LANHostConfigManagement:1 Service Template Version 1.01

For UPnP™ Version 1.0
Status: Standardized DCP
Date: November 12, 2001

This Standardized DCP has been adopted as a Standardized DCP by the Steering Committee of the UPnP™ Forum, pursuant to Section 2.1(c)(ii) of the UPnP™ Forum Membership Agreement. UPnP™ Forum Members have rights and licenses defined by Section 3 of the UPnP™ Forum Membership Agreement to use and reproduce the Standardized DCP in UPnP™ Compliant Devices. All such use is subject to all of the provisions of the UPnP™ Forum Membership Agreement.

THE UPNP™ FORUM TAKES NO POSITION AS TO WHETHER ANY INTELLECTUAL PROPERTY RIGHTS EXIST IN THE STANDARDIZED DCPS. THE STANDARDIZED DCPS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS". THE UPNP™ FORUM MAKES NO WARRANTIES, EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE STANDARDIZED DCPS, INCLUDING BUT NOT LIMITED TO ALL IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE, OF REASONABLE CARE OR WORKMANLIKE EFFORT, OR RESULTS OR OF LACK OF NEGLIGENCE.

Authors	Company
Mike Blaszczak	Microsoft Corporation
Prakash Iyer	Intel Corporation

Contents

1. OVERVIEW AND SCOPE 4

1.1. CHANGE LOG 4

2. SERVICE MODELING DEFINITIONS 6

2.1. SERVICETYPE 6
2.2. STATE VARIABLES 6
2.2.1. DHCPServerConfigurable 6
2.2.2. DHCPRelay 7
2.2.3. SubnetMask 7
2.2.4. DNSServers 7
2.2.5. DomainName 7
2.2.6. MinAddress 7
2.2.7. MaxAddress 7
2.2.8. IPRouters 7
2.2.9. ReservedAddresses 7
2.2.10. Relationships Between State Variables 7
2.3. EVENTING AND MODERATION 7
2.3.1. Event Model 8
2.4. ACTIONS 8
2.4.1. SetDHCPServerConfigurable 9
2.4.2. GetDHCPServerConfigurable 9
2.4.3. SetDHCPRelay 10
2.4.4. GetDHCPRelay 10
2.4.5. SetSubnetMask 11
2.4.6. GetSubnetMask 11
2.4.7. SetIPRouter 12
2.4.8. DeleteIPRouter 13
2.4.9. GetIPRoutersList 13
2.4.10. SetDomainName 14
2.4.11. GetDomainName 14
2.4.12. SetAddressRange 14
2.4.13. GetAddressRange 15
2.4.14. SetReservedAddress 16
2.4.15. DeleteReservedAddress 16
2.4.16. GetReservedAddresses 17
2.4.17. SetDNSServer 17
2.4.18. DeleteDNSServer 18
2.4.19. GetDNSServers 18
2.4.20. Non-Standard Actions Implemented by a UPnP Vendor 19
2.4.21. Relationships Between Actions 19
2.4.22. Common Error Codes 19
2.5. THEORY OF OPERATION 19

3. XML SERVICE DESCRIPTION 21

4. TEST 26

List of Tables

Table 1: State Variables 6

Table 2: Event Moderation....7

Table 3: Actions 8

Table 4: Arguments for `SetDHCPServerConfigurable`....9

Table 5: Arguments for `GetDHCPServerConfigurable`....10

Table 6: Arguments for `SetDHCPRelay`....10

Table 7: Arguments for `GetDHCPRelay`....11

Table 8: Arguments for `SetSubnetMask` 11

Table 9: Arguments for `GetSubnetMask` 12

Table 10: Arguments for `SetIPRouter` 12

Table 11: Arguments for `DeleteIPRouter` 13

Table 12: Arguments for `GetIPRoutersList` 13

Table 13: Arguments for `SetDomainName` 14

Table 14: Arguments for `GetDomainName` 14

Table 15: Arguments for `SetAddressRange`....15

Table 16: Arguments for `GetAddressRange`....15

Table 17: Arguments for `SetReservedAddress` 16

Table 18: Arguments for `DeleteReservedAddress` 16

Table 19: Arguments for `GetReservedAddresses`....17

Table 20: Arguments for `SetDNSServer`....17

Table 21: Arguments for `DeleteDNSServer`....18

Table 22: Arguments for `GetDNSServers` 18

Table 23: Common Error Codes....19

1. Overview and Scope

This device definition is compliant with the UPnP Architecture version 1.0.

This service definition enables the remote control, monitoring, and configuration of a Dynamic Host Configuration Protocol (DHCP) and DNS server, serving a residential LAN. Variables specified in this service are derived from DHCP options[2] standardized in the Internet Engineering Task Force (IETF).

The ***LANHostConfigManagement*** is an OPTIONAL service specified in
urn:schemas-upnp-org:device:*LANDevice*,
one or more instances of which may be specified under the root device
urn:schemas-upnp-org:device:*InternetGatewayDevice*

NOTES:

- It is assumed that each DHCP server instance defined in the context of a ***LANDevice*** and modeled by this service, manages exactly one subnet. Changes / vendor extensions may be required to the specification if this assumption is not true.
- Currently this service does not model any DNS or Dynamic DNS specific functions.
- This service essentially performs administrative operations on the DHCP server component of an Internet gateway. As such, Out-of-Band mechanisms may be needed to enforce such as administrative mode and provide authentication and/or access control. Implementation details of such mechanisms are outside the scope of this document.

1.1. Change Log

Changes from *LANHostConfigManagement:0.1*

- Updated list of SST variables and actions to comprehensively cover all DHCP and dynamic DNS drafts.

Changes from *LANHostConfigManagement:0.2* as per WC meeting on 10/17/00

- Changed list of required SST variables and corresponding actions. Others could be offered as vendor extensions (a list of possible SST variables is specified in the Theory of Operation section).
- Changed definitions and accompanying text to indicate support for only 1 subnet per DHCP server instance.

Changes from *LANHostConfigManagement:0.3*

- Changed Boolean values to 1 and 0.
- Removed white spaces from XML section.
- Changed default value for empty strings to tags with no element values

Changes from *LANHostConfigManagement:0.31*

- Updated draft status to template design complete (TDC)
- Text clarifications in various sections of the document
- Added references to standard RFCs as a footnote
- Added prefix “New” to formal parameters of actions

Changes from *LANHostConfigManagement:0.32*

- Updated to service template v1.01
- Verified against TDC checklist v1.01
- Renamed X_IPAddress to A_IPAddress

Changes from *LANHostConfigManagement:0.8*

- Removed default values for SST variables and updated XML section accordingly
- Clarified behavior of Set actions if DHCPServerConfigurable is set to 0 in section 2.2.11

[2] Refer to RFCs 2132, 2939 and 3011. Other RFCs or Internet drafts may also be applicable – Attributes from those documents may be implemented as vendor extensions.

Changes from *LANHostConfigManagement:0.81*

- Added XML comment tags to comments text in XML template

Changes from *LANHostConfigManagement:0.82*

- Deleted variable A_IPAddress
- Changed argument A_IPAddress in actions
- Changed AddIPRouter to SetIPRouter and updated the description of the action
- Changed AddDNSServer to SetDNSServer and updated the description of the action
- Changed AddReservedAddress to SetReservedAddress and updated the description of the action

Changes from *LANHostConfigManagement:0.9*

- Version changes for 1.0

Changes from *LANHostConfigManagement:0.99*

- Updated text descriptions for DeleteIPRouter, DeleteReservedAddress and DeleteDNSServer to clarify that these actions only take 1 IP address at a time as action arguments.

Changes from *LANHostConfigManagement:0.991*

- Copyright messages and document status updated.

2. Service Modeling Definitions

2.1. ServiceType

The following service type identifies a service that is compliant with this template:

urn:schemas-upnp-org:service:*LANHostConfigManagement:1*.

2.2. State Variables

Table 1: State Variables

Variable Name	Req. or Opt.[1]	Data Type	Allowed Value [3]	Default Value [2]	Eng. Units
`DHCPServerConfigurable`	R	boolean	1, 0	Not specified	N/A
`DHCPRelay`	R	boolean	1, 0	Not specified	N/A
`SubnetMask`	R	string	IP Address	Not specified	N/A
`DNSServers`	R	string	Undefined	Empty string	N/A
`DomainName`	R	string	<= 21 characters	Empty string	N/A
`MinAddress`	R	string	Undefined	Not specified	N/A
`MaxAddress`	R	string	Undefined	Not specified	N/A
`IPRouters`	R	string	Undefined	Empty string	N/A
`ReservedAddresses`	R	string	Undefined	Empty string	N/A
Non-standard state variables implemented by an UPnP vendor go here.	*X*	*TBD*	*TBD*	*TBD*	*TBD*

[1] R = Required, O = Optional, X = Non-standard.

[2] Values listed in this column are required. To specify standard optional values or to delegate assignment of values to the vendor, you must reference a specific instance of an appropriate table below.

NOTE: Default values are not specified in the DCP. A vendor may however choose to provide default values for SST variables where appropriate.

2.2.1. `DHCPServerConfigurable`

This variable enables the configuration of the DHCP server on the LAN interface. If set to 1, the DHCP server is configurable via actions defined in this service. If 0, the DHCP server configuration cannot be modified.

2.2.2. `DHCPRelay`

This variable indicates if the DHCP server performs the role of a server or a relay on the LAN interface. If this variable is set to 1, the DHCP server is configured to relay DHCP requests.

2.2.3. `SubnetMask`

This variable specifies the client's network subnet mask.

2.2.4. `DNSServers`

This variable specifies a comma-separated list of available DNS servers offered to DHCP clients. An empty string indicates that no servers are being offered.

2.2.5. `DomainName`

This variable sets the Domain name to provide to clients on the residential LAN connected to the LAN interface. An empty string means no domain name suggestion will be made.

2.2.6. `MinAddress`

This variable specifies the first address in the pool to be assigned by the DHCP server on the LAN interface.

2.2.7. `MaxAddress`

This variable specifies the last address in the pool to be assigned by the DHCP server on the LAN interface.

2.2.8. `IPRouters`

This variable specifies a comma-separated list of IP addresses of routers on this subnet. It is also referred to as DefaultGateway.

2.2.9. `ReservedAddresses`

This variable specifies a comma-separated list of addresses marked reserved from the address allocation pool. If the list is non-empty, specified addresses will not be assigned to clients on the residential LAN connected to the LAN interface. When this is empty, no addresses are excluded.

2.2.10.Relationships Between State Variables

If `DHCPServerConfigurable` is set to 0, other Set actions will return error code 501. If `DHCPRelay` is set to 1, the DHCP server will ignore other SST variable settings and forward DHCP requests upstream to another DHCP server.

2.3. Eventing and Moderation

Table 2: Event Moderation

Variable Name	Evented	Moderated Event	Max Event Rate[1]	Logical Combination	Min Delta per Event[2]
DHCPServerConfigurable	No	No	N/A	N/A	N/A
DHCPRelay	No	No	N/A	N/A	N/A
SubnetMask	No	No	N/A	N/A	N/A
DNSServers	No	No	N/A	N/A	N/A
DomainName	No	No	N/A	N/A	N/A
MinAddress	No	No	N/A	N/A	N/A
MaxAddress	No	No	N/A	N/A	N/A
IPRouters	No	No	N/A	N/A	N/A
ReservedAddresses	No	No	N/A	N/A	N/A
Non-standard state variables implemented by an UPnP vendor go here.	*TBD*	*TBD*	*TBD*	*TBD*	*TBD*

[1] Determined by N, where Rate = (Event)/(N secs).
[2] (N) * (allowedValueRange Step).

2.3.1. Event Model

No events are generated by variables on this service. This service sources no events because its intended use is to enable the configuration of a DHCP server, not to replace DHCP as a protocol. When a DHCP server configuration is changed during normal DHCP activity, the server, over the normal course of operation, will communicate that change to the appropriate clients.

2.4. Actions

Immediately following this table is detailed information about these actions, including short descriptions of the actions, the effects of the actions on state variables, and error codes defined by the actions.

Table 3: Actions

Name	Req. or Opt. [1]
SetDHCPServerConfigurable	R
GetDHCPServerConfigurable	R
SetDHCPRelay	R
GetDHCPRelay	R

SetSubnetMask	R
GetSubnetMask	R
SetIPRouter	R
DeleteIPRouter	R
GetIPRoutersList	R
SetDomainName	R
GetDomainName	R
SetAddressRange	R
GetAddressRange	R
SetReservedAddress	R
DeleteReservedAddress	R
GetReservedAddresses	R
SetDNSServer	R
DeleteDNSServer	R
GetDNSServers	R
Non-standard actions implemented by an UPnP vendor go here.	X

[1] R = Required, O = Optional, X = Non-standard.

2.4.1. SetDHCPServerConfigurable

This action enables or disables the configurability of the DHCP server for the ***LANDevice***.

2.4.1.1. Arguments

Table 4: Arguments for SetDHCPServerConfigurable

Argument	Direction	relatedStateVariable
NewDHCPServerConfigurable	IN	DHCPServerConfigurable

2.4.1.2. Dependency on State (if any)

2.4.1.3. Effect on State (if any)

This action enables or disables other actions in this service.

2.4.1.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.2. GetDHCPServerConfigurable

This action retrieves the current setting of a flag indicating if the DHCP server is configurable on the ***LANDevice***.

2.4.2.1. Arguments

Table 5: Arguments for `GetDHCPServerConfigurable`

Argument	Direction	relatedStateVariable
NewDHCPServerConfigurable	*OUT*	DHCPServerConfigurable

2.4.2.2. Dependency on State (if any)

2.4.2.3. Effect on State

None.

2.4.2.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.3. `SetDHCPRelay`

This action enables or disables DHCP relay service for the ***LANDevice***.

2.4.3.1. Arguments

Table 6: Arguments for `SetDHCPRelay`

Argument	Direction	relatedStateVariable
NewDHCPRelay	*IN*	DHCPRelay

2.4.3.2. Dependency on State (if any)

2.4.3.3. Effect on State

This service determines the functioning mode of the DHCP server on the LAN interface. It set to relay mode, other actions have no effect on the service.

2.4.3.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.4. `GetDHCPRelay`

This action retrieves the current setting of a flag indicating if the DHCP server is configured to be a relay on the ***LANDevice***.

2.4.4.1. Arguments

Table 7: Arguments for `GetDHCPRelay`

Argument	Direction	relatedStateVariable
`NewDHCPRelay`	*OUT*	`DHCPRelay`

2.4.4.2. Dependency on State (if any)

2.4.4.3. Effect on State

None.

2.4.4.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.5. `SetSubnetMask`

This action sets the subnet mask of the subnet managed by this DHCP server in the context of the ***LANDevice***.

2.4.5.1. Arguments

Table 8: Arguments for `SetSubnetMask`

Argument	Direction	relatedStateVariable
`NewSubnetMask`	*IN*	`SubnetMask`

2.4.5.2. Dependency on State (if any)

2.4.5.3. Effect on State

Initiates this server’s control of a subnet.

2.4.5.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
701	ValueAlreadySpecified	The value specified in the action is already available in the list and is consequently ignored.

2.4.6. `GetSubnetMask`

This action retrieves the subnet mask of the subnet managed by the DHCP server in the context of the ***LANDevice***.

2.4.6.1. Arguments

Table 9: Arguments for `GetSubnetMask`

Argument	Direction	relatedStateVariable
`NewSubnetMask`	*OUT*	`SubnetMask`

2.4.6.2. Dependency on State (if any)

2.4.6.3. Effect on State

None.

2.4.6.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.7. `SetIPRouter`

This action sets the list of routers specified by `IPRouters`. An empty string argument will clear the list of routers. To modify or append to an existing list, `GetIPRoutersList` should be invoked first to obtain the current list, changed locally by the control point and updated using this set action.

2.4.7.1. Arguments

Table 10: Arguments for `SetIPRouter`

Argument	Direction	relatedStateVariable
`NewIPRouters`	*IN*	`IPRouters`

2.4.7.2. Dependency on State (if any)

2.4.7.3. Effect on State

None, except for the fact that another router is marked as reachable.

2.4.7.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
701	ValueAlreadySpecified	The value specified in the action is already available in the list and is consequently ignored.

2.4.8. `DeleteIPRouter`

This action removes a router from the list of routers specified by `IPRouters`. This action accepts one IP address at a time, NOT a comma-separated list of addresses. This action must be invoked once for every address that needs to be deleted from the list.

2.4.8.1. Arguments

Table 11: Arguments for `DeleteIPRouter`

Argument	Direction	relatedStateVariable
`NewIPRouters`	*IN*	`IPRouters`

2.4.8.2. Dependency on State (if any)

2.4.8.3. Effect on State

None, except that a previously reachable router is now deleted from the list.

2.4.8.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
702	ValueSpecifiedIsInvalid	The specified value is not present in the list

2.4.9. `GetIPRoutersList`

This action retrieves the list of routers specified by `IPRouters.`

2.4.9.1. Arguments

Table 12: Arguments for `GetIPRoutersList`

Argument	Direction	relatedStateVariable
`NewIPRouters`	*OUT*	`IPRouters`

2.4.9.2. Dependency on State (if any)

2.4.9.3. Effect on State

None.

2.4.9.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.10.SetDomainName

This action sets the domain name this server will recommend for its clients.

2.4.10.1.Arguments

Table 13: Arguments for SetDomainName

Argument	Direction	relatedStateVariable
NewDomainName	*IN*	DomainName

2.4.10.2.Dependency on State (if any)

2.4.10.3.Effect on State

This action changes the domain for hosts served by the DHCP server on the LAN interface.

2.4.10.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.11.GetDomainName

This action retrieves the domain name this server will recommend for its clients.

2.4.11.1.Arguments

Table 14: Arguments for GetDomainName

Argument	Direction	relatedStateVariable
NewDomainName	*OUT*	DomainName

2.4.11.2.Dependency on State (if any)

2.4.11.3.Effect on State

None.

2.4.11.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.12.SetAddressRange

This action sets the range of addresses this server will assign on the selected subnet.

2.4.12.1.Arguments

Table 15: Arguments for `SetAddressRange`

Argument	Direction	relatedStateVariable
`NewMinAddress`	*IN*	`MinAddress`
`NewMaxAddress`	*IN*	`MaxAddress`

2.4.12.2.Dependency on State (if any)

2.4.12.3.Effect on State

This action changes the values of MinAddress and MaxAddress, which in turn define the range of addresses assigned by the DHCP server.

2.4.12.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.13.`GetAddressRange`

This action retrieves the range of addresses this server will assign on the selected subnet.

2.4.13.1.Arguments

Table 16: Arguments for `GetAddressRange`

Argument	Direction	relatedStateVariable
`NewMinAddress`	*OUT*	`MinAddress`
`NewMaxAddress`	*OUT*	`MaxAddress`

2.4.13.2.Dependency on State (if any)

2.4.13.3.Effect on State

None.

2.4.13.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.14.SetReservedAddress

This action sets the list of IP addresses to be reserved (excluded from assignment) by the DHCP server serving nodes attached to the ***LANDevice***. An empty string argument will clear the list of reserved IP addresses. To modify or append to an existing list, `GetReservedAddresses` should be invoked first to obtain the current list, changed locally by the control point and updated using this set action.

2.4.14.1.Arguments

Table 17: Arguments for `SetReservedAddress`

Argument	Direction	relatedStateVariable
NewReservedAddresses	*IN*	ReservedAddresses

2.4.14.2.Dependency on State (if any)

2.4.14.3.Effect on State

None.

2.4.14.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
701	ValueAlreadySpecified	The value specified in the action is already available in the list and is consequently ignored.

2.4.15.DeleteReservedAddress

This action removes an IP address from the list of reserved addresses at `ReservedAddresses`. This action accepts one IP address at a time, NOT a comma-separated list of addresses. This action must be invoked once for every address that needs to be deleted from the list.

2.4.15.1.Arguments

Table 18: Arguments for `DeleteReservedAddress`

Argument	Direction	relatedStateVariable
NewReservedAddresses	*IN*	ReservedAddresses

2.4.15.2.Dependency on State (if any)

2.4.15.3.Effect on State

None.

2.4.15.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
702	ValueSpecifiedIsInvalid	The specified value is not present in the list

2.4.16.`GetReservedAddresses`

This action retrieves the comma-separated list of `ReservedAddresses.`

2.4.16.1.Arguments

Table 19: Arguments for `GetReservedAddresses`

Argument	Direction	relatedStateVariable
`NewReservedAddresses`	*OUT*	`ReservedAddresses`

2.4.16.2.Dependency on State (if any)

2.4.16.3.Effect on State

None.

2.4.16.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.17.`SetDNSServer`

This action sets the list of DNS servers available to clients on the residential LAN. An empty string argument will clear the list of DNS servers. To modify or append to an existing list, `GetDNSServers` should be invoked first to obtain the current list, changed locally by the control point and updated using this set action.

2.4.17.1.Arguments

Table 20: Arguments for `SetDNSServer`

Argument	Direction	relatedStateVariable
`NewDNSServers`	*IN*	`DNSServers`

2.4.17.2.Dependency on State (if any)

2.4.17.3.Effect on State

None.

2.4.17.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
701	ValueAlreadySpecified	The value specified in the action is already available in the list and is consequently ignored.

2.4.18.`DeleteDNSServer`

This action removes a DNS server from the list of DNS servers in the variable `DNSServers`. This action accepts one IP address at a time, NOT a comma-separated list of addresses. This action must be invoked once for every address that needs to be deleted from the list.

2.4.18.1.Arguments

Table 21: Arguments for `DeleteDNSServer`

Argument	Direction	relatedStateVariable
`NewDNSServers`	*IN*	`DNSServers`

2.4.18.2.Dependency on State (if any)

2.4.18.3.Effect on State

None.

2.4.18.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22
702	ValueSpecifiedIsInvalid	The specified value is not present in the list

2.4.19.`GetDNSServers`

This action retrieves the comma-separated list of DNS servers in the variable `DNSServers.`

2.4.19.1.Arguments

Table 22: Arguments for `GetDNSServers`

Argument	Direction	relatedStateVariable
`NewDNSServers`	*OUT*	`DNSServers`

2.4.19.2.Dependency on State (if any)

2.4.19.3.Effect on State

None.

2.4.19.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See section 2.4.22
501	Action Failed	See section 2.4.22

2.4.20.Non-Standard Actions Implemented by a UPnP Vendor

To facilitate certification, non-standard actions implemented by UPnP vendors should be included in this service template. The UPnP Device Architecture lists naming requirements for non-standard actions (see the section on Description).

2.4.21.Relationships Between Actions

If `DHCPServerConfigurable` is set to 0, other actions return appropriate error status.

2.4.22.Common Error Codes

The following table lists error codes common to actions for this service type. If an action results in multiple errors, the most specific error should be returned.

Table 23: Common Error Codes

errorCode	errorDescription	Description
401	Invalid Action	See UPnP Device Architecture section on Control.
402	Invalid Args	See UPnP Device Architecture section on Control.
404	Invalid Var	See UPnP Device Architecture section on Control.
501	Action Failed	See UPnP Device Architecture section on Control.
600-699	TBD	Common action errors. Defined by UPnP Forum Technical Committee.
701-799		Common action errors defined by the UPnP Forum working committees.
800-899	*TBD*	*(Specified by UPnP vendor.)*

2.5. Theory of Operation

An ***InternetGatewayDevice*** may support a DHCP server. The settings on the DHCP server (per ***LANDevice***) may or may not be configurable via UPnP. This service template is optional, but must be supported by vendors that implement a DHCP server that is UPnP-configurable on ***InternetGatewayDevice***. For example, supporting configurability of the DHCP server enables an ISP to configure / reconfigure a DHCP server as needed, from within the residential LAN. Other security and/or access control mechanisms may be needed to support some of these actions – these are currently outside the scope of the UPnP framework and this DCP.

DHCP server settings are managed on a per LAN basis. If multiple LAN segments are bridged, the DHCP server settings apply to all of the bridged segments.

In this version of the SCP, there is no support for events on specific DHCP option settings. This feature may be introduced in a future version.

The SST only defines a partial set of possible DHCP options. Others may be added as vendor extensions.

NOTE: If `DHCPRelay` is on, the DHCP server SHOULD forward the any relay requests using the same routing decisions as for any other IP packet from the residential LAN.

3. XML Service Description

```
<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <actionList>
    <action>
    <name>SetDHCPServerConfigurable</name>
      <argumentList>
        <argument>
          <name>NewDHCPServerConfigurable</name>
          <direction>in</direction>
     <relatedStateVariable>DHCPServerConfigurable</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetDHCPServerConfigurable</name>
      <argumentList>
        <argument>
          <name>NewDHCPServerConfigurable</name>
          <direction>out</direction>
     <relatedStateVariable>DHCPServerConfigurable</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetDHCPRelay</name>
      <argumentList>
        <argument>
          <name>NewDHCPRelay</name>
          <direction>in</direction>
          <relatedStateVariable>DHCPRelay</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetDHCPRelay</name>
      <argumentList>
        <argument>
          <name>NewDHCPRelay</name>
          <direction>out</direction>
          <relatedStateVariable>DHCPRelay</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetSubnetMask</name>
      <argumentList>
        <argument>
          <name>NewSubnetMask</name>
          <direction>in</direction>
          <relatedStateVariable>SubnetMask</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetSubnetMask</name>
      <argumentList>
        <argument>
          <name>NewSubnetMask</name>
          <direction>out</direction>
          <relatedStateVariable>SubnetMask</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetIPRouter</name>
      <argumentList>
        <argument>
          <name>NewIPRouters</name>
          <direction>in</direction>
          <relatedStateVariable>IPRouters</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>DeleteIPRouter</name>
      <argumentList>
        <argument>
          <name>NewIPRouters</name>
          <direction>in</direction>
          <relatedStateVariable>IPRouters</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetIPRoutersList</name>
      <argumentList>
        <argument>
          <name>NewIPRouters</name>
          <direction>out</direction>
          <relatedStateVariable>IPRouters</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetDomainName</name>
      <argumentList>
        <argument>
          <name>NewDomainName</name>
          <direction>in</direction>
          <relatedStateVariable>DomainName</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetDomainName</name>
```

```
      <argumentList>
        <argument>
          <name>NewDomainName</name>
          <direction>out</direction>
          <relatedStateVariable>DomainName</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetAddressRange</name>
      <argumentList>
        <argument>
          <name>NewMinAddress</name>
          <direction>in</direction>
          <relatedStateVariable>MinAddress</relatedStateVariable>
        </argument>
        <argument>
          <name>NewMaxAddress</name>
          <direction>in</direction>
          <relatedStateVariable>MaxAddress</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetAddressRange</name>
      <argumentList>
        <argument>
          <name>NewMinAddress</name>
          <direction>out</direction>
          <relatedStateVariable>MinAddress</relatedStateVariable>
        </argument>
        <argument>
          <name>NewMaxAddress</name>
          <direction>out</direction>
          <relatedStateVariable>MaxAddress</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetReservedAddress</name>
      <argumentList>
        <argument>
          <name>NewReservedAddresses</name>
          <direction>in</direction>
          <relatedStateVariable>ReservedAddresses</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>DeleteReservedAddress</name>
      <argumentList>
        <argument>
          <name>NewReservedAddresses</name>
          <direction>in</direction>
          <relatedStateVariable>ReservedAddresses</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetReservedAddresses</name>
      <argumentList>
        <argument>
          <name>NewReservedAddresses</name>
          <direction>out</direction>
          <relatedStateVariable>ReservedAddresses</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>SetDNSServer</name>
      <argumentList>
        <argument>
          <name>NewDNSServers</name>
          <direction>in</direction>
          <relatedStateVariable>DNSServers</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>DeleteDNSServer</name>
      <argumentList>
        <argument>
          <name>NewDNSServers</name>
          <direction>in</direction>
          <relatedStateVariable>DNSServers</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetDNSServers</name>
      <argumentList>
        <argument>
          <name>NewDNSServers</name>
          <direction>out</direction>
          <relatedStateVariable>DNSServers</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <!-- Declarations for other actions added by UPnP vendor (if any) go
here -->
  </actionList>
  <serviceStateTable>
    <stateVariable sendEvents="no">
      <name>DHCPServerConfigurable</name>
      <dataType>boolean</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>DHCPRelay</name>
      <dataType>boolean</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>SubnetMask</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>IPRouters</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>DNSServers</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>DomainName</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>MinAddress</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>MaxAddress</name>
      <dataType>string</dataType>
    </stateVariable>
    <stateVariable sendEvents="no">
      <name>ReservedAddresses</name>
      <dataType>string</dataType>
    </stateVariable>
    <!-- Declarations for other state variables added by UPnP vendor (if any) go here -->
  </serviceStateTable>
</scpd>
```

4. Test

No semantic tests have been defined for this service.

Change History

Change Log for Version 1.0 (10-4-00)

- Revised the Title Page to call out V1.0 of the Service Template
- Changed to be consistent with Sample Designs released to the Technical Committee
- Service State Table: Variable Descriptions removed from the table and are listed in specific sections following the table.
- Actions: Reformatted the information contained in the Action Table:
    - Added overview entry point.
    - Added an Action Summary Table to specify Required or Optional
    - Added enumerated sections to specify each actions: Arguments, Effect on State, and Errors.