

## [MS-MICE]:

# Miracast over Infrastructure Connection Establishment Protocol

---

Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **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 can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.
- **No Trade Secrets.** Microsoft does not claim any trade secret rights in this documentation.
- **Patents.** Microsoft has patents that might cover your implementations of the technologies described in the Open Specifications documentation. Neither this notice nor Microsoft's delivery of this documentation grants any licenses under those patents or any other Microsoft patents. However, a given Open Specifications document might be covered by the Microsoft [Open Specifications Promise](#) or the [Microsoft Community Promise](#). If you would prefer a written license, or if the technologies described in this documentation 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).
- **License Programs.** To see all of the protocols in scope under a specific license program and the associated patents, visit the [Patent Map](#).
- **Trademarks.** The names of companies and products contained in this documentation might 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, email addresses, logos, people, places, and events that are 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 as specifically described above, whether by implication, estoppel, or otherwise.

**Tools.** The Open Specifications documentation does 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 documents are intended for use in conjunction with publicly available standards specifications and network programming art and, as such, assume that the reader either is familiar with the aforementioned material or has immediate access to it.

**Support.** For questions and support, please contact [dochelp@microsoft.com](mailto:dochelp@microsoft.com).

## Revision Summary

Date	Revision History	Revision Class	Comments
3/16/2017	1.0	New	Released new document.
6/1/2017	1.1	Minor	Clarified the meaning of the technical content.

# Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>5</b>
1.1	Glossary .....	5
1.2	References .....	6
1.2.1	Normative References .....	6
1.2.2	Informative References .....	7
1.3	Overview .....	7
1.4	Relationship to Other Protocols .....	10
1.5	Prerequisites/Preconditions .....	10
1.6	Applicability Statement .....	10
1.7	Versioning and Capability Negotiation .....	10
1.8	Vendor-Extensible Fields .....	10
1.9	Standards Assignments.....	10
<b>2</b>	<b>Messages.....</b>	<b>12</b>
2.1	Transport .....	12
2.2	Message Syntax .....	12
2.2.1	Miracast Messages .....	12
2.2.1.1	Source Ready Message .....	13
2.2.1.2	Stop Projection Message .....	13
2.2.1.3	Miracast TLVs .....	14
2.2.1.3.1	Friendly Name TLV .....	14
2.2.1.3.2	RTSP Port TLV .....	15
2.2.1.3.3	Source ID TLV .....	15
2.2.2	Multicast DNS Advertisement .....	15
2.2.3	Vendor Extension Attribute .....	16
2.2.3.1	Capability Attribute .....	16
2.2.3.2	Host Name Attribute.....	17
2.2.3.3	BSSID Attribute .....	17
2.2.3.4	Connection Preference Attribute.....	18
<b>3</b>	<b>Protocol Details .....</b>	<b>19</b>
3.1	Miracast Sink Details .....	19
3.1.1	Abstract Data Model .....	19
3.1.2	Timers .....	19
3.1.3	Initialization .....	19
3.1.4	Higher-Layer Triggered Events .....	19
3.1.5	Message Processing Events and Sequencing Rules .....	19
3.1.5.1	Receive Source Ready Message .....	19
3.1.5.2	Receive Stop Projection Message .....	19
3.1.6	Timer Events.....	19
3.1.7	Other Local Events.....	19
3.2	Miracast Source Details.....	20
3.2.1	Abstract Data Model.....	20
3.2.2	Timers .....	20
3.2.3	Initialization .....	20
3.2.4	Higher-Layer Triggered Events .....	20
3.2.5	Message Processing Events and Sequencing Rules .....	20
3.2.5.1	Send Source Ready Message .....	20
3.2.5.2	Send Stop Projection Message .....	20
3.2.6	Timer Events.....	21
3.2.7	Other Local Events.....	21
<b>4</b>	<b>Protocol Examples .....</b>	<b>22</b>
4.1	WSC Vendor Extension Attribute Example .....	22
4.2	Source Ready Message Example .....	22

4.3	Stop Projection Message Example .....	22
<b>5</b>	<b>Security Considerations.....</b>	<b>23</b>
<b>6</b>	<b>Appendix A: Product Behavior .....</b>	<b>24</b>
<b>7</b>	<b>Change Tracking.....</b>	<b>25</b>
<b>8</b>	<b>Index.....</b>	<b>26</b>

# 1 Introduction

The Miracast over Infrastructure Connection Establishment protocol specifies a connection negotiation sequence that is used to connect and disconnect from a Miracast over Infrastructure endpoint.

This protocol also defines the Miracast over Infrastructure Wi-Fi Simple Configuration (WSC) information element (IE) vendor extension attribute, which helps identify Miracast receivers (sinks) that can support Miracast sessions over infrastructure links in addition to Wi-Fi Direct (WFD) links.

Sections 1.5, 1.8, 1.9, 2, and 3 of this specification are normative. All other sections and examples in this specification are informative.

## 1.1 Glossary

This document uses the following terms:

**802.11 Access Point (AP):** Any entity that has IEEE 802.11 functionality and provides access to the distribution services, via the wireless medium for associated stations (STAs).

**basic service set identifier (BSSID):** A 48-bit structure that is used to identify an entity such as the access point in a wireless network. This is typically a MAC address.

**Beacon:** A management frame that contains all of the information required to connect to a network. In a WLAN, Beacon frames are periodically transmitted to announce the presence of the network.

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

**Domain Name System (DNS):** A hierarchical, distributed database that contains mappings of domain names to various types of data, such as IP addresses. DNS enables the location of computers and services by user-friendly names, and it also enables the discovery of other information stored in the database.

**friendly name:** A name for a user or object that can be read and understood easily by a human.

**organizationally unique identifier (OUI):** A unique 24-bit string that uniquely identifies a vendor, manufacturer, or organization on a worldwide basis, as specified in [IEEE-OUI]. The OUI is used to help distinguish both physical devices and software, such as a network protocol, that belong to one entity from those that belong to another.

**peer-to-peer (P2P):** An Internet-based networking option in which two or more computers connect directly to each other in order to communicate.

**Probe Request:** A frame that contains the advertisement IE for a device that is seeking to establish a connection with a proximate device. The Probe Request frame is defined in the Wi-Fi Peer-to-Peer (P2P) Specification v1.2 [\[WF-P2P1.2\]](#) section 4.2.2.

**Probe Response:** A frame that contains the advertisement IE for a device. The Probe Response is sent in response to a Probe Request. The Probe Response frame is defined in the Wi-Fi Peer-to-Peer (P2P) Specification v1.2 [\[WF-P2P1.2\]](#) section 4.2.3.

**Real-Time Streaming Protocol (RTSP):** A protocol used for transferring real-time multimedia data (for example, audio and video) between a server and a client, as specified in [\[RFC2326\]](#). It is a streaming protocol; this means that **RTSP** attempts to facilitate scenarios in which the multimedia data is being simultaneously transferred and rendered (that is, video is displayed and audio is played).

**subnet:** A logical division of a network. Subnets provide a multilevel hierarchical routing structure for the Internet. On TCP/IP networks, subnets are defined as all devices whose IP addresses have the same prefix. Subnets are useful for both security and performance reasons. In general, broadcast messages are scoped to within a single subnet. For more information about subnets, see [\[RFC1812\]](#).

**Transmission Control Protocol (TCP):** A protocol used with the Internet Protocol (IP) to send data in the form of message units between computers over the Internet. TCP handles keeping track of the individual units of data (called packets) that a message is divided into for efficient routing through the Internet.

**type-length-value (TLV):** A property of a network interface, so named because each property is composed of a Type field, a Length field, and a value.

**User Datagram Protocol (UDP):** The connectionless protocol within TCP/IP that corresponds to the transport layer in the ISO/OSI reference model.

**UTF-16:** A standard for encoding Unicode characters, defined in the Unicode standard, in which the most commonly used characters are defined as double-byte characters. Unless specified otherwise, this term refers to the UTF-16 encoding form specified in [\[UNICODE5.0.0/2007\]](#) section 3.9.

**UTF-8:** A byte-oriented standard for encoding Unicode characters, defined in the Unicode standard. Unless specified otherwise, this term refers to the UTF-8 encoding form specified in [\[UNICODE5.0.0/2007\]](#) section 3.9.

**Wi-Fi Direct (WFD):** A standard that allows Wi-Fi devices to connect to each other without requiring a **wireless access point (WAP)**. This standard enables WFD devices to transfer data directly among each other resulting in significant reductions in setup.

**Wi-Fi Protected Setup (WPS):** A computing standard that attempts to allow easy establishment of a secure wireless home network. This standard was formerly known as Wi-Fi Simple Config.

**wireless access point (WAP):** A wireless network access server (NAS) that implements 802.11.

**MAY, SHOULD, MUST, SHOULD NOT, MUST NOT:** These terms (in all caps) are used as defined in [\[RFC2119\]](#). All statements of optional behavior use either MAY, SHOULD, or SHOULD NOT.

## 1.2 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](#).

### 1.2.1 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.

[IANAPORT] IANA, "Service Name and Transport Protocol Port Number Registry", <http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

[IANA] IANA, "Internet Assigned Numbers Authority (IANA)", <http://www.iana.org>

[IEEE802.11-2012] IEEE, "Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY)

Specifications", ANSI/IEEE Std 802.11-2012, <http://standards.ieee.org/getieee802/download/802.11-2012.pdf>

**Note** There is a charge to download this document.

[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>

[RFC2326] Schulzrinne, H., Rao, A., and Lanphier, R., "Real Time Streaming Protocol (RTSP)", RFC 2326, April 1998, <http://www.ietf.org/rfc/rfc2326.txt>

[RFC6762] Krochmal, M. and Cheshire, S., "Multicast DNS", <http://www.rfc-editor.org/rfc/rfc6762.txt>

[RFC768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980, <http://www.rfc-editor.org/rfc/rfc768.txt>

[RFC793] Postel, J., Ed., "Transmission Control Protocol: DARPA Internet Program Protocol Specification", RFC 793, September 1981, <http://www.rfc-editor.org/rfc/rfc793.txt>

[WF-DTS1.1] Wi-Fi Alliance, "Wi-Fi Display Technical Specification v1.1", April 2014, <https://www.wi-fi.org/file/wi-fi-display-technical-specification-v11>

**Note** There is a charge to download the specification.

[WF-P2P1.2] Wi-Fi Alliance, "Wi-Fi Peer-to-Peer (P2P) Technical Specification v1.2", <https://www.wi-fi.org/wi-fi-peer-to-peer-p2p-technical-specification-v12>

**Note** There is a charge to download the specification.

[WF-WSC2.0.2] Wi-Fi Alliance, "Wi-Fi Simple Configuration Technical Specification v2.0.2", August 2011, <https://www.wi-fi.org/wi-fi-simple-configuration-technical-specification-v202>

**Note** There is a charge to download the specification.

## 1.2.2 Informative References

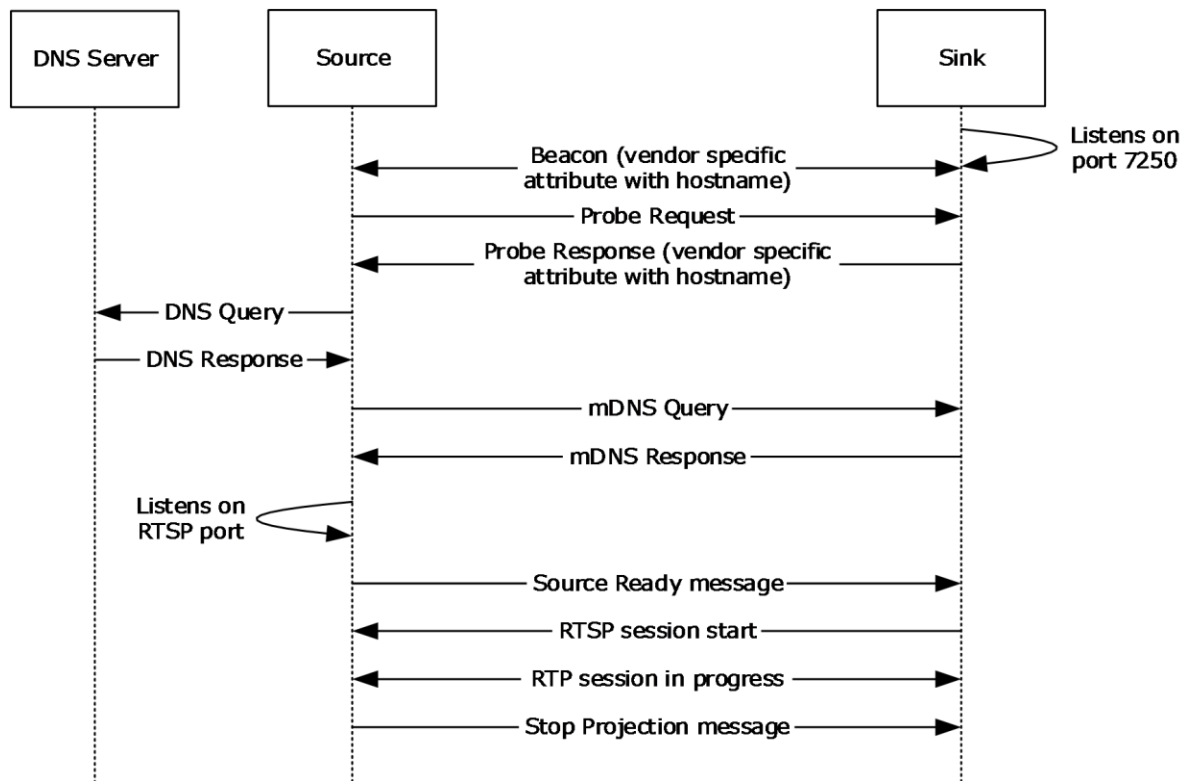
[IEEE-OUI] IEEE Standards Association, "IEEE OUI Registration Authority", February 2007, <http://standards.ieee.org/regauth/oui/oui.txt>

## 1.3 Overview

The Miracast over Infrastructure protocol defines the following actors:

- **Miracast Source:** The device that sends audio and video streams to the Miracast Sink. This device is sometimes called a "sender". Optionally, this device can also receive input signals from the Miracast Sink.
- **Miracast Sink:** The device that receives audio and video streams from the Miracast Source. This device is sometimes called a "receiver". Optionally, this device can also send input signals back to the Miracast Source.

The following diagram shows the lifelines of actors in a Miracast over Infrastructure session.



**Figure 1: Miracast over Infrastructure actor lifelines**

A session between a Miracast Source and Sink is established before projection can start. Sessions include the following phases.

1. **Wi-Fi Direct (WFD)** device discovery ([\[WF-DTS1.1\]](#) section 4.9.2).

Management frames are exchanged, including **Beacons**, **Probe Requests**, and **Probe Responses** ([\[WF-P2P1.2\]](#) section 4.2). The Wi-Fi Simple Configuration (WSC) information element (IE) vendor extension attribute (section [2.2.3](#)) is used to advertise the presence of devices that can perform WSC operations.

2. The Source resolves the host name of the target Sink device by initiating **DNS** and/or Multicast DNS (mDNS) [\[RFC6762\]](#) queries.
3. Source messages (section [2.2.1](#)) to communicate to the Sink about starting and stopping the projection.

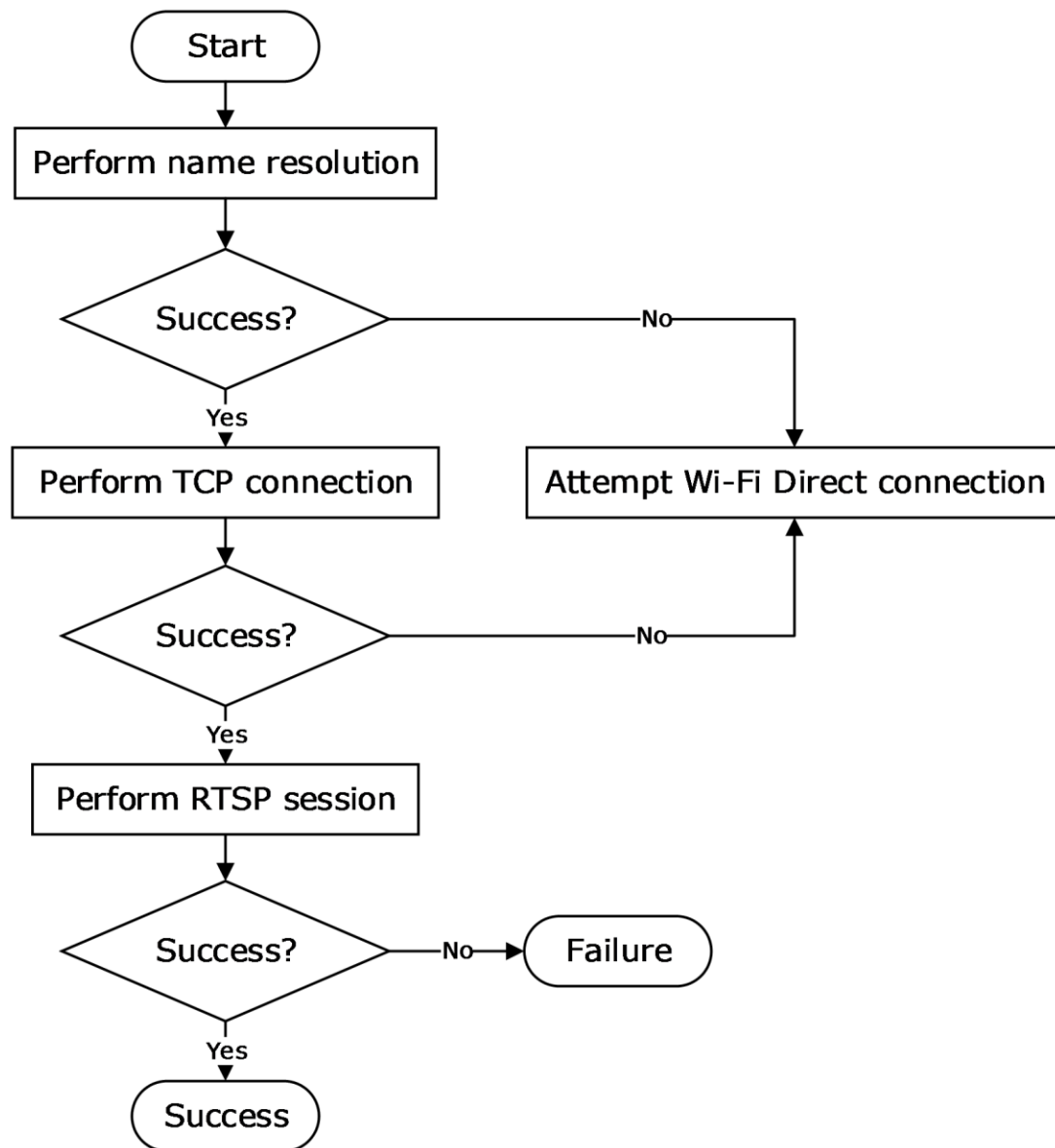
Implementing the Miracast over Infrastructure protocol requires an understanding of session establishment over standard Miracast.

First, the connection to a Sink over standard Miracast is established over Wi-Fi Direct (WFD). Because Miracast over Infrastructure does not require WFD, a way is needed to inform the Miracast Source that a Miracast over Infrastructure connection is possible, even if the target Sink isn't on the same **subnet** as the Source. That ability is provided through a WSC IE vendor extension attribute (section [2.2.3](#)).

Second, standard Miracast is triggered to start automatically when a WFD session is established. Because the Miracast over Infrastructure protocol is not triggered by WFD, messages have been defined for starting and stopping Miracast over Infrastructure sessions (section [2.2.1](#)).

WFD device discovery is performed, even if the session connection might later be made by using the Miracast over Infrastructure protocol. If the connection cannot be made by using this protocol, the Source falls back to a standard WFD Miracast session.

The following diagram illustrates the attempt to establish a Miracast over Infrastructure session, along with some possible outcomes.



**Figure 2: Establishing a Miracast over Infrastructure session**

When a Source is ready to project to a Sink, it listens on its control port for **Real-Time Streaming Protocol (RTSP)** (7236 by default) for connection requests and then sends a Source Ready message to the Sink. The Sink is expected to be listening for Source Ready messages on **TCP** port 7250, and the Source connects to port 7250 to deliver RTSP port information to the Sink in the Source Ready message (section [2.2.1.1](#)). In turn, the Sink connects to the specified RTSP Source port to establish the link.

To pause or stop the projection, the Source sends a Stop Projection message (section [2.2.1.2](#)) to notify the Sink. Upon receipt of that message, the Sink stops displaying the stream, and a disconnection follows from the Source on the socket that is connected on port 7250.

## 1.4 Relationship to Other Protocols

The Miracast over Infrastructure protocol builds upon the following standard technologies:

- Multicast DNS (mDNS) [\[RFC6762\]](#)
- **Real-Time Streaming Protocol (RTSP)** [\[RFC2326\]](#)
- **Transmission Control Protocol (TCP)** [\[RFC793\]](#)
- User Datagram Protocol [\[RFC768\]](#)
- Wi-Fi Display Protocol [\[WF-DTS1.1\]](#)
- Wi-Fi Peer-to-Peer (P2P) Protocol [\[WF-P2P1.2\]](#)
- Wi-Fi Simple Configuration (WSC) Protocol [\[WF-WSC2.0.2\]](#)

## 1.5 Prerequisites/Preconditions

The Miracast over Infrastructure protocol requires that the following both be true:

- The Miracast Source and Miracast Sink endpoints are on the same logical IP network, so they can establish a Miracast over Infrastructure connection.
- Either the Sink is on the same logical IP **subnet** as the Source, or the Sink's name is registered in a **DNS** server that the Source can resolve to.

## 1.6 Applicability Statement

The Miracast over Infrastructure protocol is applicable to projecting content from one device to another, such as PC to large-screen TV, PC to PC, phone to PC, and so on.

The protocol functions in a configuration in which Miracast Source and Miracast Sink devices share a common logical IP network and determine they can project content across that network.

## 1.7 Versioning and Capability Negotiation

This is the first version of the protocol.

## 1.8 Vendor-Extensible Fields

None.

## 1.9 Standards Assignments

The Miracast over Infrastructure protocol uses the following standard port assignments.

Parameter	Value	Reference
Multicast DNS	5353	<a href="#">[IANAPORT]</a>
RTSP requests (both UDP and TCP)	7236	[IANAPORT]

Parameter	Value	Reference
TCP	7250	<a href="#">[IANA]</a>

## 2 Messages

### 2.1 Transport

The Miracast over Infrastructure Source Ready and Stop Projection messages (section [2.2.1](#)) are sent over **TCP** port 7250.

The Multicast DNS (mDNS) [\[RFC6762\]](#) messages utilize the standard **UDP** transport [\[RFC768\]](#) on port 5353.

### 2.2 Message Syntax

In the structures defined in this section, multi-byte field values are ordered in **big-endian** format, unless specified otherwise.

#### 2.2.1 Miracast Messages

This section defines the messages used for starting and stopping Miracast over Infrastructure sessions. This is the general format for Miracast messages:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Size																Version								Command							
TLVArray (variable)																															
...																															
...																															
...																															

**Size (2 bytes):** The size of the message, in bytes.

**Version (1 byte):** The version of this protocol, which is 0x01.

**Command (1 byte):** The type of message, which determines the **TLVs** passed in the **TLVArray** field. The following messages are defined in the sections listed.

Message type	Section	Description
SOURCE_READY 0x01	<a href="#">2.2.1.1</a>	Indicates the Miracast Source is ready to accept a connection on the <b>RTSP</b> port.
STOP_PROJECTION 0x02	<a href="#">2.2.1.2</a>	Indicates the end of the projection.

**TLVArray (variable):** An array of one or more Miracast TLVs (section [2.2.1.3](#)), which specify information for the message.



- Friendly Name TLV (section [2.2.1.3.1](#))
- Source ID TLV (section [2.2.1.3.3](#))

### 2.2.1.3 Miracast TLVs

This section defines common **type-length-value (TLV)** structures that are used to pass information in messages during a Miracast session. This is the general format for the TLVs:

0	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	20	1	2	3	4	5	6	7	8	9	30	1		
Type									Length															Value (variable)									
...																																	
...																																	
...																																	

**Type (1 byte):** The type of TLV, which determines the information passed in the **Value** field. The following TLVs are defined in the sections listed.

TLV type	Section	Description
FRIENDLY_NAME 0x00	<a href="#">2.2.1.3.1</a>	Specifies the <b>friendly name</b> of the Miracast Source.
RTSP_PORT 0x02	<a href="#">2.2.1.3.2</a>	Specifies the port on which the Source is listening for <b>RTSP</b> connections.
SOURCE_ID 0x03	<a href="#">2.2.1.3.3</a>	Specifies an identifier for the Source, which is used for all messages sent during a Miracast session.

**Length (2 bytes):** The length of the **Value** field, in bytes. This value **MUST** be greater than or equal to 0x0001.

**Value (variable):** One or more bytes, which specify information for the TLV.

#### 2.2.1.3.1 Friendly Name TLV

The Friendly Name **TLV** specifies the **friendly name** of the Miracast Source in messages to the Miracast Sink.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Type									Length															Value (variable)							
...																															
...																															

**Type (1 byte):** The type of TLV, which is 0x00 for the Friendly Name TLV.

**Length (2 bytes):** The length of the **Value** field, in bytes.

**Value (variable):** The friendly name string of the Source, encoded in **UTF-16**.

#### 2.2.1.3.2 RTSP Port TLV

The RTSP Port **TLV** specifies the port on which the Miracast Source is listening. The port is used in messages for connecting sessions over **RTSP**.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Type										Length																Value					
...																															

**Type (1 byte):** The type of TLV, which is 0x02 for the RTSP Port TLV.

**Length (2 bytes):** The length of the **Value** field, in bytes, which is 0x0002.

**Value (2 bytes):** The RTSP port on which the Source is listening (7236 by default).

#### 2.2.1.3.3 Source ID TLV

The Source ID **TLV** specifies a unique identifier for the Miracast Source. That identifier is used in all messages sent during a session.

											1											2											3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1			
Type								Length														Value												
...																																		
...																																		
...																																		
...																																		

**Type (1 byte):** The type of TLV, which is 0x03 for the Source ID TLV.

**Length (2 bytes):** The length of the **Value** field, in bytes, which is 0x0010.

**Value (16 bytes):** An implementation-defined value that identifies the Source.

### 2.2.2 Multicast DNS Advertisement

During the establishment of a session with the Miracast Source, the Miracast Sink publishes a Multicast DNS (mDNS) [\[RFC6762\]](#) advertisement in the following format:

<instance name>.<service name>.<transport protocol>.<domain name>

Where: <instance name> is the **friendly name** of the Sink; <service name> is "\_display"; <transport protocol> is "\_tcp"; and <domain name> is "local".

### 2.2.3 Vendor Extension Attribute

The Miracast over Infrastructure vendor extension attribute is published in the Wi-Fi Simple Configuration (WSC) information element (IE) [\[WF-WSC2.0.2\]](#) for the Group Owner (GO) **Beacons**, **Probe Request**, and **Probe Response** frames ([\[WF-P2P1.2\]](#) section 4.2).

The attribute data is represented by one or more of the structures defined in the sections that follow.

The WSC vendor extension attribute has the following format:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
WSCVEA																Length															
OUI																								P2PATTB (variable)							
...																															
...																															

**WSCVEA (2 bytes):** The value is 0x1049 to indicate that this attribute is a WSC vendor extension.

**Length (2 bytes):** The length of the following fields in bytes.

**OUI (3 bytes):** A **Wi-Fi Protected Setup (WPS) organizationally unique identifier (OUI)** [\[IEEE-OUI\]](#). The value is 0x000137.

**P2PATTB (variable):** One or more of the **peer-to-peer (P2P)** attribute structures defined in the sections that follow. The attributes can be included in any order.

#### 2.2.3.1 Capability Attribute

The Capability attribute indicates whether a connection over Miracast over Infrastructure is possible. This attribute **MUST** be present in the vendor extension attribute.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
AttributeID																Length															
CapabilityInfo																															

**AttributeID (2 bytes):** The Capability attribute ID, which is 0x2001.

**Length (2 bytes):** The length of the **CapabilityInfo** field, in bytes, which is 0x0001.

**CapabilityInfo (1 byte):** A bit field table with capability information, which has the following structure:



**BSSID (6 bytes):** The BSSID for the associated **WAP**.

#### 2.2.3.4 Connection Preference Attribute

The **Connection Preference** attribute indicates the preference of transports for the connection of the Miracast Sink to the Miracast Source. This attribute is optional in the vendor extension attribute.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
AttributeID																Length															
ConnectionPreferenceList																															

**AttributeID (2 bytes):** The Connection Preference attribute ID, which is 0x2004.

**Length (2 bytes):** The length of the **ConnectionPreferenceList** field, in bytes, which is 0x0004.

**ConnectionPreferenceList (4 bytes):** A packed array with room for 8 connection transport IDs, in descending order of preference. The following IDs are defined:

Transport ID	Transport
0x1	Miracast over Infrastructure
0x2	<b>Wi-Fi Direct (WFD)</b>

The following is an example of a preference list buffer with Miracast over Infrastructure preferred over WFD.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0x1				0x2				0				0				0				0				0				0			

## 3 Protocol Details

### 3.1 Miracast Sink Details

The Miracast over Infrastructure attributes are published as a vendor extension attribute in the Wi-Fi Simple Configuration (WSC) information element (IE) [\[WF-WSC2.0.2\]](#) from the Miracast Sink.

#### 3.1.1 Abstract Data Model

None.

#### 3.1.2 Timers

None.

#### 3.1.3 Initialization

Upon initialization, the Miracast Sink MUST start listening on port 7250 for an inbound session.

#### 3.1.4 Higher-Layer Triggered Events

None.

#### 3.1.5 Message Processing Events and Sequencing Rules

There are two messages the Miracast Sink can receive, the Source Ready and Stop Projection messages (section [2.2.1](#)).

##### 3.1.5.1 Receive Source Ready Message

When a Miracast Sink receives a Source Ready message (section [2.2.1.1](#)), it MUST connect back to the Miracast Source over **TCP** on the **RTSP** port specified in the Source Ready message.

##### 3.1.5.2 Receive Stop Projection Message

When the Miracast Sink receives a Stop Projection message (section [2.2.1.2](#)), the Sink SHOULD stop displaying the stream and SHOULD expect a disconnection from the Miracast Source on the socket connected on port 7250.

#### 3.1.6 Timer Events

None.

#### 3.1.7 Other Local Events

A Miracast Sink might not receive a Stop Projection message (section [2.2.1.2](#)) at the end of a session. Therefore, a Sink MAY use other means to determine that a Miracast Source is no longer projecting and clean up its session.

## 3.2 Miracast Source Details

### 3.2.1 Abstract Data Model

This section describes a conceptual model of possible data organization that an implementation maintains to participate in this protocol. The described organization is provided to facilitate the explanation of how the protocol behaves. This document does not mandate that implementations adhere to this model, provided their external behavior is consistent with that described in this document.

The **Source ID** value (section [2.2.1.3.3](#)) is maintained throughout the lifetime of the Miracast session. It is included in each Miracast message (section [2.2.1](#)) to identify the Miracast Source.

### 3.2.2 Timers

This Miracast Source uses the following timers:

- **Discovery timer:** This timer is initialized when the Source attempts to resolve the host name of the Miracast Sink over DNS or mDNS (section [3.2.5](#)).[<1>](#)
- **Control channel connection timer:** This timer is initialized when the Source attempts to connect to the Miracast Sink over the control channel.[<2>](#)

### 3.2.3 Initialization

The **Source ID** member of the abstract data model is initialized to an implementation-dependent.

### 3.2.4 Higher-Layer Triggered Events

None.

### 3.2.5 Message Processing Events and Sequencing Rules

The Miracast Source discovers the Miracast Sink over **Wi-Fi Direct (WFD)**. WFD discovery is specified in the Wi-Fi Display Protocol [\[WF-DTS1.1\]](#) section 4.

During WFD discovery, the Source looks for the presence of the Wi-Fi Simple Configuration (WSC) information element (IE) vendor extension attribute (section [2.2.3](#)) and parses its contents. With information from this attribute, the Source initiates DNS and/or mDNS queries to resolve the host name of the target Sink device.

#### 3.2.5.1 Send Source Ready Message

When the Miracast Source has resolved the host name for the target Miracast Sink, the Source sends the Source Ready message (section [2.2.1.1](#)) to the Sink. This message includes a specific **RTSP** port for the Sink to connect back on.

#### 3.2.5.2 Send Stop Projection Message

When the Miracast Source ends a session, the Source sends a Stop Projection message (section [2.2.1.2](#)) to the Miracast Sink. After the message is sent, the Source closes the **TCP** session to the **RTSP** port.

### 3.2.6 Timer Events

If either of the Miracast Source timers (section [3.2.2](#)) reaches its timeout, the attempt to start a Miracast over Infrastructure session is abandoned and an WFD connection is attempted instead. This is shown in a figure in section [1.3](#).

### 3.2.7 Other Local Events

None.

## 4 Protocol Examples

### 4.1 WSC Vendor Extension Attribute Example

```
10 49 // WSC Vendor Extension
00 19 // Size = 25 Bytes
00 01 37 // Microsoft WPS OUI

20 01 // Capability Attribute
00 01 // Size = 1 Byte
88 // Capability

20 02 // HostName Attribute
00 0D // Size = 13 Bytes
57 46 44 53 75 72 66 61 63 65 48 75 62 // HostName - "WFDSurfaceHub"
```

### 4.2 Source Ready Message Example

```
00 3D // Size = 61 bytes
01 // Version = 0x1
01 // Source Ready

00 // Friendly Name TLV
00 1E // Size = 30 bytes
44 00 75 00 6D 00 6D 00 79 00 31 00 2D 00 4B 00 // Dummy1-Kabylake
61 00 62 00 79 00 6C 00 61 00 6B 00 65 00

02 // RTSP Port TLV
00 02 // Size = 2 Bytes
1C 44 // Port = 7236

03 // Source Id TLV
00 10 // Size = 16 Bytes
91 F4 AB E9 EF F5 46 4A AE E2 69 72 2A ED 11 B5 // Source ID
```

### 4.3 Stop Projection Message Example

```
00 38 // Size = 56 bytes
01 // Version = 0x1
02 // Stop Projection

00 // Friendly Name TLV
00 1E // Size = 30 bytes
44 00 75 00 6D 00 6D 00 79 00 31 00 2D 00 4B 00 // Dummy1-Kabylake
61 00 62 00 79 00 6C 00 61 00 6B 00 65 00

03 // Source Id TLV
00 10 // Size = 16 Bytes
91 F4 AB E9 EF F5 46 4A AE E2 69 72 2A ED 11 B5 // Source ID
```

## 5 Security Considerations

It is recognized that a Miracast over Infrastructure stream can occur without the benefit of link layer encryption when the connection is not over **Wi-Fi Direct (WFD)**.[<3>](#)

## 6 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 10 v1703 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 3.2.2](#): The Windows implementation uses a period of 1.5 second for the discovery timer.

[<2> Section 3.2.2](#): The Windows implementation uses a period of 5 seconds for the control channel connection timer.

[<3> Section 5](#): The Windows implementation does not attempt a Miracast over Infrastructure connection if the wireless network it is connected to does not employ link layer security (WPA2).

## 7 Change Tracking

This section identifies changes that were made to this document since the last release. Changes are classified as Major, Minor, or None.

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.
- A document revision that captures changes to protocol functionality.

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 **None** means that no new technical changes were introduced. Minor editorial and formatting changes may have been made, but the relevant technical content is identical to the last released version.

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	Description	Revision class
<a href="#">2.2.3.1</a> Capability Attribute	Editorial revision to the description for Length field - 'Is' to 'is'.	Minor
<a href="#">4.1</a> WSC Vendor Extension Attribute Example	7384 : Revised representation for Capability in example to match definition.	Minor
4.1 WSC Vendor Extension Attribute Example	7385 : Revised UTF-8 coding of bytes to match HostName in example - 'WfdSurfaceHub' to 'WFDSurfaceHub'.	Minor

## 8 Index

### A

Abstract data model  
[Miracast Sink](#) 19  
[Miracast Source](#) 20  
[Applicability](#) 10

#### Attributes

[basic service set identifier \(BSSID\)](#) 17  
[Capability](#) 16  
[Connection Preference](#) 18  
[Host Name](#) 17

### B

[Basic service set identifier \(BSSID\) attribute](#) 17

### C

[Capability attribute](#) 16  
[Capability negotiation](#) 10  
[Change tracking](#) 25  
[Connection Preference attribute](#) 18

### E

#### Examples

[Source Ready Message](#) 22  
[Stop Projection Message](#) 22  
[WSC Vendor Extension attribute](#) 22

### F

[Fields - vendor-extensible](#) 10  
[Friendly Name TLV message](#) 14

### G

[Glossary](#) 5

### H

#### Higher-layer triggered events

[Miracast Sink](#) 19  
[Miracast Source](#) 20  
[Host Name attribute](#) 17

### I

[Implementer - security considerations](#) 23  
[Informative references](#) 7

#### Initialization

[Miracast Sink](#) 19  
[Miracast Source](#) 20  
[Introduction](#) 5

### M

#### Message processing events

[Miracast Sink](#) 19  
[Miracast Source](#) 20

#### Messages

[Friendly Name TLV](#) 14  
[Miracast Messages](#) 12  
[Multicast DNS Advertisement](#) 15  
[RTSP Port TLV](#) 15  
[Source ID TLV](#) 15  
[Source Ready](#) 13  
[Stop Projection](#) 13  
[transport](#) 12  
[Vendor Extension Attribute](#) 16  
[Miracast Messages message](#) 12

#### Miracast Sink

[details](#) 19  
[overview](#) 7

#### Miracast Source

[details](#) 20  
[overview](#) 7  
[Multicast DNS Advertisement message](#) 15

### N

[Normative references](#) 6

### O

#### Other local events

[Miracast Sink](#) 19  
[Miracast Source](#) 21  
[Overview \(synopsis\)](#) 7

### P

[Preconditions](#) 10  
[Prerequisites](#) 10  
[Product behavior](#) 24  
Protocol examples  
[Source Ready Message example](#) 22  
[Stop Projection Message example](#) 22  
[WSC Vendor Extension attribute example](#) 22

### R

[Receiver](#) 7  
[References](#) 6  
[informative](#) 7  
[normative](#) 6  
[Relationship to other protocols](#) 10  
[RTSP Port TLV message](#) 15

### S

[Security - implementer considerations](#) 23  
[Sender](#) 7  
Sequencing Rules  
[Miracast Sink](#) 19  
[Miracast Source](#) 20  
[Session establishment](#) 7  
[Source ID TLV message](#) 15  
[Source Ready message](#) 13  
[Source Ready Message example](#) 22  
[Standards assignments](#) 10

[Stop Projection message](#) 13  
[Stop Projection Message example](#) 22  
[Structures – type-length value \(TLV\)](#) 14

## **T**

Timer events

[Miracast Sink](#) 19

[Miracast Source](#) 21

Timers

[Miracast Sink](#) 19

[Miracast Source](#) 20

[TLV structures](#) 14

[Tracking changes](#) 25

[Transport](#) 12

[Type-length value \(TLV\) structures](#) 14

## **V**

[Vendor Extension Attribute message](#) 16

[Vendor-extensible fields](#) 10

[Versioning](#) 10

## **W**

[WSC Vendor Extension attribute example](#) 22