Simple Device Discovery Protocol

Specification Document

Control4™

Disclaimer

The information in this document is provided in connection with Control4 products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by or in this document. Except as provided in Control4’s terms and conditions for the license of such products, Control4 Corporation and its affiliates, (“Control4”) assume no liability whatsoever, and Control4 disclaims any express or implied warranty, relating to the sale and/or use of Control4 products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Control4 products are not intended for use in medical, life saving, or life sustaining applications.

Information regarding third-party products is provided solely for educational purposes. Control4 is not responsible for the performance or support of third-party products and does not make any representations or warranties whatsoever regarding the quality, reliability, functionality or compatibility of these products. The reader is advised that third parties can have intellectual property rights that can be relevant to this document and the technologies discussed herein, and is advised to seek the advice of competent legal counsel, without obligation of Control4.

Control4 retains the right to make changes to this document or related product specifications and descriptions, at any time, without notice. Control4 makes no warranty for the use of this document and assumes no responsibility for any errors that can appear in the document nor does it make a commitment to update the information contained herein.

Trademarks

Copyright

Contact Us

Control4 Corporation
11734 S. Election Road
Salt Lake City, UT 84020 USA
http://www.control4.com

1.0 Overview

Simple Device Discover Protocol 1.0 (SDDP 1.0) is a protocol developed by Control4 to allow devices to be easily added to a control system. It has the ability to leverage other protocols and work on almost any physical layer. The protocol offers four primary functions:

1. Enable Devices to use Dynamic Host Configuration Protocol (DHCP) while still being able to identify them uniquely
2. Enable Director to discover devices residing within the Control4 system.
3. Enable Director to identify devices residing within the Control4 system.
4. Enable automatic installation of drivers for discovered devices.

Currently, SDDP is used on IP networks and assumes that only one "controller" device exists within in a system and that this device controls one or more "controlled" devices. Throughout this document, a "controller" in the system will be referred to as the Home Area Network (HAN) Controller.

This document's purpose is to explain how SDDP accomplishes the four functions enumerated earlier in this section. Most notably is the ability of a given device to search a network to find all available devices at any time. This is useful as it supports the detection of all devices that were up and running **prior** to the device performing the query became active on the network. The query itself can be narrowed to search for devices of a specific type.

SDDP messages are UDP messages that are broadcast on the multicast address 239.255.255.250 on port 1902. An implementation therefore must register as a listener to receive such broadcasts. SDDP also uses unicast messages. An example for this would be a response to a search request that is sent directly to the sender.

2.0 General SDDP message format

While the format is similar as defined in RFC 2616 (HTTP) section 4, the HTTP-Version used in SDDP 1.0 must be "SDDP/1.0". The valid method names are defined in this document. Any methods and headers defined and specified in RFC 2616 have no meaning and may be defined for different purposes in this document.

3.0 SDDP message headers

The following headers are commonly used in SDDP messages:

`Host`:

The host name of a device. This header is only used in messages originating from a device. Note that this value must be quoted for example, "`HomeController-000FFF001234`". If you have previously implemented UPnP, this value is the same as the Host value used by the SSDP protocol. Control4 recommends that you use the MAC Address of the device for this value if you have not previously implemented UPnP or SSDP.

`From:`

This value represents the address of the device that is the "sender" of the message. This value may be dynamically created if using DHCP as a network protocol or static if an I.P address is configured by a user. This must be in the format of: IP:Port, For example, "192.168.0.123:1902". This value must be quoted. Note the syntax used contains no spaces.

`Type`:

This header contains the search type as specified in the `<search_type>` tag in the c4i file for the device. Device type values are a unique identifier for a kind of device. Device types are organized in name spaces, separated by a : (colon) character, for example "`c4:HC1000HomeController`" Note that this value must be quoted.

It is important to validate that your SDDP enabled device driver contains a "Type" section. If the driver currently does not have a defined `<search_type>` element, it will need to be added in order for SSDP to work correctly. The value declared in the SDDP data packet for Type must identically match the value provided in the `<search_type>` section of your device driver.

When adding a new search type entry into a driver the naming convention used can be free form. However, Control4 recommends using a format that reflects the following: manufacturer:simplemodelnumber. For example: "`c4:HC1000HomeController`".

The following is an example of a search type driver entry for a single device type:

```
<search_type>c4:SR250remote</search_type>
```

Multiple search types can be found within one driver. The following example shows several types and how the driver `<search_type>` section should be formatted:

```
<search_types>
        <type>c4:HC1000HomeController</type>
        <type>c4:SR250remote</type>
        <type>Sony:STR5800</type>
</search_types>
```

If multiple types are listed in your driver, only one of the defined types needs to match the Type value in the data packet for SDDP to work.

`Tran`:
Tran value represents a transaction number. This is a random number that is only used in request messages. The same value will be included in the response message. This allows responses to be matched against requests. Note that the Tran header is required but it can be sent with no value.

`Timeout`:
If this header is present, responses should be sent after a random delay no longer than the number of seconds specified here. Depending on the implementation, a response may be sent without delay. The Timeout header is used to avoid immediate responses from all matching devices at the same time. However, it is optional and may not be implemented by the daemon. Even though the request may have a Timeout value, the server may respond immediately regardless. Note that delaying sending out responses should not block the daemon from responding to other requests.

`Primary-Proxy`:
This header indicates the primary proxy type defined in the device driver. For example, `<proxy>TV</proxy>`. In this example, TV would be the Primary-Proxy.
If the device driver contains several proxies defined in the `<Proxies>` section of the driver, the primary proxy value will always be the proxy listed first. For example, a TV with integrated DVD player may be defined in the c4i file as:

```
<proxies>
        <proxy>TV</proxy>
        <proxy>DVD</proxy>
<proxies/>
```

Based on our example above, “TV” will be used as the primary proxy SDDP value. Note that this value must be quoted. A supported list of Proxy types follows:

```
amplifier
aswitch
avswitch
blind
blindV2
cable
cd
discchanger
dvd
dvr
ipod
light
media_player
others
plasma
pool
projector
receiver
satellite
security
thermostat
thermostatV2
tuner
tunerXM
tv
vcr
```

`Proxies`:

This header contains a comma-separated list of proxies that are supported. This list should only contain proxies as defined in the c4i or .c4z but it may only be a subset of what is in the c4i or .c4z .

`Manufacturer`:

This header contains the name of the manufacturer of the device. This value must be quoted.

`Model`:

This header contains the model of the device. This value must be quoted.

`Driver`:

This header contains the file name of the c4i driver. This file name must match the driver file name that is used on the C4 online driver database. This value must be quoted. The structure is "device_Manufacturer_ModelName.c4i" or "tv_Control4_c4tv.c4z". Note that no special characters (&, $, #, etc.) are supported in the Driver Name.

`Max-Age:`

This value is the announcement interval in seconds. The time interval is 30 minutes. This value represents the amount of time between sending NOTIFY messages. Control4 recommends that you do not modify this value. It is also important to send ALIVE messages more often than the Max-Age value. This will avoid the Max-Age time frame expiring before an ALIVE message is sent. While this value can be omitted, Control4 recommends that you do use it to ensure proper device presence caching.

`Config-URL`

This contains the URL to the device's http accessible management user interface. If the device has no management UI, this can be omitted.

4.0 SDDP request message

The format is similar as defined in RFC 2616 (HTTP) section 5; however sections 5.2 and 5.3 do not apply. The Host header field is not mandatory, and its purpose is different than what is defined in RFC 2616. The Request-URI as specified in RFC2616 (HTTP) section 5.1.2 is used as request method Argument in SDDP, and should be interpreted as outlined in this document instead. If a method does not have any arguments, * must be used.

The following is an example of a SDDP request line:

```
METHOD Argument SDDP/1.0
```

A request message requires a response to be sent to the sender.

4.1 SEARCH request method

```
SEARCH [ * | Type ] SDDP/1.0
```

The `SEARCH` method is used to discover SDDP devices. This message is broadcast to the multicast address in order to reach all SDDP enable devices.

If * is used as argument, all devices must respond to this request. If the argument is any other value, only devices that match the Type or are within the namespace of `Type` must respond.

The following headers are required:

```
    From
    Tran
```

The following header is optional:

```
    Timeout
```

The following is an example for searching for all SDDP devices by Director:

```
SEARCH * SDDP/1.0
From: "192.168.0.111:1902"
Tran: 38359
Timeout: 10
```

The following is an example for searching for all C4 Televisions in Composer:

```
SEARCH c4:television SDDP/1.0
From: "192.168.0.21:1902"
Tran: 7129
```

In the above example, only devices within the `c4:television` namespace would respond, but not devices in the c4:projector namespace. If the search was for the entire c4 namespace, all Control4 SDDP devices would respond.

4.2 SEARCH response

Devices that are required to respond, must respond with a `200 OK` response message. The response is sent directly to the sender of the request, which is specified in the `From` header in the search request. The `Type` header contains the complete device type, including namespace.

The following headers are required:

```
      Tran
      Host
      From
      Type
      Primary-Proxy
      Proxies
      Manufacturer
      Model
      Driver
      Max-Age
```

The following header is optional:

```
      Config-URL
```

The following is an example for a response from a C4 Television:

```
SDDP/1.0 200 OK
From: "10.10.0.3:1902"
Host: "C4-VM.control4.com"
Tran:
Max-Age: 1800
Type: "acme:receiver:ACME-100"
Primary-Proxy: "receiver"
Proxies: "receiver,tuner,tunerXM,dvd"
Manufacturer: "ACME"
Model: "ACME-100"
Driver: "acme_receiver_ac100.c4i"
Config-URL: "http://10.10.0.3/netconf/"
```

5.0 SDDP notification message

An SDDP notification message is similar to a SDDP request message as defined in section 4.1, except that a response message must not be sent. Notifications are generally broadcast to the multicast address so that they can reach all listeners.

The following is an example of a SDDP notification line:

```
NOTIFY Argument SDDP/1.0
```

5.1 NOTIFY notification message

The Argument of these messages defines the type of notification. Valid types are documented in this section. The Tran header is generally not used in notifications, because no response is expected.

5.2 NOTIFY ALIVE

This notification is sent when a device comes online and in periodic intervals. How frequently this notification is sent depends on the Max-Age header. This header defines the number of seconds until an active `NOTIFY ALIVE` notification expires. The device must ensure to send `ALIVE` notifications before they expire.

The following headers are required:

```
      Host
      From
      Type
      Primary-Proxy
      Proxies
      Manufacturer
      Model
      Driver
```

The following header is optional:

```
      Config-URL
```

For example:

```
NOTIFY ALIVE SDDP/1.0
From: "10.10.0.3:1902"
Host: "C4-VM.control4.com"
Max-Age: 1800
Type: "acme:receiver:ACME-100"
Primary-Proxy: "receiver"
Proxies: "receiver,tuner,tunerXM,dvd"
Manufacturer: "ACME"
Model: "ACME-100"
Driver: "acme_receiver_ac100.c4i"
Config-URL: "http://10.10.0.3/netconf/"
```

5.3 NOTIFY IDENTIFY

This notification is sent when the device is being identified. This could be a button press, or any method determined by the manufacturer of the device. The following headers are required:

The following headers are required:

```
      Host
      From
      Type
      Primary-Proxy
      Proxies
      Manufacturer
      Model
      Driver
```

The following header is optional:

```
      Config-URL
```

For example:

```
NOTIFY IDENTIFY SDDP/1.0
From: "10.10.0.3:1902"
Host: "C4-VM.control4.com"
Type: "acme:receiver:ACME-100"
Primary-Proxy: "receiver"
Proxies: "receiver,tuner,tunerXM,dvd"
Manufacturer: "ACME"
Model: "ACME-100"
Driver: "acme_receiver_ac100.c4i"
Config-URL: "http://10.10.0.3/netconf/"
```

5.4 NOTIFY OFFLINE

This notification is sent before a device goes offline. If a device changes its IP address, this notification may be sent before the change, and a `NOTIFY ALIVE` message must be sent immediately after the change.

The following headers are required:

```
      Host
      From
      Type
```

For example:

```
NOTIFY OFFLINE SDDP/1.0
From: "10.10.0.3:1902"
Host: "C4-VM.control4.com"
Type: "acme:receiver:ACME-100"
```

6.0 SDDP response message

The format is similar as defined in RFC 2616 (HTTP) section 6, however sections 6.1.1 and 6.2 do not apply. Response messages are only sent to the sender of the request, never to the multicast address.

```
SDDP/1.0 Status-Code Reason
```

6.1 SDDP response status codes

The following list contains all SDDP response status codes and reasons:

```
200 OK
400 Bad Request
401 Unauthorized
500 Internal Server Error
501 SDDP Version Not Supported
502 Not Implemented
```

An implementation should treat status codes 2xx as success codes, 4xx as client error codes and 5xx as server error codes.