90. 00000590 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
91. 000005a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
92. 000005b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
93. 000005c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
94. 000005d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
95. 000005e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
96. 000005f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
97. 00000600 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
98. 00000610 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
99. 00000620 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
100. 00000630 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
101. 00000640 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
102. 00000650 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
103. 00000660 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
104. 00000670 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
105. 00000680 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
106. 00000690 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
107. 000006a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
108. 000006b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
109. 000006c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
110. 000006d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
111. 000006e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
112. 000006f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
113. 00000700 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
114. 00000710 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
115. 00000720 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
116. 00000730 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
117. 00000740 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
118. 00000750 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
119. 00000760 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
120. 00000770 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
121. 00000780 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
122. 00000790 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
123. 000007a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
124. 000007b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
125. 000007c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
126. 000007d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
127. 000007e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
128. 000007f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
129. 00000800 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
130. 00000810 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
131. 00000820 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
132. 00000830 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
133. 00000840 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
134. 00000850 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
135. 00000860 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
136. 00000870 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
137. 00000880 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
138. 00000890 00 00 00 00 00 00 00 00 00 00 00 00 0e 0e 0e 0e ................
139. 000008a0 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
140. 000008b0 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
141. 000008c0 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
142. 000008d0 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
143. 000008e0 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
144. 000008f0 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
145. 00000900 00 00 00 00 00 00 00 00 00 00 00 00 00 0e 0e 0e ................
146. 00000910 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e ................
147. 00000920 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
148. 00000930 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e ................
149. 00000940 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e ................
150. 00000950 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
151. 00000960 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e ................
152. 00000970 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e ................
153. 00000980 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
154. 00000990 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e ................
155. 000009a0 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e ................
156. 000009b0 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
157. 000009c0 00 00 00 00 00 00 00 00 00 00 00 00 00 0e 0e 0e ................
158. 000009d0 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
159. 000009e0 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e ................
160. 000009f0 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
161. 00000a00 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e ................
162. 00000a10 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00 03 00 00 00 ................
163. 00000a20 00 00 ..
164. 05 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_RESPONSE (5)
165. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
166. 1a 0a 00 00 -> CLIPRDR\_HEADER::dataLen = 0xa1a = 2586 bytes
167. 08 00 00 00 -> CLIPRDR\_MFPICT::mappingMode = MM\_ANISOTROPIC (0x00000008)
168. 2c 02 00 00 -> CLIPRDR\_MFPICT::xExt = 0x22c = 556
169. a7 01 00 00 -> CLIPRDR\_MFPICT::yExt = 0x1a7 = 423
170. 01 00 09 00 00 03 07 05 00 00 01 00 e1 02 00 00
171. 00 00 04 00 00 00 03 01 08 00 05 00 00 00 0c 02
172. f0 ff 15 00 05 00 00 00 0b 02 00 00 00 00 05 02
173. 00 00 f7 00 00 03 00 01 00 00 00 00 80 80 80 00
174. 80 00 00 00 80 80 00 00 00 80 00 00 00 80 80 00
175. 00 00 80 00 80 00 80 00 80 80 40 00 00 40 40 00
176. 00 80 ff 00 00 40 80 00 80 00 ff 00 80 40 00 00
177. ff ff ff 00 c0 c0 c0 00 ff 00 00 00 ff ff 00 00
178. 00 ff 00 00 00 ff ff 00 00 00 ff 00 ff 00 ff 00
179. ff ff 80 00 00 ff 80 00 80 ff ff 00 80 80 ff 00
180. ff 00 80 00 ff 80 40 00 00 00 00 00 00 00 00 00
181. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
182. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
183. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
184. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
185. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
186. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
187. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
188. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
189. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
190. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
191. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
192. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
193. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
194. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
195. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
196. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
197. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
198. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
199. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
200. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
201. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
202. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
203. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
204. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
205. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
206. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
207. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
208. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
209. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
210. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
211. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
212. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
213. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
214. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
215. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
216. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
217. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
218. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
219. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
220. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
221. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
222. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
223. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
224. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
225. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
226. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
227. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
228. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
229. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
230. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
231. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
232. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
233. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
234. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
235. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
236. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
237. 00 00 00 00 00 00 00 00 04 00 00 00 34 02 00 00
238. 03 00 00 00 35 00 e1 02 00 00 41 0b 20 00 cc 00
239. 10 00 15 00 00 00 00 00 f0 ff 15 00 00 00 00 00
240. 28 00 00 00 15 00 00 00 10 00 00 00 01 00 08 00
241. 00 00 00 00 80 01 00 00 00 00 00 00 00 00 00 00
242. 00 00 00 00 00 00 00 00 00 00 00 00 80 80 80 00
243. 00 00 80 00 00 80 80 00 00 80 00 00 80 80 00 00
244. 80 00 00 00 80 00 80 00 40 80 80 00 40 40 00 00
245. ff 80 00 00 80 40 00 00 ff 00 80 00 00 40 80 00
246. ff ff ff 00 c0 c0 c0 00 00 00 ff 00 00 ff ff 00
247. 00 ff 00 00 ff ff 00 00 ff 00 00 00 ff 00 ff 00
248. 80 ff ff 00 80 ff 00 00 ff ff 80 00 ff 80 80 00
249. 80 00 ff 00 40 80 ff 00 00 00 00 00 00 00 00 00
250. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
251. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
252. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
253. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
254. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
255. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
256. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
257. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
258. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
259. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
260. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
261. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
262. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
263. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
264. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
265. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
266. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
267. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
268. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
269. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
270. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
271. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
272. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
273. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
274. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
275. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
276. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
277. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
278. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
279. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
280. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
281. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
282. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
283. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
284. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
285. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
286. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
287. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
288. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
289. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
290. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
291. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
292. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
293. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
294. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
295. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
296. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
297. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
298. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
299. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
300. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
301. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
302. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
303. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
304. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
305. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
306. 00 00 00 00 00 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e
307. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00
308. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
309. 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e
310. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00
311. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
312. 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 00 00 00 00
313. 00 00 00 00 00 00 00 00 00 0e 0e 0e 0e 00 00 00
314. 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
315. 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e
316. 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00
317. 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
318. 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e
319. 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00
320. 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
321. 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 00 0e 0e 0e
322. 0e 0e 0e 0e 0e 0e 0e 0e 00 0e 0e 0e 0e 00 00 00
323. 0e 0e 0e 0e 00 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
324. 00 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 00 00 00 00
325. 00 00 00 00 00 00 00 00 00 0e 0e 0e 0e 00 00 00
326. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
327. 0e 0e 0e 0e 0e 00 00 00 0e 0e 0e 0e 0e 0e 0e 0e
328. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 00 00 00
329. 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e 0e
330. 0e 0e 0e 0e 0e 00 00 00 03 00 00 00 00 00 -> CLIPRDR\_MFPICT::metaFileData

### Palette Data Contained in a Format Data Response PDU

The following is an annotated dump of a [Format Data Response PDU (section 2.2.5.2)](#Section_28c193b84cec413ea07b9235e5e15f6b) that contains a 216-color palette wrapped in a [Packed Palette Payload (section 2.2.5.2.2)](#Section_45ff6ca1795c44d28936633775ea0a3c) structure.

1. 00000000 05 00 01 00 60 03 00 00 00 00 00 00 33 00 00 00 ....`.......3...
2. 00000010 66 00 00 00 99 00 00 00 cc 00 00 00 ff 00 00 00 f...............
3. 00000020 00 33 00 00 33 33 00 00 66 33 00 00 99 33 00 00 .3..33..f3...3..
4. 00000030 cc 33 00 00 ff 33 00 00 00 66 00 00 33 66 00 00 .3...3...f..3f..
5. 00000040 66 66 00 00 99 66 00 00 cc 66 00 00 ff 66 00 00 ff...f...f...f..
6. 00000050 00 99 00 00 33 99 00 00 66 99 00 00 99 99 00 00 ....3...f.......
7. 00000060 cc 99 00 00 ff 99 00 00 00 cc 00 00 33 cc 00 00 ............3...
8. 00000070 66 cc 00 00 99 cc 00 00 cc cc 00 00 ff cc 00 00 f...............
9. 00000080 00 ff 00 00 33 ff 00 00 66 ff 00 00 99 ff 00 00 ....3...f.......
10. 00000090 cc ff 00 00 ff ff 00 00 00 00 33 00 33 00 33 00 ..........3.3.3.
11. 000000a0 66 00 33 00 99 00 33 00 cc 00 33 00 ff 00 33 00 f.3...3...3...3.
12. 000000b0 00 33 33 00 33 33 33 00 66 33 33 00 99 33 33 00 .33.333.f33..33.
13. 000000c0 cc 33 33 00 ff 33 33 00 00 66 33 00 33 66 33 00 .33..33..f3.3f3.
14. 000000d0 66 66 33 00 99 66 33 00 cc 66 33 00 ff 66 33 00 ff3..f3..f3..f3.
15. 000000e0 00 99 33 00 33 99 33 00 66 99 33 00 99 99 33 00 ..3.3.3.f.3...3.
16. 000000f0 cc 99 33 00 ff 99 33 00 00 cc 33 00 33 cc 33 00 ..3...3...3.3.3.
17. 00000100 66 cc 33 00 99 cc 33 00 cc cc 33 00 ff cc 33 00 f.3...3...3...3.
18. 00000110 00 ff 33 00 33 ff 33 00 66 ff 33 00 99 ff 33 00 ..3.3.3.f.3...3.
19. 00000120 cc ff 33 00 ff ff 33 00 00 00 66 00 33 00 66 00 ..3...3...f.3.f.
20. 00000130 66 00 66 00 99 00 66 00 cc 00 66 00 ff 00 66 00 f.f...f...f...f.
21. 00000140 00 33 66 00 33 33 66 00 66 33 66 00 99 33 66 00 .3f.33f.f3f..3f.
22. 00000150 cc 33 66 00 ff 33 66 00 00 66 66 00 33 66 66 00 .3f..3f..ff.3ff.
23. 00000160 66 66 66 00 99 66 66 00 cc 66 66 00 ff 66 66 00 fff..ff..ff..ff.
24. 00000170 00 99 66 00 33 99 66 00 66 99 66 00 99 99 66 00 ..f.3.f.f.f...f.
25. 00000180 cc 99 66 00 ff 99 66 00 00 cc 66 00 33 cc 66 00 ..f...f...f.3.f.
26. 00000190 66 cc 66 00 99 cc 66 00 cc cc 66 00 ff cc 66 00 f.f...f...f...f.
27. 000001a0 00 ff 66 00 33 ff 66 00 66 ff 66 00 99 ff 66 00 ..f.3.f.f.f...f.
28. 000001b0 cc ff 66 00 ff ff 66 00 00 00 99 00 33 00 99 00 ..f...f.....3...
29. 000001c0 66 00 99 00 99 00 99 00 cc 00 99 00 ff 00 99 00 f...............
30. 000001d0 00 33 99 00 33 33 99 00 66 33 99 00 99 33 99 00 .3..33..f3...3..
31. 000001e0 cc 33 99 00 ff 33 99 00 00 66 99 00 33 66 99 00 .3...3...f..3f..
32. 000001f0 66 66 99 00 99 66 99 00 cc 66 99 00 ff 66 99 00 ff...f...f...f..
33. 00000200 00 99 99 00 33 99 99 00 66 99 99 00 99 99 99 00 ....3...f.......
34. 00000210 cc 99 99 00 ff 99 99 00 00 cc 99 00 33 cc 99 00 ............3...
35. 00000220 66 cc 99 00 99 cc 99 00 cc cc 99 00 ff cc 99 00 f...............
36. 00000230 00 ff 99 00 33 ff 99 00 66 ff 99 00 99 ff 99 00 ....3...f.......
37. 00000240 cc ff 99 00 ff ff 99 00 00 00 cc 00 33 00 cc 00 ............3...
38. 00000250 66 00 cc 00 99 00 cc 00 cc 00 cc 00 ff 00 cc 00 f...............
39. 00000260 00 33 cc 00 33 33 cc 00 66 33 cc 00 99 33 cc 00 .3..33..f3...3..
40. 00000270 cc 33 cc 00 ff 33 cc 00 00 66 cc 00 33 66 cc 00 .3...3...f..3f..
41. 00000280 66 66 cc 00 99 66 cc 00 cc 66 cc 00 ff 66 cc 00 ff...f...f...f..
42. 00000290 00 99 cc 00 33 99 cc 00 66 99 cc 00 99 99 cc 00 ....3...f.......
43. 000002a0 cc 99 cc 00 ff 99 cc 00 00 cc cc 00 33 cc cc 00 ............3...
44. 000002b0 66 cc cc 00 99 cc cc 00 cc cc cc 00 ff cc cc 00 f...............
45. 000002c0 00 ff cc 00 33 ff cc 00 66 ff cc 00 99 ff cc 00 ....3...f.......
46. 000002d0 cc ff cc 00 ff ff cc 00 00 00 ff 00 33 00 ff 00 ............3...
47. 000002e0 66 00 ff 00 99 00 ff 00 cc 00 ff 00 ff 00 ff 00 f...............
48. 000002f0 00 33 ff 00 33 33 ff 00 66 33 ff 00 99 33 ff 00 .3..33..f3...3..
49. 00000300 cc 33 ff 00 ff 33 ff 00 00 66 ff 00 33 66 ff 00 .3...3...f..3f..
50. 00000310 66 66 ff 00 99 66 ff 00 cc 66 ff 00 ff 66 ff 00 ff...f...f...f..
51. 00000320 00 99 ff 00 33 99 ff 00 66 99 ff 00 99 99 ff 00 ....3...f.......
52. 00000330 cc 99 ff 00 ff 99 ff 00 00 cc ff 00 33 cc ff 00 ............3...
53. 00000340 66 cc ff 00 99 cc ff 00 cc cc ff 00 ff cc ff 00 f...............
54. 00000350 00 ff ff 00 33 ff ff 00 66 ff ff 00 99 ff ff 00 ....3...f.......
55. 00000360 cc ff ff 00 ff ff ff 00 ........
56. 05 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_RESPONSE (5)
57. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
58. 60 03 00 00 -> CLIPRDR\_HEADER::dataLen = 0x360 = 864 bytes
59. 00 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[0] = 0x00000000
60. PALETTEENTRY::red = 0x00
61. PALETTEENTRY::green = 0x00
62. PALETTEENTRY::blue = 0x00
63. PALETTEENTRY::extra = 0x00
64. 33 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[1] = 0x00000033
65. PALETTEENTRY::red = 0x33
66. PALETTEENTRY::green = 0x00
67. PALETTEENTRY::blue = 0x00
68. PALETTEENTRY::extra = 0x00
69. 66 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[2] = 0x00000066
70. PALETTEENTRY::red = 0x66
71. PALETTEENTRY::green = 0x00
72. PALETTEENTRY::blue = 0x00
73. PALETTEENTRY::extra = 0x00
74. 99 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[3] = 0x00000099
75. PALETTEENTRY::red = 0x99
76. PALETTEENTRY::green = 0x00
77. PALETTEENTRY::blue = 0x00
78. PALETTEENTRY::extra = 0x00
79. cc 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[4] = 0x000000cc
80. PALETTEENTRY::red = 0xcc
81. PALETTEENTRY::green = 0x00
82. PALETTEENTRY::blue = 0x00
83. PALETTEENTRY::extra = 0x00
84. ff 00 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[5] = 0x000000ff
85. PALETTEENTRY::red = 0xff
86. PALETTEENTRY::green = 0x00
87. PALETTEENTRY::blue = 0x00
88. PALETTEENTRY::extra = 0x00
89. 00 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[6] = 0x00003300
90. PALETTEENTRY::red = 0x00
91. PALETTEENTRY::green = 0x33
92. PALETTEENTRY::blue = 0x00
93. PALETTEENTRY::extra = 0x00
94. 33 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[7] = 0x00003333
95. PALETTEENTRY::red = 0x33
96. PALETTEENTRY::green = 0x33
97. PALETTEENTRY::blue = 0x00
98. PALETTEENTRY::extra = 0x00
99. 66 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[8] = 0x00003366
100. PALETTEENTRY::red = 0x66
101. PALETTEENTRY::green = 0x33
102. PALETTEENTRY::blue = 0x00
103. PALETTEENTRY::extra = 0x00
104. 99 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[9] = 0x00003399
105. PALETTEENTRY::red = 0x99
106. PALETTEENTRY::green = 0x33
107. PALETTEENTRY::blue = 0x00
108. PALETTEENTRY::extra = 0x00
109. cc 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[10] = 0x000033cc
110. PALETTEENTRY::red = 0xcc
111. PALETTEENTRY::green = 0x33
112. PALETTEENTRY::blue = 0x00
113. PALETTEENTRY::extra = 0x00
114. ff 33 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[11] = 0x000033ff
115. PALETTEENTRY::red = 0xff
116. PALETTEENTRY::green = 0x33
117. PALETTEENTRY::blue = 0x00
118. PALETTEENTRY::extra = 0x00
119. 00 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[12] = 0x00006600
120. PALETTEENTRY::red = 0x00
121. PALETTEENTRY::green = 0x66
122. PALETTEENTRY::blue = 0x00
123. PALETTEENTRY::extra = 0x00
124. 33 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[13] = 0x00006633
125. PALETTEENTRY::red = 0x33
126. PALETTEENTRY::green = 0x66
127. PALETTEENTRY::blue = 0x00
128. PALETTEENTRY::extra = 0x00
129. 66 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[14] = 0x00006666
130. PALETTEENTRY::red = 0x66
131. PALETTEENTRY::green = 0x66
132. PALETTEENTRY::blue = 0x00
133. PALETTEENTRY::extra = 0x00
134. 99 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[15] = 0x00006699
135. PALETTEENTRY::red = 0x99
136. PALETTEENTRY::green = 0x66
137. PALETTEENTRY::blue = 0x00
138. PALETTEENTRY::extra = 0x00
139. cc 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[16] = 0x000066cc
140. PALETTEENTRY::red = 0xcc
141. PALETTEENTRY::green = 0x66
142. PALETTEENTRY::blue = 0x00
143. PALETTEENTRY::extra = 0x00
144. ff 66 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[17] = 0x000066ff
145. PALETTEENTRY::red = 0xff
146. PALETTEENTRY::green = 0x66
147. PALETTEENTRY::blue = 0x00
148. PALETTEENTRY::extra = 0x00
149. 00 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[18] = 0x00009900
150. PALETTEENTRY::red = 0x00
151. PALETTEENTRY::green = 0x99
152. PALETTEENTRY::blue = 0x00
153. PALETTEENTRY::extra = 0x00
154. 33 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[19] = 0x00009933
155. PALETTEENTRY::red = 0x33
156. PALETTEENTRY::green = 0x99
157. PALETTEENTRY::blue = 0x00
158. PALETTEENTRY::extra = 0x00
159. 66 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[20] = 0x00009966
160. PALETTEENTRY::red = 0x66
161. PALETTEENTRY::green = 0x99
162. PALETTEENTRY::blue = 0x00
163. PALETTEENTRY::extra = 0x00
164. 99 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[21] = 0x00009999
165. PALETTEENTRY::red = 0x99
166. PALETTEENTRY::green = 0x99
167. PALETTEENTRY::blue = 0x00
168. PALETTEENTRY::extra = 0x00
169. cc 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[22] = 0x000099cc
170. PALETTEENTRY::red = 0xcc
171. PALETTEENTRY::green = 0x99
172. PALETTEENTRY::blue = 0x00
173. PALETTEENTRY::extra = 0x00
174. ff 99 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[23] = 0x000099ff
175. PALETTEENTRY::red = 0xff
176. PALETTEENTRY::green = 0x99
177. PALETTEENTRY::blue = 0x00
178. PALETTEENTRY::extra = 0x00
179. 00 cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[24] = 0x0000cc00
180. PALETTEENTRY::red = 0x00
181. PALETTEENTRY::green = 0xcc
182. PALETTEENTRY::blue = 0x00
183. PALETTEENTRY::extra = 0x00
184. 33 cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[25] = 0x0000cc33
185. PALETTEENTRY::red = 0x33
186. PALETTEENTRY::green = 0xcc
187. PALETTEENTRY::blue = 0x00
188. PALETTEENTRY::extra = 0x00
189. 66 cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[26] = 0x0000cc66
190. PALETTEENTRY::red = 0x66
191. PALETTEENTRY::green = 0xcc
192. PALETTEENTRY::blue = 0x00
193. PALETTEENTRY::extra = 0x00
194. 99 cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[27] = 0x0000cc99
195. PALETTEENTRY::red = 0x99
196. PALETTEENTRY::green = 0xcc
197. PALETTEENTRY::blue = 0x00
198. PALETTEENTRY::extra = 0x00
199. cc cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[28] = 0x0000cccc
200. PALETTEENTRY::red = 0xcc
201. PALETTEENTRY::green = 0xcc
202. PALETTEENTRY::blue = 0x00
203. PALETTEENTRY::extra = 0x00
204. ff cc 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[29] = 0x0000ccff
205. PALETTEENTRY::red = 0xff
206. PALETTEENTRY::green = 0xcc
207. PALETTEENTRY::blue = 0x00
208. PALETTEENTRY::extra = 0x00
209. 00 ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[30] = 0x0000ff00
210. PALETTEENTRY::red = 0x00
211. PALETTEENTRY::green = 0xff
212. PALETTEENTRY::blue = 0x00
213. PALETTEENTRY::extra = 0x00
214. 33 ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[31] = 0x0000ff33
215. PALETTEENTRY::red = 0x33
216. PALETTEENTRY::green = 0xff
217. PALETTEENTRY::blue = 0x00
218. PALETTEENTRY::extra = 0x00
219. 66 ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[32] = 0x0000ff66
220. PALETTEENTRY::red = 0x66
221. PALETTEENTRY::green = 0xff
222. PALETTEENTRY::blue = 0x00
223. PALETTEENTRY::extra = 0x00
224. 99 ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[33] = 0x0000ff99
225. PALETTEENTRY::red = 0x99
226. PALETTEENTRY::green = 0xff
227. PALETTEENTRY::blue = 0x00
228. PALETTEENTRY::extra = 0x00
229. cc ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[34] = 0x0000ffcc
230. PALETTEENTRY::red = 0xcc
231. PALETTEENTRY::green = 0xff
232. PALETTEENTRY::blue = 0x00
233. PALETTEENTRY::extra = 0x00
234. ff ff 00 00 -> CLIPRDR\_PALETTE::paletteEntriesData[35] = 0x0000ffff
235. PALETTEENTRY::red = 0xff
236. PALETTEENTRY::green = 0xff
237. PALETTEENTRY::blue = 0x00
238. PALETTEENTRY::extra = 0x00
239. 00 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[36] = 0x00330000
240. PALETTEENTRY::red = 0x00
241. PALETTEENTRY::green = 0x00
242. PALETTEENTRY::blue = 0x33
243. PALETTEENTRY::extra = 0x00
244. 33 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[37] = 0x00330033
245. PALETTEENTRY::red = 0x33
246. PALETTEENTRY::green = 0x00
247. PALETTEENTRY::blue = 0x33
248. PALETTEENTRY::extra = 0x00
249. 66 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[38] = 0x00330066
250. PALETTEENTRY::red = 0x66
251. PALETTEENTRY::green = 0x00
252. PALETTEENTRY::blue = 0x33
253. PALETTEENTRY::extra = 0x00
254. 99 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[39] = 0x00330099
255. PALETTEENTRY::red = 0x99
256. PALETTEENTRY::green = 0x00
257. PALETTEENTRY::blue = 0x33
258. PALETTEENTRY::extra = 0x00
259. cc 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[40] = 0x003300cc
260. PALETTEENTRY::red = 0xcc
261. PALETTEENTRY::green = 0x00
262. PALETTEENTRY::blue = 0x33
263. PALETTEENTRY::extra = 0x00
264. ff 00 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[41] = 0x003300ff
265. PALETTEENTRY::red = 0xff
266. PALETTEENTRY::green = 0x00
267. PALETTEENTRY::blue = 0x33
268. PALETTEENTRY::extra = 0x00
269. 00 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[42] = 0x00333300
270. PALETTEENTRY::red = 0x00
271. PALETTEENTRY::green = 0x33
272. PALETTEENTRY::blue = 0x33
273. PALETTEENTRY::extra = 0x00
274. 33 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[43] = 0x00333333
275. PALETTEENTRY::red = 0x33
276. PALETTEENTRY::green = 0x33
277. PALETTEENTRY::blue = 0x33
278. PALETTEENTRY::extra = 0x00
279. 66 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[44] = 0x00333366
280. PALETTEENTRY::red = 0x66
281. PALETTEENTRY::green = 0x33
282. PALETTEENTRY::blue = 0x33
283. PALETTEENTRY::extra = 0x00
284. 99 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[45] = 0x00333399
285. PALETTEENTRY::red = 0x99
286. PALETTEENTRY::green = 0x33
287. PALETTEENTRY::blue = 0x33
288. PALETTEENTRY::extra = 0x00
289. cc 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[46] = 0x003333cc
290. PALETTEENTRY::red = 0xcc
291. PALETTEENTRY::green = 0x33
292. PALETTEENTRY::blue = 0x33
293. PALETTEENTRY::extra = 0x00
294. ff 33 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[47] = 0x003333ff
295. PALETTEENTRY::red = 0xff
296. PALETTEENTRY::green = 0x33
297. PALETTEENTRY::blue = 0x33
298. PALETTEENTRY::extra = 0x00
299. 00 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[48] = 0x00336600
300. PALETTEENTRY::red = 0x00
301. PALETTEENTRY::green = 0x66
302. PALETTEENTRY::blue = 0x33
303. PALETTEENTRY::extra = 0x00
304. 33 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[49] = 0x00336633
305. PALETTEENTRY::red = 0x33
306. PALETTEENTRY::green = 0x66
307. PALETTEENTRY::blue = 0x33
308. PALETTEENTRY::extra = 0x00
309. 66 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[50] = 0x00336666
310. PALETTEENTRY::red = 0x66
311. PALETTEENTRY::green = 0x66
312. PALETTEENTRY::blue = 0x33
313. PALETTEENTRY::extra = 0x00
314. 99 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[51] = 0x00336699
315. PALETTEENTRY::red = 0x99
316. PALETTEENTRY::green = 0x66
317. PALETTEENTRY::blue = 0x33
318. PALETTEENTRY::extra = 0x00
319. cc 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[52] = 0x003366cc
320. PALETTEENTRY::red = 0xcc
321. PALETTEENTRY::green = 0x66
322. PALETTEENTRY::blue = 0x33
323. PALETTEENTRY::extra = 0x00
324. ff 66 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[53] = 0x003366ff
325. PALETTEENTRY::red = 0xff
326. PALETTEENTRY::green = 0x66
327. PALETTEENTRY::blue = 0x33
328. PALETTEENTRY::extra = 0x00
329. 00 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[54] = 0x00339900
330. PALETTEENTRY::red = 0x00
331. PALETTEENTRY::green = 0x99
332. PALETTEENTRY::blue = 0x33
333. PALETTEENTRY::extra = 0x00
334. 33 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[55] = 0x00339933
335. PALETTEENTRY::red = 0x33
336. PALETTEENTRY::green = 0x99
337. PALETTEENTRY::blue = 0x33
338. PALETTEENTRY::extra = 0x00
339. 66 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[56] = 0x00339966
340. PALETTEENTRY::red = 0x66
341. PALETTEENTRY::green = 0x99
342. PALETTEENTRY::blue = 0x33
343. PALETTEENTRY::extra = 0x00
344. 99 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[57] = 0x00339999
345. PALETTEENTRY::red = 0x99
346. PALETTEENTRY::green = 0x99
347. PALETTEENTRY::blue = 0x33
348. PALETTEENTRY::extra = 0x00
349. cc 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[58] = 0x003399cc
350. PALETTEENTRY::red = 0xcc
351. PALETTEENTRY::green = 0x99
352. PALETTEENTRY::blue = 0x33
353. PALETTEENTRY::extra = 0x00
354. ff 99 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[59] = 0x003399ff
355. PALETTEENTRY::red = 0xff
356. PALETTEENTRY::green = 0x99
357. PALETTEENTRY::blue = 0x33
358. PALETTEENTRY::extra = 0x00
359. 00 cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[60] = 0x0033cc00
360. PALETTEENTRY::red = 0x00
361. PALETTEENTRY::green = 0xcc
362. PALETTEENTRY::blue = 0x33
363. PALETTEENTRY::extra = 0x00
364. 33 cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[61] = 0x0033cc33
365. PALETTEENTRY::red = 0x33
366. PALETTEENTRY::green = 0xcc
367. PALETTEENTRY::blue = 0x33
368. PALETTEENTRY::extra = 0x00
369. 66 cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[62] = 0x0033cc66
370. PALETTEENTRY::red = 0x66
371. PALETTEENTRY::green = 0xcc
372. PALETTEENTRY::blue = 0x33
373. PALETTEENTRY::extra = 0x00
374. 99 cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[63] = 0x0033cc99
375. PALETTEENTRY::red = 0x99
376. PALETTEENTRY::green = 0xcc
377. PALETTEENTRY::blue = 0x33
378. PALETTEENTRY::extra = 0x00
379. cc cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[64] = 0x0033cccc
380. PALETTEENTRY::red = 0xcc
381. PALETTEENTRY::green = 0xcc
382. PALETTEENTRY::blue = 0x33
383. PALETTEENTRY::extra = 0x00
384. ff cc 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[65] = 0x0033ccff
385. PALETTEENTRY::red = 0xff
386. PALETTEENTRY::green = 0xcc
387. PALETTEENTRY::blue = 0x33
388. PALETTEENTRY::extra = 0x00
389. 00 ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[66] = 0x0033ff00
390. PALETTEENTRY::red = 0x00
391. PALETTEENTRY::green = 0xff
392. PALETTEENTRY::blue = 0x33
393. PALETTEENTRY::extra = 0x00
394. 33 ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[67] = 0x0033ff33
395. PALETTEENTRY::red = 0x33
396. PALETTEENTRY::green = 0xff
397. PALETTEENTRY::blue = 0x33
398. PALETTEENTRY::extra = 0x00
399. 66 ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[68] = 0x0033ff66
400. PALETTEENTRY::red = 0x66
401. PALETTEENTRY::green = 0xff
402. PALETTEENTRY::blue = 0x33
403. PALETTEENTRY::extra = 0x00
404. 99 ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[69] = 0x0033ff99
405. PALETTEENTRY::red = 0x99
406. PALETTEENTRY::green = 0xff
407. PALETTEENTRY::blue = 0x33
408. PALETTEENTRY::extra = 0x00
409. cc ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[70] = 0x0033ffcc
410. PALETTEENTRY::red = 0xcc
411. PALETTEENTRY::green = 0xff
412. PALETTEENTRY::blue = 0x33
413. PALETTEENTRY::extra = 0x00
414. ff ff 33 00 -> CLIPRDR\_PALETTE::paletteEntriesData[71] = 0x0033ffff
415. PALETTEENTRY::red = 0xff
416. PALETTEENTRY::green = 0xff
417. PALETTEENTRY::blue = 0x33
418. PALETTEENTRY::extra = 0x00
419. 00 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[72] = 0x00660000
420. PALETTEENTRY::red = 0x00
421. PALETTEENTRY::green = 0x00
422. PALETTEENTRY::blue = 0x66
423. PALETTEENTRY::extra = 0x00
424. 33 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[73] = 0x00660033
425. PALETTEENTRY::red = 0x33
426. PALETTEENTRY::green = 0x00
427. PALETTEENTRY::blue = 0x66
428. PALETTEENTRY::extra = 0x00
429. 66 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[74] = 0x00660066
430. PALETTEENTRY::red = 0x66
431. PALETTEENTRY::green = 0x00
432. PALETTEENTRY::blue = 0x66
433. PALETTEENTRY::extra = 0x00
434. 99 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[75] = 0x00660099
435. PALETTEENTRY::red = 0x99
436. PALETTEENTRY::green = 0x00
437. PALETTEENTRY::blue = 0x66
438. PALETTEENTRY::extra = 0x00
439. cc 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[76] = 0x006600cc
440. PALETTEENTRY::red = 0xcc
441. PALETTEENTRY::green = 0x00
442. PALETTEENTRY::blue = 0x66
443. PALETTEENTRY::extra = 0x00
444. ff 00 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[77] = 0x006600ff
445. PALETTEENTRY::red = 0xff
446. PALETTEENTRY::green = 0x00
447. PALETTEENTRY::blue = 0x66
448. PALETTEENTRY::extra = 0x00
449. 00 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[78] = 0x00663300
450. PALETTEENTRY::red = 0x00
451. PALETTEENTRY::green = 0x33
452. PALETTEENTRY::blue = 0x66
453. PALETTEENTRY::extra = 0x00
454. 33 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[79] = 0x00663333
455. PALETTEENTRY::red = 0x33
456. PALETTEENTRY::green = 0x33
457. PALETTEENTRY::blue = 0x66
458. PALETTEENTRY::extra = 0x00
459. 66 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[80] = 0x00663366
460. PALETTEENTRY::red = 0x66
461. PALETTEENTRY::green = 0x33
462. PALETTEENTRY::blue = 0x66
463. PALETTEENTRY::extra = 0x00
464. 99 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[81] = 0x00663399
465. PALETTEENTRY::red = 0x99
466. PALETTEENTRY::green = 0x33
467. PALETTEENTRY::blue = 0x66
468. PALETTEENTRY::extra = 0x00
469. cc 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[82] = 0x006633cc
470. PALETTEENTRY::red = 0xcc
471. PALETTEENTRY::green = 0x33
472. PALETTEENTRY::blue = 0x66
473. PALETTEENTRY::extra = 0x00
474. ff 33 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[83] = 0x006633ff
475. PALETTEENTRY::red = 0xff
476. PALETTEENTRY::green = 0x33
477. PALETTEENTRY::blue = 0x66
478. PALETTEENTRY::extra = 0x00
479. 00 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[84] = 0x00666600
480. PALETTEENTRY::red = 0x00
481. PALETTEENTRY::green = 0x66
482. PALETTEENTRY::blue = 0x66
483. PALETTEENTRY::extra = 0x00
484. 33 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[85] = 0x00666633
485. PALETTEENTRY::red = 0x33
486. PALETTEENTRY::green = 0x66
487. PALETTEENTRY::blue = 0x66
488. PALETTEENTRY::extra = 0x00
489. 66 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[86] = 0x00666666
490. PALETTEENTRY::red = 0x66
491. PALETTEENTRY::green = 0x66
492. PALETTEENTRY::blue = 0x66
493. PALETTEENTRY::extra = 0x00
494. 99 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[87] = 0x00666699
495. PALETTEENTRY::red = 0x99
496. PALETTEENTRY::green = 0x66
497. PALETTEENTRY::blue = 0x66
498. PALETTEENTRY::extra = 0x00
499. cc 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[88] = 0x006666cc
500. PALETTEENTRY::red = 0xcc
501. PALETTEENTRY::green = 0x66
502. PALETTEENTRY::blue = 0x66
503. PALETTEENTRY::extra = 0x00
504. ff 66 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[89] = 0x006666ff
505. PALETTEENTRY::red = 0xff
506. PALETTEENTRY::green = 0x66
507. PALETTEENTRY::blue = 0x66
508. PALETTEENTRY::extra = 0x00
509. 00 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[90] = 0x00669900
510. PALETTEENTRY::red = 0x00
511. PALETTEENTRY::green = 0x99
512. PALETTEENTRY::blue = 0x66
513. PALETTEENTRY::extra = 0x00
514. 33 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[91] = 0x00669933
515. PALETTEENTRY::red = 0x33
516. PALETTEENTRY::green = 0x99
517. PALETTEENTRY::blue = 0x66
518. PALETTEENTRY::extra = 0x00
519. 66 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[92] = 0x00669966
520. PALETTEENTRY::red = 0x66
521. PALETTEENTRY::green = 0x99
522. PALETTEENTRY::blue = 0x66
523. PALETTEENTRY::extra = 0x00
524. 99 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[93] = 0x00669999
525. PALETTEENTRY::red = 0x99
526. PALETTEENTRY::green = 0x99
527. PALETTEENTRY::blue = 0x66
528. PALETTEENTRY::extra = 0x00
529. cc 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[94] = 0x006699cc
530. PALETTEENTRY::red = 0xcc
531. PALETTEENTRY::green = 0x99
532. PALETTEENTRY::blue = 0x66
533. PALETTEENTRY::extra = 0x00
534. ff 99 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[95] = 0x006699ff
535. PALETTEENTRY::red = 0xff
536. PALETTEENTRY::green = 0x99
537. PALETTEENTRY::blue = 0x66
538. PALETTEENTRY::extra = 0x00
539. 00 cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[96] = 0x0066cc00
540. PALETTEENTRY::red = 0x00
541. PALETTEENTRY::green = 0xcc
542. PALETTEENTRY::blue = 0x66
543. PALETTEENTRY::extra = 0x00
544. 33 cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[97] = 0x0066cc33
545. PALETTEENTRY::red = 0x33
546. PALETTEENTRY::green = 0xcc
547. PALETTEENTRY::blue = 0x66
548. PALETTEENTRY::extra = 0x00
549. 66 cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[98] = 0x0066cc66
550. PALETTEENTRY::red = 0x66
551. PALETTEENTRY::green = 0xcc
552. PALETTEENTRY::blue = 0x66
553. PALETTEENTRY::extra = 0x00
554. 99 cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[99] = 0x0066cc99
555. PALETTEENTRY::red = 0x99
556. PALETTEENTRY::green = 0xcc
557. PALETTEENTRY::blue = 0x66
558. PALETTEENTRY::extra = 0x00
559. cc cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[100] = 0x0066cccc
560. PALETTEENTRY::red = 0xcc
561. PALETTEENTRY::green = 0xcc
562. PALETTEENTRY::blue = 0x66
563. PALETTEENTRY::extra = 0x00
564. ff cc 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[101] = 0x0066ccff
565. PALETTEENTRY::red = 0xff
566. PALETTEENTRY::green = 0xcc
567. PALETTEENTRY::blue = 0x66
568. PALETTEENTRY::extra = 0x00
569. 00 ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[102] = 0x0066ff00
570. PALETTEENTRY::red = 0x00
571. PALETTEENTRY::green = 0xff
572. PALETTEENTRY::blue = 0x66
573. PALETTEENTRY::extra = 0x00
574. 33 ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[103] = 0x0066ff33
575. PALETTEENTRY::red = 0x33
576. PALETTEENTRY::green = 0xff
577. PALETTEENTRY::blue = 0x66
578. PALETTEENTRY::extra = 0x00
579. 66 ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[104] = 0x0066ff66
580. PALETTEENTRY::red = 0x66
581. PALETTEENTRY::green = 0xff
582. PALETTEENTRY::blue = 0x66
583. PALETTEENTRY::extra = 0x00
584. 99 ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[105] = 0x0066ff99
585. PALETTEENTRY::red = 0x99
586. PALETTEENTRY::green = 0xff
587. PALETTEENTRY::blue = 0x66
588. PALETTEENTRY::extra = 0x00
589. cc ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[106] = 0x0066ffcc
590. PALETTEENTRY::red = 0xcc
591. PALETTEENTRY::green = 0xff
592. PALETTEENTRY::blue = 0x66
593. PALETTEENTRY::extra = 0x00
594. ff ff 66 00 -> CLIPRDR\_PALETTE::paletteEntriesData[107] = 0x0066ffff
595. PALETTEENTRY::red = 0xff
596. PALETTEENTRY::green = 0xff
597. PALETTEENTRY::blue = 0x66
598. PALETTEENTRY::extra = 0x00
599. 00 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[108] = 0x00990000
600. PALETTEENTRY::red = 0x00
601. PALETTEENTRY::green = 0x00
602. PALETTEENTRY::blue = 0x99
603. PALETTEENTRY::extra = 0x00
604. 33 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[109] = 0x00990033
605. PALETTEENTRY::red = 0x33
606. PALETTEENTRY::green = 0x00
607. PALETTEENTRY::blue = 0x99
608. PALETTEENTRY::extra = 0x00
609. 66 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[110] = 0x00990066
610. PALETTEENTRY::red = 0x66
611. PALETTEENTRY::green = 0x00
612. PALETTEENTRY::blue = 0x99
613. PALETTEENTRY::extra = 0x00
614. 99 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[111] = 0x00990099
615. PALETTEENTRY::red = 0x99
616. PALETTEENTRY::green = 0x00
617. PALETTEENTRY::blue = 0x99
618. PALETTEENTRY::extra = 0x00
619. cc 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[112] = 0x009900cc
620. PALETTEENTRY::red = 0xcc
621. PALETTEENTRY::green = 0x00
622. PALETTEENTRY::blue = 0x99
623. PALETTEENTRY::extra = 0x00
624. ff 00 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[113] = 0x009900ff
625. PALETTEENTRY::red = 0xff
626. PALETTEENTRY::green = 0x00
627. PALETTEENTRY::blue = 0x99
628. PALETTEENTRY::extra = 0x00
629. 00 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[114] = 0x00993300
630. PALETTEENTRY::red = 0x00
631. PALETTEENTRY::green = 0x33
632. PALETTEENTRY::blue = 0x99
633. PALETTEENTRY::extra = 0x00
634. 33 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[115] = 0x00993333
635. PALETTEENTRY::red = 0x33
636. PALETTEENTRY::green = 0x33
637. PALETTEENTRY::blue = 0x99
638. PALETTEENTRY::extra = 0x00
639. 66 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[116] = 0x00993366
640. PALETTEENTRY::red = 0x66
641. PALETTEENTRY::green = 0x33
642. PALETTEENTRY::blue = 0x99
643. PALETTEENTRY::extra = 0x00
644. 99 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[117] = 0x00993399
645. PALETTEENTRY::red = 0x99
646. PALETTEENTRY::green = 0x33
647. PALETTEENTRY::blue = 0x99
648. PALETTEENTRY::extra = 0x00
649. cc 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[118] = 0x009933cc
650. PALETTEENTRY::red = 0xcc
651. PALETTEENTRY::green = 0x33
652. PALETTEENTRY::blue = 0x99
653. PALETTEENTRY::extra = 0x00
654. ff 33 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[119] = 0x009933ff
655. PALETTEENTRY::red = 0xff
656. PALETTEENTRY::green = 0x33
657. PALETTEENTRY::blue = 0x99
658. PALETTEENTRY::extra = 0x00
659. 00 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[120] = 0x00996600
660. PALETTEENTRY::red = 0x00
661. PALETTEENTRY::green = 0x66
662. PALETTEENTRY::blue = 0x99
663. PALETTEENTRY::extra = 0x00
664. 33 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[121] = 0x00996633
665. PALETTEENTRY::red = 0x33
666. PALETTEENTRY::green = 0x66
667. PALETTEENTRY::blue = 0x99
668. PALETTEENTRY::extra = 0x00
669. 66 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[122] = 0x00996666
670. PALETTEENTRY::red = 0x66
671. PALETTEENTRY::green = 0x66
672. PALETTEENTRY::blue = 0x99
673. PALETTEENTRY::extra = 0x00
674. 99 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[123] = 0x00996699
675. PALETTEENTRY::red = 0x99
676. PALETTEENTRY::green = 0x66
677. PALETTEENTRY::blue = 0x99
678. PALETTEENTRY::extra = 0x00
679. cc 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[124] = 0x009966cc
680. PALETTEENTRY::red = 0xcc
681. PALETTEENTRY::green = 0x66
682. PALETTEENTRY::blue = 0x99
683. PALETTEENTRY::extra = 0x00
684. ff 66 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[125] = 0x009966ff
685. PALETTEENTRY::red = 0xff
686. PALETTEENTRY::green = 0x66
687. PALETTEENTRY::blue = 0x99
688. PALETTEENTRY::extra = 0x00
689. 00 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[126] = 0x00999900
690. PALETTEENTRY::red = 0x00
691. PALETTEENTRY::green = 0x99
692. PALETTEENTRY::blue = 0x99
693. PALETTEENTRY::extra = 0x00
694. 33 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[127] = 0x00999933
695. PALETTEENTRY::red = 0x33
696. PALETTEENTRY::green = 0x99
697. PALETTEENTRY::blue = 0x99
698. PALETTEENTRY::extra = 0x00
699. 66 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[128] = 0x00999966
700. PALETTEENTRY::red = 0x66
701. PALETTEENTRY::green = 0x99
702. PALETTEENTRY::blue = 0x99
703. PALETTEENTRY::extra = 0x00
704. 99 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[129] = 0x00999999
705. PALETTEENTRY::red = 0x99
706. PALETTEENTRY::green = 0x99
707. PALETTEENTRY::blue = 0x99
708. PALETTEENTRY::extra = 0x00
709. cc 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[130] = 0x009999cc
710. PALETTEENTRY::red = 0xcc
711. PALETTEENTRY::green = 0x99
712. PALETTEENTRY::blue = 0x99
713. PALETTEENTRY::extra = 0x00
714. ff 99 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[131] = 0x009999ff
715. PALETTEENTRY::red = 0xff
716. PALETTEENTRY::green = 0x99
717. PALETTEENTRY::blue = 0x99
718. PALETTEENTRY::extra = 0x00
719. 00 cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[132] = 0x0099cc00
720. PALETTEENTRY::red = 0x00
721. PALETTEENTRY::green = 0xcc
722. PALETTEENTRY::blue = 0x99
723. PALETTEENTRY::extra = 0x00
724. 33 cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[133] = 0x0099cc33
725. PALETTEENTRY::red = 0x33
726. PALETTEENTRY::green = 0xcc
727. PALETTEENTRY::blue = 0x99
728. PALETTEENTRY::extra = 0x00
729. 66 cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[134] = 0x0099cc66
730. PALETTEENTRY::red = 0x66
731. PALETTEENTRY::green = 0xcc
732. PALETTEENTRY::blue = 0x99
733. PALETTEENTRY::extra = 0x00
734. 99 cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[135] = 0x0099cc99
735. PALETTEENTRY::red = 0x99
736. PALETTEENTRY::green = 0xcc
737. PALETTEENTRY::blue = 0x99
738. PALETTEENTRY::extra = 0x00
739. cc cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[136] = 0x0099cccc
740. PALETTEENTRY::red = 0xcc
741. PALETTEENTRY::green = 0xcc
742. PALETTEENTRY::blue = 0x99
743. PALETTEENTRY::extra = 0x00
744. ff cc 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[137] = 0x0099ccff
745. PALETTEENTRY::red = 0xff
746. PALETTEENTRY::green = 0xcc
747. PALETTEENTRY::blue = 0x99
748. PALETTEENTRY::extra = 0x00
749. 00 ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[138] = 0x0099ff00
750. PALETTEENTRY::red = 0x00
751. PALETTEENTRY::green = 0xff
752. PALETTEENTRY::blue = 0x99
753. PALETTEENTRY::extra = 0x00
754. 33 ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[139] = 0x0099ff33
755. PALETTEENTRY::red = 0x33
756. PALETTEENTRY::green = 0xff
757. PALETTEENTRY::blue = 0x99
758. PALETTEENTRY::extra = 0x00
759. 66 ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[140] = 0x0099ff66
760. PALETTEENTRY::red = 0x66
761. PALETTEENTRY::green = 0xff
762. PALETTEENTRY::blue = 0x99
763. PALETTEENTRY::extra = 0x00
764. 99 ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[141] = 0x0099ff99
765. PALETTEENTRY::red = 0x99
766. PALETTEENTRY::green = 0xff
767. PALETTEENTRY::blue = 0x99
768. PALETTEENTRY::extra = 0x00
769. cc ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[142] = 0x0099ffcc
770. PALETTEENTRY::red = 0xcc
771. PALETTEENTRY::green = 0xff
772. PALETTEENTRY::blue = 0x99
773. PALETTEENTRY::extra = 0x00
774. ff ff 99 00 -> CLIPRDR\_PALETTE::paletteEntriesData[143] = 0x0099ffff
775. PALETTEENTRY::red = 0xff
776. PALETTEENTRY::green = 0xff
777. PALETTEENTRY::blue = 0x99
778. PALETTEENTRY::extra = 0x00
779. 00 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[144] = 0x00cc0000
780. PALETTEENTRY::red = 0x00
781. PALETTEENTRY::green = 0x00
782. PALETTEENTRY::blue = 0xcc
783. PALETTEENTRY::extra = 0x00
784. 33 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[145] = 0x00cc0033
785. PALETTEENTRY::red = 0x33
786. PALETTEENTRY::green = 0x00
787. PALETTEENTRY::blue = 0xcc
788. PALETTEENTRY::extra = 0x00
789. 66 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[146] = 0x00cc0066
790. PALETTEENTRY::red = 0x66
791. PALETTEENTRY::green = 0x00
792. PALETTEENTRY::blue = 0xcc
793. PALETTEENTRY::extra = 0x00
794. 99 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[147] = 0x00cc0099
795. PALETTEENTRY::red = 0x99
796. PALETTEENTRY::green = 0x00
797. PALETTEENTRY::blue = 0xcc
798. PALETTEENTRY::extra = 0x00
799. cc 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[148] = 0x00cc00cc
800. PALETTEENTRY::red = 0xcc
801. PALETTEENTRY::green = 0x00
802. PALETTEENTRY::blue = 0xcc
803. PALETTEENTRY::extra = 0x00
804. ff 00 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[149] = 0x00cc00ff
805. PALETTEENTRY::red = 0xff
806. PALETTEENTRY::green = 0x00
807. PALETTEENTRY::blue = 0xcc
808. PALETTEENTRY::extra = 0x00
809. 00 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[150] = 0x00cc3300
810. PALETTEENTRY::red = 0x00
811. PALETTEENTRY::green = 0x33
812. PALETTEENTRY::blue = 0xcc
813. PALETTEENTRY::extra = 0x00
814. 33 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[151] = 0x00cc3333
815. PALETTEENTRY::red = 0x33
816. PALETTEENTRY::green = 0x33
817. PALETTEENTRY::blue = 0xcc
818. PALETTEENTRY::extra = 0x00
819. 66 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[152] = 0x00cc3366
820. PALETTEENTRY::red = 0x66
821. PALETTEENTRY::green = 0x33
822. PALETTEENTRY::blue = 0xcc
823. PALETTEENTRY::extra = 0x00
824. 99 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[153] = 0x00cc3399
825. PALETTEENTRY::red = 0x99
826. PALETTEENTRY::green = 0x33
827. PALETTEENTRY::blue = 0xcc
828. PALETTEENTRY::extra = 0x00
829. cc 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[154] = 0x00cc33cc
830. PALETTEENTRY::red = 0xcc
831. PALETTEENTRY::green = 0x33
832. PALETTEENTRY::blue = 0xcc
833. PALETTEENTRY::extra = 0x00
834. ff 33 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[155] = 0x00cc33ff
835. PALETTEENTRY::red = 0xff
836. PALETTEENTRY::green = 0x33
837. PALETTEENTRY::blue = 0xcc
838. PALETTEENTRY::extra = 0x00
839. 00 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[156] = 0x00cc6600
840. PALETTEENTRY::red = 0x00
841. PALETTEENTRY::green = 0x66
842. PALETTEENTRY::blue = 0xcc
843. PALETTEENTRY::extra = 0x00
844. 33 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[157] = 0x00cc6633
845. PALETTEENTRY::red = 0x33
846. PALETTEENTRY::green = 0x66
847. PALETTEENTRY::blue = 0xcc
848. PALETTEENTRY::extra = 0x00
849. 66 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[158] = 0x00cc6666
850. PALETTEENTRY::red = 0x66
851. PALETTEENTRY::green = 0x66
852. PALETTEENTRY::blue = 0xcc
853. PALETTEENTRY::extra = 0x00
854. 99 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[159] = 0x00cc6699
855. PALETTEENTRY::red = 0x99
856. PALETTEENTRY::green = 0x66
857. PALETTEENTRY::blue = 0xcc
858. PALETTEENTRY::extra = 0x00
859. cc 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[160] = 0x00cc66cc
860. PALETTEENTRY::red = 0xcc
861. PALETTEENTRY::green = 0x66
862. PALETTEENTRY::blue = 0xcc
863. PALETTEENTRY::extra = 0x00
864. ff 66 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[161] = 0x00cc66ff
865. PALETTEENTRY::red = 0xff
866. PALETTEENTRY::green = 0x66
867. PALETTEENTRY::blue = 0xcc
868. PALETTEENTRY::extra = 0x00
869. 00 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[162] = 0x00cc9900
870. PALETTEENTRY::red = 0x00
871. PALETTEENTRY::green = 0x99
872. PALETTEENTRY::blue = 0xcc
873. PALETTEENTRY::extra = 0x00
874. 33 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[163] = 0x00cc9933
875. PALETTEENTRY::red = 0x33
876. PALETTEENTRY::green = 0x99
877. PALETTEENTRY::blue = 0xcc
878. PALETTEENTRY::extra = 0x00
879. 66 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[164] = 0x00cc9966
880. PALETTEENTRY::red = 0x66
881. PALETTEENTRY::green = 0x99
882. PALETTEENTRY::blue = 0xcc
883. PALETTEENTRY::extra = 0x00
884. 99 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[165] = 0x00cc9999
885. PALETTEENTRY::red = 0x99
886. PALETTEENTRY::green = 0x99
887. PALETTEENTRY::blue = 0xcc
888. PALETTEENTRY::extra = 0x00
889. cc 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[166] = 0x00cc99cc
890. PALETTEENTRY::red = 0xcc
891. PALETTEENTRY::green = 0x99
892. PALETTEENTRY::blue = 0xcc
893. PALETTEENTRY::extra = 0x00
894. ff 99 cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[167] = 0x00cc99ff
895. PALETTEENTRY::red = 0xff
896. PALETTEENTRY::green = 0x99
897. PALETTEENTRY::blue = 0xcc
898. PALETTEENTRY::extra = 0x00
899. 00 cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[168] = 0x00cccc00
900. PALETTEENTRY::red = 0x00
901. PALETTEENTRY::green = 0xcc
902. PALETTEENTRY::blue = 0xcc
903. PALETTEENTRY::extra = 0x00
904. 33 cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[169] = 0x00cccc33
905. PALETTEENTRY::red = 0x33
906. PALETTEENTRY::green = 0xcc
907. PALETTEENTRY::blue = 0xcc
908. PALETTEENTRY::extra = 0x00
909. 66 cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[170] = 0x00cccc66
910. PALETTEENTRY::red = 0x66
911. PALETTEENTRY::green = 0xcc
912. PALETTEENTRY::blue = 0xcc
913. PALETTEENTRY::extra = 0x00
914. 99 cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[171] = 0x00cccc99
915. PALETTEENTRY::red = 0x99
916. PALETTEENTRY::green = 0xcc
917. PALETTEENTRY::blue = 0xcc
918. PALETTEENTRY::extra = 0x00
919. cc cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[172] = 0x00cccccc
920. PALETTEENTRY::red = 0xcc
921. PALETTEENTRY::green = 0xcc
922. PALETTEENTRY::blue = 0xcc
923. PALETTEENTRY::extra = 0x00
924. ff cc cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[173] = 0x00ccccff
925. PALETTEENTRY::red = 0xff
926. PALETTEENTRY::green = 0xcc
927. PALETTEENTRY::blue = 0xcc
928. PALETTEENTRY::extra = 0x00
929. 00 ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[174] = 0x00ccff00
930. PALETTEENTRY::red = 0x00
931. PALETTEENTRY::green = 0xff
932. PALETTEENTRY::blue = 0xcc
933. PALETTEENTRY::extra = 0x00
934. 33 ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[175] = 0x00ccff33
935. PALETTEENTRY::red = 0x33
936. PALETTEENTRY::green = 0xff
937. PALETTEENTRY::blue = 0xcc
938. PALETTEENTRY::extra = 0x00
939. 66 ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[176] = 0x00ccff66
940. PALETTEENTRY::red = 0x66
941. PALETTEENTRY::green = 0xff
942. PALETTEENTRY::blue = 0xcc
943. PALETTEENTRY::extra = 0x00
944. 99 ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[177] = 0x00ccff99
945. PALETTEENTRY::red = 0x99
946. PALETTEENTRY::green = 0xff
947. PALETTEENTRY::blue = 0xcc
948. PALETTEENTRY::extra = 0x00
949. cc ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[178] = 0x00ccffcc
950. PALETTEENTRY::red = 0xcc
951. PALETTEENTRY::green = 0xff
952. PALETTEENTRY::blue = 0xcc
953. PALETTEENTRY::extra = 0x00
954. ff ff cc 00 -> CLIPRDR\_PALETTE::paletteEntriesData[179] = 0x00ccffff
955. PALETTEENTRY::red = 0xff
956. PALETTEENTRY::green = 0xff
957. PALETTEENTRY::blue = 0xcc
958. PALETTEENTRY::extra = 0x00
959. 00 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[180] = 0x00ff0000
960. PALETTEENTRY::red = 0x00
961. PALETTEENTRY::green = 0x00
962. PALETTEENTRY::blue = 0xff
963. PALETTEENTRY::extra = 0x00
964. 33 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[181] = 0x00ff0033
965. PALETTEENTRY::red = 0x33
966. PALETTEENTRY::green = 0x00
967. PALETTEENTRY::blue = 0xff
968. PALETTEENTRY::extra = 0x00
969. 66 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[182] = 0x00ff0066
970. PALETTEENTRY::red = 0x66
971. PALETTEENTRY::green = 0x00
972. PALETTEENTRY::blue = 0xff
973. PALETTEENTRY::extra = 0x00
974. 99 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[183] = 0x00ff0099
975. PALETTEENTRY::red = 0x99
976. PALETTEENTRY::green = 0x00
977. PALETTEENTRY::blue = 0xff
978. PALETTEENTRY::extra = 0x00
979. cc 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[184] = 0x00ff00cc
980. PALETTEENTRY::red = 0xcc
981. PALETTEENTRY::green = 0x00
982. PALETTEENTRY::blue = 0xff
983. PALETTEENTRY::extra = 0x00
984. ff 00 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[185] = 0x00ff00ff
985. PALETTEENTRY::red = 0xff
986. PALETTEENTRY::green = 0x00
987. PALETTEENTRY::blue = 0xff
988. PALETTEENTRY::extra = 0x00
989. 00 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[186] = 0x00ff3300
990. PALETTEENTRY::red = 0x00
991. PALETTEENTRY::green = 0x33
992. PALETTEENTRY::blue = 0xff
993. PALETTEENTRY::extra = 0x00
994. 33 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[187] = 0x00ff3333
995. PALETTEENTRY::red = 0x33
996. PALETTEENTRY::green = 0x33
997. PALETTEENTRY::blue = 0xff
998. PALETTEENTRY::extra = 0x00
999. 66 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[188] = 0x00ff3366
1000. PALETTEENTRY::red = 0x66
1001. PALETTEENTRY::green = 0x33
1002. PALETTEENTRY::blue = 0xff
1003. PALETTEENTRY::extra = 0x00
1004. 99 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[189] = 0x00ff3399
1005. PALETTEENTRY::red = 0x99
1006. PALETTEENTRY::green = 0x33
1007. PALETTEENTRY::blue = 0xff
1008. PALETTEENTRY::extra = 0x00
1009. cc 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[190] = 0x00ff33cc
1010. PALETTEENTRY::red = 0xcc
1011. PALETTEENTRY::green = 0x33
1012. PALETTEENTRY::blue = 0xff
1013. PALETTEENTRY::extra = 0x00
1014. ff 33 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[191] = 0x00ff33ff
1015. PALETTEENTRY::red = 0xff
1016. PALETTEENTRY::green = 0x33
1017. PALETTEENTRY::blue = 0xff
1018. PALETTEENTRY::extra = 0x00
1019. 00 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[192] = 0x00ff6600
1020. PALETTEENTRY::red = 0x00
1021. PALETTEENTRY::green = 0x66
1022. PALETTEENTRY::blue = 0xff
1023. PALETTEENTRY::extra = 0x00
1024. 33 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[193] = 0x00ff6633
1025. PALETTEENTRY::red = 0x33
1026. PALETTEENTRY::green = 0x66
1027. PALETTEENTRY::blue = 0xff
1028. PALETTEENTRY::extra = 0x00
1029. 66 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[194] = 0x00ff6666
1030. PALETTEENTRY::red = 0x66
1031. PALETTEENTRY::green = 0x66
1032. PALETTEENTRY::blue = 0xff
1033. PALETTEENTRY::extra = 0x00
1034. 99 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[195] = 0x00ff6699
1035. PALETTEENTRY::red = 0x99
1036. PALETTEENTRY::green = 0x66
1037. PALETTEENTRY::blue = 0xff
1038. PALETTEENTRY::extra = 0x00
1039. cc 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[196] = 0x00ff66cc
1040. PALETTEENTRY::red = 0xcc
1041. PALETTEENTRY::green = 0x66
1042. PALETTEENTRY::blue = 0xff
1043. PALETTEENTRY::extra = 0x00
1044. ff 66 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[197] = 0x00ff66ff
1045. PALETTEENTRY::red = 0xff
1046. PALETTEENTRY::green = 0x66
1047. PALETTEENTRY::blue = 0xff
1048. PALETTEENTRY::extra = 0x00
1049. 00 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[198] = 0x00ff9900
1050. PALETTEENTRY::red = 0x00
1051. PALETTEENTRY::green = 0x99
1052. PALETTEENTRY::blue = 0xff
1053. PALETTEENTRY::extra = 0x00
1054. 33 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[199] = 0x00ff9933
1055. PALETTEENTRY::red = 0x33
1056. PALETTEENTRY::green = 0x99
1057. PALETTEENTRY::blue = 0xff
1058. PALETTEENTRY::extra = 0x00
1059. 66 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[200] = 0x00ff9966
1060. PALETTEENTRY::red = 0x66
1061. PALETTEENTRY::green = 0x99
1062. PALETTEENTRY::blue = 0xff
1063. PALETTEENTRY::extra = 0x00
1064. 99 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[201] = 0x00ff9999
1065. PALETTEENTRY::red = 0x99
1066. PALETTEENTRY::green = 0x99
1067. PALETTEENTRY::blue = 0xff
1068. PALETTEENTRY::extra = 0x00
1069. cc 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[202] = 0x00ff99cc
1070. PALETTEENTRY::red = 0xcc
1071. PALETTEENTRY::green = 0x99
1072. PALETTEENTRY::blue = 0xff
1073. PALETTEENTRY::extra = 0x00
1074. ff 99 ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[203] = 0x00ff99ff
1075. PALETTEENTRY::red = 0xff
1076. PALETTEENTRY::green = 0x99
1077. PALETTEENTRY::blue = 0xff
1078. PALETTEENTRY::extra = 0x00
1079. 00 cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[204] = 0x00ffcc00
1080. PALETTEENTRY::red = 0x00
1081. PALETTEENTRY::green = 0xcc
1082. PALETTEENTRY::blue = 0xff
1083. PALETTEENTRY::extra = 0x00
1084. 33 cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[205] = 0x00ffcc33
1085. PALETTEENTRY::red = 0x33
1086. PALETTEENTRY::green = 0xcc
1087. PALETTEENTRY::blue = 0xff
1088. PALETTEENTRY::extra = 0x00
1089. 66 cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[206] = 0x00ffcc66
1090. PALETTEENTRY::red = 0x66
1091. PALETTEENTRY::green = 0xcc
1092. PALETTEENTRY::blue = 0xff
1093. PALETTEENTRY::extra = 0x00
1094. 99 cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[207] = 0x00ffcc99
1095. PALETTEENTRY::red = 0x99
1096. PALETTEENTRY::green = 0xcc
1097. PALETTEENTRY::blue = 0xff
1098. PALETTEENTRY::extra = 0x00
1099. cc cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[208] = 0x00ffcccc
1100. PALETTEENTRY::red = 0xcc
1101. PALETTEENTRY::green = 0xcc
1102. PALETTEENTRY::blue = 0xff
1103. PALETTEENTRY::extra = 0x00
1104. ff cc ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[209] = 0x00ffccff
1105. PALETTEENTRY::red = 0xff
1106. PALETTEENTRY::green = 0xcc
1107. PALETTEENTRY::blue = 0xff
1108. PALETTEENTRY::extra = 0x00
1109. 00 ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[210] = 0x00ffff00
1110. PALETTEENTRY::red = 0x00
1111. PALETTEENTRY::green = 0xff
1112. PALETTEENTRY::blue = 0xff
1113. PALETTEENTRY::extra = 0x00
1114. 33 ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[211] = 0x00ffff33
1115. PALETTEENTRY::red = 0x33
1116. PALETTEENTRY::green = 0xff
1117. PALETTEENTRY::blue = 0xff
1118. PALETTEENTRY::extra = 0x00
1119. 66 ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[212] = 0x00ffff66
1120. PALETTEENTRY::red = 0x66
1121. PALETTEENTRY::green = 0xff
1122. PALETTEENTRY::blue = 0xff
1123. PALETTEENTRY::extra = 0x00
1124. 99 ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[213] = 0x00ffff99
1125. PALETTEENTRY::red = 0x99
1126. PALETTEENTRY::green = 0xff
1127. PALETTEENTRY::blue = 0xff
1128. PALETTEENTRY::extra = 0x00
1129. cc ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[214] = 0x00ffffcc
1130. PALETTEENTRY::red = 0xcc
1131. PALETTEENTRY::green = 0xff
1132. PALETTEENTRY::blue = 0xff
1133. PALETTEENTRY::extra = 0x00
1134. ff ff ff 00 -> CLIPRDR\_PALETTE::paletteEntriesData[215] = 0x00ffffff
1135. PALETTEENTRY::red = 0xff
1136. PALETTEENTRY::green = 0xff
1137. PALETTEENTRY::blue = 0xff
1138. PALETTEENTRY::extra = 0x00

## Retrieving a File List

The following is an annotated dump that shows the sequence of messages involved in obtaining a File List.

### Format List PDU

The following is an annotated dump of a [Format List PDU (section 2.2.3.1)](#Section_14e60d52e0da4e199455e8643ff17673). This format list advertises the fact that File List data is available from the peer (the FileGroupDescriptorW format is a File List).

1. 00000000 02 00 00 00 2e 00 00 00 79 c0 00 00 46 00 69 00 ....z...y...F.i.
2. 00000010 6c 00 65 00 47 00 72 00 6f 00 75 00 70 00 44 00 l.e.G.r.o.u.p.D.
3. 00000020 65 00 73 00 63 00 72 00 69 00 70 00 74 00 6f 00 e.s.c.r.i.p.t.o.
4. 00000030 72 00 57 00 00 00 r.W...
5. 02 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST (2)
6. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
7. 7a 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0x2e = 46 bytes
8. 79 c0 00 00 -> CLIPRDR\_LONG\_FORMAT\_NAME::formatId = 0xc079 = 49273
9. 46 00 69 00 6c 00 65 00 47 00 72 00 6f 00 75 00
10. 70 00 44 00 65 00 73 00 63 00 72 00 69 00 70 00
11. 74 00 6f 00 72 00 57 00 00 00 ->
12. CLIPRDR\_LONG\_FORMAT\_NAME::formatName = "FileGroupDescriptorW"

### Format List Response PDU

The following is an annotated dump of a [Format List Response PDU (section 2.2.3.2)](#Section_e3c52df2c770429e8cbb510ca55836b2).

1. 00000000 03 00 01 00 00 00 00 00 ........
2. 03 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_LIST\_RESPONSE (3)
3. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
4. 00 00 00 00 -> CLIPRDR\_HEADER::dataLen = 0 bytes

### Format Data Request PDU

The following is an annotated dump of a [Format Data Request PDU (section 2.2.5.1)](#Section_b4c6e5c827dd4d07b15fd7e40302a870). The format being requested is the File List that was advertised in section [4.5.1](#Section_adf7a15cfa5b499f90c88295f7d31c68) (the advertised ID in the Format List PDU was 49273).

1. 00000000 04 00 00 00 04 00 00 00 79 c0 00 00 ............
2. 04 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_REQUEST (4)
3. 00 00 -> CLIPRDR\_HEADER::msgFlags = 0
4. 04 00 00 00 -> CLIPRDR\_HEADER::dataLen = 4 bytes
5. 79 c0 00 00 -> CLIPRDR\_FORMAT\_DATA\_REQUEST::requestedFormatId = 0xc079 = 49273

### Format Data Response PDU

The following is an annotated dump of a [Format Data Response PDU (section 2.2.5.1)](#Section_b4c6e5c827dd4d07b15fd7e40302a870) sent in response to the File List format request in section [4.5.2](#Section_b24c2372c1d14e99bd1a66d0a8b46267).

1. 00000000 05 00 01 00 a4 04 00 00 02 00 00 00 64 40 00 00 ............d@..
2. 00000010 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
3. 00000020 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
4. 00000030 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
5. 00000040 00 00 00 00 08 5d 30 2c f3 55 ca 01 00 00 00 00 .....]0,.U......
6. 00000050 2c 00 00 00 46 00 69 00 6c 00 65 00 31 00 2e 00 ,...F.i.l.e.1...
7. 00000060 74 00 78 00 74 00 00 00 00 00 00 00 00 00 00 00 t.x.t...........
8. 00000070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
9. 00000080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
10. 00000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
11. 000000a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
12. 000000b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
13. 000000c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
14. 000000d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
15. 000000e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
16. 000000f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
17. 00000100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
18. 00000110 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
19. 00000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
20. 00000130 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
21. 00000140 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
22. 00000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
23. 00000160 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
24. 00000170 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
25. 00000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
26. 00000190 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
27. 000001a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
28. 000001b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
29. 000001c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
30. 000001d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
31. 000001e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
32. 000001f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
33. 00000200 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
34. 00000210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
35. 00000220 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
36. 00000230 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
37. 00000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
38. 00000250 00 00 00 00 00 00 00 00 00 00 00 00 64 40 00 00 ............d@..
39. 00000260 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
40. 00000270 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
41. 00000280 20 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ...............
42. 00000290 00 00 00 00 08 5d 30 2c f3 55 ca 01 00 00 00 00 .....]0,.U......
43. 000002a0 0a 00 00 00 46 00 69 00 6c 00 65 00 32 00 2e 00 ,...F.i.l.e.2...
44. 000002b0 74 00 78 00 74 00 00 00 00 00 00 00 00 00 00 00 t.x.t...........
45. 000002c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
46. 000002d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
47. 000002e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
48. 000002f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
49. 00000300 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
50. 00000310 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
51. 00000320 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
52. 00000330 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
53. 00000340 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
54. 00000350 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
55. 00000360 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
56. 00000370 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
57. 00000380 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
58. 00000390 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
59. 000003a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
60. 000003b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
61. 000003c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
62. 000003d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
63. 000003e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
64. 000003f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
65. 00000400 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
66. 00000410 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
67. 00000420 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
68. 00000430 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
69. 00000440 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
70. 00000450 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
71. 00000460 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
72. 00000470 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
73. 00000480 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
74. 00000490 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
75. 000004a0 00 00 00 00 00 00 00 00 00 00 00 00 ............
76. 05 00 -> CLIPRDR\_HEADER::msgType = CB\_FORMAT\_DATA\_RESPONSE (5)
77. 01 00 -> CLIPRDR\_HEADER::msgFlags = 0x0001 = CB\_RESPONSE\_OK
78. a4 04 00 00 -> CLIPRDR\_HEADER::dataLen = 0x4a4 = 1188 bytes
79. 02 00 00 00 -> CLIPRDR\_FILELIST::cItems = 2
80. 64 40 00 00 -> CLIPRDR\_FILEDESCRIPTOR::flags = 0x00004064
81. 0x00004064
82. = 0x00000004 |
83. 0x00000040 |
84. 0x00000020 |
85. 0x00004000
86. = FD\_ATTRIBUTES |
87. FD\_FILESIZE |
88. FD\_WRITESTIME |
89. FD\_PROGRESSUI
90. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
91. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::reserved1
92. 20 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileAttributes = 0x00000020 = FILE\_ATTRIBUTE\_ARCHIVE
93. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::reserved2
94. 08 5d 30 2c f3 55 ca 01 -> CLIPRDR\_FILEDESCRIPTOR::lastWriteTime
95. 0x01ca55f32c305d09
96. = 129,010,042,240,261,385 100-nanoseconds intervals since 1 January 1601
97. = 25 October 2009, 21:17
98. 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileSizeHigh = 0 bytes
99. 2c 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileSizeLow = 44 bytes
100. 46 00 69 00 6c 00 65 00 31 00 2e 00 74 00 78 00
101. 74 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
102. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
103. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
104. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
105. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
106. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
107. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
108. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
109. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
110. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
111. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
112. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
113. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
114. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
115. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
116. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
117. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
118. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
119. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
120. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
121. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
122. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
123. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
124. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
125. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
126. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
127. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
128. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
129. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
130. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
131. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ->
132. CLIPRDR\_FILEDESCRIPTOR::cFileName = "File1.txt"
133. 64 40 00 00 -> CLIPRDR\_FILEDESCRIPTOR::flags = 0x00004064
134. 0x00004064
135. = 0x00000004 |
136. 0x00000040 |
137. 0x00000020 |
138. 0x00004000
139. = FD\_ATTRIBUTES |
140. FD\_FILESIZE |
141. FD\_WRITESTIME |
142. FD\_PROGRESSUI
143. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
144. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::reserved1
145. 20 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileAttributes = 0x00000020 = FILE\_ATTRIBUTE\_ARCHIVE
146. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::reserved2
147. 08 5d 30 2c f3 55 ca 01 -> CLIPRDR\_FILEDESCRIPTOR::lastWriteTime
148. 0x01ca55f32c305d09
149. = 129,010,042,240,261,385 100-nanoseconds intervals since 1 January 1601
150. = 25 October 2009, 21:17
151. 00 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileSizeHigh = 0 bytes
152. 0a 00 00 00 -> CLIPRDR\_FILEDESCRIPTOR::fileSizeLow = 10 bytes
153. 46 00 69 00 6c 00 65 00 32 00 2e 00 74 00 78 00
154. 74 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
155. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
156. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
157. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
158. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
159. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
160. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
161. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
162. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
163. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
164. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
165. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
166. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
167. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
168. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
169. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
170. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
171. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
172. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
173. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
174. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
175. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
176. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
177. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
178. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
179. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
180. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
181. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
182. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
183. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
184. 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ->
185. CLIPRDR\_FILEDESCRIPTOR::cFileName = "File2.txt"

# Security

The following sections specify security considerations for implementers of the Remote Desktop Protocol: Clipboard Virtual Channel Extension.

## Security Considerations for Implementers

There are no security considerations for protocol messages because all static virtual channel traffic is secured by the underlying core Remote Desktop Protocol. An overview of the implemented security-related mechanisms is as specified in [[MS-RDPBCGR]](file:///E:\Target\Windows\Published\Books\MS-RDPECLIP\%5bMS-RDPBCGR%5d.pdf) section 5.

## Index of Security Parameters

None.

# Appendix A: Product Behavior

The information in this specification is applicable to the following Microsoft products or supplemental software. References to product versions include released service packs.

* Windows 2000 Server operating system
* Windows XP operating system
* Windows Server 2003 operating system
* Windows Vista operating system
* Windows Server 2008 operating system
* Windows 7 operating system
* Windows Server 2008 R2 operating system
* Windows 8 operating system
* Windows Server 2012 operating system
* Windows 8.1 operating system
* Windows Server 2012 R2 operating system
* Windows 10 operating system
* Windows Server 2016 Technical Preview operating system

Exceptions, if any, are noted below. If a service pack or Quick Fix Engineering (QFE) number appears with the product version, behavior changed in that service pack or QFE. The new behavior also applies to subsequent service packs of the product unless otherwise specified. If a product edition appears with the product version, behavior is different in that product edition.

Unless otherwise specified, any statement of optional behavior in this specification that is prescribed using the terms SHOULD or SHOULD NOT implies product behavior in accordance with the SHOULD or SHOULD NOT prescription. Unless otherwise specified, the term MAY implies that the product does not follow the prescription.

[<1> Section 2.2.1](#Appendix_A_Target_1): The operating systems Windows Server 2003, Windows Vista, Windows Server 2008, Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012, Windows 8.1, and Windows Server 2012 R2 append four bytes to the end of clipboard PDUs. These four bytes are not included in the PDU size specified by the **dataLen** field and can be ignored.

[<2> Section 3.1.1.2](#Appendix_A_Target_2): On Windows-based systems, the file list is encapsulated in a "File Group Descriptor" (for more information, see [[MSDN-SHELLCLIP]](http://go.microsoft.com/fwlink/?LinkId=90131), "CFSTR\_FILEDESCRIPTOR") generic data format. This format consists of an array of File Descriptors, each of which describes a single file in a collection. The Format Name of the File Group Descriptor format is "FileGroupDescriptorW" if the descriptor is in Unicode format; otherwise, it is "FileGroupDescriptor". The [Clipboard Format](#Section_f293d8e478584f5fb00e8e67323d0538) ID is assigned dynamically and is not constant across computer systems.

# Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as New, Major, Minor, Editorial, or No change.

The revision class **New** means that a new document is being released.

The revision class **Major** means that the technical content in the document was significantly revised. Major changes affect protocol interoperability or implementation. Examples of major changes are:

* A document revision that incorporates changes to interoperability requirements or functionality.
* The removal of a document from the documentation set.

The revision class **Minor** means that the meaning of the technical content was clarified. Minor changes do not affect protocol interoperability or implementation. Examples of minor changes are updates to clarify ambiguity at the sentence, paragraph, or table level.

The revision class **Editorial** means that the formatting in the technical content was changed. Editorial changes apply to grammatical, formatting, and style issues.

The revision class **No change** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the technical content of the document is identical to the last released version.

Major and minor changes can be described further using the following change types:

* New content added.
* Content updated.
* Content removed.
* New product behavior note added.
* Product behavior note updated.
* Product behavior note removed.
* New protocol syntax added.
* Protocol syntax updated.
* Protocol syntax removed.
* New content added due to protocol revision.
* Content updated due to protocol revision.
* Content removed due to protocol revision.
* New protocol syntax added due to protocol revision.
* Protocol syntax updated due to protocol revision.
* Protocol syntax removed due to protocol revision.
* Obsolete document removed.

Editorial changes are always classified with the change type **Editorially updated**.

Some important terms used in the change type descriptions are defined as follows:

* **Protocol syntax** refers to data elements (such as packets, structures, enumerations, and methods) as well as interfaces.
* **Protocol revision** refers to changes made to a protocol that affect the bits that are sent over the wire.

The changes made to this document are listed in the following table. For more information, please contact [dochelp@microsoft.com](mailto:dochelp@microsoft.com).

| Section | Tracking number (if applicable) and description | Major change (Y or N) | Change type |
| --- | --- | --- | --- |
| [6](#Section_bfa8a3bfcb24448986507c152f84aea4) Appendix A: Product Behavior | Added Windows 10 to applicability list. | Y | Content update. |

# Index

**A**

Abstract data model

client ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.2.1](#section_6906d1a0247a4401acd342aee920fd0a) 37)

server ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.3.1](#section_3a0c0d21b58b4adab406cc301365a4a7) 39)

[Annotated copy sequence examples](#section_3626af4da635452f9bc5b8e1d82d6c09) 43

[Annotated initialization sequence examples](#section_705fafbfbccd4b6e82e7008940b04607) 41

[Annotated paste sequence examples](#section_639609e6f0544b709c912396dfff620b) 45

[Applicability](#section_5746b77a69054f288a4cfcc623484ffd) 15

**C**

[Capability negotiation](#section_0dd48ed7f46742998fc0b99bac121cf3) 15

[Change tracking](#section_43bf9e62c5694132a75c0fe9ebb87180) 79

Client

abstract data model ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.2.1](#section_6906d1a0247a4401acd342aee920fd0a) 37)

higher-layer triggered events ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.2.4](#section_c3a86723ee1947bbba66159eb94d1306) 37)

initialization ([section 3.1.3](#section_59d6fb55acac4cfd9082bd150118036a) 32, [section 3.2.3](#section_87e486718ae54141baf4f4f2251d3d76) 37)

local events ([section 3.1.7](#section_c08772f8256c4b8bb8f3b207eb7ef5bc) 37, [section 3.2.7](#section_16bab796b5a44ec1b964af25ba904bc2) 39)

message processing ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.2.5](#section_b78c2072fadb42ca8476b0defec522d1) 38)

[other local events](#section_16bab796b5a44ec1b964af25ba904bc2) 39

sequencing rules ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.2.5](#section_b78c2072fadb42ca8476b0defec522d1) 38)

timer events ([section 3.1.6](#section_70a24c52d1f34859bca71bb63ea1e0e2) 37, [section 3.2.6](#section_abbe41549f574692819840a555fc1d6b) 38)

timers ([section 3.1.2](#section_6d3269df77b3443c8b56055818087ab0) 32, [section 3.2.2](#section_142af46de65849caa8d71ace2ff38800) 37)

[Client Capabilities store](#section_5a36a35177024fa3bd97c133e4fce0d5) 39

[Client Clipboard Capabilities PDU example](#section_ac9ab35ba7054e168560c9353161778a) 41

[Client Temporary Directory PDU example](#section_0689d414305040f8b50b90e112c39eb4) 42

[Client Temporary Directory store](#section_b052b5835ea2471c9771ecfd7d192865) 39

[Clipboard basics](#section_dc14f83c6f0141ab8b46feb60fea1f2d) 8

[Clipboard format](#section_f293d8e478584f5fb00e8e67323d0538) 10

[Clipboard Format ID Map](#section_680788b32bd84e2b9806589aba7cf814) 31

[Clipboard PDU Header (CLIPRDR\_HEADER) message](#section_9e97ce45059743ddb1169f62a5b34d54) 17

[Clipboard Redirection Virtual Channel](#section_a85961f670f04f7f8d81c4f7f6549b7d) 11

[CLIPRDR\_CAPS](#section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) 18

[CLIPRDR\_CAPS packet](#section_e4a5b0c4103b48bc9cfdfbc0bd85bcc3) 18

[CLIPRDR\_CAPS\_SET](#section_c160dde9a3d845d79af75f5822b2e6a8) 18

[CLIPRDR\_CAPS\_SET packet](#section_c160dde9a3d845d79af75f5822b2e6a8) 18

[CLIPRDR\_FILECONTENTS\_REQUEST](#section_cbc851d34e6845f4929226872a9209f2) 29

[CLIPRDR\_FILECONTENTS\_REQUEST packet](#section_cbc851d34e6845f4929226872a9209f2) 29

[CLIPRDR\_FILECONTENTS\_RESPONSE](#section_df87c178ab02471aacdebb921aa1af85) 30

[CLIPRDR\_FILECONTENTS\_RESPONSE packet](#section_df87c178ab02471aacdebb921aa1af85) 30

[CLIPRDR\_FILEDESCRIPTOR packet](#section_a765d7842b394b889faa88f8666f9c35) 27

[CLIPRDR\_FILELIST packet](#section_3570c2e4cdd744608a7e1a4595f5ebdc) 26

[CLIPRDR\_FORMAT\_DATA\_REQUEST](#section_b4c6e5c827dd4d07b15fd7e40302a870) 24

[CLIPRDR\_FORMAT\_DATA\_REQUEST packet](#section_b4c6e5c827dd4d07b15fd7e40302a870) 24

[CLIPRDR\_FORMAT\_DATA\_RESPONSE](#section_28c193b84cec413ea07b9235e5e15f6b) 24

[CLIPRDR\_FORMAT\_DATA\_RESPONSE packet](#section_28c193b84cec413ea07b9235e5e15f6b) 24

[CLIPRDR\_FORMAT\_LIST](#section_14e60d52e0da4e199455e8643ff17673) 21

[CLIPRDR\_FORMAT\_LIST packet](#section_14e60d52e0da4e199455e8643ff17673) 21

[CLIPRDR\_GENERAL\_CAPABILITY](#section_7718c8c9798d4788bb7564afdc913869) 19

[CLIPRDR\_GENERAL\_CAPABILITY packet](#section_7718c8c9798d4788bb7564afdc913869) 19

[CLIPRDR\_HEADER](#section_9e97ce45059743ddb1169f62a5b34d54) 17

[CLIPRDR\_HEADER packet](#section_9e97ce45059743ddb1169f62a5b34d54) 17

[CLIPRDR\_LOCK\_CLIPDATA packet](#section_150bac72bc7f42e59e8ecb5a0ddc7dbc) 23

[CLIPRDR\_LONG\_FORMAT\_NAME](#section_feee15ee787d4ca2bb41136cee50d3de) 22

[CLIPRDR\_LONG\_FORMAT\_NAME packet](#section_feee15ee787d4ca2bb41136cee50d3de) 22

[CLIPRDR\_LONG\_FORMAT\_NAMES](#section_704b2025336f424da81c42395c85808d) 22

[CLIPRDR\_LONG\_FORMAT\_NAMES packet](#section_704b2025336f424da81c42395c85808d) 22

[CLIPRDR\_MFPICT](#section_051ca890f6f14a9aa86bb59827b698bc) 25

[CLIPRDR\_MFPICT packet](#section_051ca890f6f14a9aa86bb59827b698bc) 25

[CLIPRDR\_MONITOR\_READY](#section_04d53575ba9e482887c268d88e034b69) 20

[CLIPRDR\_MONITOR\_READY packet](#section_04d53575ba9e482887c268d88e034b69) 20

[CLIPRDR\_PALETTE](#section_45ff6ca1795c44d28936633775ea0a3c) 26

[CLIPRDR\_PALETTE packet](#section_45ff6ca1795c44d28936633775ea0a3c) 26

[CLIPRDR\_SHORT\_FORMAT\_NAME](#section_07765daa65da4b0087eada32b155e12d) 21

[CLIPRDR\_SHORT\_FORMAT\_NAME packet](#section_07765daa65da4b0087eada32b155e12d) 21

[CLIPRDR\_SHORT\_FORMAT\_NAMES](#section_6a20969b779d452eab7c5c8ad6dc90b8) 21

[CLIPRDR\_SHORT\_FORMAT\_NAMES packet](#section_6a20969b779d452eab7c5c8ad6dc90b8) 21

[CLIPRDR\_TEMP\_DIRECTORY](#section_7a0a0433d65a4d39a5b3931ca889633e) 20

[CLIPRDR\_TEMP\_DIRECTORY packet](#section_7a0a0433d65a4d39a5b3931ca889633e) 20

[CLIPRDR\_UNLOCK\_CLIPDATA packet](#section_2ae0ff2619b24eb8af0a4811a6a18906) 23

Copy Sequence ([section 1.3.2.2.1](#section_887747ab9bad490f8ddb68c6365e58dd) 13, [section 2.2.3](#section_ba286327ef484057b0299479f4c9601a) 20, [section 3.1.5.2](#section_6fe7e1410b3e46e3b9663411ceb9b4fc) 33, [section 4.2](#section_3626af4da635452f9bc5b8e1d82d6c09) 43)

[Copy Sequence message](#section_ba286327ef484057b0299479f4c9601a) 20

**D**

Data model - abstract

client ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.2.1](#section_6906d1a0247a4401acd342aee920fd0a) 37)

server ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.3.1](#section_3a0c0d21b58b4adab406cc301365a4a7) 39)

Data types

[file stream](#section_6119fce572b84ec38871f0b785ededd3) 9

[generic](#section_21ba2e40b4eb4edfbd76519907af6bba) 9

[metafile](#section_184cbfe69ccb44778127d97563ccfdd4) 9

[overview](#section_041e7c8470c54910854d2f985bcc9eb9) 9

[palette](#section_83bb9339fdb24d408f394fbcd47abf17) 9

[Delayed rendering](#section_fa309d1b803444bfb927adfc753e69c1) 11

[Direct file access](#section_d66378b9575b4239aa52c32ea67d97e5) 32

**E**

Examples

[annotated copy sequence examples](#section_3626af4da635452f9bc5b8e1d82d6c09) 43

[annotated initialization sequence examples](#section_705fafbfbccd4b6e82e7008940b04607) 41

[annotated paste sequence examples](#section_639609e6f0544b709c912396dfff620b) 45

[client Clipboard Capabilities PDU example](#section_ac9ab35ba7054e168560c9353161778a) 41

[client Temporary Directory PDU example](#section_0689d414305040f8b50b90e112c39eb4) 42

[Format Data Request PDU example](#section_78d20ab3767149e1bbda6ee76180dba0) 45

[Format Data Response PDU example](#section_539f2ca7ee6746ba8995b2274fd8bcea) 45

Format List PDU example ([section 4.1.5](#section_c80226ff32814efd8e09d5751a45605f) 43, [section 4.2.1](#section_ba1464da316b4869b1d29b9c6eb418f4) 44)

Format List Response PDU example ([section 4.1.6](#section_d0f29c1d5ed34c728447ff2e48fcc583) 43, [section 4.2.2](#section_db8dab8c286f4563a73319d15cb12f14) 44)

[server Clipboard Capabilities PDU example](#section_017fbfda2f6d449ab9cf396b2cc58591) 41

[server Monitor Ready PDU example](#section_de4edca63bde45d58a4e752200f91911) 41

**F**

[Fields - vendor-extensible](#section_b405e97ca717406288424aed1d9c1fbf) 15

[File List](#section_c40918d1ab3840b4ae3ab8758e148926) 31

[File stream data types](#section_6119fce572b84ec38871f0b785ededd3) 9

[Format Data Request PDU example](#section_78d20ab3767149e1bbda6ee76180dba0) 45

[Format Data Response PDU example](#section_539f2ca7ee6746ba8995b2274fd8bcea) 45

Format List PDU example ([section 4.1.5](#section_c80226ff32814efd8e09d5751a45605f) 43, [section 4.2.1](#section_ba1464da316b4869b1d29b9c6eb418f4) 44)

Format List Response PDU example ([section 4.1.6](#section_d0f29c1d5ed34c728447ff2e48fcc583) 43, [section 4.2.2](#section_db8dab8c286f4563a73319d15cb12f14) 44)

[FORMAT\_LIST\_RESPONSE](#section_e3c52df2c770429e8cbb510ca55836b2) 22

[FORMAT\_LIST\_RESPONSE packet](#section_e3c52df2c770429e8cbb510ca55836b2) 22

**G**

[Generic data types](#section_21ba2e40b4eb4edfbd76519907af6bba) 9

[Glossary](#section_44250f524a9c4c1daa7041585b41f33a) 7

**H**

Higher-layer triggered events

client ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.2.4](#section_c3a86723ee1947bbba66159eb94d1306) 37)

server ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.3.4](#section_b290bcba1b5648e88c95ff2ab1de1a40) 39)

**I**

[Implementer - security considerations](#section_9a452576c16e46abb47eb3608c8868ef) 77

[Index of security parameters](#section_a471b48f320346fc96bbdeb31bf3c259) 77

[Informative references](#section_eb2fd57d68434a72ba8bf4952bc3f56c) 8

Initialization

client ([section 3.1.3](#section_59d6fb55acac4cfd9082bd150118036a) 32, [section 3.2.3](#section_87e486718ae54141baf4f4f2251d3d76) 37)

server ([section 3.1.3](#section_59d6fb55acac4cfd9082bd150118036a) 32, [section 3.3.3](#section_1a17abadf60a4b2e8a1e3019b39c8970) 39)

Initialization Sequence ([section 1.3.2.1](#section_a5cae3c9170c4154992d9ac8a149cc7e) 11, [section 2.2.2](#section_cdf6a3b839264336a30420d336bd727d) 17, [section 3.2.5.1](#section_c34bdfe6f907408196bc2f6d9bf65525) 38, [section 3.3.5.1](#section_b9f266d425fb48609773789644129898) 39, [section 4.1](#section_705fafbfbccd4b6e82e7008940b04607) 41)

[Initialization Sequence message](#section_cdf6a3b839264336a30420d336bd727d) 17

[Interacting with local Clipboard and applications](#section_9a784d09bf534362995da20d0eb5f074) 13

[Introduction](#section_1e36fc8d593a4c81815e9afb2c3ee37a) 7

**L**

[Local Clipboard update](#section_5025ebbbb4dd4524a6c12d01967382ae) 32

Local events

client ([section 3.1.7](#section_c08772f8256c4b8bb8f3b207eb7ef5bc) 37, [section 3.2.7](#section_16bab796b5a44ec1b964af25ba904bc2) 39)

server ([section 3.1.7](#section_c08772f8256c4b8bb8f3b207eb7ef5bc) 37, [section 3.3.7](#section_f70e63ce6ed94ba0ae71bdcc9f8940c8) 40)

[Local paste operation](#section_d329e0f292164e4e9181645e9f8a2187) 32

**M**

Message processing

client ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.2.5](#section_b78c2072fadb42ca8476b0defec522d1) 38)

server ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.3.5](#section_8542af96ef9a40ecb9c0d4a547ff056b) 39)

Messages

[Clipboard PDU Header (CLIPRDR\_HEADER)](#section_9e97ce45059743ddb1169f62a5b34d54) 17

[Copy Sequence](#section_ba286327ef484057b0299479f4c9601a) 20

[Initialization Sequence](#section_cdf6a3b839264336a30420d336bd727d) 17

[Paste Sequence](#section_3095a07fe55847ca966ea54bc9a6c41d) 24

[syntax](#section_6b192801f47545b787a24fe1c36bdba3) 17

[transport](#section_a373b2b937374c5fa296bc91a2f53344) 17

[Metafile data types](#section_184cbfe69ccb44778127d97563ccfdd4) 9

[Monitoring Clipboard updates](#section_1373cf9e64ce4a2a9f01bc7c5425baaa) 10

**N**

[Normative references](#section_2eae1b4b24604180b10e6a2d6eac2773) 8

**O**

Other local events

[client](#section_16bab796b5a44ec1b964af25ba904bc2) 39

[server](#section_f70e63ce6ed94ba0ae71bdcc9f8940c8) 40

[Overview](#section_e6a638f5195044d398d67f850969861a) 8

[Overview (synopsis)](#section_e6a638f5195044d398d67f850969861a) 8

**P**

[Palette data types](#section_83bb9339fdb24d408f394fbcd47abf17) 9

[PALETTEENTRY](#section_6e0dc592ee78488286bce050533f6675) 26

[PALETTEENTRY packet](#section_6e0dc592ee78488286bce050533f6675) 26

[Parameters - security](#section_a471b48f320346fc96bbdeb31bf3c259) 77

[Parameters - security index](#section_a471b48f320346fc96bbdeb31bf3c259) 77

Paste Sequence ([section 1.3.2.2.3](#section_30688d0996b646f8af18ea1998bb7987) 13, [section 2.2.5](#section_3095a07fe55847ca966ea54bc9a6c41d) 24, [section 3.1.5.4](#section_fc1254ec394b4c79bb9734d4ce1da969) 35, [section 4.4](#section_639609e6f0544b709c912396dfff620b) 45)

[Paste Sequence message](#section_3095a07fe55847ca966ea54bc9a6c41d) 24

[Preconditions](#section_c68cc20117a24147b262ba883639dbcd) 15

[Prerequisites](#section_c68cc20117a24147b262ba883639dbcd) 15

[Processing a Client Clipboard Capabilities PDU](#section_a75719fd796e476f9c80d3be6b7bfe0f) 40

[Processing a File Contents Request PDU](#section_d19de33e402743deb150668ee21bced8) 36

[Processing a File Contents Response PDU](#section_8d4c49cd92d0491ba3499fcf48233e44) 37

[Processing a Format Data Request PDU](#section_96840dc2bb534934b34dfaf893a39234) 35

[Processing a Format Data Response PDU](#section_37ea0f4642c14725a7d4d972b4e28606) 36

[Processing a Monitor Ready PDU](#section_3cd7e4d0bd8e40699974d2961a20129e) 38

[Processing a Server Clipboard Capabilities PDU](#section_8cf0af8e28dc46719831ca33ee95aee6) 38

[Processing a Temporary Directory PDU](#section_60012e81366e4cf5af4d768de1a8bc7b) 40

[Processing Clipboard PDU](#section_ed6118444eb34b29ac8c2576151b4d0c) 33

[Processing Format List PDU](#section_4e41486468804b0d8af71838ca000c2d) 33

[Processing Format List Response PDU](#section_1ac0265a779b481da95b809f28d957dd) 34

[Product behavior](#section_bfa8a3bfcb24448986507c152f84aea4) 78

**R**

[References](#section_8169da3cee784a2587e7dad476a5bae8) 7

[informative](#section_eb2fd57d68434a72ba8bf4952bc3f56c) 8

[normative](#section_2eae1b4b24604180b10e6a2d6eac2773) 8

[Relationship to other protocols](#section_dd0f20c07e884c7ab682243f797afde2) 15

**S**

Security

[implementer considerations](#section_9a452576c16e46abb47eb3608c8868ef) 77

[overview](#section_a3ca66f8b885468d8be4335dbf75b727) 77

[parameter index](#section_a471b48f320346fc96bbdeb31bf3c259) 77

[parameters](#section_a471b48f320346fc96bbdeb31bf3c259) 77

[Sending a Client Clipboard Capabilities PDU](#section_1f3e2433fed14ff8a757ad1196437ac9) 38

[Sending a File Contents Request PDU](#section_5bd7dface4684c01bc42e034bb4fa51a) 36

[Sending a File Contents Response PDU](#section_5d3991e9440d4e4d8b8d391c324c4007) 36

[Sending a Format Data Request PDU](#section_b52ef2d644424a1085b5f631775a1487) 35

[Sending a Format Data Response PDU](#section_dc95d66a9b5f4ec28878357c623fcf4c) 35

[Sending a Monitor Ready PDU](#section_a9089520044c4390a5ac8a7f8fa6d2a1) 40

[Sending a Server Clipboard Capabilities PDU](#section_0e6097613aa848fbaae77582418c002d) 39

[Sending a Temporary Directory PDU](#section_de92863f69e84f6681e8fcbddb20b6e0) 38

[Sending Format List PDU](#section_84bc5151fe6544c2a696216c4fd25c9c) 33

[Sending Format List Response PDU](#section_59c53bcde7324603b0a8e8694b444c16) 34

Sequencing rules

client ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.2.5](#section_b78c2072fadb42ca8476b0defec522d1) 38)

server ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.3.5](#section_8542af96ef9a40ecb9c0d4a547ff056b) 39)

Server

abstract data model ([section 3.1.1](#section_2cd395cbf94c4906a9e2d4b67f110245) 31, [section 3.3.1](#section_3a0c0d21b58b4adab406cc301365a4a7) 39)

higher-layer triggered events ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.3.4](#section_b290bcba1b5648e88c95ff2ab1de1a40) 39)

initialization ([section 3.1.3](#section_59d6fb55acac4cfd9082bd150118036a) 32, [section 3.3.3](#section_1a17abadf60a4b2e8a1e3019b39c8970) 39)

local events ([section 3.1.7](#section_c08772f8256c4b8bb8f3b207eb7ef5bc) 37, [section 3.3.7](#section_f70e63ce6ed94ba0ae71bdcc9f8940c8) 40)

message processing ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.3.5](#section_8542af96ef9a40ecb9c0d4a547ff056b) 39)

[other local events](#section_f70e63ce6ed94ba0ae71bdcc9f8940c8) 40

sequencing rules ([section 3.1.5](#section_b57642ccdaba43d588f9e88804b819c7) 33, [section 3.3.5](#section_8542af96ef9a40ecb9c0d4a547ff056b) 39)

timer events ([section 3.1.6](#section_70a24c52d1f34859bca71bb63ea1e0e2) 37, [section 3.3.6](#section_1c0e229764e84078970fc845276d8517) 40)

timers ([section 3.1.2](#section_6d3269df77b3443c8b56055818087ab0) 32, [section 3.3.2](#section_6aeeb91ffa5b4f149f1e7b04a43a641c) 39)

[Server Capabilities store](#section_ff0fbd69743b407a9959497bdafa3297) 37

[Server Clipboard Capabilities PDU example](#section_017fbfda2f6d449ab9cf396b2cc58591) 41

[Server Monitor Ready PDU example](#section_de4edca63bde45d58a4e752200f91911) 41

[Standards assignments](#section_beb14cd8abb94fd4a1cf722df28a92c9) 16

[Syntax - message](#section_6b192801f47545b787a24fe1c36bdba3) 17

**T**

Timer events

client ([section 3.1.6](#section_70a24c52d1f34859bca71bb63ea1e0e2) 37, [section 3.2.6](#section_abbe41549f574692819840a555fc1d6b) 38)

server ([section 3.1.6](#section_70a24c52d1f34859bca71bb63ea1e0e2) 37, [section 3.3.6](#section_1c0e229764e84078970fc845276d8517) 40)

Timers

client ([section 3.1.2](#section_6d3269df77b3443c8b56055818087ab0) 32, [section 3.2.2](#section_142af46de65849caa8d71ace2ff38800) 37)

server ([section 3.1.2](#section_6d3269df77b3443c8b56055818087ab0) 32, [section 3.3.2](#section_6aeeb91ffa5b4f149f1e7b04a43a641c) 39)

[Tracking changes](#section_43bf9e62c5694132a75c0fe9ebb87180) 79

[Transport](#section_a373b2b937374c5fa296bc91a2f53344) 17

[Transport - message](#section_a373b2b937374c5fa296bc91a2f53344) 17

Triggered events - higher-layer

client ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.2.4](#section_c3a86723ee1947bbba66159eb94d1306) 37)

server ([section 3.1.4](#section_6a9679b9de81478791d1ba9e19be99b1) 32, [section 3.3.4](#section_b290bcba1b5648e88c95ff2ab1de1a40) 39)

**V**

[Vendor-extensible fields](#section_b405e97ca717406288424aed1d9c1fbf) 15

[Versioning](#section_0dd48ed7f46742998fc0b99bac121cf3) 15