Cisco Edge 340 系列软件配置指南， 1.0 版

2013 年 11 月 14 日

Cisco Systems, Inc.
www.cisco.com

思科在全球设有 200 多个办事处。
地址、电话号码和传真号码
在思科网站上列出，网址为：
www.cisco.com/go/offices。

文本部件号：OL-29491-01

Cisco Edge 340 系列软件配置指南， 1.0 版

前言

本文档介绍如何在网络中配置 Cisco Edge 340 系列。

本指南不介绍如何安装 Cisco Edge 340 系列。有关如何安装 Cisco Edge 340 系列的信息，请参阅与您的设备相关的硬件安装指南。

约定

本出版物采用如下约定表示指令和信息：

对于命令描述：

- 命令和关键字以**粗体**文本表示。
- 由您提供值的参数以*斜体*表示。
- 方括号 ([ ]) 表示可选元素。
- 花括号 ({ }) 将必选项组合在一起，竖线 (|) 用于分隔可替代元素。
- 方括号内的花括号和竖线 ([{ | }]) 表示可选元素中的某个必选项。

对于交互示例：

- 终端会话和系统显示内容以屏幕字体显示。
- 输入的信息以**粗体屏幕**字体显示。
- 非打印字符（如密码或制表符）放在尖括号 (< >) 中。

注释、注意事项和警告使用如下约定和符号：

注 表示*读者需要记录*。注释中包含有用的建议或包含对本手册中所没有的材料的引用。

注意事项 表示*读者应当小心*。在这种情况下，您的操作可能会导致设备损坏或数据丢失。

警告 **重要安全性说明**

此警告符号表示存在危险。您目前所处情形有可能遭受身体伤害。在操作任何设备之前，请务必意识到触电危险并熟悉标准工作程序，以免发生事故。请根据每个警告结尾处的声明号来查找此设备随附的安全警告的翻译文本。声明 1071

请妥善保存这些说明

相关出版物

- *Cisco Edge 340 系列安装指南*
- *Cisco Edge 340 系列发行说明*

获取文档和提交服务请求

有关获取文档、提交服务请求和收集更多信息的相关内容，请参阅*思科产品文档更新*，网址为：http://www.cisco.com/en/US/docs/general/whatsnew/whatsnew.html。

通过 RSS 源的方式订阅*思科产品文档更新*（其中包括所有新的和修改过的思科技术文档），并将相关内容通过阅读器应用程序直接发送至您的桌面。 RSS 源是一种免费服务。

Cisco Edge 340 系列概述

Cisco Edge 340 系列概述

Cisco Edge 340 系列是用于进行纵向和企业互连空间部署的下一代设备。这些部署可以提供丰富的媒体支持功能和特定纵向应用支持。该系列具有丰富的连接功能，可通过以太网 LAN 上行链路、无线接入、富媒体和应用计算，在数字互连空间体验中支持所有重要组件。它还是一个开放式应用平台，您可以自定义该平台以启用纵向解决方案。

数字媒体播放器

Cisco Edge 340 系列设备可用作以下解决方案的下一代数字媒体播放器：

- 数字媒体标牌
- 体育与娱乐
- iServcies

在数字媒体标牌解决方案中，Cisco Edge 340 系列设备可作为数字媒体播放器 (DMP) 提供视频和音频播放器的功能。此外，它还可提供较高的计算能力和用于此类内容的本地存储功能。

应用支持

Cisco Edge 340 系列可支持纵向应用并提供计算能力、存储和简单的连接功能。

家庭自动化网关

Cisco Edge 340 系列可以用作在家中部署的家庭服务网关，位于住宅网关设备后面，通过 WiFi 提供家庭自动化服务并连接和控制智能电器。

Cisco Edge 340 系列的功能和应用支持

Cisco Edge 340 系列可提供以下功能和应用支持：

- 视频会议
- 视频播放器
- 文档查看器
- VLC 播放器
- VPN 客户端支持
- SNMP 支持
- 屏幕捕获

有关这些功能和应用的详细信息，请参阅适合您的版本的 *Cisco Edge 340 系列发行说明*。

管理和配置支持

Cisco Edge 340 系列支持以下管理和配置方法：

- 本地 CLI—CLISH
- Web GUI
- HTTP API

本地 CLI—CLISH

您可以使用 CLISH 配置用于本地 CLI 配置的 Cisco Edge 340 系列。CLI 仅使用 Cisco Edge 340 系列专用的命令。尽管语法与 Cisco IOS CLI 类似，但命令与 Cisco IOS 命令*并不兼容*。有关详细信息，请参阅第 2 章，“ 配置本地 CLI - CLISH”。

Web GUI

您可以使用 Web GUI 配置 Cisco Edge 340 系列并以本地或远程方式监控其状态。有关详细信息，请参阅第 3 章，“ 配置 Web GUI”。

HTTP API

可以在 Cisco Edge 340 系列上以本地或远程方式运行应用，来通过 HTTP API 管理该设备。设备管理包括配置设备、监控 Cisco Edge 340 系列的状态以及安装和升级软件。有关详细信息，请参阅第 4 章，“ 配置 HTTP API”。

系统安装和升级

Cisco Edge 340 系列支持以下安装和升级类型：

- USB 模式安装和升级
- 远程升级
- BIOS 升级

USB 模式安装和升级

Cisco Edge 340 系列软件发布了一个自解压安装程序。文件名是 Cisco-Edge-*version*-i386-DVD.bin。这是一个执行文件，可帮助您自动执行安装。当您执行自解压安装程序时，会将相关的安装文件解压到 Cisco Edge 340 系列的硬盘驱动器，并在内部 USB 中创建 livecd。然后系统从内部 USB（也称为出厂模式）启动并自动执行安装。

如果系统已将内部 USB 创建为 livecd，您可以按 Cisco Edge 340 系列的前面板上的 factory 模式针孔进入 factory 模式，之后安装程序将自动执行。

注　通常，内部 USB 在出厂时创建为 livecd。执行自解压安装程序将会覆盖原始 livecd 并新建一个 livecd。

远程升级

如果您有自解压安装程序的下载地址，可以使用 Web GUI 对 Cisco Edge 340 系列执行远程升级。如果您选择执行删除升级，系统将从您提供的 URL 自动下载并执行自解压安装程序，以完成安装。

BIOS 升级

在 Linux 环境中，只能通过手动安装程序包并执行命令来执行 BIOS 升级。BIOS 是系统的关键组成部分，因此如果 BIOS 崩溃，就无法进行软件恢复。为确保 BIOS 升级成功，请确保外部电源始终连接，*不要*在升级过程中执行任何关机开机操作。

配置本地 CLI - CLISH

本章包含以下部分：

配置指南

CLISH 用于本地 CLI 配置，您可以在其中配置 Cisco Edge 340 系列。CLI 仅使用 Cisco Edge 340 系列专用的命令。尽管语法与 Cisco IOS CLI 类似，但这些命令与 Cisco IOS 命令 *不兼容*。

您可以在以下两种模式下使用 CLISH：

- 用户模式 - 以普通用户身份登录到 Cisco Edge 340 系列时，将进入用户模式。要进入特权模式，请输入 **enable** 命令，然后输入根用户的密码。
- 全局（特权）模式 - 以根用户身份登录到 Cisco Edge 340 系列时，不必输入 **enable** 命令即可直接进入全局模式。

使用 CLI 可以配置下面的设备设置：

- 基本设备设置 - 主机名、 MAC 地址、蓝牙设置、密码、网络时间协议 (NTP) 服务器和设备语言
- 以太网接口设置 - 状态、速度和服务质量 (QoS)
- 无线接口设置 - 状态、无线电、无线模式、信道、无线分离、传输功率、 Wi-Fi 多媒体 (WMM) 和高级无线设置
- 服务集标识符 (SSID) 安全设置 - 广播、身份验证和加密

在使用 CLISH 时，请遵循以下配置指南：

- 在 PC 的命令提示符下输入 **ssh** *username@ip-address* 或 **ssh root@***ip-address*，在欢迎屏幕中输入密码。输入 **mgtcmd** 命令以启动 CLISH 进程。
- 如果您以普通用户身份登录，请输入 **enable** 命令以及根用户的密码以切换到全局模式。
- 使用 **configure terminal** 全局命令启动 Cisco Edge 配置。使用 **exit** 全局命令结束 Cisco Edge 配置文件。

注　如果您以普通用户身份登录到 Cisco Edge 340 系列，但希望以**根**用户身份进入 CLISH，请使用 Linux 命令 **su -** ,其中 - 意味着将普通用户切换到根用户并使用根用户的环境变量。如果进入特权模式之后超过 10 分钟仍没有任何活动，您将自动退出特权模式。请注意，提示符 > 和 #; > 意味着用户模式，# 意味着特权模式。

- 从系统配置模式中，您可以进入以下配置模式：
  - 以太网配置模式

    使用 **interface** 系统配置命令进入此模式。使用 **exit** 全局配置命令返回系统配置模式。
  - WiFi AP 接口配置模式

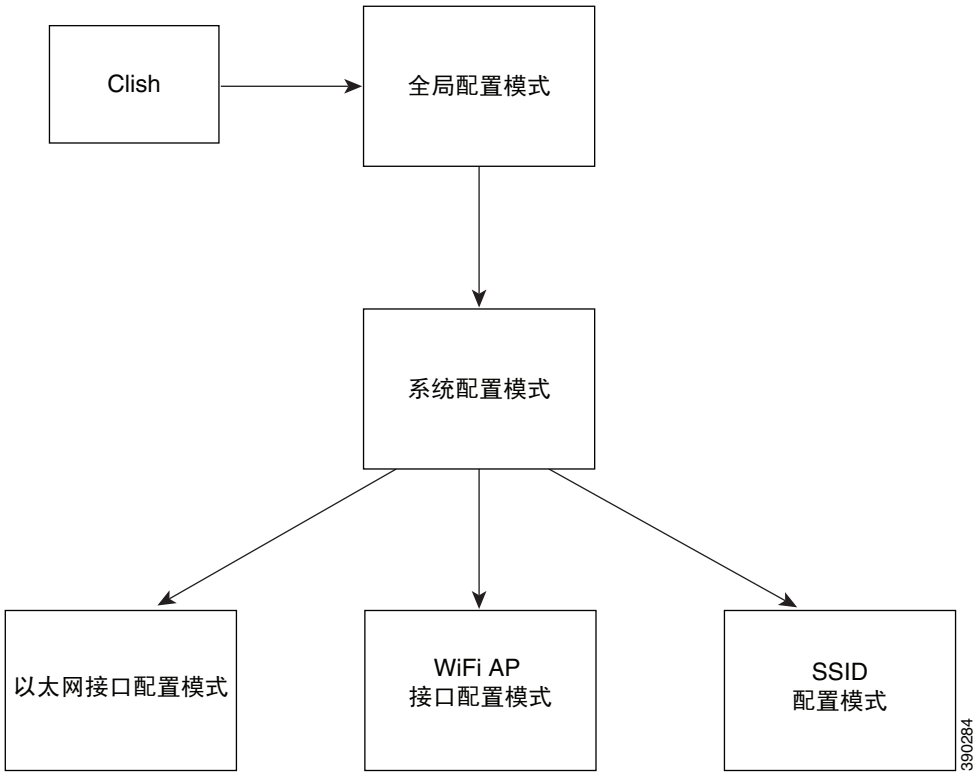
    使用 **interface** 系统配置命令进入此模式。我们建议在配置任何无线设置之前，先使用 **wireless-mode** WiFi 配置命令来设置 802.11 无线模式。使用 **exit** 全局配置命令返回系统配置模式。
  - SSID 配置模式

    使用 **ssid** 系统配置命令进入此模式。使用 **exit** 全局配置命令返回系统配置模式。
- 必须以小写字母输入所有命令。参数可以包括大写字母。
- 如果存在配置冲突，则使用最近的配置。在下例中，不广播 SSID：

```
ssid NEWAP1
    broadcast ssid on
    broadcast ssid off
exit
```

图 2-1 显示 CLISH 功能结构的逻辑顺序。

图 2-1　　CLISH 功能结构的逻辑顺序



命令参考

本节包含以下模式的命令：

- 用户配置模式命令
- 全局配置模式命令
- 系统配置模式命令
- 以太网接口配置模式命令
- WiFi AP 接口配置模式命令
- SSID 配置模式
- show 命令

注　我们仅为不能自我解释的命令提供了语法说明、命令默认模式、命令模式、使用指南和示例。

用户配置模式命令

本节包含用户配置模式命令。这些命令提供表 2-1 中所显示的基本功能。

表 2-1　用户配置模式命令

命令	功能
enable	进入全局配置模式。
exit	从 CLI 中退出。
help	显示交互式帮助系统的说明。
ping	诊断基本网络连接并验证远程设备是否可访问。
show	显示运行系统信息。
traceroute	将路由数据包跟踪打印到网络主机。

enable

要进入全局配置模式，请在用户配置模式下使用 **enable** 命令。

enable

命令模式　用户配置

使用指南　使用 **enable** 命令并输入**根**用户的密码可切换到全局配置模式。

exit

要退出所处的配置模式，请在任意配置模式中使用 **exit** 命令。

exit

命令模式

用户配置
全局配置
系统配置
以太网接口配置
WiFi AP 接口配置
SSID 配置

使用指南

使用 **exit** 来离开某个配置模式并返回到以前的配置模式。

help

要显示帮助系统的简短说明，请在用户配置模式下使用 **help** 命令。

help

命令模式

用户配置

全局配置

使用指南

help 命令显示可用命令及其简短说明的列表。要显示特定命令的其他详细信息，输入命令名称，紧接着输入 -? 选项。

ping

要在 Cisco Edge 340 系列设备上诊断基本的网络连接，请在用户配置模式或全局配置模式下使用 **ping** 命令。

ping {[ip | ipv6 | arp] ***hostname*** **|** ***ip_address*****}**

语法说明

IP	向网络主机发送互联网控制消息协议 (ICMP) IPv4 消息（默认设置）。
ipv6	向网络主机发送 ICMP IPv6 消息。
ARP	向邻近主机发送 ARP 请求。
hostname	要 ping 的主机名。
ip_address	要 ping 的 IP 地址。

命令模式

用户配置

全局配置

使用指南

ping 命令向某个地址发送回应请求数据包并等待回复。Ping 输出可帮助您评估路径到主机的可靠性、该路径中的延迟以及主机是否可以访问或者主机是否正常工作。

traceroute

要发现数据包在到达其目标地址时实际通过的路由，请在用户配置模式或全局配置模式下使用 **traceroute** 命令。

traceroute [*protocol*] ***destination*** [[**resolve**] **source** *ip_address* | **interface** *interface_name*]

语法说明

protocol	（可选）协议； IP 或 IPv6。如果未指定，将根据软件对目标格式的检查确定协议参数。默认协议是 IP。
destination	要跟踪的路由的目标地址或主机名。
resolve	指定要解析主机名。
source *ip_address*	指定源 IP 地址。
interface *interface_name*	指定源接口。

命令模式

用户配置

全局配置

使用指南

traceroute 命令通过利用在数据报超过其跃点限制值时由设备生成的错误消息来工作。

traceroute 命令首先发送跃点限制为 1 的探测数据报。在探测数据报中包括跃点限制 1 会导致相邻的设备放弃该探测数据报并发回错误消息。 **traceroute** 命令会发送多个探测（这些探测的跃点限制逐渐增加），并显示每个探测的往返时间。

traceroute 命令一次发出一个探测。每个传出数据包可能导致一个或多个错误消息。`time-exceeded` 错误消息指示中间设备已经发现并放弃了探测。`destination unreachable` 错误消息指示目标节点已经接收到探测，但由于数据包的跳数限制达到了 0 而放弃了该探测。如果计时器在收到响应之前结束， **traceroute** 命令会打印星号 (*)。

traceroute 命令会在以下情况下终止：目标响应时；超过跃点限制时；用户中断对转义序列的跟踪时。默认情况下，要调用转义序列，请同时按下并松开 **Ctrl**、 **Shift** 和 **6** 键，然后按 **X** 键。

全局配置模式命令

本节包含全局配置模式命令。这些命令提供表 2-2 中所显示的特权模式功能。

表 2-2　　全局配置模式命令

命令	功能
configure terminal	启动 Cisco Edge 配置文件，并进入全局配置模式。
copy running-config startup-config	将运行配置保存为启动配置文件。
exit	退出全局配置模式。
export	将 running-config 或 startup-config 导出到目标路径。
help	显示交互式帮助系统的说明。
import	将配置文件导入 running-config 或 startup-config。
ping	诊断基本网络连接并验证远程设备是否可访问。
reboot	关机并重新冷启动。
restore	恢复默认出厂配置。
show	显示运行系统信息。
traceroute	将路由数据包跟踪打印到网络主机。

configure terminal

要进入全局配置模式，请在全局配置模式下使用 **configure terminal**。

configure terminal

命令模式　全局配置

copy running-config startup-config

要将运行配置保存为启动配置文件，请在全局配置模式下使用 **copy running-config startup-config** 命令。

copy running-config startup-config

命令模式　全局配置

export

要将配置文件导出到 USB 存储或本地目录，请在全局配置模式下使用 **export-config** 命令。

export startup-config to *destination*

语法说明

destination	您要将配置文件导出到的目标。目标可以是 USB 或本地目录。

命令模式

全局配置

使用指南

您可以将配置文件导出到 USB 或本地目录。如果您选择将配置文件导出到 USB，则会自动检测和挂载配置，并将其导出到 USB。

import

要从 USB 或本地目录导入配置文件，请在全局配置模式下使用 **import-config** 命令。

import startup-config from *source*

语法说明

source	要导入的配置文件所在的位置。源可以是 USB 或本地目录。

命令模式

全局配置

使用指南

您可以从 USB 或本地目录导入配置文件。如果选择从 USB 导入配置文件，则会自动检测和挂载配置，并将其从 USB 中导入。

reboot

要暂停和执行冷启动，请在全局配置模式下使用 **reboot** 命令。

reboot

命令模式　全局配置

restore

要恢复默认出厂配置，请在全局配置模式下使用 **restore** 命令。

restore factory-default

命令模式　　全局配置模式

show

要显示运行系统信息，请在全局配置模式下使用 **show** 命令。

show

命令模式

用户配置

全局配置

系统配置模式命令

本节包含系统配置模式命令。这些命令提供表 2-3 中所显示的系统配置模式功能。

表 2-3　　系统配置模式命令

命令	功能
auto-login	允许或拒绝自动登录系统。
bluetooth	启用或禁用设备上的蓝牙。
clock	配置时区。
display	配置在连接了两个显示器时，HDMI 和 VGA 之间的关系。
do	在全局配置模式或其他配置模式或子模式下执行用户 EXEC 或特权 EXEC 命令。
exit	退出系统配置模式。
hdmi	配置 HDMI 分辨率和旋转。
hostname	配置设备的主机名。
interface	进入以太网接口配置模式以配置千兆以太网接口，或进入 WiFi AP 接口配置模式以配置无线接口。
language support	配置设备的语言。
log	配置日志大小。
monitor	启用或禁用 HDMI 或 VGA。
ntp	配置由设备使用的 NTP 服务器。
proxy-server	配置代理服务器。
ssh	配置 SSH 用户。
ssid	配置 SSID 名称，并进入 SSID 配置模式以配置接入点的安全设置。
vga	配置 VGA 分辨率和旋转。
wifi-mode	设置 WiFi 模式。

auto-login

要配置系统的自动登录，请在系统配置模式下使用 **auto-login** 命令。

auto-login {**enable** | **disable**}

语法说明

enable	允许自动登录系统。
disable	禁止自动登录系统。

命令默认

自动登录处于禁用状态。

命令模式

系统配置

bluetooth

要在 Cisco Edge 340 系列设备上启用或禁用蓝牙，请在系统配置模式下使用 **bluetooth** 命令。

bluetooth {**on** | **off**}

命令默认　蓝牙处于开启 (on) 状态。

命令模式　系统配置

clock

要设置仅供显示的时区，请在系统配置模式下使用 **clock** 命令。

clock timezone *timezone*

语法说明

timezone	Continent 或 ocean。有效值为 **Africa**、 **America**、 **Antarctica**、 **Arctic**、 **Asia**、 **Atlantic**、 **Australia**、 **Europe**、 **Indian**、 **Mideast** 和 **Pacific**。

命令模式

系统配置

display

要配置两个显示器之间的关系，请在系统配置模式下使用 **display** 命令。

display type {**hdmi** | **vga**} **relation** *type* {**hdmi** | **vga**}

语法说明

type	选择显示器类型：**HDMI** 或 **VGA**。
relation *type*	配置两个显示器之间的关系。有效值为 **same-as**、**right**、**left**、**below** 和 **above**。

命令模式

系统配置

do

要在全局配置模式或其他配置模式下执行用户配置或全局配置命令，请在任何配置模式下使用 **do** 命令。

do *command*

语法说明		
	command	要执行的用户配置命令或全局配置命令。

命令默认　不从配置模式执行用户配置命令或全局配置命令。

命令模式　所有配置模式。

使用指南　使用此命令可以在配置路由设备时执行用户配置命令或全局配置命令（如 **show**、 **copy** 和 **export**)。在执行命令后，系统将返回到您使用的配置模式。

hdmi

要配置高清多媒体 (HDMI) 分辨率或旋转，请在系统配置模式下使用 **hdmi** 命令。

hdmi {resolution *resolution_value* | **rotation** *rotation_value}*

语法说明

resolution_value	要设置的分辨率，采用 *xx@yy* 形式。
rotation_value	要设置的旋转。有效值为 **normal**、 **right**、 **inverted** 和 **left**。

命令模式

系统配置

hostname

要配置 Cisco Edge 340 系列设备的主机名，请在系统配置模式下使用 **hostname** 命令。

hostname *name*

语法说明

name	您分配给设备的名称。

命令模式

系统配置

使用指南

更改主机名需要重启。

interface

要进入以太网接口配置模式以配置千兆以太网接口，或者要进入 WiFi AP 接口配置模式以配置无线接口，请在系统配置模式下使用 **interface** 命令。

interface {ethernet ge | wireless bvi1}

语法说明

ethernet ge	配置千兆以太网接口。
wireless bvi1	配置无线接口。

命令模式

系统配置

使用指南

使用 **interface** 命令进入以太网接口配置模式或 WiFi AP 接口配置模式。

相关命令

使用 **exit** 命令离开以太网接口配置模式或 WiFi AP 接口配置模式。

第 2-37 页上的表 2-4 列出了以太网接口配置模式命令。

第 2-46 页上的表 2-5 列出了 WiFi AP 接口配置模式命令。

language support

要配置设备语言，请在系统配置模式下使用 **language support** 命令。

language support *language_value*

语法说明

language_value	设备的语言。有效值为 **zh_CH.utf8**、 **en_US.utf8**、 **ko_KR.utf8** 和 **ja_JP.utf8**。

命令默认

默认值为英语 (en_US.utf8)。

命令模式

系统配置

使用指南

更改语言需要重启。

log

要设置日志大小，请在系统配置模式下使用 **log 命令**。

log size *value*

语法说明

size *value*	设置日志大小。默认单位是 MB。有效范围为 1 至 10000。默认大小为 10 MB。

命令模式

系统配置

monitor

要启用或禁用显示器类型（HDMI 或 VGA），请在系统配置模式下使用 **monitor** 命令。

monitor type {**hdmi** | **vga**} {**on** | **off**}

语法说明

type	设置显示器类型。
hdmi	将显示器类型设置为 HDMI。
vga	将显示器类型设置为 VGA。
on	启用显示器。
off	禁用显示器。

命令模式

系统配置

no

要移除命令的配置或将命令设置为默认值，请在系统配置模式下使用 **no** 命令。

no

命令模式　系统配置

ntp

要配置由 Cisco Edge 340 系列设备使用的网络时间协议 (NTP) 服务器，请在系统配置模式下使用 **ntp** 命令。

ntp {**refresh** {**on** | **off**} | **server** *ip_address*}

语法说明

refresh	配置 NTP 服务器的自动同步。
on	启用 NTP 服务器的自动同步。
off	禁用 NTP 服务器的自动同步。
server *ip_address*	配置 NTP 服务器的 IP 地址。

命令模式

系统配置

proxy-server

要配置代理服务器，请在系统配置模式下使用 **proxy-server** 命令。

proxy-server *server* **[*type*] [port *port_number*]**

语法说明

server	代理服务器的主机名或 IP 地址。
type	（可选）代理服务器的类型。有效值为 **no_for**、**all**、**http**、**ftp** 和 **https**。
port *port_number*	（可选）指定代理端口号。范围为 0 至 65535。

命令模式

系统配置

ssh

要配置安全外壳 (SSH) 用户，请在系统配置模式下使用 **ssh** 命令。

ssh {**add** *user* | **delete** *user*}

语法说明

add *user*	添加 SSH 用户。
delete *user*	删除 SSH 用户。

命令模式

系统配置

ssid

要设置服务集标识符 (SSID) 名称，并进入 SSID 配置模式以配置设备接入点的安全设置，请在系统配置模式下使用 **ssid** 命令。

ssid *ssid*

语法说明

ssid	接入点的 SSID 名称。名称可以包含除 '、\、"、?、=、, 和空格以外的所有 ASCII 字符。

命令默认

默认的 SSID 名称为 CISCO_EDGE。

命令模式

系统配置

相关命令

使用 **exit** 命令可离开 SSID 配置模式。

第 2-77 页上的表 2-6 列出了 SSID 配置模式命令。

vga

要配置视频图形阵列 (VGA) 分辨率或旋转，请在系统配置模式下使用 **vga** 命令。

vga {resolution *resolution* | **rotation** *rotation*}

语法说明

resolution	要设置的分辨率，采用 *xx@yy* 形式。
rotation	要设置的旋转。有效值为 **normal**、 **right**、 **inverted** 和 **left**。

命令模式

系统配置

wifi-mode

要设置 Cisco Edge 340 系列设备的 WiFi 模式，请在全局配置模式下使用 **wifi-mode** 命令。

wifi-mode {ap | client | off}

语法说明

ap	在重新启动后将 WiFi 模式设置为接入点 (AP)。
client	在重新启动后将 WiFi 模式设置为 client。
off	将 WiFi 模型设置为 off。

命令模式

系统配置

使用指南

如果选择 AP 模式，Cisco Edge 340 系列设备将立即在 AP 模式下工作，并且只有特定于 AP 模式的命令可见。如果选择 client 模式，Cisco Edge 340 系列设备将立即在 client 模式下工作，并且只有特定于 client 模式的命令可见。

以太网接口配置模式命令

本节包含以太网接口配置模式命令。这些命令提供表 2-4 中所显示的以太网接口配置模式功能。

表 2-4　以太网接口配置模式命令

命令	功能
do	从全局配置模式或其他配置模式执行用户配置命令或全局配置命令。
duplex	配置千兆以太网 (GE) 接口的双工模式。
exit	退出以太网接口配置模式。
ip address	配置接口的 IP 地址。
ip default-gateway	配置默认网关。
ip name-server	配置 DNS 服务器。
ipv6 address	配置接口的 IPv6 地址。
ipv6 default-gateway	配置 IPv6 默认网关。
ipv6 name-server	配置 IPv6 DNS 服务器。
speed	配置 GE 接口的速度。

duplex

要配置千兆以太网 (GE) 接口的双工模式，请在以太网接口配置模式下使用 **duplex** 命令。

duplex {auto | half | full}

语法说明

auto	配置自动感知双工模式。
half	配置半双工模式。
full	配置全双工模式。

默认值

默认值为自动感知双工模式。

ip address

要设置接口的 IP 地址，请在以太网接口配置模式下使用 **ip address** 命令。

ip address {***dhcp*** | ***ip_address}***

语法说明

dhcp	通过动态主机配置协议 (DHCP) 协商的 IP 地址。
ip_address	接口的 IP 地址。

命令默认

默认值为 dhcp。

ipv6 address

要设置接口的 IPv6 地址，请在以太网接口配置模式下使用 **ipv6 address** 命令。

ipv6 address {***dhcp*** | ***ipv6_address***}

语法说明

dhcp	通过 DHCP 协商的 IPv6 地址。
ipv6_address	接口的 IPv6 地址。

命令默认

默认值为 dhcp。

ip default-gateway

要指定 Cisco Edge 340 系列设备的默认网关，请在以太网接口配置模式下使用 **ip default-gateway** 命令。

ip default-gateway ***ip_address***

语法说明

ip_address	默认网关的 IP 地址。

ipv6 default-gateway

要指定 Cisco Edge 340 系列设备的 IPv6 默认网关，请在以太网接口配置模式下使用 **ipv6 default-gateway** 命令。

ipv6 default-gateway ***ipv6_address***

语法说明

ip_address	默认网关的 IPv6 地址。

ip name-server

要指定域名系统 (DNS) 服务器，请在以太网接口配置模式下使用 **ip name-server** 命令。

ip name-server ***ip_address***

语法说明

ip_address	DNS 服务器的 IP 地址。

ipv6 name-server

要指定 IPv6 DNS 服务器，请在以太网接口配置模式下使用 **ipv6 name-server** 命令。

ipv6 name-server ***ipv6_address***

语法说明

ip_address	DNS 服务器的 IPv6 地址。

speed

要配置接口的速度，请在以太网配置模式下使用 **speed** 命令。

speed {**auto** | **10** | **100** | **1000**}

语法说明

auto	配置自动感知速度。
10	配置 10 Mbps 速度。
100	配置 100 Mbps 速度。
1000	配置 1000 Mbps 速度和全双工模式。

命令默认

默认值为 auto。

WiFi AP 接口配置模式命令

本节包含 WiFi AP 接口配置模式命令。这些命令提供表 2-5 中所显示的 WiFi AP 接口配置模式功能。

表 2-5　　WiFi AP 接口配置模式命令

命令	功能
aggregation-msdu	启用或禁用聚合 MAC 服务数据单元 (MSDU)。
ap-isolation	配置连接同一 SSID 的客户端的无线分离。
apsd	配置接入点的 Wi-Fi 多媒体 (WMM) 省电模式。
auto-block	启用或禁用自动块。
ba-decline	允许或禁止拒绝 ba 请求。
beacon-interval	配置接入点的信标间隔。
bg-protection	配置接入点的 CTS-to-self 保护。
channel bandwidth	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时的信道带宽。
channel number	配置接入点的信道号（用于设置频率）。
data-beacon-rate	配置接入点的传输流量指示消息 (DTIM) 间隔。
do	从全局配置模式或其他配置模式执行用户配置命令或全局配置命令。
exit	退出 WiFi AP 接口配置模式。
extension channel	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时用于扩展信道或辅助信道的控制端频段。
frag-threshold	配置 Frag 阈值。
guard-interval	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时数据包之间的时段。
igmp-snoop	启用或禁用互联网组管理协议 (IGMP) 监视。
mcs	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时高吞吐量调制和编码方案 (MCS) 的速率。
multicast-mcs	配置组播帧的高吞吐量 MCS 速率。
multicast-phy-mode	配置组播帧上的 PHY 模式。
operating-mode	配置接入点在 802.11n 模式下工作时的绿地模式或混合模式。
packet aggregation	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时的聚合 MAC 服务数据单元 (A-MSDU) 数据包聚合。
rdg	配置接入点在 802.11n 模式或 802.11n 混合模式下工作时的反向授权 (RDG)。
rts-threshold	设置 RTS 阈值。
short-slot	配置接入点在 802.11g 模式或 802.11g 混合模式下工作时的短碰撞槽时间。
stbc	配置空时块码 (STBC)。
transmit burst	配置接入点的传输脉冲（Tx 脉冲）。

表 2-5　WiFi AP 接口配置模式命令 （续）

命令	功能
transmit preamble	配置接入点的前导信号。
transmit power	配置接入点无线电传输其无线信号的功率。
wireless-mode	配置接入点的 802.11 无线模式。
wmm	配置接入点的 Wi-Fi 多媒体 (WMM)。

aggregation-msdu

要启用或禁用 MAC 服务数据单元 (MSDU) 聚合，请在 WiFi AP 接口配置模式下使用 **aggregation-msdu** 命令。

aggregation-msdu {on | off}

语法说明

on	启用聚合 MSDU。
off	禁用聚合 MSDU。

命令模式

WiFi AP 接口配置

ap-isolation

要为连接到同一服务集标识符 (SSID) 的客户端配置无线分离，请在 WiFi AP 接口配置模式下使用 **ap-isolation** 命令。

ap-isolation {on | off}

语法说明

on	启用无线分离。这避免了连接到同一 SSID 的无线客户端相互通信。
off	禁用无线分离。连接到同一 SSID 的无线客户端可以相互通信。

命令默认

禁用无线分离。

相关命令

WiFi AP 接口配置。

apsd

要配置接入点的 Wi-Fi 多媒体 (WMM) 省电模式，请在 WiFi AP 接口配置模式下使用 **apsd** 命令。

apsd {on | off}

语法说明

on	启用 WMM 省电模式。
off	禁用 WMM 省电模式。

命令默认

禁用 WMM 省电模式。

命令模式

WiFi AP 接口配置

使用指南

仅当启用了 WMM 时才可以配置 **apsd** 命令。

相关命令

使用 **wmm** 命令启用 WMM。

auto-block

要配置自动块，请在 WiFi AP 接口配置模式下使用 **auto-block** 命令。

auto-block {on | off}

语法说明

on	启用自动块。
off	禁用自动块。

相关命令

WiFi AP 接口配置

ba-decline

要启用或禁用 BA 请求的拒绝任务，请在 WiFi AP 接口配置模式下使用 **ba-decline** 命令。

ba-decline {**on** | **off**}

语法说明

on	启用 BA 请求的拒绝任务。
off	禁用 BA 请求的拒绝任务。

命令模式

WiFi AP 接口配置

beacon-interval

要配置接入点的信标间隔，请在 WiFi AP 接口配置模式下使用 **beacon-interval** 命令。

beacon-interval ***interval***

语法说明		
	interval	要用来配置信标间隔的时段。范围为 20 至 1000 毫秒。默认值为 100 毫秒。

命令默认

默认时段为 100 毫秒。

命令模式

WiFi AP 接口配置

使用指南

默认设置应能够满足大多数网络需求。

配置长间隔可以：

- 提高接入点的吞吐量性能。
- 减少客户端的发现时间并降低漫游效率。
- 减少客户端的能耗。

配置短间隔可以：

- 将客户端的发现时间减至最短并提高漫游效率
- 降低接入点的吞吐性能。
- 增加客户端的能耗。

bg-protection

要配置接入点的 CTS-to-self 保护，请在 WiFi AP 接口配置模式下使用 **bg-protection** 命令。

bg-protection {auto | on | off}

注　此命令适用于 802.11b/g 混合模式、 802.11n/g 混合模式和 802.11b/g/n 混合模式。

语法说明

auto	配置自动选择 CTS-to-self 保护。
on	启用 CTS-to-self 保护。
off	禁用 CTS-to-self 保护。

命令默认　默认值是自动选择 CTS-to-self 保护。

命令模式　WiFi AP 接口配置

使用指南　CTS-to-self 保护可最大限度减少混合模式环境中客户端之间的冲突，但会降低吞吐性能。

channel bandwidth

要配置接入点在 802.11n 模式下工作时的信道带宽，请在 WiFi AP 接口配置模式下使用 **channel bandwidth** 命令。

channel bandwidth {**20** | **20/40**}

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

20	配置 20-MHz 信道带宽。
20/40	配置自动选择 20-MHz 或 40-MHz 信道带宽。

命令默认

默认值为自动选择 20-MHz 或 40-MHz 信道带宽。

命令模式

WiFi AP 接口配置

使用指南

默认设置应能够满足大多数网络需求。

40-MHz 信道可为 802.11n 客户端提供较高的吞吐性能。

802.11b 和 802.11g 客户端只能在 20-MHz 信道下工作。

相关命令

channel bandwidth 命令的设置会影响 **mcs** 命令的选项。

channel number

要配置信道号（用于为接入点设置频率），请在 WiFi AP 接口配置模式下使用 **channel number** 命令。

channel number {**auto** | *number*}

语法说明

auto	配置自动选择信道号。
number	信道号。值为 1 到 13、 149、 153、 157、 161 和 165。默认值为 6。

命令默认

默认信道号为 6。

命令模式

WiFi AP 接口配置

使用指南

我们建议使用默认信道号或自动选择信道号，并且仅在网络中发生干扰时才更改信道号。

如果必须更改信道号，请根据您所在的位置使用以下号码：

- 中国和欧洲：1 至 13
- 美国：1 至 11

data-beacon-rate

要配置接入点的传输流量指示消息 (DTIM) 时间间隔，请在 WiFi AP 接口配置中使用 **data-beacon-rate** 命令。

data-beacon-rate *rate*

语法说明

rate	范围为 1 至 255 毫秒。默认值为 1 毫秒。

命令默认

默认速率是 1 毫秒。

命令模式

WiFi AP 接口配置

使用指南

DTIM 间隔是信标间隔的倍数。在更改 DTIM 的时间间隔之前，请考虑网络中客户端的类型：笔记本电脑使用短间隔时性能较好，而移动电话使用长间隔时性能较好。

长间隔可让客户端节省能源，但可能会延迟多播和广播通信。

短间隔可以缩短多播和广播通信的传输时间，但可能会增加客户端的能耗。

相关命令

beacon-interval 命令的设置会影响 **data-beacon-rate** 命令。

extension channel

要配置接入点在 802.11n 模式下工作时用于扩展信道或辅助信道的控制边带，请在 WiFi AP 接口配置模式下使用 **extension channel** 命令。

extension channel {**upper** | **lower**}

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

upper	配置较高扩展信道。
lower	配置较低扩展信道。

命令默认

配置较低扩展信道。

命令模式

WiFi AP 接口配置

使用指南

此命令仅在配置 40-MHz 信道带宽时才生效。

当主信道号处于较低范围时（例如，在 1 至 4 范围内），请使用较高扩展信道。

当主信道号处于较高范围时（例如，在 10 至 13 范围内），请使用较低扩展信道。

当主信道号处于中间范围时（例如，在 5 至 9 范围内），请使用较高扩展信道或较低扩展信道。

相关命令

使用 **channel bandwidth** 命令配置信道带宽。

使用 **channel number** 命令配置主信道号。

frag-threshold

要配置 Frag 阈值，请在 WiFi AP 接口配置模式下使用 **frag-threshold** 命令。

frag-threshold ***value***

语法说明

value	配置 Frag 阈值。范围为 256 至 2346。

命令模式

WiFi AP 接口配置

guard-interval

要配置接入点在 802.11n 模式下工作时数据包之间的保护间隔时段，请在 WiFi AP 接口配置模式下使用 **guard-interval** 命令。

guard-interval {**400** | **800**}

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

400	将短保护间隔配置为 400 毫微秒 (ns)。
800	将长保护间隔配置为 800 毫微秒。

命令默认

默认值为 400 毫微秒。

命令模式

WiFi AP 接口配置

使用指南

使用 400 毫微秒间隔可增加 802.11n 客户端的吞吐性能，但可能会导致某些数据包错误和多路径干扰。

使用 800 毫微秒间隔可将数据包错误和多路径干扰减至最低，但会降低 802.11n 客户端吞吐性能。

相关命令

guard-interval 命令的设置影响 **mcs** 命令的选项。

igmp-snoop

要启用或禁用互联网组管理协议 (IGMP) 在无线接口上的监听，请在 WiFi AP 接口配置模式下使用 **igmp-snoop** 命令。

igmp-snoop {**on** | **off**}

语法说明

on	IGMP 监听处于打开状态。
off	IGMP 监听处于关闭状态。

命令默认

IGMP 监听处于关闭状态。

命令模式

WiFi AP 接口配置。

mcs

要配置接入点在 802.11n 模式下工作时的高吞吐量调制和编码方案 (MCS) 速率，请在 WiFi AP 接口配置模式下使用 **mcs** 命令。

mcs *index_number*

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

index_number	范围为 0 至 15 和 33（自动选择）。

命令默认

默认值为 33（自动配置速率）。

命令模式

WiFi AP 接口配置

使用指南

此表根据 MCS、保护间隔和信道带宽显示了 MCS 索引号及其潜在数据速率 (Mbps)。

	保护间隔 800 毫微秒		**保护间隔 400 毫微秒**	
索引号	**20-MHz 信道带宽**	**40-MHz 信道带宽**	**20-MHz 信道带宽**	**40-MHz 信道带宽**
0	6.5	13.5	7 2/9	15
1	13	27	14 4/9	30
2	19.5	40.5	21 2/3	45
3	26	54	28 8/9	60
4	39	81	43 1/3	90
5	52	109	57 5/9	120
6	58.5	121.5	65	135
11	52	108	57 7/9	120
12	78	162	86 2/3	180
13	104	216	115 5/9	240
14	117	243	130	270
15	130	270	144 4/9	300
33	配置自动选择 MCS 索引号。			

我们建议使用自动选择 MCS 索引号。仅当网络中客户端接收的信号强度指示 (RSSI) 可以支持选择的 MCS 索引号时，才将 MCS 索引更改为固定的号。

相关命令

channel bandwidth 命令的设置影响 **mcs** 命令的选项。

guard-interval 命令的设置影响 **mcs** 命令的选项。

multicast-mcs

要配置接入点在 802.11n 模式下工作时多播帧的高吞吐量调制和编码方案 (MCS) 速率，请在 WiFi AP 接口配置模式下使用 **multicast-mcs** 命令。

multicast-mcs *index_number*

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

index_number	范围为 0 至 15。

命令默认

默认值为 2。

使用指南

此表根据 MCS、保护间隔和信道带宽显示了 MCS 索引号及其潜在数据速率 (Mbps)。

	保护间隔 800 毫微秒		保护间隔 400 毫微秒	
索引号	**20-MHz 信道带宽**	**40-MHz 信道带宽**	**20-MHz 信道带宽**	**40-MHz 信道带宽**
0	6.5	13.5	7 2/9	15
1	13	27	14 4/9	30
2	19.5	40.5	21 2/3	45
3	26	54	28 8/9	60
4	39	81	43 1/3	90
5	52	109	57 5/9	120
6	58.5	121.5	65	135
7	65	135	72 2/9	152.5
8	13	27	14 4/9	30
9	26	54	28 8/9	60
10	39	81	43 1/3	90
11	52	108	57 7/9	120
12	78	162	86 2/3	180
13	104	216	115 5/9	240
14	117	243	130	270
15	130	270	144 4/9	300

multicast-phy-mode

要配置接入点在 802.11n 模式下工作时多播帧上的 PHY 模式，请在 WiFi AP 接口配置模式下使用 **multicast-phy-mode** 命令。

multicast-phy-mode {**0** | **1** | **2** | **3**}

语法说明

0	指定已禁用模式。
1	指定补码键控 (CCK) (802.11b)。
2	指定正交频分复用 (OFDM) (802.11g)。这是默认值。
3	指定 HTMIX (802.11b/g/n)。

命令默认

默认值为 2。

命令模式

WiFi AP 接口配置

operating-mode

要配置接入点在 802.11n 模式下工作时的绿地模式或混合模式，请在 WiFi AP 接口配置模式下使用 **operating-mode** 命令。

operating-mode {**greenfield** | **mixed**}

注　此命令适用于 802.11n 模式。

语法说明

greenfield	配置绿地模式，这可以提高 802.11n 吞吐性能，但会阻止覆盖区域中存在的 802.11b 和 802.11g 客户端识别 802.11n 通信。
mixed	配置混合模式，这可以让覆盖区域的 802.11b 和 802.11g 客户端识别 802.11n 通信。这是默认值。

命令默认

默认值为 **mixed**。

命令模式

WiFi AP 接口配置

使用指南

如果覆盖区域仅有 802.11n 客户端，请使用绿地模式。如果当 802.11b、802.11g 和 802.11n 客户端在同一覆盖区域内共存时使用绿地模式，可能会发生数据包冲突。

当 802.11b、 802.11g 和 802.11n 客户端在同一覆盖区域共存时，请使用混合模式。

packet aggregation

要配置接入点在 802.11n 模式下工作时的聚合 MAC 服务数据单元 (A-MSDU) 数据包聚合，请在 WiFi AP 接口配置模式下使用 **packet aggregation** 命令。

packet aggregation {on | off}

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

on	启用数据包聚合。
off	禁用数据包聚合。

命令默认

数据包聚合处于关闭 (off) 状态。

命令模式

WiFi AP 接口配置

使用指南

如果网络通信主要包含数据，则启用数据包聚合。

如果网络通信主要包含语音、视频或其他多媒体通信，则禁用数据包聚合。

rdg

要配置接入点在 802.11n 模式下工作时的反向授权 (RDG)，请在 WiFi AP 接口配置模式下使用 **rdg** 命令。

rdg {on | off}

注　此命令适用于 802.11n 模式或 802.11n 混合模式。

语法说明

on	启用 RDG。
off	禁用 RDG。

命令默认

禁用 RDG。

命令模式

WiFi AP 接口配置

使用指南

在启用 RDG 时，保留信道传输机会的发射器允许接收器按保留的方向发送数据包。在禁用 RDG 时，数据包在信道传输机会保留期间只能按一个方向传输。

启用 RDG 可提高 802.11n 通信的吞吐性能。

rts-threshold

要配置请求发送 (RTS) 阈值，请在 WiFi AP 接口配置模式下使用 **rts-threshold** 命令。

rts-threshold *value*

语法说明

value	设置 RTS 阈值。范围为 1 至 2347。

命令模式

WiFi AP 接口配置

short-slot

要配置接入点在 802.11g 模式或 802.11g 混合模式下工作时的短碰撞槽时间，请在 WiFi AP 接口配置模式下使用 **short-slot** 命令。

short-slot {**on** | **off**}

注　此命令适用于 802.11g 模式或 802.11g 混合模式。

语法说明

on	启用短碰撞槽时间。
off	禁用短碰撞槽时间。

命令默认

启用短碰撞槽时间。

命令模式

WiFi AP 接口配置

使用指南

启用短碰撞槽时间可以提高 802.11g 客户端的吞吐性能。

如果网络中大多数为 802.11b 客户端，请禁用短碰撞槽时间。

stbc

要配置空时块码 (STBC)，请在 WiFi AP 接口配置模式下使用 **stbc** 命令。

stbc {on | off}

语法说明

on	启用 STBC。
off	禁用 STBC。

相关命令

WiFi AP 接口配置

transmit burst

要配置接入点的传输脉冲（Tx 脉冲），请在 WiFi AP 接口配置模式下使用 **transmit burst** 命令。

transmit burst {**on** | **off**}

语法说明

on	启用 Tx 脉冲。
off	禁用 Tx 脉冲。

命令默认

启用 Tx 脉冲。

命令模式

WiFi AP 接口配置

使用指南

启用 Tx 脉冲可以提高吞吐性能。

如果发现网络中存在无线干扰，请禁用 Tx 脉冲。

transmit preamble

要配置接入点的前导信号，请在 WiFi AP 接口配置模式下使用 **transmit preamble** 命令。

transmit preamble {**long** | **short** | **auto**}

语法说明

long	配置长前导。
short	配置短前导。
auto	配置自动选择前导信号。

命令默认

默认值是长前导。

命令模式

WiFi AP 接口配置

使用指南

使用长前导设置可以与使用 1 和 2 Mb/s 的传统 802.11 系统兼容。

配置短前导设置可以提高吞吐性能。

transmit power

要配置接入点无线电传输其无线信号的功率，请在 WiFi AP 接口配置模式下使用 **transmit power** 命令。

transmit power *percentage*

语法说明

percentage	传输功率百分比。范围为 1 至 100。

命令默认

默认值为 100%。

命令模式

WiFi AP 接口配置

使用指南

要远距离传输无线信号，请使用 100% 设置。

要近距离传输无线信号，例如当所有客户端都在一个小房间时，请降低该百分比。

wireless-mode

要配置接入点的 802.11 无线模式，请在 WiFi AP 接口配置模式下使用 **wireless-mode** 命令。

wireless-mode {**0** | **1** | **2** | **4** | **6** | **7** | **8** | **9** | **11**|}

语法说明

0	配置 802.11b/g 混合模式。
1	配置 802.11b 模式。
2	配置仅限 5GHz 的 802.11a 模式。
4	配置 802.11g 模式。
6	配置仅限 2GHz 的 802.11n 模式。
7	配置 802.11n/g 混合模式。
8	配置仅限 5GHz 的 802.11a/n 混合模式。
9	配置 802.11b/g/n 混合模式。
11	配置仅限 5GHz 的 802.11n 模式。

命令默认

默认值为 802.11b/g/n 混合模式。

命令模式

WiFi AP 接口配置

使用指南

802.11b/g 混合模式 - 如果网络中的设备支持 802.11b 和 802.11g，请选择此模式。

802.11b 模式 - 如果无线网络中的所有设备仅支持 802.11b，请选择此模式。

仅限 5GHz 的 802.11a 模式 - 如果无线网络中的所有设备仅在 5GHz 频段支持 802.11a，则选择此模式。

802.11g 模式 - 如果无线网络中的所有设备仅支持 802.11g，请选择此模式。

仅限 2GHz 的 802.11n 模式 - 如果无线网络中的所有设备仅在 2GHz 频段支持 802.11n，则选择此模式。

802.11b/g/n 混合模式 - 如果网络中的设备支持 802.11b、 802.11g 和 802.11n，请选择此模式。

802.11b/g 混合模式 - 如果网络中的设备支持 802.11b 和 802.11g，请选择此模式。

wmm

要配置接入点的 Wi-Fi 多媒体 (WMM)，请在 WiFi AP 接口配置模式下使用 **wmm** 命令。

wmm {**on** | **off**}

语法说明

on	启用 WMM。
off	禁用 WMM。

命令默认

禁用 WMM。

命令模式

WiFi AP 接口配置

使用指南

WMM 为无线通信提供 QoS。如果存在许多混合媒体通信（语音、视频、数据），请启用 WMM。

相关命令

使用 **apsd** 命令可配置 WMM 省电模式。

SSID 配置模式

本节包含服务集标识符 (SSID) 配置模式命令。这些命令提供表 2-5 中所显示的 SSID 配置模式功能。

表 2-6　SSID 配置命令

命令	功能
broadcast ssid	启用或禁用对服务集标识符 (SSID) 名称的广播。
do	在全局配置模式或其他配置模式或子模式下执行用户 EXEC 或特权 EXEC 命令。
encryption mode（open、shared 或 WEP 配置）	配置接入点的开放、共享、 Wi-Fi 保护接入 (WPA)、WPA1WPA2、 WPA2、 WPA2PSK、 WPAPSK 和 WPAPSKWPA2PSK 身份验证和关联加密。
encryption mode（WPA 配置）	
exit	退出 SSID 配置模式。
no	删除命令的配置或将命令设置为默认值。
radius-server	配置 RADIUS 服务器的名称。
wep-keyid	配置有线等效保密 (WEP) 加密。

注　SSID 的配置将在退出 SSID 配置模式后生效。

broadcast ssid

要启用或禁用 SSID 名称的广播，请在 SSID 配置模式下使用 **broadcast ssid** 命令。

broadcast ssid {**on** | **off**}

语法说明

on	启用 SSID 名称的广播。
off	禁用 SSID 名称的广播。

命令默认

广播 SSID。

命令模式

SSID 配置

使用指南

禁用 SSID 广播以增强安全性。只有知道 SSID 的无线客户端可以连接到接入点。

启用 SSID 广播可提供更广泛的可用性和更方便的接入。

encryption mode(open、 shared 或 WEP 配置)

要配置接入点的开放、共享或有线等效加密 (WEP) 身份验证和关联加密，请在 SSID 配置模式下使用 **encryption mode** 命令。

encryption mode {**open** | **shared**} **type** {**none** | **wep** {**key** {**1** | **2** | **3** | **4**} {**hex** *number* | **ascii** *phrase*}}}

语法说明

open	配置无需身份验证的开放接入。
shared	配置带有共享密钥的身份验证。
none	配置无加密。
wep	配置 WEP 加密。
key 1 **key 2** **key 3** **key 4**	配置 WEP 加密的密钥号。 (只能使用四个密钥之一)
hex *number*	配置带有十六进制密钥的身份验证或带有十六进制密钥的身份验证和加密： • 在选择 **none** 关键字时，请配置带有十六进制密钥的身份验证。 • 在选择 **wep** 关键字时，请配置带有十六进制密钥的身份验证和加密。 对于 *number*，请输入 10 或 26 个十六进制数字。
ascii *phrase*	配置带有密码的身份验证或带有密码的身份验证和加密： • 在选择 **none** 关键字时，请配置带有密码的身份验证。 • 在选择 **wep** 关键字时，请配置带有密码的身份验证和加密。 对于 *phrase*，输入 5 或 13 个字母数字字符。支持短划线 (-) 和下划线 (_) 字符。

命令默认

默认值为开放接入和无加密。

命令模式

SSID 配置

使用指南

对于无加密的共享接入， WEP 十六进制数或密码仅用于身份验证。

对于使用 WEP 加密的共享接入， WEP 十六进制数或密码同时用于身份验证和加密。

示例

下例显示如何使用 key 3 和密码 3uifsfis-_0r5 配置共享身份验证和 WEP 加密：

```
encryption mode shared type wep key 3 ascii 3uifsfis-_0r5
```

encryption mode（WPA 配置）

要配置接入点的 Wi-Fi 保护接入 (WPA) 身份验证和关联加密，请在 SSID 配置模式下使用 **encryption mode** 命令。

encryption mode {**wpapsk** | **wpa2psk** | **wpapskwpa2psk**} **type** {**tkip** | **aes** | **tkipaes**} **pass-phrase** *phrase*

语法说明

wpapsk	配置带有预共享密钥 (PSK) 身份验证的 WPA。
wpa2psk	配置带有 PSK 身份验证的 WPA2。
wpapskwpa2psk	配置带有 PSK 身份验证的合并 WPA 和 WPA2。
tkip	配置临时密钥完整性协议 (TKIP) 加密。
aes	配置高级加密标准 (AES) 的加密。
tkipaes	配置合并 TKIP 和 AES 加密。
pass-phrase *phrase*	配置密码。对于 *phrase*，输入 8 至 63 个字母数字字符。支持短划线 (-) 和下划线 (_) 字符。

命令默认

默认值为开放接入和无加密。

命令模式

SSID 配置

示例

下例显示如何使用密码 safE478_Ty33Yep- 配置带有合并 TKIP 和 AES 加密的合并 WPA 和 WPA2 身份验证：

```
encryption mode wpapskwpa2psk type tkipaes pass-phrase safE478_Ty33Yep-
```

加密模式 (802.1x)

要配置接入点的 Wi-Fi 保护接入 (WPA) 身份验证和关联加密，请在 SSID 配置模式下使用 **encryption mode** 命令。

encryption mode {**wpa** | **wpa2** | **wpa1wpa2**} **type** {**tkip** | **aes** | **tkipaes**}

注　加密模式 (802.1x) 应该与 RADIUS 服务器一起使用。

语法说明

wpa	配置带有 802.1x 身份验证的 WPA。
wpa2	配置带有 802.1x 身份验证的 WPA2。
wpa1wpa2	配置带有 802.1x 身份验证的合并 WPA 和 WPA2。
tkip	配置临时密钥完整性协议 (TKIP) 加密。
aes	配置高级加密标准 (AES) 的加密。
tkipaes	配置合并 TKIP 和 AES 加密。

命令默认

默认模式是 wpa2psk 接入、 tkipaes 加密，密码为 Cisco123。

命令模式

SSID 配置

示例

下例显示如何使用 802.1x 身份验证方法配置带有合并 TKIP 和 AES 加密的合并 WPA 和 WPA2 身份验证：

```
encryption mode wpa1wpa2 type tkipaes
```

radius-server

要配置 RADIUS 服务器的相关信息，请在 SSID 配置模式下使用 **radius-server**。

radius-server ***hostname*** [**auth-port** ***port_number***] [**key** *secret*]

语法说明

hostname	RADIUS 服务器的主机名或 IP 地址。
auth-port	指定 RADIUS 服务器的身份验证端口号。
port_number	RADIUS 服务器的身份验证端口号。范围为 0 至 65535。默认值为 1812。
key	指定 RADIUS 服务器上身份验证服务的密码。
secret	RADIUS 服务器上身份验证服务的密码。

命令默认

port_number 的默认值是 1812。

secret 的默认值为空。

命令模式

SSID 配置

示例

下例显示如何配置 RADIUS 服务器的相关信息：

```
radius-server 192.168.1.1 auth-port 1812 key pass1234
```

wep-keyid

要配置有效的有线等效保密 (WEP) 加密，请在 SSID 配置模式下使用 **wep-keyid**。

wep-keyid ***wep_key***

语法说明

wep_key	有效的 WEP 密钥。范围为 1 至 4。

命令默认

wep_key 的默认值为 1。

命令模式

SSID 配置

使用指南

WEP 加密支持四组密码。您应该使用 **wep-keyid** 命令来配置哪组密码生效。例如，四组 WEP 密码为 **cisco1**、 **cisco2**、 **cisco3** 和 **cisco4**。命令 **wep-keyid 3** 意味着密码 **cisco3** 将生效。

show 命令

用户配置模式

在用户配置模式下使用以下 **show** 命令可显示 Cisco Edge 340 系列设备上的配置：

- **show cpu** - 显示 CPU 信息。
- **show mac** - 显示 MAC 地址。
- **show memory** - 显示内存使用情况信息。
- **show mount** - 显示挂载信息。
- **show os-build-time** - 显示发行版生成时间。
- **show os-version** - 显示发行版本。
- **show os-install-time** - 显示发行版安装时间。
- **show storage** - 显示存储信息。

全局配置模式

在全局配置模式下使用以下 **show** 命令可以显示 Cisco Edge 340 系列设备上的配置：

- **show all-running-config** - 显示有关正在运行的配置的所有信息。
- **show all-startup-config** - 显示有关启动配置的所有信息。
- **show running-config** - 显示 RAM 中所保存的配置。
- **show startup-config** - 显示数据库中所保存的配置。
- **show bluetooth** - 显示蓝牙信息。
- **show hdmi dev-name** - 显示所连接的 HDMI 显示器的设备名称。
- **show hdmi current-resolution** - 显示所连接的 HDMI 显示器的当前分辨率。
- **show hdmi support-resolution** - 显示所连接的 HDMI 显示器支持的分辨率。
- **show hostname** - 显示主机名。
- **show ip interface** - 显示为 IP 配置的接口的状态。
- **show log-size** - 显示日志大小。
- **show monitor-full** - 显示当前所有的显示器信息。
- **show ssid** - 显示 AP 无线 SSID 设置。
- **show wifi-mode** - 显示 WiFi 模式。
- **show vga dev-name** - 显示所连接的 VGA 显示器的设备名称。
- **show vga current-resolution** - 显示所连接的 VGA 显示器的当前分辨率。
- **show vga support-resolution** - 显示所连接的 VGA 显示器支持的分辨率。

配置 Web GUI

使用基于 Web 的 GUI，可以配置 Cisco Edge 340 系列设备并以本地或远程方式监控 Cisco Edge 340 系列的状态。

要使用基于 Web 的 GUI 配置 Cisco Edge 340 系列设备，请按照下列各节介绍的步骤操作：

登录 Web GUI

在 https://[*Cisco Edge 340 的 IP 地址*] 访问基于 Web 的 GUI，然后输入用户名 admin 和 admin 帐户密码以本地或远程方式登录 Web 门户。

图 3-1　　登录页

语言设置

登录后，选择要用于 Web GUI 的语言。在 GUI 顶部，从下拉列表中选择语言(图 3-2)。

图 3-2　　Web GUI 的语言设置

系统配置

登录 Web GUI 后，会看到系统配置窗口。您可以通过单击左侧窗格中的对应链接来配置基本信息、账户、分辨率、日期与时间、系统日志以及吐核。

配置基本信息

单击左侧窗格中的“ **基本**”可配置主机名、启用或禁用“ 自动登录”、选择区域设置以及查看模式、操作系统版本和 RPM 版本（ 图 3-3 ）。单击“ **应用**”可保存更改，单击“ **重置**”可恢复默认值。

注　更改区域设置需要重新启动设备才能使更改生效。

图 3-3　基本信息

配置帐户信息

单击左侧窗格中的“ **账户**”可更改用户，例如根用户（图 3-4）的密码。单击“ **应用**”可保存更改，单击“ **重置**”可恢复默认值。

图 3-4　　帐户信息

注　更改密码时需要遵守以下规则：
密码至少需要 8 个字符；
新密码至少应包含以下类别中三类：小写字母，大写字母，数字和特殊字符；
新密码中不能包含连续重复三遍以上的字符；
新密码不能与用户名相同，也不能是用户名的逆字符串。

配置分辨率

单击左侧窗格中的“ **分辨率**”可配置所连接的显示器的“ 分辨率”、“ 旋转”、“ 反射”和“ 状态”，并查看显示器的“ 屏幕模型”和“ 输出端口”（图 3-5）。

图 3-5　　分辨率信息

Cisco Edge 340 系列支持同时使用 VGA 和 HDMI 输出端口。在连接了两个显示器时，您将看到显示出双屏模式配置（图 3-6）。

图 3-6　　双屏设置

在双屏设置窗口中，您可以在一个位置配置两个屏幕的“ 分辨率”、“ 旋转”、“ 反射” 和“ 状态” ，并查看显示器的“ 屏幕模型” 和“ 输出端口”。注意，两个屏幕的配置可能有所不同。

双屏模式包含两种：

- Extend - 主屏幕显示系统的主视图，屏幕 2 会扩展主屏幕以显示更多视图。
- Duplicate - 两个屏幕都显示系统的相同视图。

您可以从“ **双屏**” 模式下拉列表中选择双屏模式类型 （ 图 3-7 ）。对于 Extend 类型的双屏模式，单击“ **切换位置**” 可以将主屏幕与屏幕 2 互换。

图 3-7　　双屏模式类型

配置日期与时间

单击左侧窗格中的“ **日期与时间**” 可配置 Cisco Edge 340 系列设备的日期与时间信息。

步骤 1　选择配置日期和时间的模式：**自动与 NTP 服务器同步**或**手动设置时间**。如果选择“ **手动设置时间**” ，请转到步骤 4。

步骤 2　如果选择“ 自动与 NTP 服务器同步” ，则“ 日期” 和“ 时间” 字段处于禁用状态；您无法对它们进行更改。

步骤 3　在“ **NTP 服务器**” 字段中输入 NTP 服务器地址并转到步骤 7。

步骤 4　如果您为步骤 1 中的模式选择“ **手动设置时间**” ，则“ **NTP 服务器**” 字段处于禁用状态；您无法对它进行修改。

步骤 5　单击“ **日期**” 字段右侧的日历图标，从下拉列表中选择月份和年份，然后单击对应的日期。

步骤 6　在“ **时间**” 字段中，分别从左侧下拉列表和右侧下拉列表中选择小时和 分钟。

步骤 7　从“ **时区**” 下拉列表中选择时区。

步骤 8　单击“ **应用**” 可保存更改，单击“ **重置**” 可恢复默认值。

图 3-8　　日期与时间信息

配置系统日志

单击左侧窗格中的“ **系统日志**”可配置系统日志的设置。

步骤 1　在“ 路径”字段中输入您用于存储系统日志的路径。系统将会自动检查您输入的路径是否有效。

步骤 2　从“ **级别**”下拉列表中选择系统日志的级别：

- 调试
- 信息
- 注意
- 警告
- 错误
- 严重
- 提示
- 紧急
- *（以上全部）

步骤 3　在“ 容量”字段中，在左侧栏中输入系统日志的大小，从右侧的下拉列表中选择日志大小的单位。

容量有三个单位：KB、 MB 和 GB。最大日志大小为 50 MB，默认大小为 10 MB。如果您提供的值超过 50 MB，更改将失败而且将保留原始大小。您不能使用 GB 作为单位。如果选择 KB 或 MB，日志大小值不能为 0。

步骤 4　单击“ **应用**”可保存更改，单击“ **重置**”可恢复默认值。

图 3-9　　系统日志信息

配置吐核

单击左侧窗格中的“ **吐核**” 可配置吐核的设置。

步骤 1　在“ 路径” 字段中输入文件名和您用于存储吐核的路径。文件可以直接通过“ 路径” 字段创建。

步骤 2　在 Size 字段中，在左侧栏中输入吐核的大小，从右侧的下拉列表中选择吐核大小的单位。

步骤 3　单击“ **应用**” 可保存更改，单击“ **重置**” 可恢复默认值。

图 3-10　　吐核信息

网络配置

可以使用“ 网络”选项卡配置有线、无线、 SNMP 和 VPN 设置。

配置有线设置

在“ 网络”选项卡中，单击左侧窗格中的“ **有线网络**”可配置“ 链路设置”、启用或禁用 Wake on Lan 以及配置 IPv4 模式和 IPv6 模式。请参阅图 3-11。单击“ **应用**”可保存更改，单击“ **重置**”可恢复默认值。

图 3-11　有线信息

配置 Wake on Lan

要启用 Wake on Lan 功能，请从“ **模式**”下拉列表中选择“ **手动**”，从 **Wake on Lan** 下拉列表中选择 **Magic-Packet**。然后您可以配置“ 速度”和“ 双工”设置。请参阅图 3-12。

图 3-12　Wake on Lan 设置

配置 IPv4 设置

要查看和配置 IPv4 的 IPv4 地址、子网掩码、网关和 DNS，请从“ **模式**”下拉列表中选择“ **手动**”，此时将显示“ IPv4 设置”屏幕（图 3-13）。

图 3-13　IPv4 设置

配置 IPv6 设置

要查看和配置 IPv6 的 IPv6 地址、子网掩码、网关和 DNS，请从“ **模式**”下拉列表中选择“ **手动**”，此时将显示“ IPv6 设置”屏幕（图 3-14）。

图 3-14　IPv6 设置

配置无线设置

Cisco Edge 340 系列设备支持以下无线模式：

- 接入点 (AP) — 允许通过 Wi-Fi 将无线设备连接到有线网络的设备。
- 站点 — 连接到其他网络的无线连接。
- 关 — 无线功能处于禁用状态。

要选择所需的无线模式，请单击左侧窗格中的“ **无线网络**”并配置每个模式的设置。

配置 AP 模式

如果从“ **当前模式**”下拉列表中选择 **AP**，您可以为 AP 模式配置 SSID 名称、广播 SSID、无线模式、信道带宽、信道号和安全设置。请参阅图 3-15。单击“ **应用**”可保存更改，单击“ **重置**”可恢复默认值。

图 3-15　AP 模式设置

单击图 3-15 中“ 设置” 部分末端的“ **高级**” 按钮，此时将显示“ AP 模式高级设置” 屏幕（ 图 3-16 ）。

图 3-16　AP 模式高级设置

单击屏幕底端的“ **隐藏**” 可返回简单模式。

配置站点模式

如果从“ **当前模式**” 下拉列表中选择“ **站点**” ，您可以在站点模式下添加、编辑、删除、建立和刷新无线连接。

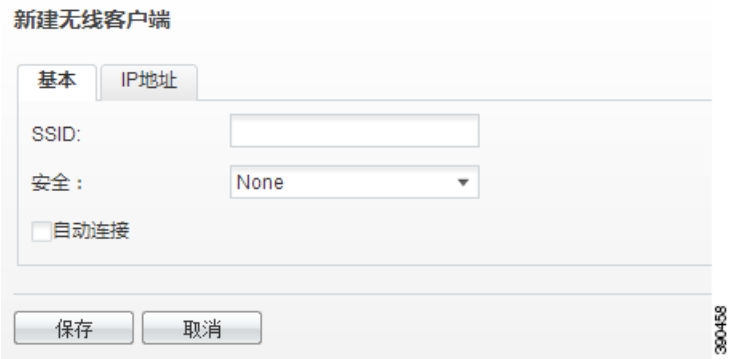
添加无线连接

要添加无线连接，请按照下列步骤操作：

步骤 1　单击“ 连接” 区域中的 ＋。

步骤 2　此时将显示“ 添加无线连接” 屏幕，如图 3-17 所示。

图 3-17　添加无线连接



步骤 3　在“ 基本” 选项卡下面，在 SSID 字段中输入无线连接的名称，从“ 安全” 下拉列表中选择安全类型。

您可以配置下列安全类型：

- “ None — 如果您从“ 安全” 下拉列表中选择 **None**，此无线连接的安全类型将在“ 连接” 区域中显示为 Open，如图 3-18 所示。

图 3-18　安全类型 — None

- WEB 40/128 位密钥（十六进制或 ASCII）— 可以为 WEB 40/128 位密钥（十六进制或 ASCII）安全类型配置 SSID、密钥、 WEP 索引和身份验证。
- WPA 和 WPA2 个人 — 此类型通常用于个人用户。可以为 WPA & WPA2 Personal 安全类型配置密码。
- WPA 和 WPA2 Enterprise — 此类型通常用于企业用户。可以为 WPA & WPA2 Enterprise 安全类型配置身份验证、用户名、用户证书、 CA 证书、私钥和私钥密码。

步骤 4　（可选）如果没有 DHCP 服务器，请单击“ **IP 地址**” 选项卡输入 IPv4 或 IPv6 地址信息。

步骤 5　如果选择“ **自动连接**”，无线连接将是下次连接的首选。

步骤 6　单击“ **保存**” 以添加无线连接。

编辑无线连接

要编辑无线连接，请按照下列步骤操作：

步骤 1　选择要编辑的无线连接并单击“ 连接” 区域中的 ，此时将显示“ 编辑无线连接” 屏幕，如图 3-19 所示。

图 3-19　编辑无线连接

步骤 2　根据需要编辑安全选项或 IP 地址信息，然后单击“ **保存**”。

注　有关安全选项的详细信息，请参阅第 3-12 页上的“ 添加无线连接” 一节。

注　如果无线网络连接成功，那么，除非断开连接，否则无法对其进行编辑或删除。

删除无线连接

要删除无线连接，请按照下列步骤操作：

步骤 1　从“ 连接” 区域中选择选择您要删除的无线连接。

步骤 2　单击“ 连接” 区域中的 以删除该无线连接。

注　如果无线网络连接成功，那么，除非断开连接，否则无法对其进行编辑或删除。

连接无线网络

要连接到无线网络，请按照下列步骤操作：

步骤 1　从 Connections 区域中选择您要连接的无线连接。

步骤 2　单击 Connections 区域中的 以连接该无线网络。

如果您在添加或编辑无线连接时输入错误的设置信息，系统会用红色显示消息 `Connection failed, please correct settings and click save to reconnect`。更正设置并单击“ **保存**”以连接该无线网络。

刷新无线网络

要刷新无线网络，请单击“ 连接” 区域中的 。

禁用无线功能

要禁用无线功能，请从“ 无线网络” 选项卡的“ **当前模式**” 下拉列表中选择 **off**。

配置 SNMP

要启用或禁用 SNMP，或者添加、删除和查看 SNMP 用户，请在“ 网络” 选项卡的左侧窗格中单击 **SNMP**，如图 3-20 所示。

要启用 SNMP，请选中“ **打开 SNMP**” 复选框。

图 3-20　启用 SNMP

添加 SNMP 用户

要添加 SNMP 用户，请按照下列步骤操作：

步骤 1　单击“ SNMP 用户” 区域 。

步骤 2　此时将显示“ 添加 SNMP 用户” 屏幕，如图 3-21 所示。

步骤 3　在“ 用户名” 字段中输入用户名。用户名只能包含字母、数字和 _。第一个字符必须是字母。用户名长度应介于 4 和 33 之间。

步骤 4　在 Authpass 和 Privpass 字段中输入密码。密码只能包含字母、数字、@、#、$、%、^、&、+、= 和 _。密码长度应介于 8 和 32 之间。

步骤 5　单击“ **保存**” 以创建 SNMP 用户。

图 3-21　添加 SNMP 用户

删除 SNMP 用户

要删除 SNMP 用户，请按照下列步骤操作：

步骤 1　在“ SNMP 用户” 区域中选择要删除的 SNMP 用户。

步骤 2　单击“ SNMP 用户” 区域中的 以删除 SNMP 用户。

配置 VPN

Cisco Edge 340 系列支持以下 VPN 连接类型：

- PPTP
- IPSEC/L2TP/PSK
- IPSEC/L2TP/RSA
- CISCO（对于身份验证模式支持 PSK 和混合类型）

要添加、编辑和删除 VPN 连接，或者连接到 VPN，请在“ 网络”选项卡下的左侧窗格中单击 **VPN**，如图 3-22 所示。

图 3-22　VPN 信息

添加 PPTP 类型的 VPN 连接

要添加 PPTP 类型的 VPN 连接，请按照下列步骤操作：

步骤 1　“ 网络”选项卡下的左侧窗格中单击 **VPN**。

步骤 2　单击右侧窗格中的 ＋。

步骤 3　此时将显示“ 添加 VPN 连接”屏幕，如图 3-23 所示。

图 3-23　添加 VPN 连接 — PPTP

步骤 4　在“ 添加 VPN 连接” 窗口中，在“ 名字” 字段中输入要添加的 VPN 连接的名称。

步骤 5　如果您希望 Cisco Edge 340 系列设备自动连接到此 VPN，请选中**“ 自动连接”** 复选框。

步骤 6　从“ 类型” 下拉列表中为 VPN 连接类型选择 **PPTF**：

步骤 7　在“ 服务器”、“ 用户名” 和“ 密码” 字段中输入 VPN 服务器地址、用户名和密码。如果您希望以纯文本形式显示密码，请选中“ 显示密码” 复选框。

步骤 8　输入最大传输单位 (MTU) 和最大接收单位 (MRU) 的值。

步骤 9　选择要用于此 VPN 连接的协议。

步骤 10　从 MPPE 下拉列表中选择 MPPE 加密类型：

- 无
- MPPE
- MPPE40
- MPPE128

步骤 11　单击**“ 保存”** 以添加 VPN 连接。

步骤 12　VPN 窗格将进行更新以显示新 VPN 连接的名称和状态。

添加 IPSEC/L2TP/PSK 类型的 VPN 连接

要添加 IPSEC/L2TP/PSK 类型的 VPN 连接，请按照下列步骤操作：

步骤 1　在“ 网络”选项卡下的左侧窗格中单击 **VPN**。

步骤 2　单击右侧窗格中的 ＋。

步骤 3　此时将显示“ 添加 VPN 连接”屏幕，如图 3-24 所示。

步骤 4　在“ 添加 VPN 连接”窗口，在“ 名字”字段中输入要添加的 VPN 连接的名称。

步骤 5　如果您希望 Cisco Edge 340 系列设备自动连接到此 VPN，选中“ 自动连接”复选框。

步骤 6　从“ 类型”下拉列表中为 VPN 连接类型选择 **IPSEC/L2TP/PSK**。

图 3-24　添加 VPN 连接 — IPSEC/L2TP/PSK

步骤 7　在“ 服务器”、“ 用户名”和“ 密码”字段中输入 VPN 服务器地址、用户名和密码。如果您希望以纯文本形式显示密码，请选中“ **显示密码**”复选框。

步骤 8　输入 MTU 和 MRU 的值。

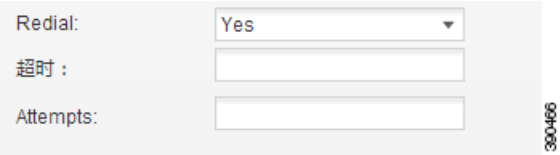
步骤 9　选择要用于此 VPN 连接的协议。

步骤 10　从 MPPE 下拉列表中选择 MPPE 加密类型：

步骤 11　在 Pre-shared Key 字段中输入预共享密钥。

步骤 12　从 Redial 下拉列表中选择 Yes 或 No。如果您选择 Yes，应为“ 超时”和“ Attempts”字段输入值（图 3-25）。

图 3-25　Redial 信息



- 超时 — 每次连接 VPN 服务器的最大时间。
- Attempts — 尝试连接 VPN 服务器的最大次数。

超时值乘以尝试次数的结果就是连接 VPN 服务器所用的时间，最多为 20 秒。

步骤 13　单击“ **保存**”以添加 VPN 连接。

步骤 14　VPN 窗格将进行更新以显示新 VPN 连接的名称和状态。

添加 IPSEC/L2TP/RSA 类型的 VPN 连接

要添加 IPSEC/L2TP/RSA 类型的 VPN 连接，请按照下列步骤操作：

步骤 1　在“ 网络”选项卡下的左侧窗格中单击 VPN。

步骤 2　单击右侧窗格中的 。

步骤 3　此时将显示“ 添加 VPN 连接”窗口（图 3-26）。

步骤 4　在“ 添加 VPN 连接”窗口中，在“ 名字”字段中输入要添加的 VPN 连接的名称。

步骤 5　如果您希望 Cisco Edge 340 系列设备自动连接到此 VPN，请选中“ **自动连接**”复选框。

步骤 6　从“ 类型”下拉列表中为 VPN 连接类型选择 **IPSEC/L2TP/RSA**。

步骤 7　在“ 服务器”、“ 用户名”和“ 密码”字段中输入 VPN 服务器地址、用户名和密码。如果您希望以纯文本形式显示密码，请选中“ **显示密码**”复选框。

步骤 8　输入 MTU 和 MRU 的值。

步骤 9　选择要用于此 VPN 连接的协议。

步骤 10　从 MPPE 下拉列表中选择 MPPE 加密类型：

步骤 11　输入私钥文件、客户端证书文件、 CA 证书文件和服务器证书文件的路径。

步骤 12　从 Redial 下拉列表中选择 Yes 或 No。如果您选择 Yes，请在“ 超时”和“ Attempts”字段中输入相关值。超时值乘以尝试次数的结果就是连接 VPN 服务器所用的时间，最多为 20 秒。

步骤 13　单击“ **保存**”以添加 VPN 连接。

步骤 14　VPN 窗格将进行更新以显示新 VPN 连接的名称和状态。

图 3-26　添加 VPN 连接 — IPSEC/L2TP/RSA

添加 CISCO 类型的 VPN 连接

要添加 CISCO 类型的 VPN 连接，请按照下列步骤操作：

步骤 1　在“ 网络”选项卡下的左侧窗格中单击 **VPN**。

步骤 2　单击右侧窗格中的 ＋。

步骤 3　此时将显示“ 添加 VPN 连接”窗口（图 3-27）。

步骤 4　在“ 添加 VPN 连接”窗口中，在“ 名字”字段中输入要添加的 VPN 连接的名称。

步骤 5　如果您希望 Cisco Edge 340 系列设备自动连接到此 VPN，请选中“ **自动连接**”复选框。

步骤 6　从“ 类型”下拉列表中为 VPN 连接类型选择 **CISCO**。

步骤 7　在“ 服务器”、“ 用户名”和“ 密码”字段中输入 VPN 服务器地址、用户名和密码。如果您希望以纯文本形式显示密码，请选中“ **显示密码**”复选框。

步骤 8　输入 MTU 和 MRU 的值。

步骤 9　从“ 认证模式” 下拉列表中选择 PSK 或 Hybrid。

步骤 10　在“ 组名”、“ 组命名”、“ 域名”、“ 加密”、“ NAT 穿透” 和“ IKE DH Group” 字段中输入值。如果选择 Hybrid 作为身份验证模式，请输入 CA 证书文件的路径。

步骤 11　单击**“ 保存”** 以添加 VPN 连接。

步骤 12　VPN 窗格将进行更新以显示新 VPN 连接的名称和状态。

图 3-27　添加 VPN 连接 — CISCO

编辑 VPN 连接

要编辑 VPN 连接，请按照下列步骤操作：

步骤 1　从“ 连接” 区域选择要编辑的 VPN 连接，然后单击 。

步骤 2　编辑“ VPN 连接” 屏幕中的配置。

步骤 3　单击“ **保存**”。

删除 VPN 连接

要删除 VPN 连接，请在“ 连接” 区域中选择要从连接列表中删除的 VPN 连接，然后单击 。

连接 VPN 连接

要连接 VPN 连接，请在“ 连接” 区域中选择要从连接列表中连接的 VPN 连接，然后单击 。

监控平台和网络状态

可以使用“ 监控” 选项卡监控 Cisco Edge 340 平台和网络的状态。图 3-28 显示“ 监控” 选项卡下的“ 平台” 窗格。

图 3-28　平台信息

图 3-29 显示“ 监控” 选项卡下的“ 网络” 窗格。

图 3-29　网络信息

维护

在“ 维护” 选项卡中，可以执行镜像升级和配置存档，还可以重新启动或重置 Cisco Edge 340 系列设备。

镜像升级

使用“ 镜像升级” 窗格，可以查看与设备型号和当前版本相关的信息，检查 /opt 目录下是否有可用的升级镜像文件 (.bin)。如果有，请单击“ **应用**” 升级系统，如图 3-30 所示。

图 3-30　镜像升级信息

配置存档

使用“ 配置存档” 窗格，可以通过单击“ 下载” 将配置文件下载到 USB 存储或本地目录，如图 3-31 所示。

图 3-31　配置存档信息

要将配置文件从一个 Cisco Edge 340 系列设备复制到另一个 Cisco Edge 340 系列设备，请按照下列步骤操作：

步骤 1　单击“ 下载” 以从原来的 Cisco Edge 340 系列设备下载配置文件。

步骤 2　通过以本地或远程方式复制配置文件来将配置文件保存到另一个 Cisco Edge 340 系列设备上。

步骤 3　从第二个 Cisco Edge 340 系列设备中打开 Web GUI，然后在“ 维护” 选项卡下的左侧窗格中单击**“ 配置存档”**。

步骤 4　单击**“ 选择文件”** 以选择已在步骤 2 中保存的配置文件。该文件的名称将显示在“ 选择文件” 按钮旁边。

步骤 5　单击**“ 应用”**。

步骤 6　Cisco Edge 340 系列设备将重新启动。

重启或重设

使用“ 重启 / 重设” 窗格，可以重新启动具有最新设置的 Cisco Edge 340 系列设备，也可以将 Cisco Edge 340 系列设备重置为出厂默认设置，然后重新启动它。

图 3-32　重启 / 重设信息

配置 HTTP API

可以在 Cisco Edge 340 系列设备上以本地或远程方式运行应用，通过 HTTP API 管理该设备。设备管理包括配置设备、监控状态以及安装和升级软件。

本章将介绍每个 HTTP API，包括请求、回复、参数限制和错误代码。

要使用 HTTP API 配置 Cisco Edge 340 系列设备，请参阅以下各节：

系统 API

使用此节中的命令可配置系统 API。

注　Curl 用作请求 API 的一个工具示例。 **Localhost** 用作 Cisco Edge 340 系列的一个 IP 地址示例，**cisco123!** 用作 一个管理员密码示例。

设置帐户

示例：将根密码设置为 cisco123!

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"account"
: "root cisco123!"}' https://localhost/api/2.0/sys/account
```

回复

```
{"success":"true","updatedAt":"2013-05-10 14:36:09"}
```

参数限制

帐户的第一个参数应为现有用户。新密码至少应包含三个以下类别的字符：

小写字母、大写字母、数字和特殊字符。

新密码中的任何字符均不能连续重复三次以上。新密码不能与相关用户名和反写的用户名相同。密码长度应遵循相应限制（介于最小长度和最大长度之间）。

获取 CPU

示例：获取 CPU

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/cpu
```

回复

```
{"cpu":"0.1%us  0.1%sy","success":"true","getAt":"2013-05-08 11:43:40"}
```

获取内存

示例：获取内存

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/memory
```

回复

```
{"memory":"Mem  Total:2001 Used:1236 Free:764\nSwap Total:0 Used:0
Free:0","success":"true","getAt":"2013-05-08 11:43:46"}
```

获取存储

示例：获取存储

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/storage
```

回复

```
{"storage":"Root[/]     Size:7.4G Used:2.7G Free:4.4G Used38%\nHOME[/home] Size:7.4G
Used:2.7G Free:4.4G Used38%","success":"true","getAt":"2013-05-08 11:44:02"}
```

获取主机名

示例：获取主机名

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/hostname
```

回复

```
{"hostname":"cisco.com","success":"true","getAt":"2013-05-08 11:44:21"}
```

设置主机名

示例：将主机名设置为 cisco

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"hostname" : "cisco"}' https://localhost/api/2.0/sys/hostname
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

参数限制

主机名必须是有效的域名。

获取 RPM 版本

示例：获取 Python 数据包的 RPM 版本

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/rpmVersion/python
```

回复

```
{"python":"2.7.3","success":"true","getAt":"2013-05-08 11:56:24"}
```

设置 NTP 服务器

示例：将 NTP 服务器设置为 time-a.nist.gov

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"server":"time-a.nist.gov"}' https://localhost/api/2.0/sys/ntp/server
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

参数限制

有效参数为 full 或 half。

获取 NTP 服务器

示例：获取 NTP 服务器

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/ntp/server
```

回复

```
{"ntp":"time-a.nist.gov","success":"true","getAt":"2013-05-09 12:22:26"}
```

获取蓝牙状态

示例：获取蓝牙状态

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/bluetooth
```

回复

```
{"bluetooth":"on","success":"true","getAt":"2013-05-10 15:04:35"}
```

设置蓝牙状态

示例：将蓝牙状态设置为 off

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"bluetooth" : "off"}' https://localhost/api/2.0/sys/bluetooth
```

回复

```
{"success":"true","updatedAt":"2013-05-10 15:04:57"}
```

参数限制

有效参数为 on 或 off。

获取时区

示例：获取时区

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/timeZone
```

回复

```
{"timeZone":"8","success":"true","getAt":"2013-05-10 15:38:15"}
```

设置时区

示例：设置时区

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"timeZone":"Asia/Shanghai"}' https://localhost/api/2.0/sys/timeZone
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

参数限制

有效参数为 /usr/share/zoneinfo/posix/ 文件夹下的任何城市。

获取 Auto-Login 状态

示例：获取 Auto-Login 状态

请求

```
curl -k -X GET -H 'password: cisco123!' https:// localhost /api/2.0/sys/autologin
```

回复

```
{"autologin":"enable","success":"true","getAt":"2013-05-10 15:04:35"}
```

参数限制

系统报告了 r。

设置 Auto-Login

示例：将 Auto-Login 设置为 disable

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"autologin":"disable"}' https:// localhost /api/2.0/sys/autologin
```

回复

```
{"success":"true","updatedAt":"2013-05-10 15:05:54"}
```

参数限制

有效参数为 enable 或 disable。

设置时间

示例：将时间设置为 2012-11-06 19:49:21

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"time" :
"2012-11-06 19:49:21"}' https://localhost/api/2.0/sys/time
```

回复

```
{"success":"true","updatedAt":"2012-11-06 19:49:21"}
```

参数限制

参数应采用 YYYY-MM-DD HH:MM:SS 格式。否则系统会报告 004 错误。

获取时间

示例：获取时间

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/time
```

回复

```
{"time":"2012-11-07 15:22:15","success":"true","getAt":"2012-11-07 15:22:15"}
```

存储库

列出所有存储库文件

示例：列出正使用的所有存储库文件

请求

```
curl -k -X GET -H 'password: cisco123!'  https://localhost/api/2.0/repository/reposList
```

回复

```
{"reposList":"fedora-updates.repo fedora-updates-testing.repo google-chrome.repo
fedora.repo","success":"true","getAt":"2013-08-14 14:29:28"}
```

设置存储库备份

示例：将存储库备份设置为目标存储库

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"backup":"/tmp"}' https://localhost/api/2.0/repository/backup
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

删除指定的存储库文件

示例：从存储库目录中删除指定的存储库文件

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"removing":" google-chrome.repo"}' https://localhost/api/2.0/repository/removing
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

添加指定的存储库文件

示例：添加指定的存储库文件

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"update":"/home/user/google-chrome.repo"}' https://localhost/api/2.0/repository/update
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

添加指定文件夹下的所有存储库文件

示例：添加指定文件夹下的所有存储库文件

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"updateAll":"/home/user/"}' https://localhost/api/2.0/repository/updateAll
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

吐核

设置吐核限制

示例：将吐核限制设置为 unlimit

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"limit":"unlimited"}' https://localhost/api/2.0/cordDump/limit
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取吐核限制

示例：获取吐核限制

请求

```
curl -k -X GET -H 'password: cisco123!' https:// localhost /api/2.0/coreDump/limit
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51","limit":"unlimited"}
```

设置吐核位置

示例：将吐核文件的位置设为 /tmp

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"pos":"/tmp"}' https://localhost/api/2.0/coreDump/pos
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取吐核位置

示例：获取吐核文件的位置

请求

```
curl -k -X PUT -H 'password: cisco123!' https://localhost/api/2.0/coreDump/pos
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51","pos":"/tmp"}
```

日志

获取日志路径

示例：获取日志路径

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/log/path
```

回复

```
{"path":"/var/log","success":"true","getAt":"2013-05-08 17:00:25"}
```

设置日志路径

示例：将日志路径设置为 /tmp

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"path" :
"/tmp"}' https://localhost/api/2.0/log/path
```

回复

```
{"success":"true","updatedAt":"2013-05-08 17:12:04"}
```

获取日志级别

示例：获取日志级别

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/sys/log/level
```

回复

```
{"level":"*","success":"true","getAt":"2013-05-08 17:48:22"}
```

设置日志级别

示例：将日志级别设置为 info

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"level" :
"info"}' https://localhost/api/2.0/log/level
```

回复

```
{"level":"info","success":"true","getAt":"2013-05-08 17:54:28"}
```

获取日志大小

示例：获取日志大小

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/log/size
```

回复

```
{"size":"5k","success":"true","getAt":"2013-05-08 17:59:41"}
```

设置日志大小

示例：将日志大小设置为 10M

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"size? :
"10M"}' https://localhost/api/2.0/log/size
```

回复

```
{"success":"true","updatedAt":"2013-04-09 10:27:52"}
```

以太网 API

使用本节中的命令可配置以太网 API。

获取速度

示例：获取速度

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/speed
```

回复

```
{"speed":"100","success":"true","getAt":"2013-04-08 10:58:49"}
```

设置速度

示例：将速度设置为 1000

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"speed" :
"1000"}' https://localhost/api/2.0/eth/speed
```

回复

```
{"success":"true","updatedAt":"2012-11-08 09:37:31"}
```

参数限制

仅 1000、 100 和 10 是有效参数。

获取双工

示例：获取双工

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/duplex
```

回复

```
{"duplex":"full","success":"true","getAt":"2013-04-08 13:08:20"}
```

设置双工

示例：将双工设置为 half

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"duplex"
: "half"}' https://localhost/api/2.0/eth/duplex
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

参数限制

仅 full 和 half 是有效参数。

获取 Link-Detected

示例：获取 Link-Detected

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/linkDetected
```

回复

```
{"linkDetected":"yes","success":"true","getAt":"2013-04-08 13:08:20"}
```

获取 MAC

示例：获取 MAC

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/mac
```

回复

```
{"mac":"","success":"true","getAt":"2013-06-25 18:00:51"}
```

获取 Address4

示例：获取 address4

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/address4
```

回复

```
{"address4":"10.140.28.90/255.255.255.0","success":"true","getAt":"2013-06-25 18:04:42"}
```

获取 Address6

示例：获取 address6

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/address6
```

回复

```
{"address6":"","success":"true","getAt":"2013-06-25 18:05:29"}
```

获取 DNS4

示例：获取 DNS4

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/dns4
```

回复

```
{"dns4":"64.104.123.144\n171.70.168.183\n8.8.8.8\n8.8.4.4\n4.4.4.4","success":"true","getA
t":"2013-06-25 18:10:48"}
```

获取 DNS6

示例：获取 DNS6

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/dns6
```

回复

```
{"address6":"","success":"true","getAt":"2013-06-25 18:05:29"}
```

获取 Gateway4

示例：获取 gateway4

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/gateway4
```

回复

```
{"gateway4":"10.140.28.1","success":"true","getAt":"2013-06-25 18:09:25"}
```

获取 Gateway6

示例：获取 gateway6

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/gateway6
```

回复

```
{"gateway6":"","success":"true","getAt":"2013-06-25 18:09:46"}
```

获取 IPv4

示例：获取 IPv4

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/ipv4
```

回复

```
{"ipv4":"auto","success":"true","getAt":"2013-06-25 18:08:11"}
```

获取 IPv6

示例：获取 IPv6

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/ipv6
```

回复

```
{"ipv6":"auto","success":"true","getAt":"2013-06-25 18:08:55"}
```

设置 IPv4

示例：设置 IPv4

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"ipv4" :
"auto"}' https://localhost/api/2.0/eth/ipv4
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 IPv6

示例：设置 IPv6

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"ipv6" :
"0"}' https://localhost/api/2.0/eth/ipv6
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 Address4

示例：设置 address4

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"address4" : "8.8.8.8/24"}' https://localhost/api/2.0/eth/address4
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 Address6

示例：设置 address6

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"
address6" : "2001::1/64"}' https://localhost/api/2.0/eth/address6
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 Gateway4

示例：设置 gateway4

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"
gateway4" : "10.0.0.1"}' https://localhost/api/2.0/eth/gateway4
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 Gateway6

示例：设置 gateway4

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"
gateway6" : "2001:2"}' https://localhost/api/2.0/eth/gateway6
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 DNS4

示例：设置 DNS4

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{" dns4" :
"0"}' https://localhost/api/2.0/eth/dns4
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

设置 DNS6

示例：设置 DNS6

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{" dns4" :
"0"}' https://localhost/api/2.0/eth/dns6
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

应用设置

示例：获取 run

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/eth/run
```

回复

```
{"run":"","success":"true","getAt":"2013-06-25 18:00:51"}
```

发出一个命令

重新启动 CE340

示例：重新启动 CE340

请求

```
curl -k -X GET -H 'password: cisco123!' httpss://10.140.44.134/api/1.0/cmd/reboot
```

回复

```
{"reboot":"true","success":"true","getAt":"2012-11-12 05:10:43"}
```

发出管理命令

示例：将管理命令设置为 mv /tmp/test /tmp/test1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"mngCmd"
: "mv /tmp/test /tmp/test1"}' httpss://10.140.44.134/api/1.0/cmd/mngCmd
```

回复

```
{"success":"true","updatedAt":"2012-11-12 05:23:49"}
```

镜像版本信息

获取操作系统版本

示例：获取操作系统版本信息

请求

```
curl -k -X GET -H 'password: cisco123!' httpss://10.140.44.134/api/1.0/image/os
```

回复

```
{"os":"1.3.9.1","success":"true","getAt":"2012-11-12 05:51:56"}
```

获取第三方应用程序版本

示例：获取第三方应用程序版本信息

请求

```
curl -k -X GET -H 'password: cisco123!' httpss://10.140.44.134/api/1.0/image/3rdapp
```

回复

```
{"3rdapp":"1.3.9.1","success":"true","getAt":"2012-11-12 05:56:39"}
```

获取操作系统和第三方应用程序版本

示例：获取操作系统版本和第三方应用程序版本信息

请求

```
curl -k -X GET -H 'password: cisco123!' httpss://10.140.44.134/api/1.0/image
```

回复

```
{"os":"1.3.9.1","3rdapp":"1.3.9.1","success":"true","getAt":"2012-11-12 08:18:58"}
```

AP 信息

设置 AP SSID

示例：将 SSID 设置为 cisco

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"ssid" :
"cisco"}' https://localhost/api/2.0/wifi/ap/ssid
```

回复

```
{"success":"true","updatedAt":"2012-11-12 06:25:47"}
```

参数限制

AP SSID 的长度应介于 1 到 32 个字符之间，否则系统会报告 004 错误。

获取 AP SSID

示例：获取 SSID

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/ssid
```

回复

```
{"ssid":"BluebirdAP","success":"true","getAt":"2013-05-15 10:46:49"}
```

获取 SSID 隐藏状态

示例：获取 SSID 隐藏状态

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/ssidHiding
```

回复

```
{"ssidHiding":"0","success":"true","getAt":"2013-05-15 10:46:49"}
```

参数限制

仅 0 或 1 是有效值，否则系统会报告 004 错误。

设置 SSID 隐藏状态

示例：设置 SSID 隐藏状态

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"ssidHiding" : "0"}' https://localhost/api/2.0/wifi/ap/ssidHiding
```

回复

```
{"success":"true","updatedAt":"2013-05-15 15:07:03"}
```

获取不转发状态

示例：获取不转发状态

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/noForwarding
```

回复

```
{"noForwarding":"0","success":"true","getAt":"2013-05-15 14:23:10"}
```

设置不转发状态

示例：设置不转发状态

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"noForwarding":"0"}' https://localhost/api/2.0/wifi/ap/noForwarding
```

回复

```
{"success":"true","updatedAt":"2013-05-15 14:41:58"}
```

参数限制

仅 0 或 1 是有效值，否则系统会报告 004 错误。

获取无线网模式

示例：获取无线网模式

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/wirelessMode
```

回复

```
{"wirelessMode":"9","success":"true","getAt":"2013-05-15 15:05:23"}
```

设置无线模式

示例：将无线模式设置为 9

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wirelessMode" : "9"}' https://localhost/api/2.0/wifi/ap/wirelessMode
```

回复

```
{"success":"true","updatedAt":"013-05-15 15:11:35"}
```

参数限制

仅 0、 1、 2、 4、 6、 7、 8、 9、 10 和 11 是有效参数，否则系统会报告 004 错误。

设置信道号

示例：将信道号设置为 9

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"channelNumber" : "9"}' https://localhost/api/2.0/wifi/ap/channelNumber
```

回复

```
{"success":"true","updatedAt":"2012-11-12 07:32:04"}
```

参数限制

表 4-1 和表 4-2 显示可基于地区代码设置的信道。

注　无线模式设置为 11 时，只能使用 5G 信道。

表 4-1　　2.4 GHz 信道

地区	信道
0	1-11
1	1-13
2	10 – 11
3	10-13
4	14
5	1-14

表 4-2　　5 GHz 信道

地区	信道
0	36、 40、 44、 48、 52、 56、 60、 64、 149、 153、 157、 161、 165
1	36、40、44、48、52、56、60、64、100、104、108、112、116、120、124、128、132、 136、 140
2	36、 40、 44、 48、 52、 56、 60、 64
3	52、 56、 60、 64、 149、 153、 157、 161
4	149、 153、 157、 161、 165
5	149、 153、 157、 161
6	36、 40、 44、 48
7	36、40、44、48、52、56、60、64、100、104、108、112、116、120、124、128、132、 136、 140、 149、 153、 157、 161、 165
8	52、 56、 60、 64
9	36、40、44、48、52、56、60、64、100、104、108、112、116、132、136、140、149、 153、 161、 165
10	36、 40、 44、 48、 149、 153、 157、 161、 165
11	36、40、44、48、52、56、60、64、100、104、108、112、116、120、149、153、157、 161

获取信道号

示例：获取信道号

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/channelNumber
```

回复

```
{"channelNumber":"6","success":"true","getAt":"2013-05-15 17:35:56"}
```

获取操作模式

示例：获取操作模式

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/operatingMode
```

回复

```
{"operatingMode":"1","success":"true","getAt":"2013-05-15 17:48:56"}
```

设置操作模式

示例：设置操作模式

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"operatingMode" : "1"}' https://localhost/api/2.0/wifi/ap/operatingMode
```

回复

```
{"success":"true","updatedAt":"2012-11-12 07:32:04"}
```

参数限制

对于混合模式，有效参数为 0 ；对于绿地，有效参数为 1。否则系统会报告 004 错误。

设置信道带宽

示例：将信道带宽设置为 20 MHz

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"channelBandwidth" : "1"}' https://localhost/api/2.0/wifi/ap/channelBandwidth
```

回复

```
{"success":"true","updatedAt":"2012-11-12 07:32:04"}
```

参数限制

对于信道带宽 20 MHz，有效参数为 0 ；对于信道带宽 20/40 MHz，有效参数为 1。否则系统会报告 004 错误。

获取信道带宽

示例：获取信道带宽

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/channelBandwidth
```

回复

```
{"channelBandwidth":"1","success":"true","getAt":"2013-05-16 10:45:31"}
```

设置保护间隔

示例：将保护间隔设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"guardInterval" : "1"}' https://localhost/api/2.0/wifi/ap/guardInterval
```

回复

```
{"success":"true","updatedAt":"2012-11-12 06:34:20"}
```

参数限制

对于长保护间隔（800 毫微秒），有效参数为 0 ；对于短保护间隔（400 毫微秒），有效参数为 1。否则系统会报告 004 错误。

获取保护间隔

示例：获取保护间隔

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/guardInterval
```

回复

```
{"guardInterval":"1","success":"true","getAt":"2013-05-16 10:45:31"}
```

设置 MCS

示例：将 mcs 设置为 33

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"mcs" :
"33"}' https://localhost/api/2.0/wifi/ap/mcs
```

回复

```
{"success":"true","updatedAt":"2013-05-16 13:45:04"}
```

参数限制

仅 0 到 15 的值以及 33 是有效参数，否则系统会报告 004 错误。对于 2*2 MIMO，有效参数为 0 到 15 的值；对于自动速率适配，有效参数为 33（建议）。

获取 MCS

示例：获取 mcs

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/mcs
```

回复

```
{"mcs":"33","success":"true","getAt":"2013-05-16 13:50:17"}
```

设置 RDG

示例：将 rdg 设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"rdg" :
"0"}' https://localhost/api/2.0/wifi/ap/rdg
```

回复

```
{"success":"true","updatedAt":"2013-05-16 13:45:04"}
```

参数限制

有效参数为 0（建议，用于禁用 RDG ）或 1（用于启用 RDG)。否则系统会报告 004 错误。

获取 RDG

示例：获取 rdg

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/rdg
```

回复

```
{"rdg":"0","success":"true","getAt":"2013-05-16 14:09:25"}
```

获取扩展信道

示例：获取扩展信道

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/extensionChannel
```

回复

```
{"extensionChannel":"0","success":"true","getAt":"2012-11-12 06:33:20"}
```

设置扩展信道

示例：将扩展信道设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"extensionChannel" : "0"}' https://localhost/api/2.0/wifi/ap/extensionChannel
```

回复

```
{"success":"true","updatedAt":"2013-05-16 14:49:39"}
```

获取 AMSDU

示例：获取 AMSDU

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/amsdu
```

回复

```
{"amsdu":"0","success":"true","getAt":"2013-05-16 15:18:02"}
```

设置 AMSDU

示例：将 AMSDU 设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"amsdu" :
"0"}' https://localhost/api/2.0/wifi/ap/amsdu
```

回复

```
{"success":"true","updatedAt":"2013-05-16 13:45:39"}
```

参数限制

有效参数为 0（用于禁用 Tx AMSDU）或 1（用于启用 Tx AMSDU）。否则系统会报告 004 错误。

设置自动块确认

示例：将自动块确认设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"autoBa"
: "0"}' https://localhost/api/2.0/wifi/ap/autoBa
```

回复

```
{"success":"true","updatedAt":"2013-05-16 13:45:39"}
```

参数限制

有效参数为 0 和 1。否则系统会报告 004 错误。

- 0 — 禁用，手动设置 BA 会话。
- 1 — 启用，连接后自动设置 BA 会话（建议）。

获取自动块确认

示例：获取自动块确认

请求

```
curl -k -X GET -H 'password: cdisco123' https://localhost/api/2.0/wifi/ap/autoBa
```

回复

```
{"autoBa":"1","success":"true","getAt":"2013-05-16 15:25:51"}
```

设置 Disallow TKIP

示例：将 disallowTKIP 设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"disallowTKIP" : "0"}' https://localhost/api/2.0/wifi/ap/disallowTKIP
```

回复

```
{"disallowTKIP":"1","success":"true","getAt":"2013-05-16 15:25:51"}
```

参数限制

有效参数为 0 和 1。否则系统会报告 004 错误。

- 0 — 允许将 HT 速率与 TKIP 和 WEP 加密一起使用。
- 1 — 不允许将 HT 速率与 TKIP 和 WEP 加密一起使用。

获取 Disallow TKIP

示例：获取 disallowTKIP

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/disallowTKIP
```

回复

```
{"disallowTKIP":"1","success":"true","getAt":"2013-05-16 15:25:51"}
```

设置 authMode

示例：将 authMode 设置为 OPEN

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"authMode" : "0"}' https://localhost/api/2.0/wifi/ap/authMode
```

回复

```
{"authMode":"open","success":"true","getAt":"2013-05-16 15:25:51"}
```

参数限制

仅 OPEN、 SHARED、 WPAPSK、 WPA2PSK、 WPAPSKWPA2PSK、 WPA、 WPA2 和 WPA1WPA2 是有效参数，否则系统会报告 004 错误。

获取 authMode

示例：获取 authMode

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/authMode
```

回复

```
{"authMode":"OPEN","success":"true","getAt":"2013-05-16 15:25:51"}
```

设置 mcastMcs

示例：将 mcastMcs 设置为 2

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"mcastMcs" : "2"}' https://localhost/api/2.0/wifi/ap/mcastMcs
```

回复

```
{"success":"true","updatedAt":"2013-05-23 16:50:51"}
```

参数限制

有效参数是从 0 到 15 的值。否则系统会报告 004 错误。

获取 mcastMcs

示例：获取 mcastMcs

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/mcastMcs
```

回复

```
{"authMode":"OPEN","success":"true","getAt":"2013-05-16 15:25:51"}
```

获取密钥 1 的类型

示例：获取密钥 1 的类型

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/key1Type
```

回复

```
{"key1Type":"0","success":"true","getAt":"2013-05-22 13:19:39"}
```

设置密钥 1 的类型

示例：将密钥 1 的类型设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"key1Type" : "1"}' https://localhost/api/2.0/wifi/ap/key1Type
```

回复

```
{"success":"true","updatedAt":"2013-05-22 13:26:04"}
```

获取加密类型

示例：获取加密类型

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/encryptionType
```

回复

```
{"encryptionType":"AES","success":"true","getAt":"2013-05-22 13:19:39"}
```

设置加密类型

示例：将加密类型设置为 WEP

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"encryptionType" : "WEP"}' https://localhost/api/2.0/wifi/ap/key1Type
```

回复

```
{"success":"true","updatedAt":"2013-05-22 13:26:04"}
```

设置 RADIUS 服务器

示例 1：将 RADIUS 服务器主机设置为 192.168.1.10

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"radiusServer" : "192.168.1.10"}' https://localhost/api/2.0/wifi/ap/radiusServer
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

IP 地址必须是有效地址。

获取 RADIUS 服务器

示例：获取 Radius 服务器

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/radiusServer
```

回复

```
{"radiusServer":"192.168.2.3","success":"true","getAt":"2013-05-22 14:25:17"}
```

设置 RADIUS 端口

示例 1：将 RADIUS 端口设置为 8000

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"radiusPort" : "8000"}' https://localhost/api/2.0/wifi/ap/radiusPort
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

端口应是介于 0 到 65535 之间的整数。

获取 RADIUS 端口

示例：获取 RADIUS 端口

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/radiusPort
```

回复

```
{"radiusPort":"1812","success":"true","getAt":"2013-05-22 14:25:17"}
```

设置 RADIUS 密钥

示例 1：将 RADIUS 密钥设置为 cisco

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"radiusKey" : "cisco"}' https://localhost/api/2.0/wifi/ap/radiusKey
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

获取 RADIUS 密钥

示例：获取 RADIUS 密钥

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/radiusKey
```

回复

```
{"radiusKey":"cisco","success":"true","getAt":"2013-05-22 14:25:17"}
```

设置自己的 IP 地址

示例 1：将自己的 IP 地址设置为 192.168.1.2

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"ownIpAddr" : "192.168.1.2"}' https://localhost/api/2.0/wifi/ap/ownIpAddr
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

IP 地址必须是有效地址。

获取自己的 IP 地址

示例：获取自己的 IP 地址

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/ownIpAddr
```

回复

```
{"ownIpAddr":"192.168.1.2","success":"true","getAt":"2013-05-22 14:25:17"}
```

获取背景保护

示例：获取背景保护

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/bgProtection
```

回复

```
{"bgProtection":"0","success":"true","getAt":"2013-05-22 14:25:17"}
```

将 bgProtection 设置为 2

示例：将 bgProtection 设置为 2

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"bgProtection" : "2"}' https://localhost/api/2.0/wifi/ap/bgProtection
```

回复

```
{"success":"true","updatedAt":"2013-05-22 14:25:17"}
```

参数限制

有效参数为 0、 1 和 2。

设置信标周期

示例：将信标周期设置为 22

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"beaconPeriod" : "22"}' https://localhost/api/2.0/wifi/ap/beaconPeriod
```

回复

```
{"success":"true","updatedAt":"2012-11-12 07:49:35"}
```

参数限制

仅 20 到 1000 之间的整数是有效参数。否则系统会报告 004 错误。

获取信标周期

示例：获取信标周期

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/beaconPeriod
```

回复

```
{"beaconPeriod":"100","success":"true","getAt":"2012-11-12 07:48:15"}
```

设置 DTIM 周期

示例：将 DTIM 周期设置为 200

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"dtimPeriod" : "200"}' https://localhost/api/2.0/wifi/ap/dtimPeriod
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

参数必须是介于 1 到 255 之间的整数。

获取 DTIM 周期

示例：获取 DTIM 周期

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/dtimPeriod
```

回复

```
{"dtimPeriod":"200","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置分段阈值

示例：将分段阈值设置为 256

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"fragThreshold" : "256"}' https://localhost/api/2.0/wifi/ap/fragThreshold
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

参数必须是介于 256 到 2346 之间的整数。

获取分段阈值

示例：获取分段阈值

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/fragThreshold
```

回复

```
{"fragThreshold":"2346","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置 RTS 阈值

示例：将 RTS 阈值设置为 2347

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"rtsThreshold" : "2347"}' https://localhost/api/2.0/wifi/ap/rtsThreshold
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

参数必须是介于 1 到 2347 之间的整数。

获取 RTS 阈值

示例：获取 RTS 阈值

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/rtsThreshold
```

回复

```
{"rtsThreshold":"2347","success":"true","getAt":"2013-05-22 08:06:44"}
```

获取 TX 功率

示例：获取 TX 功率

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/txPower
```

回复

```
{"txPower":"100","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置 TX 功率

示例：将 TX 功率设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"txPower"
: "1"}' https://localhost/api/2.0/wifi/ap/txPower
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

参数必须是介于 1 到 100 之间的整数。

获取 TX 前导

示例：获取 TX 前导

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/txPreamble
```

回复

```
{"txPreamble":"0","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置 TX 前导

示例：将 TX 前导设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"txPreamble" : "1"}' https://localhost/api/2.0/wifi/ap/txPreamble
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

参数必须是介于 0 到 2 之间的整数。

获取短碰撞槽时间

示例：获取短碰撞槽时间

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/shortSlot
```

回复

```
{"shortSlot":"1","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置短碰撞槽时间

示例：将短碰撞槽时间设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"shortSlot" : "1"}' https://localhost/api/2.0/wifi/ap/shortSlot
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

设置 TX 脉冲

示例：将 TX 脉冲设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"txBurst"
: "1"}' https://localhost/api/2.0/wifi/ap/txBurst
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

获取 TX 脉冲

示例：获取 TX 脉冲

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/txBurst
```

回复

```
{"txBurst":"1","success":"true","getAt":"2013-05-22 08:06:44"}
```

获取数据包聚合

示例：获取数据包聚合

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/pktAggregate
```

回复

```
{"pktAggregate":"1","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置数据包聚合

示例：将数据包聚合设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"pktAggregate" : "1"}' https://localhost/api/2.0/wifi/ap/pktAggregate
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

获取国家 / 地区代码

示例：获取国家 / 地区代码

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/countryCode
```

回复

```
{"countryCode":"1","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置国家 / 地区代码

示例：将国家 / 地区代码设置为 CN

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"countryCode" : "CN"}' https://localhost/api/2.0/wifi/ap/countryCode
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数只能包含 2 个字符，例如 TW 指中国台湾。

获取 WMM 支持

示例：获取 WMM 支持

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/wmmCapable
```

回复

```
{"wmmCapable":"0","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置 WMM 支持

示例：将 WMM 支持设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wmmCapable" : "1"}' https://localhost/api/2.0/wifi/ap/wmmCapable
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

获取 APSD 支持

示例：获取 APSD 支持

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/apsdCapable
```

回复

```
{"apsdCapable":"1","success":"true","getAt":"2013-05-22 08:06:44"}
```

设置 APSD 支持

示例：将 APSD 支持设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"apsdCapable" : "0"}' https://localhost/api/2.0/wifi/ap/apsdCapable
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

设置 IGMP SN 启用

示例：将 IGMP SN 启用设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"igmpSnEnable" : "0"}' https://localhost/api/2.0/wifi/ap/igmpSnEnable
```

回复

```
{"success":"true","updatedAt":"2013-05-22 08:08:05"}
```

参数限制

有效参数为 0 或 1。

获取 IGMP SN 启用

示例：获取 IGMP SN 启用

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/ap/igmpSnEnable
```

回复

```
{"igmpSnEnable":"0","success":"true","getAt":"2013-05-22 08:06:44"}
```

Wi-Fi 客户端信息

注　对于本节的命令，假设 Wi-Fi 客户端的 IP 地址为 10.140.44.147。

获取 AP 列表

示例：获取 AP 列表

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/client/list/ap
```

回复

```
{"ap":"BSSID / FLAG / FREQUENCY / STRENGTH / SSID / ACTIVE\nD8:24:BD:76:8D:A7  [  0,  0]
2437   14  cisco-8DA5  \n0C:D9:96:76:74:5F  [  0,288]  5805   22  blizzard
\n10:BD:18:AA:02:48  [  0,188]  2437   10  ciscosb1  \n64:AE:0C:F0:BB:D0  [14C,14C]  5200
29  sbbu-alpha","success":"true","getAt":"2013-04-24 13:30:17"}
```

获取连接列表

示例：获取连接列表

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/list/connection
```

回复

```
{"connection":"blizzard","success":"true","getAt":"2013-04-24 13:28:56"}
```

获取 WiFi 客户端 AP 标志

示例：获取 WiFi 客户端 AP 标志，假设 ssid 为 blizzard

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/flags
```

回复

```
{"flags":"1","success":"true","getAt":"2013-04-24 16:20:56"}
```

获取 WiFi 客户端 AP WPA 标志

示例：获取 WiFi 客户端 AP WPA 标志，假设 ssid 为 blizzard

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/wpaflags
```

回复

```
{"wpaflags":"0","success":"true","getAt":"2013-04-24 16:27:47"}
```

获取 WiFi 客户端 AP RSN 标志

示例：获取 WiFi 客户端 AP RSN 标志，假设 ssid 为 blizzard

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/rsnflags
```

回复

```
{"rsnflags":"288","success":"true","getAt":"2013-04-24 16:28:29"}
```

获取 WiFi 客户端 AP 频率

示例：获取 WiFi 客户端 AP 频率

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/frequency
```

回复

```
{"frequency":"5745","success":"true","getAt":"2013-04-24 16:29:18"}
```

获取 WiFi 客户端 AP 最大比特率

示例：获取 WiFi 客户端 AP 最大比特率

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/maxbitrate
```

回复

```
{"maxbitrate":"44000","success":"true","getAt":"2013-04-24 16:30:22"}
```

获取 WiFi 客户端 AP 强度

示例：获取 WiFi 客户端 AP 强度

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/strength
```

回复

```
{"strength":"71","success":"true","getAt":"2013-04-24 16:32:41"}
```

获取 WiFi 客户端 AP SSID

示例：获取 WiFi 客户端 AP SSID

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/ssid
```

回复

```
{"ssid":"blizzard","success":"true","getAt":"2013-04-24 16:33:35"}
```

获取 WiFi 客户端 AP 活动

示例：获取 WiFi 客户端 AP 活动

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/activity
```

回复

```
{"activity":"1","success":"true","getAt":"2013-04-24 16:36:05"}
```

获取 WiFi 客户端 AP BSSID

示例：获取 WiFi 客户端 AP BSSID

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/bssid
```

回复

```
{"bssid":"0C:D9:96:76:5B:EF","success":"true","getAt":"2013-04-24 16:40:47"}
```

获取 WiFi 客户端 AP 信道

示例：获取 AP 名称 blizzard 的 WiFi 客户端信道

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/channel
```

回复

```
{"channel":"6","success":"true","getAt":"2013-04-24 16:40:47"}
```

获取 WiFi 客户端 AP 链路状态

示例：获取 AP 名称 blizzard 的 WiFi 客户端链路状态

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/isconnected
```

回复

```
{"isconnected":"false","success":"true","getAt":"2013-04-24 16:40:47"}
```

获取当前 AP 的 WiFi 客户端信息

示例：获取当前 AP 的 WiFi 客户端信息

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/client/active/ap
```

回复

```
{"ap":"ssid=CMCC\nstrength=100\n?","success":"true","getAt":"2013-04-24 16:40:47"}
```

获取当前连接的 WiFi 客户端信息

示例：获取当前连接的 WiFi 客户端信息

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/active/connection
```

回复

```
{"connection":"ssid=CMCC\nmode=infrastructure\n...","success":"true","getAt":"2013-04-24
16:40:47"}
```

获取当前链路的 WiFi 客户端信息

示例：获取当前链路的 WiFi 客户端信息

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/client/active/link
```

回复

```
{"link":"firmware-version=\nip-interface=ra0\n ?","success":"true","getAt":"2013-04-24
16:40:47"}
```

获取所有的 WiFi 客户端 AP 信息

示例：获取所有的 WiFi 客户端 AP 信息

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/ap/blizzard/*
```

回复

{"*":"maxbitrate=44000\nssid=blizzard\nrsnflags=288\nwpaflags=0\nbssid=0C:D9:96:76:5B:EF\nfrequency=5745\nflags=1\nactive=1\nstrength=71","success":"true","getAt":"2013-04-24 16:45:02"}

获取所有的 WiFi 客户端连接信息

示例：获取 ssid blizzard 的所有 WiFi 客户端连接信息

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/wifi/client/con/blizzard
```

回复

{"*":"key-mgmt=wpa-eap\nproto=wpa,rsn\ntimestamp=1367030784\ntype=802-11-wireless\nid=blizzard\nuuid=48fc378d-9645-4ad6-9fd1-86c1a1f2fca8\nipv4-routes=\nipv4-addresses=\nipv4-dns=\nipv4-method=auto\neap=peap\npassword=123456\nidentity=ciscoId\nphase2-auth=mschapv2\nipv6-routes=\nipv6-addresses=\nipv6-dns=\nipv6-method=auto\nsecurity=802-11-wireless-security\nssid=blizzard\nmac-address=C0:62:6B:4A:17:08\nmode=infrastructure","success":"true","getAt":"2013-04-27 10:46:32"}

设置 WiFi 客户端连接 ID

示例：将 WiFi 客户端连接 ID 从 blizzard1 设置为 blizzard

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/update
```

回复 1

{"update":"blizzard1","success":"true","getAt":"2013-04-28 15:30:34"}

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"id" :
"blizzard"}' https://localhost/api/2.0/wifi/client/connection/id
```

回复 2

{"success":"true","updatedAt":"2013-04-28 15:33:00"}

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

{"id":"blizzard","success":"true","getAt":"2013-04-24 17:17:42"}

获取 WiFi 客户端连接 ID

示例：获取 WiFi 客户端连接 ID

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/connection/id
```

回复

```
{"id":"blizzard","success":"true","getAt":"2013-04-24 17:17:42"}
```

设置 WiFi 客户端连接 SSID

示例：将 WiFi 客户端连接 SSID 从 blizzard1 设置为 blizzard

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/update
```

回复 1

```
{"update":"blizzard1","success":"true","getAt":"2013-04-28 15:30:34"}
```

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"ssid" :
"blizzard"}' https://localhost/api/2.0/wifi/client/connection/ssid
```

回复 2

```
{"success":"true","updatedAt":"2013-04-28 15:33:10"}
```

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

```
{"id":"blizzard","success":"true","getAt":"2013-04-24 17:17:42"}
```

获取 WiFi 客户端连接 - 自动连接

示例：获取 WiFi 客户端连接 - 自动连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/connection/autoconnection
```

回复

```
{"autoconnection":"True","success":"true","getAt":"2013-04-24 17:29:38"}
```

将 WiFi 客户端连接设置为自动连接

示例：将 WiFi 客户端连接设置为自动连接

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard/update
```

回复 1

```
{"update":"blizzard","success":"true","getAt":"2013-04-28 16:05:22"}
```

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"autoconnection" : "True"}'
https://localhost/api/2.0/wifi/client/connection/autoconnection
```

回复 2

```
{"success":"true","updatedAt":"2013-04-28 16:05:48"}
```

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

```
{"saving":"","success":"true","getAt":"2013-04-28 16:06:21"}
```

获取 WiFi 客户端连接 SSID

示例：获取 WiFi 客户端连接 SSID

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/connection/ssid
```

回复

```
{"ssid":"blizzard","success":"true","getAt":"2013-04-27 09:59:27"}
```

获取 WiFi 客户端连接 EAP

示例：获取 WiFi 客户端连接 EAP

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/eap
```

回复

```
{"eap":"peap","success":"true","getAt":"2013-04-27 10:25:20"}
```

设置 WiFi 客户端连接 EAP

示例：将 WiFi 客户端连接 EAP 设置为 PEAP

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard/update
```

回复 1

```
{"update":"blizzard","success":"true","getAt":"2013-04-28 16:05:22"}
```

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"eap" :
"peap"}' https://localhost/api/2.0/wifi/client/wireless/eap
```

回复 2

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

```
{"saving":"","success":"true","getAt":"2013-04-28 16:06:21"}
```

获取 WiFi 客户端连接身份

示例：获取 WiFi 客户端连接身份

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/identity
```

回复

```
{"identity":"ciscoId","success":"true","getAt":"2013-04-27 10:26:22"}
```

设置 WiFi 客户端连接身份

示例：将 WiFi 客户端身份设置为 cisco

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard/update
```

回复 1

```
{"update":"blizzard","success":"true","getAt":"2013-04-28 16:05:22"}
```

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"identity" : "cisco"}' https://localhost/api/2.0/wifi/client/wireless/identity
```

回复 2

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

```
{"saving":"","success":"true","getAt":"2013-04-28 16:06:21"}
```

设置 WiFi 客户端连接 Anonymous-Identity

示例：将 WiFi 客户端 Anonymous-Identity 设置为 cisco

请求 1

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard/update
```

回复 1

```
{"update":"blizzard","success":"true","getAt":"2013-04-28 16:05:22"}
```

请求 2

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"anonymous-identity" : "cisco"}'
https://localhost/api/2.0/wifi/client/wireless/anonymous-identity
```

回复 2

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

请求 3

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/blizzard1/saving
```

回复 3

```
{"saving":"","success":"true","getAt":"2013-04-28 16:06:21"}
```

获取 WiFi 客户端连接 Anonymous-Identity

示例：获取 WiFi 客户端连接 Anonymous-Identity

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/anonymous-identity
```

回复

```
{"anonymous-identity":"","success":"true","getAt":"2013-04-27 10:27:19"}
```

设置 WiFi 客户端连接 pac-file

示例：将 WiFi 客户端 pac-file 设置为 /tmp/pac

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"pac-file" : "/tmp/pac"}' https://localhost/api/2.0/wifi/client/wireless/pac-file
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 pac-file

示例：获取 WiFi 客户端连接 pac-file

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/pac-file
```

回复

```
{"pac-file":"","success":"true","getAt":"2013-04-27 10:31:28"}
```

设置连接的 WiFi CA 证书

示例：将 WiFi 客户端 ca-cert 设置为 /tmp/ca-cert.pem

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"ca-cert"
: "/tmp/ca-cert.pem"}' https://localhost/api/2.0/wifi/client/wireless/ca-cert
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取连接的 WiFi CA 证书

示例：获取连接的 WiFi CA 证书

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/ca-cert
```

回复

```
{"ca-cert":"","success":"true","getAt":"2013-04-27 10:49:33"}
```

设置连接的 WiFi 客户端 CA 证书

示例：将 WiFi client-cert 设置为 /tmp/client-cert.pem

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"client-cert" : "/tmp/client-cert.pem"}'
https://localhost/api/2.0/wifi/client/wireless/client-cert
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取连接的 WiFi 客户端 CA 证书

示例：获取连接的 WiFi 客户端 CA 证书

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/client-cert
```

回复

```
{"client-cert":"","success":"true","getAt":"2013-04-27 10:51:24"}
```

设置 WiFi 客户端连接阶段 1 快速调配

示例：将 WiFi 客户端 phase1-fast-provisioning 设置为 0

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase1-fast-provisioning" : "0"}'
https://localhost/api/2.0/wifi/client/wireless/phase1-fast-provisioning
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接阶段 1 快速调配

示例：获取 WiFi 客户端连接阶段 1 快速调配

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase1-fast-provisioning
```

回复

```
{"phase1-fast-provisioning":"","success":"true","getAt":"2013-04-27 10:52:41"}
```

设置 WiFi 客户端连接 Phase1-Peapver

示例：将 WiFi 客户端 phase1-peapver 设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase1-peapver" : "1"}' https://localhost/api/2.0/wifi/client/wireless/phase1-peapver
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 Phase1-Peapver

示例：获取 WiFi 客户端连接 Phase1-Peapver

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase1-peapver
```

回复

```
{"phase1-peapver":"","success":"true","getAt":"2013-04-27 10:52:41"}
```

设置 WiFi 客户端连接 Phase2-Autheap

示例：将 WiFi 客户端连接 Phase2-Autheap 设置为 mschapv2

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-autheap" : "mschapv2"}'
https://localhost/api/2.0/wifi/client/wireless/phase2-autheap
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 Phase2-Autheap

示例：获取 WiFi 客户端连接 Phase2-Autheap

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-autheap
```

回复

```
{"phase2-autheap":"","success":"true","getAt":"2013-04-27 10:53:48"}
```

设置 WiFi 客户端连接 Phase2-Auth

示例：将 WiFi 客户端连接 Phase2-Auth 设置为 tls

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-auth" : "tls"}' https://localhost/api/2.0/wifi/client/wireless/phase2-auth
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 Phase2-Auth

示例：获取 WiFi 客户端连接 Phase2-Auth

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-auth
```

回复

```
{"phase2-auth":"mschapv2","success":"true","getAt":"2013-04-27 10:54:42"}
```

设置 WiFi 客户端连接 Phase2-Ca-Cert

示例：将 WiFi 客户端连接 Phase2-Ca-Cert 设置为 /tmp/ca-cert.pem

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-ca-cert" : "/tmp/ca-cert.pem"}'
https://localhost/api/2.0/wifi/client/wireless/phase2-ca-cert
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 Phase2-Ca-Cert

示例：获取 WiFi 客户端连接 Phase2-Ca-Cert

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-ca-cert
```

回复

```
{"phase2-ca-cert":"","success":"true","getAt":"2013-04-27 10:55:35"}
```

设置 WiFi 客户端连接 Phase2-Client-Cert

示例：将 WiFi 客户端连接 Phase2-Client-Cert 设置为 /tmp/client-cert.pem

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-client-cert" : "/tmp/client-cert.pem"}'
https://localhost/api/2.0/wifi/client/wireless/phase2-client-cert
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接 Phase2-Client-Cert

示例：获取 WiFi 客户端连接 Phase2-Client-Cert

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-client-cert
```

回复

```
{"phase2-client-cert":"","success":"true","getAt":"2013-04-27 10:57:37"}
```

设置 WiFi 客户端连接密码

示例：将 WiFi 客户端连接密码设置为 12345678

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"password" : "12345678"}' https://localhost/api/2.0/wifi/client/wireless/password
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接密码

示例：获取 WiFi 客户端连接密码

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/password
```

回复

```
{"password":"123456","success":"true","getAt":"2013-04-27 11:02:20"}
```

设置 WiFi 客户端连接私钥

示例：将 WiFi 客户端连接私钥设置为 /tmp/private-key

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"private-key" : "/tmp/private-key"}'
https://localhost/api/2.0/wifi/client/wireless/private-key
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接私钥

示例：获取 WiFi 客户端连接私钥

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/private-key
```

回复

```
{"private-key":"","success":"true","getAt":"2013-04-27 11:04:36"}
```

设置 WiFi 客户端连接私钥密码

示例：将 WiFi 客户端连接私钥密码设置为 12345678

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"private-key-password" : "12345678"}'
https://localhost/api/2.0/wifi/client/wireless/private-key-password
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接私钥密码

示例：获取 WiFi 客户端连接私钥密码

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/private-key-password
```

回复

```
{"private-key-password":"","success":"true","getAt":"2013-04-27 11:09:51"}
```

设置 WiFi 客户端连接阶段 2 私钥

示例：将 WiFi 客户端连接阶段 2 私钥设置为 /tmp/private-key

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-private-key" : "/tmp/private-key"}'
https://localhost/api/2.0/wifi/client/wireless/phase2-private-key
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接阶段 2 私钥

示例：获取 WiFi 客户端连接阶段 2 私钥

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-private-key
```

回复

```
{"phase2-private-key":"","success":"true","getAt":"2013-04-27 11:12:57"}
```

设置 WiFi 客户端连接阶段 2 私钥密码

示例：将 WiFi 客户端连接阶段 2 私钥密码设置为 12345678

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"phase2-private-key-password" : "12345678"}'
https://localhost/api/2.0/wifi/client/wireless/phase2-private-key-password
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接阶段 2 私钥密码

示例：获取 WiFi 客户端连接阶段 2 私钥密码

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/phase2-private-key-password
```

回复

```
{"phase2-private-key-password":"","success":"true","getAt":"2013-04-27 11:17:38"}
```

设置 WiFi 客户端连接密钥管理

示例：将 WiFi 客户端连接 key-mgmt 设置为 wpa-psk

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"key-mgmt" : "wpa-psk"}' https://localhost/api/2.0/wifi/client/wireless/key-mgmt
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端连接密钥管理

示例：获取 WiFi 客户端连接密钥管理

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/key-mgmt
```

回复

```
{"key-mgmt":"wpa-eap","success":"true","getAt":"2013-04-27 11:19:01"}
```

设置 WiFi 客户端 wep-tx-keyidx

示例：将 WiFi 客户端连接 wep-tx-keyidx 设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-tx-keyidx" : "1"}' https://localhost/api/2.0/wifi/client/wireless/wep-tx-keyidx
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-tx-keyidx

示例：获取 WiFi 客户端 wep-tx-keyidx

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-tx-keyidx
```

回复

```
{"wep-tx-keyidx":"","success":"true","getAt":"2013-04-27 11:21:32"}
```

设置 WiFi 客户端 auth-alg

示例：将 WiFi 客户端连接 auth-alg 设置为 open

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"auth-alg" : "open"}' https://localhost/api/2.0/wifi/client/wireless/auth-alg
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 auth-alg

示例：获取 WiFi 客户端 auth-alg

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/auth-alg
```

回复

```
{"auth-alg":"","success":"true","getAt":"2013-04-27 11:23:20"}
```

设置 WiFi 客户端配对

示例：将 WiFi 客户端配对设置为 tkip

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"pairwise" : "tkip"}' https://localhost/api/2.0/wifi/client/wireless/pairwise
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取成对的 WiFi 客户端

示例：获取成对的 WiFi 客户端

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/pairwise
```

回复

```
{"pairwise":"","success":"true","getAt":"2013-04-27 11:24:31"}
```

设置 WiFi 客户端组

示例：将 WiFi 客户端连接组设置为 tkip

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"group" :
"tkip"}' https://localhost/api/2.0/wifi/client/wireless/group
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端组

示例：获取 WiFi 客户端组

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/group
```

回复

```
{"group":"","success":"true","getAt":"2013-04-27 11:25:15"}
```

设置 WiFi 客户端 wep-key0

示例：将 WiFi 客户端链路 wep-key0 设置为 12345

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-key0" : "12345"}' https://localhost/api/2.0/wifi/client/wireless/wep-key0
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-key0

示例：获取 WiFi 客户端 wep-key0

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-key0
```

回复

```
{"wep-key0":"","success":"true","getAt":"2013-04-27 11:53:44"}
```

设置 WiFi 客户端 wep-key1

示例：将 WiFi 客户端 wep-key1 设置为 12345

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-key1" : "12345"}' https://localhost/api/2.0/wifi/client/wireless/wep-key1
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-key1

示例：获取 WiFi 客户端 wep-key1

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-key1
```

回复

```
{"wep-key1":"","success":"true","getAt":"2013-04-27 11:53:44"}
```

设置 WiFi 客户端 wep-key2

示例：将 WiFi 客户端 wep-key2 设置为 12345

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-key2" : "12345"}' https://localhost/api/2.0/wifi/client/wireless/wep-key2
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-key2

示例：获取 WiFi 客户端 wep-key2

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-key2
```

回复

```
{"wep-key2":"","success":"true","getAt":"2013-04-27 14:01:02"}
```

设置 WiFi 客户端 wep-key3

示例：将 WiFi 客户端 wep-key3 设置为 12345

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-key3" : "12345"}' https://localhost/api/2.0/wifi/client/wireless/wep-key3
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-key3

示例：获取 WiFi 客户端 wep-key3

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-key3
```

回复

```
{"wep-key3":"","success":"true","getAt":"2013-04-27 14:13:50"}
```

设置 WiFi 客户端 wep-key-type

示例：将 WiFi 客户端连接 wep-key-type 设置为 1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"wep-key-type" : "1"}' https://localhost/api/2.0/wifi/client/wireless/wep-key-type
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 wep-key-type

示例：获取 WiFi 客户端 wep-key-type

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/wep-key-type
```

回复

```
{"wep-key-type":"","success":"true","getAt":"2013-04-27 14:15:22"}
```

设置 WiFi 客户端 PSK

示例：将 WiFi 客户端 PSK 设置为 12345678

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"psk" :
"12345678"}' https://localhost/api/2.0/wifi/client/wireless/psk
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 PSK

示例：获取 WiFi 客户端 PSK

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/wireless/psk
```

回复

```
{"psk":"","success":"true","getAt":"2013-04-27 14:16:47"}
```

设置 WiFi 客户端 IPv4 方法

示例：将 WiFi 客户端 IPv4 方法设置为手动

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"method"
: "manual"}' https://localhost/api/2.0/wifi/client/ipv4/method
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv4 方法

示例：获取 WiFi 客户端 IPv4 方法

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv4/method
```

回复

```
{"method":"auto","success":"true","getAt":"2013-04-27 14:21:11"}
```

设置 WiFi 客户端 IPv4 DNS

示例：将 WiFi 客户端 IPv4 DNS 设置为 192.168.0.1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"dns" :
"192.168.0.1"}' https://localhost/api/2.0/wifi/client/ipv4/dns
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv4 DNS

示例：获取 WiFi 客户端 DNS

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv4/dns
```

回复

```
{"dns":"","success":"true","getAt":"2013-04-27 14:22:09"}
```

设置 WiFi 客户端 IPv4 地址

示例：将 WiFi 客户端 IPv4 地址设置为 192.168.0.100

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"addresses" : "192.168.0.100,24,192.168.0.1"}'
https://localhost/api/2.0/wifi/client/ipv4/addresses
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv4 地址

示例：获取 WiFi 客户端 IPv4 地址

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv4/addresses
```

回复

```
{"addresses":"","success":"true","getAt":"2013-04-27 14:26:19"}
```

设置 WiFi 客户端 IPv4 路由

示例：将 WiFi 客户端 IPv4 路由设置为 192.168.0.1

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"routes"
: "192.168.0.1,24,192.168.0.0,1"}' https://localhost/api/2.0/wifi/client/ipv4/routes
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv4 路由

示例：获取 WiFi 客户端 IPv4 路由

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv4/routes
```

回复

```
{"routes":"","success":"true","getAt":"2013-04-27 14:50:53"}
```

设置 WiFi 客户端 IPv6 方法

示例：将 WiFi 客户端 IPv6 方法设置为手动

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"method"
: "manual"}' https://localhost/api/2.0/wifi/client/ipv6/method
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 方法

示例：获取 WiFi 客户端 IPv6 方法

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/method
```

回复

```
{"method":"auto","success":"true","getAt":"2013-04-27 14:52:13"}
```

设置 WiFi 客户端 IPv6 DNS

示例：将 WiFi 客户端 IPv6 DNS 设置为 1001::1002

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"dns" :
"1001::1002"}' https://localhost/api/2.0/wifi/client/ipv6/dns
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 DNS

示例：获取 WiFi 客户端 IPv6 DNS

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/dns
```

回复

```
{"dns":"","success":"true","getAt":"2013-04-27 14:58:41"}
```

设置 WiFi 客户端 IPv6 地址

示例：将 WiFi 客户端 IPv6 地址设置为 1000::1001

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"addresses" : "1000::1001,64,1000::1"}'
https://localhost/api/2.0/wifi/client/ipv6/addresses
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 地址

示例：获取 WiFi 客户端 IPv6 地址

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/addresses
```

回复

```
{"addresses":"","success":"true","getAt":"2013-04-27 15:13:29"}
```

设置 WiFi 客户端 IPv6 路由

示例：将 WiFi 客户端 IPv6 路由设置为 1001::1002

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d '{"routes"
: "1001::1002,64,1001::1000,1"}' https://localhost/api/2.0/wifi/client/ipv6/routes
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 路由

示例：获取 WiFi 客户端 IPv6 路由

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/routes
```

回复

```
{"routes":"","success":"true","getAt":"2013-04-27 15:14:15"}
```

设置 WiFi 客户端 IPv6 ignore-auto-routes

示例：将 WiFi 客户端 IPv6 ignore-auto-routes 设置为 False

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"ignore-auto-routes" : "False"}'
https://localhost/api/2.0/wifi/client/ipv6/ignore-auto-routes
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 ignore-auto-routes

示例：获取 WiFi 客户端 IPv6 ignore-auto-routes

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/ignore-auto-routes
```

回复

```
{"ignore-auto-routes":"","success":"true","getAt":"2013-04-27 15:58:38"}
```

设置 WiFi 客户端 IPv6 ignore-auto-dns

示例：将 WiFi 客户端 IPv6 ignore-auto-dns 设置为 False

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"ignore-auto-dns" : "False"}' https://localhost/api/2.0/wifi/client/ipv6/ignore-auto-dns
```

回复

```
{"success":"true","updatedAt":"2013-04-28 16:50:13"}
```

获取 WiFi 客户端 IPv6 ignore-auto-dns

示例：获取 WiFi 客户端 IPv6 ignore-auto-dns

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/con/blizzard/ipv6/ignore-auto-dns
```

回复

```
{"ignore-auto-dns":"","success":"true","getAt":"2013-04-27 16:00:00"}
```

Wi-Fi 客户端 — 添加连接

示例：添加名称为 test 的连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/adding
```

回复

```
{"adding":"test","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端 — 更新连接

示例：更新名称为 test 的连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/update
```

回复

```
{"update":"test","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端 — 保存连接

示例：保存名称为 test 的连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/saving
```

回复

```
{"saving":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端 — 启用连接

示例：启用名称为 test 的连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/enablement
```

回复

```
{"enable":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端 — 删除连接

示例：删除名称为 test 的连接

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/removal
```

回复

```
{"remove":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端断开

示例：从 AP 断开

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/disconnection
```

回复

```
{"disconnection":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端 - 选择一个连接作为默认设置

示例：选择一个名称为 test 的连接作为默认设置

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/selection
```

回复

```
{"selection":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

Wi-Fi 客户端扫描

示例：发送扫描请求

请求

```
curl -k -X GET -H 'password: cisco123!'
https://localhost/api/2.0/wifi/client/execution/test/scan
```

回复

```
{"scan":"","success":"true","getAt":"2013-08-14 12:27:53"}
```

PCAMAP 信息

获取 MAC 地址

示例：获取 MAC 地址

请求

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/sys/pcamap/macAddr.php
```

回复

```
{"macAddr":"C0:62:6B:4A:16:70","success":"true","getAt":"2013-06-18 13:14:29"}
```

获取主板组件号

示例：获取主板组件号

请求

```
curl -k -X GET -H 'password: admin123'
https://localhost/api/2.0/sys/pcamap/mbAssemblyNum.php
```

回复

```
{"mbAssemblyNum":"FOC123","success":"true","getAt":"2013-06-18 13:16:26"}
```

获取主板序列号

示例：获取主板序列号

请求

```
curl -k -X GET -H 'password: admin123'
https://localhost/api/2.0/sys/pcamap/mbSerialNum.php
```

回复

```
{"mbSerialNum":"abc","success":"true","getAt":"2013-06-18 13:17:10"}
```

获取主板修订号

示例：获取主板修订号

请求

```
curl -k -X GET -H 'password: admin123'
https://localhost/api/2.0/sys/pcamap/mbRevisionNum.php
```

回复

```
{"mbRevisionNum":"123","success":"true","getAt":"2013-06-18 13:17:43"}
```

获取型号

示例：获取主板型号

请求

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/sys/pcamap/modelNum.php
```

回复

```
{"modelNum":"CS-E340W-G32-C-K9","success":"true","getAt":"2013-06-18 13:18:18"}
```

获取系统序列号

示例：获取系统序列号

请求

```
curl -k -X GET -H 'password: admin123'
https://localhost/api/2.0/sys/pcamap/systemSerialNum.php
```

回复

```
{"systemSerialNum":"123","success":"true","getAt":"2013-06-18 13:19:23"}
```

获取模型修订号

示例：获取模型修订号

请求

```
curl -k -X GET -H 'password: admin123'
https://localhost/api/2.0/sys/pcamap/modelRevisionNum.php
```

回复

```
{"modelRevisionNum":"123","success":"true","getAt":"2013-06-18 13:20:00"}
```

获取版本 ID

示例：获取版本 ID

请求

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/sys/pcamap/versionId.php
```

回复

```
{"versionId":"111","success":"true","getAt":"2013-06-18 13:20:38"}
```

分辨率

获取分辨率

示例 1：获取 DVI1 的分辨率

请求 1

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/resolution/DVI1
```

回复 1

```
{"DVI1":"device_name=AOC\nrelation=same-as\nsupport_resolution_list=1360x768@60,1280x768@5
9,1024x768@60,800x600@60,800x600@56,640x480@60\ncurrent_resolution=1360x768@60\nsetting_re
solution=1360x768@60\npreferred_resolution=1280x768@59\nrotation=normal*,right,inverted,le
ft\nreflection=normal*,x,y,xy","success":"true","getAt":"2013-06-13 11:20:20"}
```

示例 2：获取 VGA1 的分辨率

请求 2

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/resolution/VGA1
```

回复 2

```
{"VGA1":"device_name=None\nrelation=same-as\nsupport_resolution_list=1024x768@60,800x600@6
0,800x600@56,848x480@60,640x480@59\ncurrent_resolution=1024x768@60\nsetting_resolution=102
4x768@60\npreferred_resolution=1024x768@60\nrotation=normal*,right,inverted,left\nreflecti
on=normal*,x,y,xy","success":"true","getAt":"2013-06-13 11:22:14"}
```

示例 3：获取所有分辨率

请求 3

```
curl -k -X GET -H 'password: admin123' https://localhost/api/2.0/resolution/fullInfo
```

回复 3

```
{"fullInfo":"Screen 0: minimum 320 x 240, current 1360 x 768, maximum 8192 x 8192\n
359mm x 203mm\nCrtcs: 2\nSizes @ Refresh Rates:\n  [0] 1024 x 768 @ [60]\n  [1] 800 x 600
@ [60, 56]\n  [2] 848 x 480 @ [60]\n  [3] 640 x 480 @ [60]\nRotations: normal right
inverted left \n\nOutputs:\n  DVI1 (410mm x 230mm)\n    Modes:\n      [0] 1360 x 768 @
60*\n\n      [1] 1280 x 768 @ 59(preferred)\n\n      [2] 1024 x 768 @ 60\n\n      [3] 800
x 600 @ 60\n\n      [4] 800 x 600 @ 56\n\n      [5] 640 x 480 @ 60\n\n    Rotations:normal
right inverted left\n\n  LVDS1 (not connected)\n  DP1 (not connected)\n  VGA1 (0mm x
0mm)\n    Modes:\n      [0] 1024 x 768 @ 60*(preferred)\n\n      [1] 800 x 600 @ 60\n\n
[2] 800 x 600 @ 56\n\n      [3] 848 x 480 @ 60\n\n      [4] 640 x 480 @ 59\n\n
Rotations:normal right inverted left","success":"true","getAt":"2013-06-13 11:23:34"}
```

设置分辨率

示例 1：将 VGA1 的分辨率设置为 800x600@60

请求 1

```
curl -k -X PUT -H 'password:admin123' -H 'Content-Type: application/json' -d
'{"resolution":"VGA1,800x600@60"}' https://localhost/api/2.0/resolution/resolution
```

回复 1

```
{"success":"true","updatedAt":"2013-06-13 11:36:19"}
```

示例 2：将 VGA1 的旋转方式设置为倒转

请求 2

```
curl -k -X PUT -H 'password:admin123' -H 'Content-Type: application/json' -d
'{"rotation":"VGA1,inverted"}' https://localhost/api/2.0/resolution/rotation
```

回复 2

```
{"success":"true","updatedAt":"2013-06-13 11:39:43"}
```

示例 3：将 VGA1 的反射方式设置为正常

请求 3

```
curl -k -X PUT -H 'password:admin123' -H 'Content-Type: application/json' -d
'{"reflection":"VGA1,normal"}' https://localhost/api/2.0/resolution/reflection
```

回复 3

```
{"success":"true","updatedAt":"2013-06-13 11:41:33"}
```

示例 4：将 VGA1 和 DVI1 的关系设置为 3

请求 4

```
curl -k -X PUT -H 'password:admin123' -H 'Content-Type: application/json' -d
'{"relation":"VGA1,3,DVI1"}' https://localhost/api/2.0/resolution/relation
```

回复 4

```
{"success":"true","updatedAt":"2013-06-13 11:43:46"}
```

代理

设置 HTTP 代理

示例：设置 HTTP 代理

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"http":"http://cisco.com:8080"}' https://localhost/api/2.0/proxy/http
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取 HTTP 代理

示例：获取核心 HTTP 代理

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/proxy/http
```

回复

```
{"http":"http://cisco.com:8080","success":"true","getAt":"2013-08-14 17:13:05"}
```

设置 HTTPS 代理

示例：设置 HTTPS 代理

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"https":"http://cisco.com:8080"}' https://localhost/api/2.0/proxy/https
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取 HTTPS 代理

示例：获取 HTTPS 代理

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/proxy/https
```

回复

```
{"https":"http://cisco.com:8080","success":"true","getAt":"2013-08-14 17:13:05"}
```

设置 FTP 代理

示例：设置 FTP 代理

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"ftp":"http://cisco.com:8080"}' https://localhost/api/2.0/proxy/ftp
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取 FTP 代理

示例：获取 FTP 代理

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/proxy/ftp
```

回复

```
{"ftp":"http://cisco.com:8080","success":"true","getAt":"2013-08-14 17:13:05"}
```

设置所有代理

示例：设置所有代理

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"all":"http://cisco.com:8080"}' https://localhost/api/2.0/proxy/all
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取所有代理

示例：获取所有代理

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/proxy/all
```

回复

```
{"all":"http://cisco.com:8080","success":"true","getAt":"2013-08-14 17:13:05"}
```

设置代理旁路列表

示例：设置代理旁路列表

请求

```
curl -k -X PUT -H 'password: cisco123!' -H 'Content-Type: application/json' -d
'{"no_for":"127.0.0.1"}' https://localhost/api/2.0/proxy/no_for
```

回复

```
{"success":"true","updatedAt":"2013-04-08 13:14:51"}
```

获取代理旁路列表

示例：获取代理旁路列表

请求

```
curl -k -X GET -H 'password: cisco123!' https://localhost/api/2.0/proxy/no_for
```

回复

```
{"no_for":"127.0.0.1","success":"true","getAt":"2013-08-14 17:13:05"}
```

错误代码

表 4-3 显示错误代码及其说明。

表 4-3　　错误代码和说明

错误代码	说明
001	密码无效
002	内容类型错误
003	此资源不支持所使用的 HTTP 方法
004	非法参数
005	执行命令时出错
006	无效的资源
101	Json：超出最大栈深度
102	Json：下溢或模式不匹配
103	Json：发现异常控制字符
104	Json：语法错误，JSON 格式不正确
105	Json：UTF-8 字符格式不正确，可能编码错误
106	Json：未知错误

故障排除

本附录提供有关以下 Cisco Edge 340 系列设备问题的故障排除信息：

引导和登录

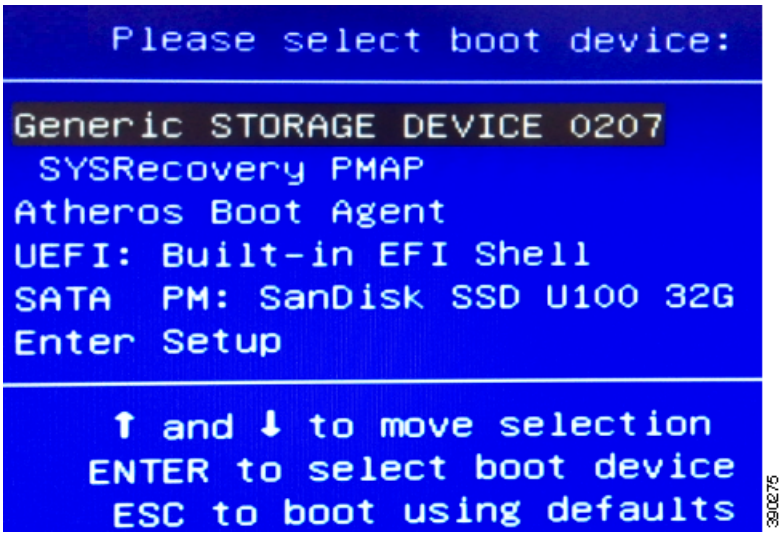
本节提供有关引导和登录问题的故障排除信息。

忘记根密码

如果您忘记了 Cisco Edge 340 系列设备的根密码，请按照下列步骤操作：

步骤 1 重新启动 Cisco Edge 340 系列设备并按 **F12** 进入 privileged 模式，如图 A-1 所示。

图 A-1　　选择 Boot Device



步骤 2　选择 **SYSRecovery PMAP** 进入 Troubleshooting 屏幕。

步骤 3　选择语言和键盘类型，对网络禁用项选择 **NO**，然后单击 **Continue** 进入 Rescue 模式。

步骤 4　选择 **Shell** 进入根 shell，然后运行命令 **chroot /mnt/sysimage/** 以启动系统。

步骤 5　输入 **passwd** 命令重置您的根密码，如下例所示：

```
Starting shell...
bash-4.2# chroot /mnt/
install/ sysimage/
bash-4.2# chroot /mnt/sysimage/
sh-4.2# passwd
Changing password for user root.
New password:
BAD PASSWORD: it is based on a dictionary word
Retype new password:
passwd: all authentication tokens updated successfully.
sh-4.2#
```

系统启动缓慢

如果系统启动超过一到两分钟，请按照下列步骤操作：

步骤 1　插入 Cisco Edge 340 系列设备的控制台电缆。

步骤 2　查看印在终端上的日志，验证此问题是否由外部设备的电力供应不足引起的。

步骤 3　如果是，请断开外部设备并重新启动系统。或者切断 Cisco Edge 340 系列设备的电源并重新启动它。

步骤 4　如果此问题不是由外部设备引起的，请记录系统日志的内容并与思科支持代表联系。

使用错误密码五次后系统锁定

如果在登录时键入错误密码超过五次，系统会被锁定 15 秒。 15 秒后屏幕解锁，您可以重新键入密码。

重置和升级

本节提供有关重置和升级问题的故障排除信息。

更新系统时出现问题

如果您在更新 Cisco Edge 340 系列设备的系统时出现问题，请执行下列操作之一：

- 如果您可以登录系统，请执行 ***.bin** 文件以自动更新系统。
- 如果您无法登录系统，请在启动 Cisco Edge 340 系列设备时按 **F12**，进入 privileged 模式并选择 **SYSRecovery PMAP** 作为引导设备，如图 A-1 所示。
- 通过 ***.bin /dev/sdb1** 命令创建 USB 恢复磁盘。在启动 Cisco Edge 340 系列设备时按 **F12**，进入 privileged 模式并选择 USB 设备作为引导设备。

在 Web GUI 中恢复 Factory Settings 失败

如果在 Web GUI 中恢复 Factory Settings 失败，请在 Maintenance 选项卡下方单击左侧窗格中的 **Restart/Reset** 重试。

显示问题

本节提供有关显示问题的故障排除信息。

无信号输出

如果您在连接显示器后未发现信号输出，并且 Cisco Edge 340 系列设备的网络状态为 Disconnected，请按照下列步骤操作：

步骤 1　检查显示器是否已通电。

步骤 2　检查 VGA 或 HDMI 连接器是否正确连接。

步骤 3　如果电源和连接都正常，请使用控制台端口排除故障。使用根权限登录系统。输入 **DISPLAY:=0.0 xrandr** 命令，检查 Cisco Edge 340 系列设备是否检测到显示器。

注　如果此问题未能解决，请重新制作该设备的镜像。

改变分辨率后屏幕变得模糊

如果您在改变分辨率后发现屏幕变得模糊，请执行下列操作之一：

- 检查 Cisco Edge 340 系列设备和显示器的连接，并重新启动系统。
- 在 Web GUI 中将当前分辨率改为新值。

网络问题

本节提供有关网络问题的故障排除信息。

连接状态在 WiFi 站点模式下未刷新

在 WiFi station 模式下，如果您在连接 AP 后发现连接状态未刷新，请在 Web GUI 中单击 **Refresh** 按钮。如果问题仍然存在，请单击左侧导航窗格中的 **Wireless** 选项以刷新屏幕。

Wake On LAN 功能无效

如果 Wake On LAN 功能无效，请执行下列操作之一：

- 验证是否在 Cisco Edge 340 系列设备上启用了 Wake On LAN 功能。
- 验证远程设备和 Cisco Edge 340 系列设备是否处于同一个广播域中。

DNS 未解析

Cisco Edge 340 系列最多支持三个 DNS 服务器。如果前三个 DNS 服务器无法解析，即使其他的 DNS 服务器有效，也不能使用。

如果 DNS 无法解析，请输入 **#vi /etc/resolve.config** 命令以编辑 **resolve.config** 文件，将前三个 DNS 服务器替换为后面的有效服务器。

第三方设备无法连接

如果 station 模式下的第三方设备无法连接到 AP 模式下的 Cisco Edge 340 系列设备，请执行下列操作之一：

- 验证第三方设备与 Cisco Edge 340 系列之间的加密和身份验证机制是否匹配。
- 验证第三方设备中的网卡是否支持 5G。除非加密和身份验证机制匹配，否则无法连接网络。

电源问题

本节提供有关电源问题的故障排除信息。

外围设备电力不足

如果外围设备的电力不足并且 Cisco Edge 340 系列设备通过以太网供电 (PoE) ,请验证 Cisco Edge 340 系列设备是否由 PoE 802.3AF 供电。如果是，将它更改为 802.3AT 模式，因为 802.3AT 提供的电源比 802.3AF 提供的电源稳定。

后面板上的 USB 端口不工作

如果仅有前面板上的两个 USB 端口正在运行，而后面板上的 USB 端口没有电源，请验证 Cisco Edge 340 系列设备是否由 PoE 供电。要启用后面板上的 USB 端口，请使用外部电源。