ASA REST API v1.2.2 简介

首次发布日期： 2014 年 12 月 16 日
修订日期： 2015 年 2 月 5 日

目录

[隐藏]

概述

此 REST API 提供用于配置传统 ASA（从 9.3.2 版开始）的基于编程模型的界面。术语“传统 ASA”是指不包含 CX 或 SourceFire 传感器，或者 NGFW（下一代防火墙）服务的集成功能的设备。另请注意，当其他安全模块与传统 ASA 配合使用时，没有面向这些模块的 API。

REST API 可在配置 ASA 时与以下现有管理界面和应用配合使用：命令行界面 (CLI)、自适应安全设备管理器 (ASDM) 和思科安全管理器 (CSM)。

REST API 1.2.2 的新增功能：

- 智能许可。
- 支持 IP 审核和其他应用检查协议（FTP、NetBIOS、RTSP、SIP、SQL*Net）。
- 能够查询 ASA 的序列号。
- REST API 1.2.2.200 版包含一个面向 CSCux92088 的修复：将 Bulk API 请求条目限制增大到 1000。

REST API 1.2.1 的新增功能：

- 对多情景模式的监控支持。
- 支持以下 ASA 功能：DHCP 服务器和中继、DNS 客户端和动态 DNS、协议超时 (PTO) 和 GTP 检测。

REST API 1.1.1 的新增功能：

- 支持基于令牌的身份验证。
- 支持以下 ASA 功能：应用检查协议（DNS over UDP、HTTP、ICMP、ICMP ERROR、RTSP、DCERPC、IP 选项）、备份和恢复、连接限制、多情景（有限支持）、NTP 和写内存命令 API。

REST API 1.0.1 的功能：

- 支持以下 ASA 功能：AAA、访问规则、故障切换、接口、侦听（永久和激活密钥许可证）、共享密钥许可证、日志记录、管理访问、监控、NAT（两次 NAT 和对象 NAT）、对象、静态路由、服务策略和站点到站点 VPN。
- Bulk API。
- Generic CLI Command Executor API，意味着可以使用 REST API 发送任何 CLI 命令。

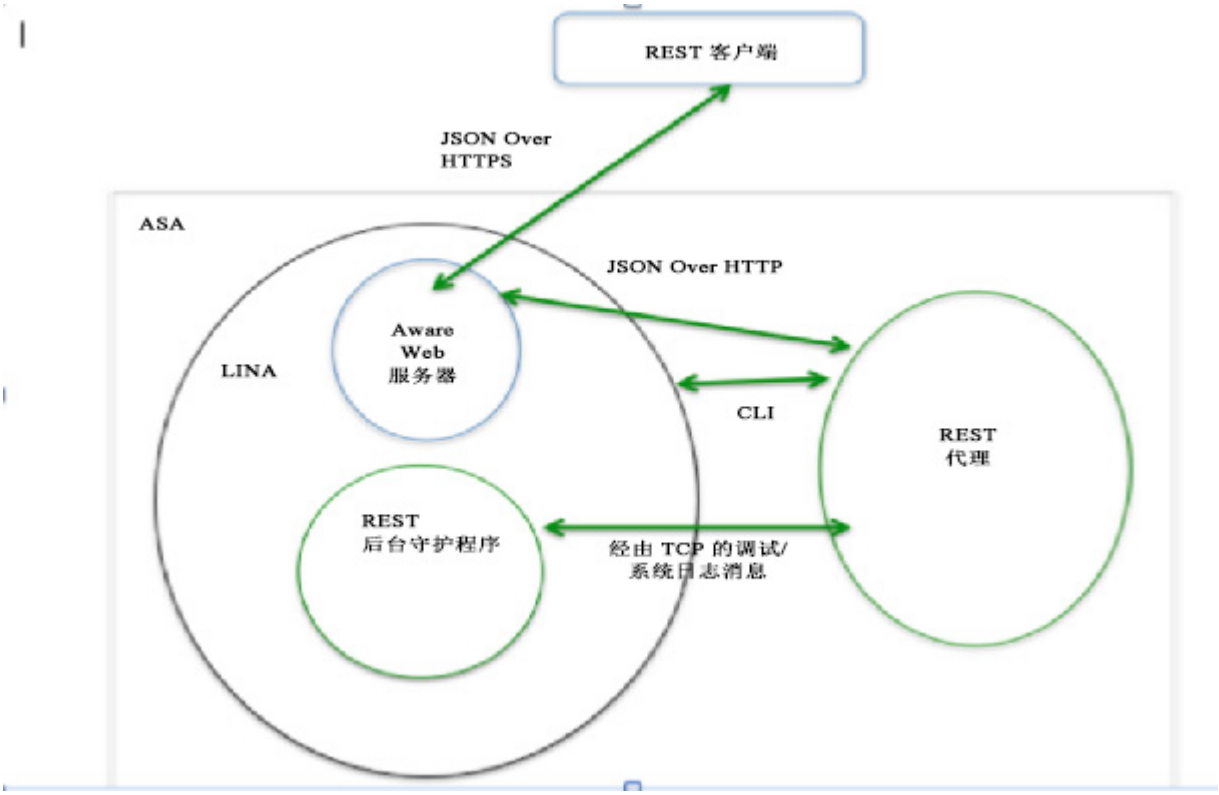
支持的平台

REST API 仅支持 5500-X 系列（包括 5585-X）和 ASAv 平台以及 Firepower 9300 ASA 安全模块；它不支持 ASA 服务模块 (ASA-SM)。有关详细信息，请参阅 ASA REST API 兼容性。

支持的模式

REST API 当前不支持在多模式下直接配置任何选项。在多情况模式下仅支持通用 CLI 命令执行器 API（CLI 透传）、令牌身份验证 API 和监控。有关详细信息，请参阅“多情景模式”一节。

高级架构



典型请求流程

以下是面向任何 REST PUT/POST/DELETE API 请求的流程：

- REST 客户端建立到 ASA 的 SSL 连接。
- REST 客户端向 ASA 发送含有基本身份验证报头的 API 请求。
- ASA HTTP 服务器验证和处理客户端请求。
- ASA HTTP 服务器使用 TCP 通道打开到 REST 代理的连接，并将 HTTP 请求写入 REST 代理。
- ASA HTTP 服务器等待 REST 代理进程的响应。
- REST 代理处理 API 请求，选取会话/用户信息，并调用 CLI 命令，请求 LINA 侦听 ASA 的“localhost”端口。REST 代理在请求中包含会话/用户信息。
- LINA 管理处理程序处理 CLI 命令并收集生成的输出。
- LINA 向 REST 代理发送对 CLI 命令请求的响应。
- REST 代理准备对 REST API 请求的响应，然后将响应发送到 ASA HTTP 服务器。
- ASA HTTP 服务器将响应转发到客户端。服务器对从 REST 代理进程收到的响应不做任何处理。

资源标识

所有资源将有一个唯一标识符“objectId”，该 ID 要么是指定类型的固有唯一名称（由用户分配），要么是从复合型唯一属性生成的散列。注意，因为 CLI 没有唯一标识符 (UID) 的概念，并且因为 REST 代理是无状态的，因此 REST 代理不可能生成明显的唯一标识符。

示例：

```
{
  "kind": "object#AccessGroup",
  "selfLink": "https://<asa_ip>/api/access/in/inside",
  "ACLName": "inside_in_acl",
 " direction": "IN",
  "interface": {
    "kind": "objectRef#Interface",
    "refLink": "https://<asa_ip>/api/interfaces/physical/GigabitEthernet0_API_SLASH_1",
    "objectId": "GigabitEthernet0_API_SLASH_1",
    "name": "inside"
  }
}
```

资源 URL：“selfLink”属性

“selfLink”属性是一个资源的完整 URL，在该对象的 JSON 定义中指定。此属性允许直接访问对象元素集合，而无需从其 objectID 构建 URL。此属性将在每个资源对象的 JSON 定义中指定。

selfLink 的 objectId 部分将采用 URL 编码，无论 selfLink 是 JSON 响应的一部分还是位置报头的一部分。

收到 API 请求后，对请求 URL 执行双重或混合编码规范化检查。如果 URL 是双重编码，则返回“400 错误请求”(400 bad request)。如果 URL 通过规范化检查，则对请求 URL 进行解码并将其发送以执行进一步处理。

注意： JSON 响应内的 objectId 从未进行 URL 编码。因此，如果使用来自 JSON 响应的 objectId 显式构建 URL（与使用 selfLink 相反），则应在对 objectId 进行相应的 URL 编码之后构建 URL。

资源类型：“kind”属性

所有 JSON 对象都有一个“kind”属性，指示对象内容的类型 - 如果对象表示一个列表，则它的 kind 属性为“collection#{type}”；否则将是某种形式的“object#{type}”或一个原始类型，如下一节所述。

示例：

kind: collection#accessPolicySet => represents a list of ACL entries
kind: object#networkobject => represents an object of type ‘networkobject’
kind: objectref#networkobject => represents a reference to an object of type ‘networkobject’
kind: IPAddress => represents a primitive resource of type ‘ipAddress’

原始类型

在与其他资源类型混合时，可以使用“kind”表示某些原始类型，例如 IP 地址、网络、FQDN、服务类型等。在这些情况下，“kind”没有符号“#”，并且将直接指定。此类资源非常简单，除了“kind”以外，仅包含指定值的“value”属性。

示例：

```
{
"kind: "IPv4Address "
"value": "1.1.1.1"
}
```

资源关联

可以从指定资源引用其他资源。引用类型有两种：

1. 通过内联对象，其中完整的引用对象是整个存在的。很少使用此方法，并且只有某些 API 才支持此方法。
2. 引用其他资源的最常用方法是通过其资源标识符，而资源标识符可能是 objectId 或 refLink。

示例：

```
{
"kind": "objectref#networkObjectGroup" ,
"refLink": "http://host/api/object/networkObjectGroups/548292" ,
"objectId":"548292"
}
OR
{
"kind": "objectref#networkObjectGroup" ,
"refLink": "http://host/api/object/networkObjectGroup/Lab%20Printers" ,
"objectId": "Lab Printers"
}
```

“rangeInfo”对象

大多数集合资源将包含一个“rangeInfo”对象，该对象提供有关集合所含项目的详细信息。GET 请求和查询 API 支持分页，并且返回的项目数量绝不大于已定义的最大数量。因此，如果您有 20000 个网络对象，您不能在一次调用中获取所有这些对象。在 API 请求中，您也可以指定偏移和从该偏移计起应在结果中返回的数量限制。此结果始终包含一个“rangeInfo”条目，指定偏移、返回的限制以及项目总数。

```
"rangeInfo": {
"offset": "integer",
"limit": "integer",
"total": "integer",
},
```

限制的最大接受值为 100。如果 REST 客户端查询超过 100 个项目，并且有超过 100 个项目可用，则仅返回 100 个项目，而“total”将指示可用项目数量。

REST API 身份验证

支持 HTTP 基本身份验证或带安全 HTTPS 传输的基于令牌的身份验证，并且每一次请求都会执行身份验证。

注意： 建议在 ASA 上使用证书颁发机构 (CA) 颁发的证书，这样 REST API 客户端就能够在建立 SSL 连接时验证 ASA 服务器证书。

要调用监控 API，需要权限 3 或更高权限。要调用 GET API，需要权限 5 或更高权限。要调用 PUT/POST/DELETE 操作，需要权限 15 或更高权限。

REST API 约定

- HTTP PUT 请求用于替换、更新或修改现有资源，而 HTTP POST 用于创建新的资源（或者 PUT 未涵盖的任何操作）。您不能使用 HTTP PUT 来创建资源。
- HTTP PUT 请求的正文必须包含资源的强制属性的完整表示。
- 一条 HTTP PUT 接收一个完整的资源。它在响应中不返回更新后的版本。如果在响应中不发送修改后的资源，则在 HTTP 报头响应中，HTTP 状态代码为 204（不是 200 OK）。
- 在适用于对资源进行部分更新的情况下，支持 HTTP PATCH。未指定的任何属性都将采用服务器的值。
- HTTP POST 请求包含要以 JSON 格式创建的新资源的详细信息。
- “创建”(Create) 请求的 HTTP POST 响应将有一个 201 返回代码以及一个在 HTTP 报头中包含新建资源的 URI 的 Location 报头。
- 自动创建的配置（资源）不支持创建和删除 REST 操作；即，没有 HTTP POST 和 DELETE 请求。例如，您无法创建或删除与日志记录有关的配置，但是可以修改 (PUT) 或检索 (GET) 该配置。
- HTTP GET 或 HTTP DELETE 都没有请求正文。
- 不支持对资源集合进行 HTTP DELETE 操作，因为您可能正在删除该 URL 识别的资源。如果该资源已被删除，则您无法创建子资源（集合中的“项目”）。
- HTTP GET 响应有一个“kind”属性，用于指示对象或对象集合的名称。
- 所有 REST API 请求和响应必须采用 JSON 格式。
- 所有 JSON 属性必须采用“CamelCase”命名约定；例如“policyType”。

- 字符串类型的 JSON 值必须使用双引号引起来；布尔或数值类型的值不需要使用双引号。布尔值要么为 true，要么为 false（小写）。
- 预期收到的每个 HTTP 请求在其 HTTP 报头中都有此“Accept: application/json”语句，指示 REST 客户端预期 REST 响应采用 JSON 格式。
- 每个 HTTP POST 请求必须包含一个 JSON 正文（一个属性）。
- HTTP 响应中的 Location 报头将包含所有 POST（创建）情景的完整 URL。
- 括号，如某个方案的 JSON 表示中的 [<items>]，表示项目列表。
- 除非另有指定，HTTP GET 返回当前配置的状态。
- 如果某个属性没有值，则是否显示该属性取决于它是否为可选属性。如果是可选属性，则可以在 HTTP GET 响应中忽略该属性。如果不是可选属性，若属性为字符串类型，则其值表示为一个空字符串，若属性为数值类型，则其值为 0 (零)。
- 支持分页，并且分页限制为能够通过 GET 或查询 API 调用获取的最大项目数量。

REST API 代码

将依据以下标准报告 HTTP 错误代码：

HTTP 报头中的 HTTP 错误代码	说明
400 错误请求 (400 Bad Request)	无效的查询参数：无法识别的参数、缺少参数或无效值。
404 未找到 (404 Not Found)	URL 与现有资源不匹配。例如，因为资源不可用，对某个资源进行的 HTTP DELETE 操作失败。
405 不允许使用此方法 (405 Method not Allowed)	不允许对只读资源使用 HTTP 谓词，例如 POST。
500 内部服务器错误 (500 Internal Server Error)	服务器错误。全部捕获任何其他故障 - 当没有其他响应代码有意义时，这会是最后的选择。

除了错误代码以外，返回的响应还可能包含一个正文，该正文包含一个错误对象，提供有关错误的详细信息。

HTTP 成功代码将依据以下标准报告：

HTTP 报头中的 HTTP 成功代码	说明
200 成功 (200 Success OK)	已使用 GET 方法成功检索资源。
201 已创建 (201 Created)	已使用 POST 方法成功创建资源。
204 无内容 (204 No Content)	已使用 PUT 或 PATCH 方法成功更新资源，或者已成功删除 (DELETE)。

JSON 错误/警告响应方案

```
{
  "level"  : "string",
  "code"   : "string",
  "context": "string",
  "details": "string"
}
```

属性	类型	说明
level	字符串	“错误”、“警告”或“信息”。
code	字符串	响应代码，例如“READ-ONLY-FIELD”，或者某个功能的特定代码。
context	字符串	此错误/警告/信息响应所要应用到的属性的名称。
details	字符串	此错误/警告/信息响应的详细信息。

ASA 中的 REST API 代理

安装和启用 ASA REST API 代理

REST API 代理不随 cisco.com 上的 ASA 映像一起发布。即提供的 ASA 映像不包括 REST API 插件包。REST API 软件包下载到闪存并使用“rest-api image”命令安装。使用“rest-api agent”命令启用 REST API 代理。

在多情景模式中，REST API 代理命令仅在系统情景中可用。

注意： REST API 代理是基于 Java 的应用。Java Runtime Environment (JRE) 捆绑在 REST API 代理软件包中。

“rest-api image”命令

如果出现问题，此命令将执行兼容性/验证检查并通知您。如果所有检查都通过，则它将安装 REST API 映像。要卸载，请使用命令的“no”形式。

```
[no] rest-api image disk0:/<package>
```

image - 使用此关键字在 ASA 上安装/卸载 REST API 映像；提供目标（在本例中为表示 ASA 闪存的“disk0:”）和 REST API 映像包的名称。

安装/更新 REST API 软件包不会触发 ASA 重启。

此配置将保存在启动配置文件中。

示例

此示例从 TFTP 服务器下载并安装 REST API 软件包：

copy tftp://<tftpserver>/asa-restapi-121-lfbff-k8.SPA disk0:
rest-api image disk0:/asa-restapi-121-lfbff-k8.SPA

支持的模式

单/多情景，路由/透明

REST API 代理需要单独启动

- 启用 HTTP 服务器并通过管理接口让客户端连接：
  http server enable
  http 0.0.0.0 0.0.0.0 <mgmt interface nameif>
- 配置（静态）路由。
- 如果启用了命令授权，则确保具有权限级别 15 的本地用户“enable_1”可用（REST API 代理使用此帐户与 ASA 进行通信）：
  username enable_1 password <pass> encrypted privilege 15

“rest-api agent”命令

要在安装 REST API 映像之后启用 REST API 代理，请使用“rest-api agent”命令。要禁用 REST API 代理，请使用此命令的“no”形式。

```
[no] rest-api agent
```

agent - 在 ASA 上启动 REST API 代理进程。

必备条件：必须启用 HTTP 服务器才能让 REST API 代理正常工作。

当 REST API 代理启用时，“/api”URL 请求将从 ASA HTTP 服务器重定向到 REST API 代理。

支持的模式

单/多情景，路由/透明

“show rest-api agent”命令

“show rest-api agent”命令显示 REST API 代理的当前状态：

```
ciscoasa(config)# show rest-api agent
The REST API Agent is currently enabled
```

或

```
ciscoasa(config)# show rest-api agent
The REST API Agent is currently disabled
```

当禁用 REST API 代理时，“/api”URL 请求被 ASA HTTP 服务器拒绝，并返回 404 状态代码响应。

支持的模式

单/多情景，路由/透明

“show version”命令

“show version”命令的输出列出 REST API 代理的版本：

```
ciscoasa(config)# show version
Cisco Adaptive Security Appliance Software Version 9.4(1)
REST API Agent Version <version number>
```

REST API 代理调试

“debug rest-api”命令可在 CLI 终端中启用 REST API 代理调试跟踪。

当调用时，命令会触发从 REST API 后台守护程序到 REST API 代理的一条消息，以启用并转发调试日志。随后，REST API 代理通过 TCP 将日志转发到 REST API 后台守护程序，并且这些日志会在 CLI 会话期间显示。当 CLI 会话关闭时，或者当发布“no debug rest-api”命令时，REST API 后台守护程序会通知 REST API 代理禁用会话的日志记录。

```
debug rest-api [agent | cli | client | daemon | process | token-auth] {event, error}
```

agent - 面向 REST API 代理操作的调试信息。

cli - 面向 REST API CLI 后台守护程序到 REST API 代理通信的调试信息。

client - 面向在 REST API 客户端和 REST API 代理之间路由邮件的调试信息。

daemon - 面向 REST API 后台守护程序到 REST API 代理通信的调试信息。

process - 面向 REST API 代理启动/停止处理的调试信息。

token-auth - 面向 REST API 令牌身份验证处理的调试信息。

支持的模式

单/多情景，路由/透明

Show 命令的输出

“debug rest-api agent is enabled”或“debug rest-api agent is disabled”

“debug rest-api cli is enabled”或“debug rest-api cli is disabled”

“debug rest-api daemon is enabled”或“debug rest-api daemon is disabled”

“debug rest-api http is enabled”或“debug rest-api http is disabled”

“debug rest-api process is enabled”或“debug rest-api process is disabled”

“debug rest-api token-auth is enabled”或“debug rest-api token-auth is disabled”

系统日志

系统日志 #342001

说明/原因/概述

REST API 代理已成功启动。

默认级别：

7

系统日志编号和格式：

%ASA-7-342001: REST API Agent started successfully.

说明：

在 REST API 客户端可以配置 ASA 之前，必须成功启动 REST API 代理。

建议/操作：

无

系统日志 #342002

说明/原因/概述

REST API 代理失败。

默认级别：

3

系统日志编号和格式：

%ASA-3-342002: REST API Agent failed, reason: *<reason>*

<reason> REST API 代理失败的原因。

说明：

REST API 代理可能由于各种原因无法启动或者崩溃。其中一个原因可能是 REST API 代理需要的内存不足。另一个原因可能是用于启用/禁用 REST API 代理的消息传输失败。

建议/操作：

管理员应尝试禁用（“no rest-api agent”），然后使用“rest-api agent”重新启用 REST API 代理。

系统日志 #342003

说明/原因/概述

REST API 代理已失败并正在重新启动的通知。

默认级别：

3

系统日志编号和格式：

%ASA-3-342003: REST API Agent failure notification received.Agent will be restarted automatically.

说明：

REST API 代理已失败，并且正尝试重新启动此代理。

建议/操作：

无

系统日志 #342004

说明/原因/概述

REST API 代理在数次尝试之后仍然无法成功启动。

默认级别：

3

系统日志编号和格式：

%ASA-3-342004: Failed to automatically restart the REST API Agent after five unsuccessful attempts.Use the 'no rest-api agent' and 'rest-api agent' commands to manually restart the Agent.

说明：

REST API 代理在多次尝试后仍然无法启动。

建议/操作：

请参阅系统日志 %ASA-3-342002（如果记录）以确定失败原因。尝试禁用（“no rest-api agent”），然后再次重新启用 REST API 代理（“rest-api agent”）。

系统日志 #342005

说明/原因/概述

REST API 映像已成功安装。

默认级别：

7

系统日志编号和格式：

%ASA-7-342005: REST API image has been installed successfully.

说明：

必须成功安装 REST API 映像之后才能启动 REST API 代理。

建议/操作：

无

系统日志 #342006

说明/原因/概述

无法安装 REST API 映像。

默认级别：

3

系统日志编号和格式：

%ASA-3-342006: Failed to install REST API image, reason: <reason>

<reason> REST API 代理安装失败的原因。

说明：

REST API 映像可能会因以下原因而无法安装：

版本检查失败 | 映像验证失败 | 找不到映像文件 | 闪存空间不足 | 安装失败

建议/操作：

管理员应修复失败原因并尝试使用“rest-api image <image>”重新安装映像。

系统日志 #342007

说明/原因/概述

REST API 映像已成功卸载。

默认级别：

7

系统日志编号和格式：

%ASA-7-342007: REST API image has been uninstalled successfully.

说明：

必须成功卸载旧的 REST API 映像之后才能安装新的映像。

建议/操作：

无

系统日志 #342008

说明/原因/概述

无法卸载 REST API 映像。

默认级别：

3

系统日志编号和格式：

%ASA-3-342008: Failed to uninstall REST API image, reason: <reason>.

说明：

REST API 映像可能会因以下原因而无法卸载：

卸载失败 | 已启用 REST 代理

建议/操作：

在卸载 REST API 映像之前，管理员应禁用 REST 代理。

带外更改处理

如果在处理 REST API 请求时发现带外配置更改，则在尝试处理请求之前将配置重新加载到 REST API 代理。

支持的 ASA 功能

AAA

AAA API 支持身份验证、授权和命令权限的 AAA 相关功能。

尚不支持 AAA 服务器组和记帐。

身份验证

api/aaa/authentication

配置网络身份验证。

限制：

当前仅支持“本地”(LOCAL) 服务器组。

授权

api/aaa/authorization

配置网络授权。

限制：

当前仅支持“本地”(LOCAL) 服务器组。

命令权限

api/aaa/commandprivileges

配置本地命令权限级别。

限制：

不适用

访问规则

/api/access

使用访问 API 配置路由和透明防火墙模式下的网络访问。

使用 REST API，您可以 GET（获得）访问组访问规则。为特定接口和方向创建第一个访问规则时，系统会自动创建访问组。同样地，当删除最后一个访问规则时，也会删除访问组。同时还支持全局访问规则。

使用 REST API，您可以 GET（获得）/POST（发布）/PUT（推送）/PATCH（修补）/DELETE（删除）访问规则。访问 URI 是按接口和方向分组的，并且有一个通用 URI 根 /access。

限制：

无限制；支持与 ASDM 应用相同的功能。

备份和恢复

使用此 API 在 ASA 中备份配置: **/api/backup**

使用此 API 恢复 ASA 中的配置: **/api/restore**

限制：

不适用

DHCP

使用这些 API 配置 DHCP 客户端和 DHCP 中继：**/api/dhcp**

限制：

在透明模式下不支持 DHCP 中继。

DNS

使用以下 API 配置 DNS：**/api/dns**

限制：

不适用

故障切换

/api/failover

限制：

不适用

接口

有六组 API 可用于提供与接口相关的配置。分别面向物理接口（**/api/interfaces/physical**）、VLAN 接口（**/api/interfaces/vlan**）、端口-通道接口（**/api/interfaces/portchannel**）、冗余接口（**/api/interfaces/redundant**）、网桥组接口（BVI）（**/api/interfaces/bvi**）（在透明模式下可用）和全局接口设置（**/api/interfaces/setup**）。

限制：

不适用

IP 审核

/api/firewall/ipaudit

限制：

不适用

许可

永久许可证和激活密钥许可证

api/licensing/activation

这些 API 用于查看和配置基于密钥的许可证。永久许可证通过 GET 方法获取，正如激活许可证一样。

限制：

在更改激活许可证配置之后必须手动重新加载 ASA；例如，添加了新的许可证，或者启用/禁用许可证。

共享许可证

api/licensing/shared

这些 API 用于支持配置共享许可证设置，包括客户端或服务器共享许可证，如当前生效的许可证所定义。

限制：

不适用

智能许可证

api/licensing/smart

此 API 用于配置智能许可证及监控支持平台上的授权。

注意，即使令牌 ID 无效，到 api/licensing/smart/asav/register 的 POST 请求也会返回代码 201（成功）。ASAv 本身无法验证令牌 ID；它依赖许可证服务器进行验证。但是，在 ASAv 接受令牌 ID 后，对许可证服务器调用的发布和处理是异步的。

限制：

不适用

日志记录

系统日志服务器

api/logging/syslogserver

此 API 用于支持系统日志服务器的 CRUD 操作。

限制：

不适用

系统日志服务器设置

/api/logging/syslogserversettings

此 API 用于支持系统日志服务器的高级设置，包括配置日志记录队列和在系统日志服务器关闭时允许 TCP 日志记录。

限制：

不适用

系统日志消息配置

/api/logging/syslogconfig

此 API 用于支持配置系统日志消息的详细信息，包括级别和启用/禁用消息。

限制：

不适用

系统日志消息设置

/api/logging/syslogconfigsettings

此 API 用于支持配置系统日志消息的设置，例如，包含非 EMBLEM 格式的设备 ID、时间戳或集群 IP（适用时）。

限制：

不适用

NetFlow 配置

/api/logging/netflow

此 API 用于支持 Netflow 配置的 CRUD 操作。

限制：

不适用

Netflow 收集器设置

此 API 用于支持 Netflow 收集器设置的 CRUD 操作。

限制：

不支持含有 Netflow 的服务策略规则

管理访问

一般管理访问

api/mgmtaccess

使用此 API 配置与 telnet、SSH 和 HTTPS (ASDM) 有关的 ASA 访问设置。

限制：

不适用

主机数

/api/mgmtaccess/hosts

允许对用于 telnet、SSH 和 HTTPS (ASDM) 连接的管理访问主机进行 CRUD 操作。

限制：

不适用

监控

/api/monitoring/

这些 API 可用于获取运行状况、性能和 REST API 代理监控统计信息。

在多情景模式下，要获取指定情景（包括系统情景）的监控统计信息，请附加一个含有“context”参数的查询：https: //<asa_admin_context_ip>/api/cli?context=<context_name>。如果在监控请求中不存在“context”查询参数，则 REST API 代理会尝试自行确定目标情景。对于仅在系统情景中可用的资源（例如，CPU 进程使用），请求会被定向到系统情景。其余命令会被定向到管理情景。

限制：

不适用

多情景模式

多情景模式支持限制为通用 CLI 命令执行器 API、令牌身份验证 API 和监控。此时，除了通过 CLI 命令执行器 API 以外，REST API 不支持在多情景模式下配置 ASA。

注意：

- 可以在多情景模式下启用 REST API 代理。REST API 代理 CLI 仅存在于系统情景中。
- 如果使用令牌身份验证，则在发出任何 REST API 命令之前，需要通过 https://<asa_admin_context_ip>/api/tokenservices 获得身份验证令牌。

  注意，为管理情景接收的令牌也可用于配置/监控任何其他情景。
- 通用 CLI 命令执行器 API 可用于将任何情景配置为 https://<asa_admin_context_ip>/api/cli?context=<context_name>。如果“context”查询参数不存在，则请求会被定向到管理情景。
- 如果在监控请求中不存在“context”查询参数，则 REST API 代理会尝试自行确定目标情景。对于仅在系统情景中可用的资源（例如，CPU 进程使用），请求会被定向到系统情景。其余命令会被定向到管理情景。

限制：

REST API 命令仅在系统情景中可用。当 ASA 从单情景模式切换到多情景模式时必，须重新启动 REST API 代理，反之亦然。

NTP

/api/devicesetup/ntp/

限制：

不适用

NAT

/api/nat

NAT API 支持 TwiceNAT（也称为手动 NAT）和 ObjectNAT（也称为 AutoNAT）。每个 NAT 类型都有唯一 URI。完全支持 Before AutoNAT 和 After AutoNAT（路由和透明模式）。

API 也包含用于配置 InterfacePAT、DynamicPAT（隐藏）和 PAT Pool 的属性。

不支持在同一列表中显示所有 NAT 类型（Twice 和 Auto）。

ObjectNAT (AutoNAT)

限制：

不支持创建含有 NAT 规则的内联网络对象。要为现有网络对象创建对象 NAT，源地址应指向要转换的网络对象。

TwiceNAT（手动 NAT）

Before NAT 和 After NAT 分成两个列表，并且分别有各自的 URI。不支持将 Before NAT 规则移动到 After NAT 规则，反之亦然。

限制：

不适用

对象

/api/objects/

对象是可重复使用的配置组件。可以在 ASA 配置中定义和使用对象来代替内联 IP 地址、服务、名称等。REST API 为以下类型的对象提供支持：

- 扩展 ACL。与访问规则类似，在创建它们的第一个 ACE 时会创建扩展 ACL，在删除最后一个 ACE 时也会删除扩展 ACL。
- 本地用户和用户组。
- 网络对象和对象组。
- 网络服务（包括预定义的网络服务）和服务器组。不能更改或删除预定义的服务对象。它们可用于剪切和粘贴内联服务，或者在创建服务对象时使用。
- 正则表达式。
- 安全对象组。

- 时间范围。
- 用户对象。

与 ASDM 类似，REST API 支持在访问、NAT 和服务策略规则中使用内联对象和对象组。

限制：

仅支持本地用户。

协议超时

/api/firewall/timeouts

这些 API 用于配置全局协议和会话超时。

限制：

不适用

路由

/api/routing/static

目前仅支持静态路由。

限制：

不适用

服务政策

/api/servicepolicy/

REST API 支持以下协议检测：

DCERPC
DNS over UDP
FTP
HTTP
ICMP
ICMP ERROR
IP 选项
NetBIOS
RTSP
SIP
SQL*Net

作为单独的资源 URI 支持正则表达式和连接限制。

限制：

不适用

VPN

/api/vpn/

在 REST API 中仅支持站点到站点 VPN 配置。支持 IPv4 和 IPv6。不支持站点到站点 VPN 监控。

限制：

仅支持站点到站点配置。不支持如在 ASDM 中看到的证书管理。

特殊 API

Bulk API

为方便起见，此 API 让您将面向不同资源的多个 POST、PUT、PATCH 和 DELETE 请求分组到一个 HTTP POST 调用。这意味着您进行一次请求即可以修改多个资源，并且按在负载中出现的顺序处理包含的每个请求。但是请注意，批请求的内容被视为原子配置更改：如果其内含的任何请求失败，则整个负载会被拒绝，并且不对 ASA 配置做出任何更改。

请求负载和响应的详细信息如下：

POST URL：/api

请求负载格式：[{}, {}, {}, ...] 其中每个 JSON 对象是一个操作包装器：

```
{
 method:<HTTP_REQUEST_METHOD_FOR_RESOURCE >,
 resourceUri:<RESOURCE_URI>,
 data:<POST_CONTENT_FOR_THIS_URI_IF_APPLICABLE>
}
```

属性	类型	说明
method	字符串	支持“GET”、“POST”、“DELETE”、“PATCH”调用。
resourceUri	字符串	资源 URI，如果独立做出请求。
data	字符串	作为原始正文发送的 JSON 数据，如果单独做出请求。对于“DELETE”方法，这不是必需的。

批请求响应格式为：

```
{
entryMessages:[{}, {}, ...],
commonMessages: []
}
```

entryMessages 是一个对象数组，每个对象对应一个批请求条目。

通用 CLI 命令执行器 API

此特殊 API 可使用单行或多行 CLI 命令并将 CLI 输出作为 API 响应呈现。

POST URL： **/api/cli**

请求负载的格式：

```
{
"commands": ["command-1", "command-2",…, "command-n"]
}
```

响应格式：

```
{
"response": ["command-1 response", "command-2 response",…, "command-n response"]
}
```

限制

在 CLI 透传中不支持调试命令。所有调试命令都是针对每个终端会话，并不是一个全局配置。因此，如果经由 CLI 透传发送调试命令，则可能会返回一个错误或成功响应，但是对设备没有任何影响。

令牌身份验证 API

REST API 客户端需要向“/api/tokenservices”发送 POST 请求，在基本身份验证报头中包含用户信息，从而为该用户获取令牌。之后，REST API 客户端可以在任何随后发生的 REST API 调用的“X-Auth-Token”请求报头中使用此令牌。该“令牌”将一直有效，直到“DELETE /api/tokenservices/<token>”请求使用基本身份验证报头中的用户信息显式地使该令牌失效，或者直到会话超时。

POST URL: /api/tokenservices

请求负载为空。基本身份验证报头应包含用户信息。

响应可能如下：

原因	HTTP 状态代码
AAA 验证失败/授权报头不存在。	401 未授权 (401 Unauthorized)
身份验证成功。	204 无内容 (204 No Content) + X-Auth-Token <token id>（报头）
无法从报头获取用户名/密码，或者任何其他完整性检查失败。	400 错误请求 (400 Bad Request)
达到最大会话数量。 **注意：** 每个情景的最大会话数量为 25。	503 服务不可用 (503 Service unavailable)

要删除令牌，请向 URL: /api/tokenservices/<token> 发出 DELETE 请求

请求负载为空。基本身份验证报头应包含用户信息。

响应可能如下：

原因	HTTP 状态代码
AAA 验证失败/无效令牌。	401 未授权（401 Unauthorized）
成功。	204 无内容（204 No Content）
无法从报头获取用户名/密码，或者任何其他完整性检查失败。	400 错误请求（400 Bad Request）。

注意：

- 现有系统日志 605004 和 605005 用于创建/删除令牌。
- 现有系统日志 109033 用于身份验证服务器请求进行“质询”以通知用户它“不受支持”的情况。
- 当收到 REST API 请求时，请先检查是否有“X-Auth-Token”报头；如果没有，则服务器会回退到基本身份验证。
- 令牌身份验证不符合 Oauth 2.0 RFC 6749 规范。
- 生成的令牌数据库将存储在 ASA 的内存中，并且在整个故障切换配对或集群中者不会重复。换言之，如果在故障切换配对内发生故障切换，或者集群主设备更改，则需要重新执行身份验证。
- 对于多情景设备，接收管理情景的令牌，然后可以将该令牌用于配置其他任何情景。

写内存 API

REST API 调用对 ASA 配置所做的更改不会持久存留在启动配置中；也就是说，只将该更改分配到正在运行的配置中。此“写内存 API”可用于将当前运行的配置保存到启动配置。

POST URL：/api/commands/writemem

请求负载为空。

REST API 在线文档

在线文档界面（“Doc-UI”）将用户界面的功能与包含在嵌入式 API 文档中的所有信息组合在一起。Doc-UI 可以在以下任何浏览器中运行：Chrome（当前版本）、Firefox（当前版本）、Internet Explorer 9+、Safari 5.1+、Opera（当前版本）。旧版本可能会正常运行，但 Internet Explorer 8 及以下版本不会正常运行。

必须启用 REST API 代理才能访问 Doc-UI；可从 `https://<asa management interface ip>/doc/` 访问 Doc-UI（注意，要访问 Doc-UI，末尾的“/”必不可少）。

注意： 当访问本地 REST API 文档页面时，您的浏览器会向这些页面的 ASA 发送访问请求，并且也会向各个 Web 位置请求某些 jQuery 和 JSON 文件。`https://cdnjs.cloudflare.com` 是其中一个位置。

但是，当通过 ASA 并启用 FirePOWER 服务时，如果配置了“类别：广告门户”(Categories: ad portal) 阻止过滤器，则此类请求可能被 FirePOWER 模块阻止。要取消对 `cloudflare` 网站的阻止，请创建一个显式允许此网站的访问控制规则，并将其放在包含阻止“广告门户”应用条件的规则之上。

请参阅 http://www.cisco.com/c/en/us/support/docs/security/firesight-management-center/117956-technote-sourcefire-00.html#anc9 以了解有关排除网站/Web 应用，避免其因 URL 过滤或应用控制而被阻止的信息。

脚本类型

可从 Doc-UI 生成三种类型的脚本，从而实现 REST API 操作自动化：Javascript、Python 和 Perl。

使用生成脚本的必备条件

JavaScript 脚本需要安装 Node.js，可从 http://nodejs.org/ 下载。Node.js 让您就像使用命令行脚本（例如 Python 或 Perl）一样使用通常为浏览器编写的 JavaScript 应用。只需遵循安装说明，然后使用以下命令运行您的脚本：

```
node script.js
```

Python 脚本要求您安装 Python，可在 https://www.python.org/ 找到。安装 Python 后，即可使用以下命令运行您的脚本：

```
Python script.py <username> <password>
```

Perl 脚本需要某些额外的设置。您需要五个组件：Perl 本身和四个 Perl 库：

Perl，可在 http://www.perl.org/ 找到

Bundle::CPAN，可在 http://search.cpan.org/~andk/Bundle-CPAN-1.861/CPAN.pm 找到

REST::Client，可在 http://search.cpan.org/~mcrawfor/REST-Client-88/lib/REST/Client.pm 找到

MIME::Base64，位于 http://perldoc.perl.org/MIME/Base64.html

JSON，可在 http://search.cpan.org/~makamaka/JSON-2.90/lib/JSON.pm 找到

以下是 Macintosh 的一个 Perl 安装示例：

```
Boot strapping for MAC:
$ sudo perl -MCPAN e shell
cpan> install Bundle::CPAN
cpan> install REST:: Client
cpan> install MIME::Base64
cpan> install JSON
```

在安装依赖项后，即可使用以下命令运行脚本：

```
perl script.pl <username> <password>
```

法律信息

本手册中有关产品的规格和信息如有更改，恕不另行通知。本手册中的所有声明、信息和建议均准确可靠，但我们不为其提供任何明示或暗示的担保。用户必须承担使用产品的全部责任。

随附产品的软件许可和有限担保在随产品一起提供的信息包中提供，且构成本文的一部分。如果您无法找到软件许可或有限担保，请与思科代表联系以获取副本。

思科所采用的 TCP 报头压缩是加州大学伯克莱分校 (UCB) 开发的一个程序的改版，是 UCB 的 UNIX 操作系统公共域版本的一部分。保留所有权利。版权所有 1981，加州大学董事会。

无论在该手册中是否作出了其他担保，来自这些供应商的所有文档文件和软件都按“原样”提供且仍有可能存在缺陷。思科和上述供应商不承诺所有明示或暗示的担保，包括（但不限于）对特定用途的适销性、适用性、非侵权性以及因交易、使用或商业惯例所衍生的担保。

在任何情况下，对于任何间接、特殊、连带发生或偶发的损坏，包括（但不限于）因使用或无法使用本手册而导致的任何利润损失或数据损失或损坏，思科及其供应商概不负责，即使思科及其供应商已获知此类损坏的可能性也不例外。

本文档中使用的任何互联网协议 (IP) 地址和电话号码并非实际地址和电话号码。本文档中所含的任何示例、命令显示输出、网络拓扑图和其他图形仅供说明之用。说明性内容中用到的任何真实 IP 地址或电话号码纯属巧合，并非有意使用。

所有打印副本和软拷贝均被视为非受控副本，应以原始在线版本为最新版本。

思科在全球设有 200 多个办事处。www.cisco.com/go/offices 中列有各办事处的地址、电话和传真。

思科商标

思科和思科徽标是思科和/或其附属公司在美国和其他国家/地区的商标或注册商标。要查看思科商标的列表，请访问此 URL：www.cisco.com/go/trademarks。文中提及的第三方商标为其相应所有者的财产。“合作伙伴”一词的使用并不意味着思科和任何其他公司之间存在合作伙伴关系。(1110R)