iControl® REST API User Guide

Version 15.1

Contents

REST

About Representational State Transfer

Representational State Transfer (REST) describes an architectural style of web services where clients and servers exchange representations of resources. The REST model defines a resource as a source of information, and also defines a representation as the data that describes the state of a resource. REST web services use the HTTP protocol to communicate between a client and a server, specifically by means of the POST, GET, PUT, and DELETE methods, to create, read, update, and delete elements or collections. In general terms, REST queries resources for the configuration objects of a BIG-IP® system, and creates, deletes, or modifies the representations of those configuration objects. xyz

The iControl® REST implementation follows the REST model by:

- Using REST as a resource-based interface, and creating API methods based on nouns.
- Employing a stateless protocol and MIME data types, as well as taking advantage of the authentication mechanisms and caching built into the HTTP protocol.
- Supporting the JSON format for document encoding.
- Representing the hierarchy of resources and collections with a Uniform Resource Identifier (URI) structure.
- Returning HTTP response codes to indicate success or failure of an operation.
- Including links in resource references to accommodate discovery.

Important changes in iControl REST API

This version of iControl® REST includes the changes described here that may impact existing iControl REST scripts written for version 11.6. The changes are described as `tmsh` commands.

Changes in the BIG-IP DNS module:

configurationModule->cli_cmd->@{id:gtm-pool-member}->keyword->@{id:order}->$->id	Changed: <codeph>order</codeph> TO <codeph>member-order</codeph>
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:max-address-returned}	Removed: "<keyword id="max-address-returned" attribute="answers_to_return"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:canonical-name}	Removed: "<keyword id="canonical-name" attribute="cname"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:fallback-ipv4}	Removed: "<keyword id="fallback-ipv4" attribute="fallback_ip"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:fallback-ipv6}	Removed: "<keyword id="fallback-ipv6" attribute="fallback_ipv6"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:monitor}	Removed: "<keyword id="monitor" attribute="monitor_rule" parser="gtm::pool-monitor-rule"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-bps}	Removed: "<keyword id="limit-max-bps" attribute="limits.bits_per_sec"/>"

configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-bps-status}	Removed: "<keyword id="limit-max-bps-status" attribute="limits.bits_per_sec_enabled" cli_enum="cli_enable_disable"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-pps}	Removed: "<keyword id="limit-max-pps" attribute="limits.pkts_per_sec"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-pps-status}	Removed: "<keyword id="limit-max-pps-status" attribute="limits.pkts_per_sec_enabled" cli_enum="cli_enable_disable"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-connections}	Removed: "<keyword id="limit-max-connections" attribute="limits.connections"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->keyword->@{id:limit-max-connections-status}	Removed: "<keyword id="limit-max-connections-status" attribute="limits.connections_enabled" cli_enum="cli_enable_disable"/>"
configurationModule->cli_cmd->@{id:gtm-pool}->association->@{id:members}	Removed: "<association id="members" command="gtm-pool-member"> <attribute set="gtm_pool_member.pool_name" from="gtm_pool.name"/> </association>"
configurationModule->cli_cmd->@{id:gtm-wideip}->keyword->@{id:ipv6-no-error-response}->$->id	Changed: "ipv6-no-error-response" TO "failure-rcode-response"
configurationModule->cli_cmd->@{id:gtm-wideip}->keyword->@{id:ipv6-no-error-neg-ttl}->$->id	Changed: "ipv6-no-error-neg-ttl" TO "failure-rcode-ttl"
configurationModule->cli_cmd->@{id:gtm-wideip}->keyword->@{id:last-resort-pool}	Removed: "<keyword id="last-resort-pool" attribute="last_resort_pool"/>"
configurationModule->cli_cmd->@{id:gtm-distributed-app}->association_list	Removed: "<association_list id="wideips" target="gtm_application_wideip.wip_name"> <attribute set="gtm_application_wideip.application_name" from="gtm_application.name"/> </association_list>"

Changes in the LTM module:

configurationModule->cli_cmd->@{id:urldb_feed_list}->keyword->@{id:file}	Removed: "<keyword id="file" attribute="custom_urldb_file"/>"
configurationModule->cli_cmd->@{id:profile-diameter}->keyword->@{id:subscriber-aware}	Removed: "<keyword id="subscriber-aware" attribute="subscriber_aware" cli_enum="cli_enable_disable"/>"
configurationModule->cli_cmd->@{id:profile-radius}->keyword->@{id:subscriber-aware}->$->id	Changed: "subscriber-aware" TO "subscriber-discovery"
configurationModule->cli_cmd->@{id:profile-radius}->keyword->@{id:subscriber-id-type}	Removed: "<keyword id="subscriber-id-type" attribute="subscriber_id_type"/>"
configurationModule->cli_cmd->@{id:profile-tcp}->keyword->@{id:nagle}->$->cli_enum	Removed: "cli_enable_disable"
configurationModule->cli_cmd->@{id:profile-classification}->keyword->@{id:description}	Removed: "<keyword id="description" attribute="description"/>"

configurationModule->cli_cmd->@{id:profile-classification}->keyword->@{id:smtp-server}	Removed: "<keyword id="smtp-server" attribute="smtp_config_name"/>"
configurationModule->cli_cmd->@{id:dns-tsig-key}->keyword->@{id:algorithm}	Removed: "<keyword id="algorithm" attribute="algorithm" cli_enum="tmm_dns_tsig_key_algorithm_t"/>"
configurationModule->cli_cmd->@{id:dns-tsig-key}->keyword->@{id:secret}->$->id	Removed: "<keyword id="secret" attribute="secret"/>"
configurationModule->cli_cmd->@{id:dns-cache}->keyword->@{id:dnssec-on-miss}	Removed: "<keyword id="dnssec-on-miss" attribute="dnssec_on_miss" cli_enum="cli_yes_no"/>"
configurationModule->cli_cmd->@{id:dns-cache-resolver}->keyword->@{id:dnssec-on-miss}	Removed: "<keyword id="dnssec-on-miss" attribute="dnssec_on_miss" value="no"/>"
configurationModule->cli_cmd->@{id:dns-cache-resolver-validator}->keyword->@{id:dnssec-on-miss}	Removed: "<keyword id="dnssec-on-miss" attribute="dnssec_on_miss" value="no"/>"

Changes in the PEM module:

configurationModule->cli_cmd->@{id:pem-forwarding-endpoint}->keyword->@{id:persistence}->$->id	Changed: "persistence" TO "persistence.type"
configurationModule->cli_cmd->@{id:pem-globals__analytics}->keyword	Removed: "<keyword id="mode" attribute="avr_reporting_mode" cli_enum="cli_enable_disable"/>"
configurationModule->cli_cmd->@{id:pem-globals__analytics}->keyword	Removed: "<keyword id="logging.hsl.endpoint-id" attribute="hsl_endpoint_id"/>"

Changes in the APM module:

configurationModule->cli_cmd->@{id:agent-aaa-ocsp}->keyword	Removed: "<keyword id="ocsp-responder" attribute="ocsp_responder"/>"

Changes in the Security modules:

configurationModule->cli_cmd->@{id:fw-user-list}->keyword	Removed: "<keyword id="description" attribute="description"/>"
configurationModule->cli_cmd->@{id:fw-user-list}->association	Removed: "<association id="users" command="fw-user-list-entry" operations="add delete modify replace-all-with"> <attribute set="fw_user_list_entry.list_name" from="fw_user_list.name"/> </association>"
configurationModule->cli_cmd->@{id:fw-user-list}->association	Removed: "<association id="user-groups" command="fw-user-group-list-entry" operations="add delete modify replace-all-with"> <attribute set="fw_user_group_list_entry.list_name" from="fw_user_list.name"/> </association>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.mode}	Removed: "<keyword id="latency-based.mode" attribute="latency_based_mode"/>"

configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:behavior-based.mode}	Removed: "<keyword id="behavior-based.mode" attribute="behavior_based_mode"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.latency-increase-rate}	Removed: "<keyword id="latency-based.latency-increase-rate" attribute="latency_increase_rate"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.maximum-latency}	Removed: "<keyword id="latency-based.maximum-latency" attribute="maximum_latency"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.minimum-latency}	Removed: "<keyword id="latency-based.minimum-latency" attribute="minimum_latency"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-client-side-defense}->$->id	Changed: "latency-based.ip-client-side-defense" TO "stress-based.ip-client-side-defense"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-client-side-defense}->$->id	Changed: "latency-based.geo-client-side-defense" TO "stress-based.geo-client-side-defense"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-client-side-defense}->$->id	Changed: "latency-based.url-client-side-defense" TO "stress-based.url-client-side-defense"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-client-side-defense}->$->id	Changed: "latency-based.site-client-side-defense" TO "stress-based.site-client-side-defense"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-captcha-challenge}->$->id	Changed: "latency-based.ip-captcha-challenge" TO "stress-based.ip-captcha-challenge"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-captcha-challenge}->$->id	Changed: "latency-based.geo-captcha-challenge" TO "stress-based.geo-captcha-challenge"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-captcha-challenge}->$->id	Changed: "latency-based.url-captcha-challenge" TO "stress-based.url-captcha-challenge"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-captcha-challenge}->$->id	Changed: "latency-based.site-captcha-challenge" TO "stress-based.site-captcha-challenge"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-rate-limiting}->$->id	Changed: "latency-based.ip-rate-limiting" TO "stress-based.ip-rate-limiting"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-request-blocking-mode}	Removed: "<keyword id="latency-based.ip-request-blocking-mode" attribute="latency_based_source_ip_based_request_blocking_mode"/>"

configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-rate-limiting}->$->id	Changed: "latency-based.geo-rate-limiting" TO "stress-based.geo-rate-limiting"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-request-blocking-mode}	Removed: "<keyword id="latency-based.geo-request-blocking-mode" attribute="latency_based_geolocation_based_request_blocking_mode"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-rate-limiting}->$->id	Changed: "latency-based.url-rate-limiting" TO "stress-based.url-rate-limiting"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-rate-limiting}->$->id	Changed: "latency-based.site-rate-limiting" TO "stress-based.site-rate-limiting"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-tps-increase-rate}	Removed: "<keyword id="latency-based.ip-tps-increase-rate" attribute="latency_based_ip_tps_increase_rate"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-maximum-tps}	Removed: "<keyword id="latency-based.ip-maximum-tps" attribute="latency_based_ip_maximum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.ip-minimum-tps}	Removed: "<keyword id="latency-based.ip-minimum-tps" attribute="latency_based_ip_minimum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-share-increase-rate}	Removed: "<keyword id="latency-based.geo-share-increase-rate" attribute="latency_based_geolocation_traffic_share_increase_rate"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.geo-minimum-share}	Removed: "<keyword id="latency-based.geo-minimum-share" attribute="latency_based_geolocation_traffic_minimum_share"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-tps-increase-rate}	Removed: "<keyword id="latency-based.url-tps-increase-rate" attribute="latency_based_url_tps_increase_rate"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-maximum-tps}	Removed: "<keyword id="latency-based.url-maximum-tps" attribute="latency_based_url_maximum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.url-minimum-tps}	Removed: "<keyword id="latency-based.url-minimum-tps" attribute="latency_based_url_minimum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-tps-increase-rate}	Removed: "<keyword id="latency-based.site-tps-increase-rate" attribute="latency_based_site_wide_tps_increase_rate"/>"

configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-maximum-tps}	Removed: "<keyword id="latency-based.site-maximum-tps" attribute="latency_based_site_wide_maximum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.site-minimum-tps}	Removed: "<keyword id="latency-based.site-minimum-tps" attribute="latency_based_site_wide_minimum_tps"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.escalation-period}	Removed: "<keyword id="latency-based.escalation-period" attribute="latency_based_escalation_period"/>"
configurationModule->cli_cmd->@{id:dos-application}->keyword->@{id:latency-based.de-escalation-period}	Removed: "<keyword id="latency-based.de-escalation-period" attribute="latency_based_deescalation_period"/>"
configurationModule->cli_cmd->@{id:profile-httpsecurity}->keyword->@{id:methods.values}->$->tabc	Removed: "asm_http_method"
configurationModule->cli_cmd_mode->@{id:firewall-mode}->command->@{id:fw-user-group-entity}->$->keyword	Changed: "user-group-entity" TO "fqdn-entity"

Overview: URI format and structure

A principle of the REST architecture describes the identification of a resource by means of a Uniform Resource Identifier (URI). A URI identifies the name of a web resource; in this case, the URI also represents the tree structure of modules and components in `tmsh`. You can specify a URI with a web service request to create, read, update, or delete some component or module of a BIG-IP® system configuration. In the context of the REST architecture, the system configuration is synonymous with the representation of a resource, and web service requests read and write that representation using the iControl® REST API.

Tip: Use admin, the default administrative account, for requests to iControl REST. Once you are familiar with the API, you can create user accounts for iControl REST users with various permissions.

For the URI snippet shown here, the `management-ip` component of the URI is the fully qualified domain name (FQDN) or IP address of a BIG-IP device.

```
https://<management-ip>/mgmt/tm/...
```

In iControl REST, the URI structure for all requests includes the string `/mgmt/tm/` to identify the namespace for traffic management. Any identifiers that you append to that string specify collections.

```
https://<management-ip>/mgmt/tm/...
```

The ellipsis in the snippet indicates the location where you specify an *organizing collection*, which is a collection of links to other resources in iControl REST. Organizing collections are the functional equivalent of modules in `tmsh`. In other words, the organizing collection `apm` in iControl REST is the `apm` module. In iControl REST, you can use the following URI to access all of the resources in the `apm` collection:

```
https://192.168.25.42/mgmt/tm/apm
```

Expanding on that approach, the URI in the following example designates all of the resources in the `report` collection. You can think of a collection as the equivalent of a `tmsh` sub-module. An iControl REST collection contains collections or resources.

```
https://192.168.25.42/mgmt/tm/apm/report
```

The URI in the following example designates a resource, which is a set of entities. In iControl REST, an *entity* is a property that you can configure, such as `"destAddrMax":2048`. A resource may also contain sub-collections. In the parlance of `tmsh`, a resource is the equivalent of a component.

```
https://192.168.25.42/mgmt/tm/apm/report/default-report
```

Important: iControl REST only supports secure access through HTTPS, so you must include credentials with each REST call. Use the same credentials you use for the BIG-IP device manager interface.

About reserved ASCII characters

To accommodate the BIG-IP® configuration objects that use characters, which are not part of the unreserved ASCII character set, use a percent sign (%) and two hexadecimal digits to represent them in a URI. The unreserved character set consists of: `[A - Z] [a - z] [0 - 9] dash (-), underscore (_), period (.), and tilde(~).`

You must encode any characters that are not part of the unreserved character set for inclusion in a URI scheme. For example, an IP address in a non-default route domain that contains a percent sign to indicate an address in a specific route domain, such as `192.168.25.90%3`, should be encoded to replace the `%` character with `%25` .

About REST resource identifiers

A URI is the representation of a resource that consists of a protocol, an address, and a path structure to identify a resource and optional query parameters. Because the representation of folder and partition names in `tmsh` often includes a forward slash (/), URI encoding of folder and partition names must use a different character to represent a forward slash in iControl® REST. To accommodate the forward slash in a resource name, iControl REST maps the forward slash to a tilde (~) character. When a resource name includes a forward slash (/) in its name, substitute a tilde (~) for the forward slash in the path. For example, a resource name, such as `/Common/plist1`, should be modified to the format shown here: `https://management-ip/mgmt/tm/security/firewall/port-list/~Common~plist1`

About HTTP method semantics

Hypertext Transfer Protocol (HTTP 1.1) describes the methods and headers that build on the Uniform Resource Identifier (URI) that identifies a collection or resource. The portion of a URI that makes up an absolute path includes endpoints, such as `/mgmt`, that specify the path to a resource or collection. With the exception of the X-F5-REST-Coordination-ID header that identifies a transaction, iControl® REST does not define any additional HTTP headers. A collection is a set of resources of the same type, and a collection is either a collection of resources or an organizing collection of links to resources. In the context of an HTTP method, a URI identifies a resource or collection as the target of a request.

In addition to the path of a resource, query parameters allow refinement of the result set for a GET request. A query string begins with a question mark (?) character and consists of expressions that refine the response data. The iControl REST query parameters are implementations of the OData query parameters as well as several custom query parameters. To distinguish the custom query parameters from OData query parameters, iControl REST custom query parameters omit the dollar sign ($) as the first character of the parameter.

The semantics of iControl REST methods behave differently depending on the URI. For a POST request, a URI indicates a resource under which the request creates a subordinate resource. HTTP considers the subordinate resource to be a new entity and not a modification of an existing entity. If the subordinate resource already exists, the protocol considers a request to create the same resource as an error. For a PUT request, a URI refers to an existing resource

and the request modifies the existing resource. For a PATCH request, a URI refers to an existing resource and the request merges changes into the resource.

To address different requirements, iControl REST implements both PATCH and PUT methods. In iControl REST, the PATCH method modifies only the properties that you specify in a request. The PUT method modifies the properties that you specify in a request and sets the remaining properties to either default values or empty values.

The semantics of iControl REST methods behave differently for collections and resources, as described in the following table.

Method	Description
GET	For both collections and resources, iControl REST supports the GET method. Also supports query strings.
POST	For both collections and resources, iControl REST supports the POST method.
DELETE	For collections, iControl REST does not support the DELETE method. For resources, iControl REST supports the DELETE method.
PUT	For collections, iControl REST does not support the PUT method. For resources, iControl REST supports the PUT method. For versions 11.6 and earlier, iControl REST only partially supports the PUT method for resources.
PATCH	For collections, iControl REST does not support the PATCH method. For resources, iControl REST supports the PATCH method.

About JSON request and response semantics

When iControl REST processes a GET request, it generates a response code and a tJSONbody. Likewise, an error response contains additional descriptive text in JSON format. To indicate the format of the text body in a response, iControl REST sets the HTTP `Content-Type` header as `application/json`. A response from iControl REST contains properties which describe a configuration object or the statistics for a resource. In iControl REST, the term *property* refers to a name/value, or key/value, pair in a JSON object.

The JSON terminology consists of two structures: objects and arrays. An object is a collection of one or more name/value pairs, as shown:

```
{ "partition":"Common" }
```

For a GET request, the properties consist of JSON objects or arrays, or both. Note that the name and value appear in double quotes (" "), with a colon (:) separator between the name and the value. For objects that contain multiple name pairs, additional name/value pairs are separated by a comma (,). An example of a typical, albeit slightly dated, response from iControl REST illustrates the JSON body formatting.

```
{
   "kind":"tm:ltm:ltmcollectionstate",
   "selfLink":"https://localhost/mgmt/tm/ltm?ver=11.5.0",
   "items":[
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/auth?ver=11.5.0"
         }
      },
      {
         "reference":{
```

```
            "link":"https://../mgmt/tm/ltm/classification?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/data-group?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/dns?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/global-settings?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/html-rule?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/message-routing?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/monitor?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/persistence?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/profile?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/default-node-monitor?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/ifile?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/lsn-pool?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/nat?ver=11.5.0"
         }
      },
      {
```

```
            "reference":{
               "link":"https://../mgmt/tm/ltm/node?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/policy?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/policy-strategy?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/pool?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/rule?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/snat?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/snat-translation?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/snatpool?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/traffic-class?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/virtual?ver=11.5.0"
            }
         },
         {
            "reference":{
               "link":"https://../mgmt/tm/ltm/virtual-address?ver=11.5.0"
            }
         }
      ]
}
```


About additional iControl REST properties

The iControl® REST implementation includes some document properties not present in Traffic Management Shell (tmsh) output. The differences are noted in the table and appear in a response to a GET request of a collection or resource, as shown in the example.

PropertyName	Description
kind	A unique type identifier.
generation	A generation number for a resource. Modification of a resource, or a related resource, changes the value. The value does not necessarily increase monotonically. For example, if you modify a resource in a sub-collection, the modification may cause a change in the parent object.
selfLink	A link to this resource.

```
{
   "kind":"tm:sys:software:image:imagecollectionstate",
   "selfLink":"https://localhost/mgmt/tm/sys/software/image?ver=11.5.0",
   "items":[
      {
         "kind":"tm:sys:software:image:imagestate",
         "name":"BIGIP-11.5.0.0.0.191.iso",
         "fullPath":"BIGIP-11.5.0.0.0.191.iso",
         "generation":38,
         "selfLink":"https://../mgmt/tm/sys/software/image/
BIGIP-11.5.0.0.0.191.iso?ver=11.5.0",
         "build":"0.0.191",
         "buildDate":"Wed Nov 27 14 03 09 PST 2013",
         "checksum":"fab5b673486ccc1ec20fbe6cea51df50",
         "fileSize":"1751 MB",
         "lastModified":"Tue Dec  3 01:30:32 2013",
         "product":"BIG-IP",
         "verified":"yes",
         "version":"11.5.0"
      },
      {
         "kind":"tm:sys:software:image:imagestate",
         "name":"BIGIP-tmos-bugs-staging-11.5.0.0.0.237.iso",
         "fullPath":"BIGIP-tmos-bugs-staging-11.5.0.0.0.237.iso",
         "generation":37,
         "selfLink":"https://../software/image/BIGIP-tmos-bugs-
staging-11.5.0.0.0.237.iso?ver=11.5.0",
         "build":"0.0.237",
         "buildDate":"Wed Dec 4 14 14 44 PST 2013",
         "checksum":"bb4ae4838a5743fa209f67a1b56dedef",
         "fileSize":"1843 MB",
         "lastModified":"Wed Dec  4 15:32:28 2013",
         "product":"BIG-IP",
         "verified":"yes",
         "version":"11.5.0"
      }
   ]
}
```

```
root@(BIG-IP1)(...)(tmos)# list sys software image
sys software image BIGIP-11.4.0.321.0.iso {
    build 321.0
    build-date "Mon Feb 11 07 23 24 PST 2013"
    checksum f9411fde01d6a3521d4ae393e9bb077c
    file-size "1522 MB"
    last-modified "Mon Feb 11 09:35:50 2013"
    product BIG-IP
    verified yes
```

```
    version 11.4.0
}
root@(BIG-IP1)(...)(tmos)#
```

About null values and properties

Flags are typically composed as a bit set by software to indicate state, such as `0` or `1`, and indicate on or off, respectively. iControl® REST displays flags that are set with the flag name and a value of `null`. If the value of a flag is none, iControl REST omits the property from the output.

Note: To POST or PUT a flag with only a single value, enter the property name in the JSON body with a value of null.

```
{
   "kind":"tm:sys:software:volume:volumecollectionstate",
   "selfLink":"https://localhost/mgmt/tm/sys/software/volume?ver=11.5.0",
   "items":[
      {
         "kind":"tm:sys:software:volume:volumestate",
         "name":"MD1.1",
         "fullPath":"MD1.1",
         "generation":34,
         "selfLink":"https://localhost/mgmt/tm/sys/software/volume/MD1.1?
ver=11.5.0",
         "basebuild":"0.0.191",
         "build":"0.0.191",
         "product":"BIG-IP",
         "status":"complete",
         "version":"11.5.0",
         "media":[
            {
               "name":"MD1.1",
               "media":"array",
               "size":"default"
            }
         ]
      },
      {
         "kind":"tm:sys:software:volume:volumestate",
         "name":"MD1.2",
         "fullPath":"MD1.2",
         "generation":35,
         "selfLink":"https://localhost/mgmt/tm/sys/software/volume/MD1.2?
ver=11.5.0",
         "active":null,
         "apiRawValues":{

         },
         "basebuild":"0.0.237",
         "build":"0.0.237",
         "product":"BIG-IP",
         "status":"complete",
         "version":"11.5.0",
         "media":[
            {
               "name":"MD1.2",
               "defaultBootLocation":null,
               "media":"array",
               "size":"default"
            }
         ]
      },
      {
```

```
         "kind":"tm:sys:software:volume:volumestate",
         "name":"MD1.3",
         "fullPath":"MD1.3",
         "generation":36,
         "selfLink":"https://localhost/mgmt/tm/sys/software/volume/MD1.3?
ver=11.5.0",
         "status":"complete",
         "media":[
            {
               "name":"MD1.3",
               "media":"array",
               "size":"default"
            }
         ]
      }
   ]
}
```

About reserved property names

iControl® REST reserves several property names, most notably, the words `name` and `generation`. Some `tmsh` components include properties with reserved property names. When iControl REST encounters a reserved name in the JSON body, it replaces the reserved names with the corresponding replacement, `tmName` or `tmGeneration`.

About property name format differences

Property and option names in iControl® REST use a different naming convention than *Traffic Management (tmsh) Shell*. In `tmsh`, property names consist of lowercase characters. For property names that contain multiple words, hyphens separate the words. iControl REST uses camel case convention for property names, where the first word of a property is lowercase, and all additional words in the name are capitalized.

For example, the property `build-date`, as shown in `tmsh`, appears as `buildDate` in iControl REST.

About JSON formats and encodings

iControl® REST supports the following specifications for string encodings:

- W3C XML Schema for numbers
- ISO 3166 for countries and territories
- ISO 6709 for latitude and longitude
- ISO for currency
- RFC 3339 for dates and times
- Olson Time Zone Database for time zones
- Time durations can be expressed as seconds since Unix Epoch (00:00:00 UTC on January 1, 1970), up to one microsecond of fractional time.

For dates and times that are specific to a property in the configuration, a property name that incorporates the time unit into the name, such as `checkIntervalDays`, provides a hint about the units of time.

About API versions

Over time, modifications to the iControl® REST API may necessitate that a release is assigned a new version number. To limit requests to a particular version of the API, iControl REST accepts an API version parameter as an option to a URI. To use a particular API version, specify the ver parameter, an API version number, such as 11.5.0, and append the string to the end of the URI, as you would with any query parameter.

```
GET https://192.168.25.42/mgmt/tm/ltm?ver=11.5.0
```

The JSON body for a response includes an API version number in the `selfLink` property, as well as any links. For iControl REST, the version number of a resource in a response matches the version number sent in a request. If you do not specify the version of the API, the version defaults to the current version. To maintain backward compatibility with future releases of the API, a response will contain resources that match the version number specified in the request. If iControl REST cannot generate a response that is compatible with the request, it returns an error code.

Note: Although some REST implementations use HTTP headers to manage version information, iControl REST does not use any HTTP headers to identify an API.

Password change in iControl REST

When the password expires, F5® iControl® REST blocks access to resources on a BIG-IP® system for the currently logged-in user. To continue to use the BIG-IP system, the user must create a new password before accessing resources on the system. The programmatic approach to password changes uses a PATCH request to the `/mgmt/tm/auth/user` endpoint, which you can access to change the password.

If you choose to change a password using iControl REST, you must supply a password in a JSON body. The password must adhere to the established password policy for your organization. You can use `tmsh` to find the password policy in effect for your organization. If the password you supply meets the requirements for a password, iControl REST generates a `200 OK` message in response to your request. If the password does not satisfy the requirements, then iControl REST rejects the password change request with an HTTP error response.

Changing a password by using an iControl REST request

When your password expires, a BIG-IP® system blocks access to resources. To re-enable access to resources, you can access the `/mgmt/tm/auth/user` endpoint to change your password.

1. To change your password, supply a new password in a JSON body.

```
{
  "password": "<password>"
}
```

   If the password you supply fails to meet the requirements for a password, such as password length or uniqueness, the request fails. Otherwise, the request returns a success message. You can use the `tmsh` command `list auth password-policy` to find the password policy.

2. Make a PATCH request to the `/mgmt/tm/auth/user` endpoint and include the JSON body.

```
PATCH https://192.168.25.42/mgmt/tm/auth/user
```

In this task, you changed your password by making an iControl® REST request.

About iControl and authentication for user accounts

The iControl® REST no longer requires that you grant permissions on iControl REST resources for individual user accounts. As of version 12.0, a user automatically has access to REST resources, but all user must acquire a token for authentication and include that token in all REST requests. Administrators of a BIG-IP® system can still make REST requests by using basic authentication. The basic authentication requires a Base64 encoded string that consists of a user ID, a colon (:), and a password, this string can expose credentials for use by attackers in CSRF attacks. It is recommended to disable the basic authentication using database variable `httpd.basic_auth`. If the database variable `httpd.basic_auth` is set to disable, then all users including administrators are required to acquire a token for authentication and include that token in all REST requests.

Requesting a token for iControl REST authentication

As an administrator of a BIG-IP® system, you can use the basic authentication to make iControl REST calls. For users that lack administrator privileges, the user must request a token that can be used to authenticate the user making REST API requests.

1. To create an authentication token, make a POST request to the BIG-IP® system. You must enclose both the name and password values in double quotes (" "), as with any JSON string.

```
POST https://172.68.25.42/mgmt/shared/authn/login

{
  "username": <user name>,
  "password": <user password>,
  "loginProviderName": "tmos"
}
```

   The BIG-IQ® documentation specifies `loginReference`, which takes a reference to a login provider. In the example, the `loginProviderName` property allows you to specify a name instead of a reference. For most situations, use the `loginProviderName` and specify `tmos`.

2. To use the token in a REST request, copy the string for the `token` property and save it.

   The token consists of a string of random letters and digits. In this example, the string is `492D3316E5456378B4AC9B5E2FA923595F0DA65A`. The lifetime of the token is eight hours.

3. To make a REST request, add the token to request header. You must enclose the token within double quotes (" "), as with any JSON string.

```
GET https://172.68.25.42/mgmt/tm/ltm

{ "X-F5-Auth-Token": "492D3316E5456378B4AC9B5E2FA923595F0DA65A" }
```

In this example, you acquired a token to include in an iControl REST request.

Overview: Fundamentals of Cross-Origin Resource Sharing

The same origin policy in browsers controls interactions between two different origins, such as requests with `XMLHttpRequest` (XHR) objects. Furthermore, the same origin policy states that a browser that is downloading data from a particular web site cannot interact with another resource that does not originate from the same web site, where protocol, port number, and host name identify the web site. While there are mechanisms to implement a safe cross-site data transfer, *Cross-Origin Resource Sharing (CORS)* enables secure cross-site data transfers by adding new HTTP headers to describe or enumerate a set of origins, as well as to determine the viability of a request prior to the transmission of client data. The CORS headers permit communication between a client and server to establish the limits of such requests.

CORS supports two types of requests: simple and preflight. A simple request consists of a GET, HEAD, or POST request. For POST requests, the Content-Type of the data sent to a server must be `application/x-www-form-urlencoded`, `multipart/form-data`, or `text/plain`. One final condition for a simple request is that the request does not set custom headers.

For HTTP methods that modify a web resource, the CORS standard defines a *preflight* capability that enables a client to determine if a server allows a request. A client uses the preflight mechanism if a request contains a method other than GET, HEAD, or POST, or specifies a Content-Type header other than `application/x-www-form-urlencoded`, `multipart/form-data`, or `text/plain` with a POST request. Before the client sends a request with data, the client makes a request with the OPTIONS method to query the server.

Finally, a client initiates a cross-origin request by including the `Origin` HTTP header in a request. A client also includes the `Access-Control-Request-Method` and `Access-Control-Request-Headers` headers in the cross-origin request. A server that allows a cross-origin request responds with an HTTP `Access-Control-Allow-Origin` header and the value of the requesting origin, an `Access-Control-Request-Method` header and supported methods, and an `Access-Control-Request-Headers` header and supported values.

Cross-Origin Resource Sharing request headers

This table lists the request headers sent by a client, according to the Cross-Origin Resource Sharing (CORS) specification.

HTTP header	Description
Origin	Specifies a URI that indicates the source of the cross-origin or preflight request.
Access-Control-Request-Method	Specifies the HTTP method that the client will send in a request.
Access-Control-Request-Headers	Specifies the HTTP headers that the client will include in a request.

Cross-Origin Resource Sharing response headers

This table lists the response headers sent by a server, in response to a preflight request, according to the Cross-Origin Resource Sharing (CORS) specification.

HTTP header	Description
Access-Control-Allow-Origin	Specifies a URI that is allowed to access a resource. For iControl® REST users, this header lists origins for which you allow requests. The iControl REST implementation does not allow wild card characters (*).
Access-Control-Expose-Headers	Specifies a list of HTTP headers that are safe to expose. For iControl REST users, this header is a list of F5®-specific headers that clients can access.
Access-Control-Max-Age	Specifies the length of time to cache the results of a preflight request. The client should discard the results after this time period expires. The value is either the lesser of the session timeout value or one day.
Access-Control-Allow-Credentials	Indicates whether to expose the response if the credentials setting is true. For iControl REST users, this header indicates the allowance of authentication cookies in a CORS request. Specify the value as true. If you do not need cookies for authentication, do not specify this header. You must also set the `withCredentials` property of the `xmlHttpRequest` object to true for a CORS request to succeed.
Access-Control-Allow-Methods	Specifies only the methods for which the server allows cross-origin access.
Access-Control-Allow-Headers	Specifies the headers that the server allows.

About external authentication providers with iControl REST

iControl® REST supports external authentication to other providers, such as Active Directory (AD) or RADIUS. Authentication with a provider other than the local authentication provider on a BIG-IP® system requires a token that you can use to access resources in iControl REST. A token consists of 32 random characters, primarily digits and uppercase ASCII characters, valid for a period of time. Until the token expires, a server validates your identity based on the authentication token you submit. When the token expires, you simply acquire a new token from a provider.

Note: Before you make a REST request using token-based authentication, you must obtain a token from an external authentication provider.

You create a token by calling a user authentication method in the F5® REST API. Prior to making a token creation request, you must obtain a login reference from your system administrator that identifies an external authentication provider. To create the authentication token, make a POST request and specify user name, password, and login reference in the JSON body of the request. This request associates an authentication token with a user name. If the token creation request is successful, the response contains a JSON body similar to this.

```
{
   "username":"auser",
   "loginReference":{
      "link":"https://localhost/mgmt/cm/system/authn/providers/
ldap/298d4aa5-d255-438f-997d-7f984109dd5d/login"
   },
   "token":{
      "uuid":"69c4b1c8-efdc-429a-b50c-723e92703a2b",
      "name":"492D3316E5456378B4AC9B5E2FA923595F0DA65A",
      "token":"492D3316E5456378B4AC9B5E2FA923595F0DA65A",
      "userName":"USERNAME",
      "user":{
         "link":"https://localhost/mgmt/cm/system/authn/providers/
ldap/298d4aa5-d255-438f997d7f984109dd5d/users/a25e2147-92e0-4349-
ac99-7c844b3d30c2"
      },
      "groupReferences":[

      ],
      "timeout":1200,
      "startTime":"2014-07-08T17:14:34.305-0700",
      "address":"192.168.2.2",
      "partition":"[All]",
      "generation":1,
      "lastUpdateMicros":1404864874295548,
      "expirationMicros":1404866074305000,
      "kind":"shared:authz:tokens:authtokenitemstate",
      "selfLink":"https://localhost/mgmt/shared/authz/tokens/69c4b1c8-
efdc-429a-b50c-723e92703a2b"
   },
   "generation":0,
   "lastUpdateMicros":0
}
```

The `token` property identifies the value to include in a request. In the JSON body, the `token` is the string `492D3316E5456378B4AC9B5E2FA923595F0DA65A`, inside of the `token` object. To be authenticated by the resource, you must include the `X-F5-Auth-Token` header in a REST request and specify the token value in the header. If you prefer to authenticate locally, you can leave the `Authorization` header blank. For more information about obtaining and using an authentication token, see *BIG-IQ® Systems: REST API Reference*.

GET Requests

Discovering modules and components

iControl® REST supports discovery through a GET request. The structure of resources becomes more obvious as you investigate the organizing collections. One other benefit of discovering the organizing collections is the relationship between iControl REST and `tmsh`.

To discover the structure, make a request to iControl REST with the GET method and specify an organizing collection, as shown in this example.

```
GET https://192.168.25.42/mgmt/tm/ltm
```


```
{
   "items":[
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/ltm/auth?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/classification?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/data-group?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/ltm/dns?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/global-settings?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/html-rule?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/message-routing?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/monitor?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/persistence?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/profile?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/default-node-monitor?ver=11.5.0"
         }
      },
```

```
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/ifile?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/lsn-pool?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/nat?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/node?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/policy?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/policy-strategy?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/pool?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/rule?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/snat?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/snat-translation?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/snatpool?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/traffic-class?ver=11.5.0"
         }
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/virtual?ver=11.5.0"
         }
```

```
      },
      {
         "reference":{
            "link":"https://../mgmt/tm/ltm/virtual-address?ver=11.5.0"
         }
      }
   ],
   "kind":"tm:ltm:ltmcollectionstate",
   "selfLink":"https://localhost/mgmt/tm/ltm?ver=11.5.0"
}
```

If you are familiar with command-line tools, use `curl`, or a similar utility, to make a request to iControl REST. In the URI, specify an organizing collection. For example, the command: `curl -k -u admin:admin -X GET https://192.168.25.42/mgmt/tm/ltm` makes a request of the `ltm` organizing collection.

Note: The contents of an iControl REST resource may not have all of the properties and options of its `tmsh` counterpart below the sub-collection level.

Note: A module that is not provisioned on a BIG-IP® system will not appear in the output.

About paging properties

iControl® REST supports pagination options for large collections. The implementation of pagination utilizes the Open Data Protocol (OData) query parameters to provide information that you can use to navigate a large result set. When you request a large collection, the iControl REST response includes properties to identify the URI for the collection, the next page of the result set, the previous page of the result set, as well as the total number of items in the result, total number of pages, the current page, the number of items per page, and a count of the number of items in the current page. iControl® REST calculates these values on the filtered result set.

Property	Description
selfLink	The URI of the collection, including any query parameters.
nextLink	The next set of data in the result set. Includes the `$skip` query parameter in the link.
previousLink	The previous set of data in the result set. Not present in the first set of data.
currentItemCount	A count of the number of items in the result set, either as the value of the `$top` query parameter, or the remaining number of items if less than the number requested.
itemsPerPage	The number of items to display per page.
pageIndex	The current page in the result set.
totalPages	The total number of pages in the result set, equal to the result of (totalItems / itemsPerPage), rounded up to the next integer value.
startIndex	The index of the first item in the result set.
totalItems	The number of items in the result set, as calculated by the $inlinecount=allpages query parameter.

About query parameters

iControl® REST implements a subset of the *Open Data Protocol (OData)* recommendations for query languages and system query options. The OData protocol defines System Query Options that are query string parameters to manage the presentation of data in a result set identified by a URL. For example, you can include or exclude rows from a

result set, constrain a query to resources contained within an administrative partition, or specify a particular version of iControl REST. With the exception of the `asm` module, query parameters are limited to GET requests.

To use a query parameter, append a query parameter expression to the end of a request URI. All query parameter expressions begin with a question mark (?), followed by a query parameter name, a comparison or logical operator, and a value. A value adheres to the camel case naming convention for iControl REST. OData query parameters begin with a dollar sign ($), whereas custom query parameters do not. For example, you can specify that the response only include the name property in the following request:

```
GET https://localhost/mgmt/tm/ltm/pool/?$select=name
```

To specify additional query parameters, precede each additional query parameter with an ampersand (&), then specify the query parameter expression. The following table lists the parameters that are iControl REST implementations of the OData query parameters. All OData query parameters begin with a dollar sign ($). Note that the $filter parameter, if used, limits the result set to a specific administrative partition.

Parameter	**Description**
$filter	Specifies an administrative partition to query for a result set. This parameter filters the result set by partition name and does not fully implement the corresponding OData query parameter. The `asm` module fully implements the OData query parameter.
$select	Specifies a subset of the properties that will appear in the result set.
$skip	Specifies the number of rows to skip in the result set. The result set is chosen from the remaining rows.
$top	Specifies the first N rows of the result set.

iControl REST supports comparison and logical operators as described in the OData recommendation.

Operator	**Description**
eq	Equal to
ne	Not equal to
lt	Less than
le	Less than or equal to
gt	Greater than
ge	Greater than or equal to
and	True if both operands are true
or	True if either operand is true
not	Negation of operand

Note: iControl REST supports only the `eq` operator with the $filter parameter.

iControl REST includes several custom query parameters. The custom query parameters do not include a dollar sign ($) character in the parameter name.

Parameter	**Description**
expandSubcollections	Specifies that iControl REST expand any references to sub collections when set to `true`. By default, the

Parameter	Description
	response to a GET request only contains links for sub collection reference properties.
options	Specifies the options to a query request. This parameter takes values that are compatible with the `tmsh` command-line options, for example, `# tmsh list apm oauth token-details db-instance /Common/oauth-api-db` is equal to `https://localhost/mgmt/tm/apm/oauth/token-details?options="db-instance","/Common/oauth-api-db"` or `https://localhost/mgmt/tm/apm/oauth/token-details?options="db-instance"+"/Common/oauth-api-db"`.
ver	Specifies the version number of the iControl® REST API to use when making a request. Defaults to the current version if you do not specify a value.

Paging through large collections

Collections that contain a large number of items consume a great deal of network bandwidth and processing power if processed in a single GET request. Query parameters allow you to manage multi page responses. iControl® REST supports the OData system query parameters $top and $skip to return pages items sets.

Use the `$top` query parameter to specify the maximum number of items for the BIG-IP® device to return. If you use curl and run this command from a Unix command line, precede the dollar sign character ($) with a backslash character (\) to prevent shell interpretation of the character.

```
curl -k -u admin:admin -X GET https://192.168.25.42/mgmt/tm/sys?\$top=4
```

To query for the first `n` data items, specify the URI, and append the `$top` query parameter to the URI. This query displays the first four items in the `sys` collection output. The response indicates the `nextLink` and `previousLink` properties that serve as navigation markers to the next page and previous page, respectively.

```
https://192.168.25.42/mgmt/tm/sys?$top=4
```

```
{ "currentItemCount" : 4,
  "items" : [
    { "reference" :
       { "link" : "https://../mgmt/tm/sys/application?ver=11.5.0" } }
    { "reference" :
       { "link" : "https://../mgmt/tm/sys/crypto?ver=11.5.0" } }
    { "reference" :
       { "link" : "https://../mgmt/tm/sys/daemon-log-settings?
ver=11.5.0" } }
    { "reference" :
       { "link" : "https://../mgmt/tm/sys/disk?ver=11.5.0" } }
  ],
  "itemsPerPage" : 4,
  "kind" : "tm:sys:syscollectionstate",
  "nextLink" : "https://localhost/mgmt/tm/sys?$top=4&$skip=4&ver=11.5.0",
  "pageIndex" : 1,
  "selfLink" : "https://localhost/mgmt/tm/sys?$top=4&ver=11.5.0",
  "startIndex" : 1,
  "totalItems" : 36,
  "totalPages" : 9
```

```
}
```

To request the next n data items, use the same URI as the previous example and append the `$skip` query parameter to the URI. This example displays the next four items in the `sys` collection output. The response also indicates the `nextLink` and `previousLink` properties that serve as navigation markers into the data.

```
https://192.168.25.42/mgmt/tm/sys?$top=4&$skip=4
```

```
{ "currentItemCount" : 4,
  "items" : [
    { "reference" :
      { "link" : "https://../mgmt/tm/sys/file?ver=11.5.0" } },
    { "reference" :
      { "link" : "https://../mgmt/tm/sys/icall?ver=11.5.0" } },
    { "reference" :
      { "link" : "https://../mgmt/tm/sys/log-config?ver=11.5.0" } },
    { "reference" :
      { "link" : "https://../mgmt/tm/sys/sflow?ver=11.5.0" } }
  ],
  "itemsPerPage" : 4,
  "kind" : "tm:sys:syscollectionstate",
  "nextLink" : "https://localhost/mgmt/tm/sys?$top=4&$skip=8&ver=11.5.0",
  "pageIndex" : 2,
  "previousLink" : "https://localhost/mgmt/tm/sys?$top=4&ver=11.5.0",
  "selfLink" : "https://localhost/mgmt/tm/sys?$top=4&$skip=4&ver=11.5.0",
  "startIndex" : 5,
  "totalItems" : 36,
  "totalPages" : 9
}
```

About sub-collection expansion

iControl® REST supports the `expandSubcollections` query parameter. In `tmsh`, configuration components contain properties, child components, and associated, non-child components. For example, you can create an associated component independently from the component that contains it,such as a virtual server (the `ltm virtual` component in `tmsh`) that contains an LTM® pool, even though you create the LTM pool as a separate task.

If set to `true`, the `expandSubcollections` query parameter displays all child components but omits any associated non-child components from the response.

Although the command creates a lengthy output block, the query parameter displays the properties of the sub-collection, in addition to the properties of the component. As with other query parameters, the `expandSubcollections` parameter does not support requests other than a GET request.

```
https://192.168.25.42/mgmt/tm/ltm/virtual/my-VS/?expandSubcollections=true
```

```
{
   "kind":"tm:ltm:virtual:virtualstate",
   "name":"my-VS",
   "fullPath":"my-VS",
   "generation":1,
   "selfLink":"https://../tm/ltm/virtual/my-VS?
expandSubcollections=true&ver=11.5.0",
   "autoLasthop":"default",
   "cmpEnabled":"yes",
   "connectionLimit":0,
   "destination":"/Common/10.2.1.189:0",
   "enabled":null,
   "gtmScore":0,
   "ipProtocol":"tcp",
```

```
   "mask":"255.255.255.255",
   "mirror":"disabled",
   "mobileAppTunnel":"disabled",
   "nat64":"disabled",
   "pool":"/Common/my-Pool",
   "rateLimit":"disabled",
   "rateLimitDstMask":0,
   "rateLimitMode":"object",
   "rateLimitSrcMask":0,
   "source":"0.0.0.0/0",
   "sourceAddressTranslation":{
      "type":"automap"
   },
   "sourcePort":"preserve",
   "synCookieStatus":"not-activated",
   "translateAddress":"enabled",
   "translatePort":"disabled",
   "vlansDisabled":null,
   "vsIndex":2,
   "policiesReference":{
      "link":"https://../tm/ltm/virtual/~Common~my-VS/policies?ver=11.5.0",
      "isSubcollection":true,
      "items":[
         {
            "kind":"tm:ltm:virtual:policies:policiesstate",
            "name":"asm_auto_l7_policy__my-VS",
            "partition":"Common",
            "fullPath":"/Common/asm_auto_l7_policy__my-VS",
            "generation":1,
            "selfLink":"https://../~Common~my-VS/policies/
~Common~asm_auto_l7_policy__my-VS?ver=11.5.0"
         }
      ]
   },
   "securityLogProfiles":[
      "\"/Common/Log illegal requests\""
   ],
   "fwRulesReference":{
      "link":"https://../tm/ltm/virtual/~Common~my-VS/fw-rules?ver=11.5.0",
      "isSubcollection":true
   },
   "profilesReference":{
      "link":"https://../tm/ltm/virtual/~Common~my-VS/profiles?ver=11.5.0",
      "isSubcollection":true,
      "items":[
         {
            "kind":"tm:ltm:virtual:profiles:profilesstate",
            "name":"http",
            "partition":"Common",
            "fullPath":"/Common/http",
            "generation":1,
            "selfLink":"https://../tm/ltm/virtual/~Common~my-VS/profiles/
~Common~http?ver=11.5.0",
            "context":"all"
         },
         {
            "kind":"tm:ltm:virtual:profiles:profilesstate",
            "name":"tcp",
            "partition":"Common",
            "fullPath":"/Common/tcp",
            "generation":1,
            "selfLink":"https://../tm/ltm/virtual/~Common~my-VS/profiles/
~Common~tcp?ver=11.5.0",
            "context":"all"
```

```
         },
         {
            "kind":"tm:ltm:virtual:profiles:profilesstate",
            "name":"websecurity",
            "partition":"Common",
            "fullPath":"/Common/websecurity",
            "generation":1,
            "selfLink":"https://../tm/ltm/virtual/~Common~my-VS/profiles/
~Common~websecurity?ver=11.5.0",
            "context":"all"
         }
      ]
   }
}
```

Expanding a sub-collection reference

The responses from iControl® REST can include references to sub collections. The `expandSubcollections` query parameter expands references to sub-collections.

View the details of a particular resource, including the details of its sub-collections, append the string `expandSubcollections=true` to the URI. Do not prepend a dollar sign ($) to this query parameter.

To see the differences, this example shows a GET request for a resource with sub-collection expansion. The response contains the `isSubcollection` property, set to true, to indicate a sub-collection. The output only contains a reference to the sub-collection.

```
https://192.168.42.25/mgmt/tm/ltm/pool/~Common~my-Pool
```

```
{ "allowNat" : "yes",
  "allowSnat" : "yes",
  "description" : "sdfds",
  "fullPath" : "/Common/my-Pool",
  "generation" : 1,
  "ignorePersistedWeight" : "disabled",
  "ipTosToClient" : "pass-through",
  "ipTosToServer" : "pass-through",
  "kind" : "tm:ltm:pool:poolstate",
  "linkQosToClient" : "pass-through",
  "linkQosToServer" : "pass-through",
  "loadBalancingMode" : "round-robin",
  "membersReference" : { "isSubcollection" : true,
      "link" : "https://../mgmt/tm/ltm/pool/~Common~my-Pool/members?
ver=11.5.0"
    },
  "minActiveMembers" : 0,
  "minUpMembers" : 0,
  "minUpMembersAction" : "failover",
  "minUpMembersChecking" : "disabled",
  "name" : "my-Pool",
  "partition" : "Common",
  "queueDepthLimit" : 0,
  "queueOnConnectionLimit" : "disabled",
  "queueTimeLimit" : 0,
  "reselectTries" : 0,
  "selfLink" : "https://../mgmt/tm/ltm/pool/~Common~my-Pool?ver=11.5.0",
  "slowRampTime" : 10
}
```

To see the expanded sub-collection, this example uses the `expandSubcollections` query parameter. iControl® REST supports the custom `expandSubcollections` query parameter, which omits the dollar sign ($) from its name.

```
https://192.168.25.42/mgmt/tm/ltm/pool/~Common~my-Pool/?
expandSubcollections=true
```

```
{ "allowNat" : "yes",
  "allowSnat" : "yes",
  "description" : "sdfds",
  "fullPath" : "/Common/my-Pool",
  "generation" : 1,
  "ignorePersistedWeight" : "disabled",
  "ipTosToClient" : "pass-through",
  "ipTosToServer" : "pass-through",
  "kind" : "tm:ltm:pool:poolstate",
  "linkQosToClient" : "pass-through",
  "linkQosToServer" : "pass-through",
  "loadBalancingMode" : "round-robin",
  "membersReference" : { "isSubcollection" : true,
      "items" : [ { "address" : "1.1.1.1",
            "connectionLimit" : 0,
            "dynamicRatio" : 1,
            "fullPath" : "/Common/block:0",
            "generation" : 1,
            "inheritProfile" : "enabled",
            "kind" : "tm:ltm:pool:members:membersstate",
            "logging" : "disabled",
            "monitor" : "default",
            "name" : "block:0",
            "partition" : "Common",
            "priorityGroup" : 0,
            "rateLimit" : "disabled",
            "ratio" : 1,
            "selfLink" : "https://../tm/ltm/pool/~Common~my-Pool/members/
~Common~block:0?ver=11.5.0",
            "session" : "user-enabled",
            "state" : "unchecked"
          } ],
      "link" : "https://../tm/ltm/pool/~Common~my-Pool/members?ver=11.5.0"
    },
  "minActiveMembers" : 0,
  "minUpMembers" : 0,
  "minUpMembersAction" : "failover",
  "minUpMembersChecking" : "disabled",
  "name" : "my-Pool",
  "partition" : "Common",
  "queueDepthLimit" : 0,
  "queueOnConnectionLimit" : "disabled",
  "queueTimeLimit" : 0,
  "reselectTries" : 0,
  "selfLink" : "https://../tm/ltm/pool/~Common~my-Pool?
expandSubcollections=true&ver=11.5.0",
  "slowRampTime" : 10
}
```

Returning resources from an administrative partition

To access an administrative partition, use the `$filter` query parameter in a GET request to specify a resource in a partition.

1. Access a partition other than `Common`, using the `$filter` query option at the end of the URI.

2. Encode the URI by creating the following string: `?$filter=partition%20eq%20fw_objs`

To use a filter parameter, this example shows a GET request that uses a filter setting to limit the query to a specific partition. The response from the request appears in the second block.

```
GET https://192.168.25.42/mgmt/tm/ltm/pool/?$filter=partition eq fw_objs
```

```
{
   "kind":"tm:ltm:pool:poolcollectionstate",
   "selfLink":"https://../mgmt/tm/ltm/pool?$filter=partition%20eq
%20fw_objs&ver=11.5.0",
   "items":[
      {
         "kind":"tm:ltm:pool:poolstate",
         "name":"tcb-pool2",
         "partition":"fw_objs",
         "fullPath":"/fw_objs/tcb-pool2",
         "generation":9587,
         "selfLink":"https://../mgmt/tm/ltm/pool/~fw_objs~tcb-pool2?
ver=11.5.0",
         "allowNat":"yes",
         "allowSnat":"yes",
         "description":"This pool exists in the fw_objs partition.",
         "ignorePersistedWeight":"disabled",
         "ipTosToClient":"pass-through",
         "ipTosToServer":"pass-through",
         "linkQosToClient":"pass-through",
         "linkQosToServer":"pass-through",
         "loadBalancingMode":"round-robin",
         "minActiveMembers":0,
         "minUpMembers":0,
         "minUpMembersAction":"failover",
         "minUpMembersChecking":"disabled",
         "queueDepthLimit":0,
         "queueOnConnectionLimit":"disabled",
         "queueTimeLimit":0,
         "reselectTries":0,
         "slowRampTime":10,
         "membersReference":{
            "link":"https://../mgmt/tm/ltm/pool/~fw_objs~tcb-pool2/members?
ver=11.5.0",
            "isSubcollection":true
         }
      }
   ]
}
```

Use iControl REST to obtain statistical output

iControl® REST supports the generation of statistical output by using a GET request. The output consists of read-only statistics, displayed in JSON format. Use of the `/stats` endpoint produces statistical output equivalent to the `tmsh show` command.

To obtain statistical results for a resource, append the endpoint `/stats` to the URI.

```
GET https://192.168.25.42/mgmt/tm/ltm/pool/stats
```

```
{
 "kind": "tm:ltm:pool:poolstats",
 "generation": 9,
 "selfLink": "https://localhost/mgmt/tm/ltm/pool/members/stats?ver=13.0.0",
```

```
 "entries": {
  "https://localhost/mgmt/tm/ltm/pool/members/~Common~pool1/stats": {
   "nestedStats": {
    "kind": "tm:ltm:pool:poolstats",
    "selfLink": "https://localhost/mgmt/tm/ltm/pool/members/~Common~pool1/
stats?ver=13.0.0",
    "entries": {
     "activeMemberCnt": {
      "value": 0
     },
     "availableMemberCnt": {
      "value": 1
     },
         ...(truncated for readability) ...

     "curSessions": {
      "value": 0
     },
     "memberCnt": {
      "value": 1
     },
     "minActiveMembers": {
      "value": 0
     },
     "monitorRule": {
      "description": "none"
     },
     "tmName": {
      "description": "/Common/pool1"
     },
     "serverside.bitsIn": {
      "value": 0
     },
     "serverside.bitsOut": {
      "value": 0
     },
     "serverside.curConns": {
      "value": 0
     },
     "serverside.maxConns": {
      "value": 0
     },
     "serverside.pktsIn": {
      "value": 0
     },
     "serverside.pktsOut": {
      "value": 0
     },
     "serverside.totConns": {
      "value": 0
     },
     "status.availabilityState": {
      "description": "unknown"
     },
     "status.enabledState": {
      "description": "enabled"
     },
     "status.statusReason": {
      "description": "The children pool member(s) either don't have service
 checking enabled, or service check results are not available yet"
     },
     "totRequests": {
      "value": 0
     },
```

```
     "https://localhost/mgmt/tm/ltm/pool/members/~Common~pool1/members/
stats": {
      "nestedStats": {
       "kind": "tm:ltm:pool:members:membersstats",
       "selfLink": "https://localhost/mgmt/tm/ltm/pool/members/
~Common~pool1/members/stats?ver=13.0.0",
       "entries": {
        "https://localhost/mgmt/tm/ltm/pool/members/~Common~pool1/members/
~Common~1.1.1.1:80/stats": {
         "nestedStats": {
          "kind": "tm:ltm:pool:members:membersstats",
          "selfLink": "https://localhost/mgmt/tm/ltm/pool/members/
~Common~pool1/members/~Common~1.1.1.1:80/stats?ver=13.0.0",
          "entries": {
           "addr": {
            "description": "1.1.1.1"
           },
           "connq.ageEdm": {
            "value": 0
           },
           "connq.ageEma": {
            "value": 0
           },
           "connq.ageHead": {
            "value": 0
           },
           "connq.ageMax": {
            "value": 0
           },
           "connq.depth": {
            "value": 0
           },
           "connq.serviced": {
            "value": 0
           },
           "curSessions": {
            "value": 0
           },
           "monitorRule": {
            "description": "none"
           },
           "monitorStatus": {
            "description": "unchecked"
           },
           "nodeName": {
            "description": "/Common/1.1.1.1"
           },
           "poolName": {
            "description": "/Common/pool1"
           },
           "port": {
            "value": 80
           },
             ... (truncated for readability) ...

           "sessionStatus": {
            "description": "enabled"
           },
           "status.availabilityState": {
            "description": "unknown"
           },
           "status.enabledState": {
             "description": "enabled"
           },
```

```
            "status.statusReason": {
             "description": "Pool member does not have service checking
 enabled"
            },
            "totRequests": {
             "value": 0
            }
           }
          }
         }
        }
       }
      }
     }
    }
   }
  }
 }
}
```

Statistics are organized in a response as nested objects. At each level of nesting, the response includes a `nestedStats` object that contains the `entries` for an object. The metadata (`kind`, `selfLink`) for each object makes up part of the block for each `nestedStats` object.

Note: Prior to version 11.5, a response object did not contain the `nestedStats` object.

POST and PUT requests

About JSON format for POST and PUT

Unlike a GET request, a POST or PUT request includes a JSON body. When you create or modify a resource, you use the same JavaScript Object Notation (JSON) format as shown in a GET request to define the configuration of an object. Use POST to create a new configuration object from a JSON body, and use PUT or PATCH to edit an existing configuration object with a JSON body.

The format of the JSON body consists of objects that follow the model for an object, as shown:

```
{ "partition":"Common" }
```

Both the name and value appear in double quotes, and a colon separates the name and the value in the pair. For objects that contain multiple name pairs, a comma (,) separates additional name/value pairs. A JSON value must be an object, array, number, string, or one of three literal names: false, null, or true. The other structure is a JSON array, or collection, which is an ordered list of values, as shown:

```
[ { "components":8, "isSubcomponent":"true" } ]
```

In JSON format, square brackets enclose the objects in an array. The objects in the array follow the JSON standard for name/value pairs. Collectively, the name/value pairs are the properties of a BIG-IP® system configuration. For iControl REST, the name/value pairs can be thought of as property name and property value.

In a REST call, declare the format of the object to post . For iControl REST, specify the format `application/json`. In a `curl` command, for example, specify the HTTP header `-H "Content-Type: application/json"` to declare JSON format:

```
curl -k -u username:password -H "Content-Type: application/json"
            -X http-method uri
```

Within the JSON body, define the name of the configuration object. Then include the property names and values for the object, using the same names and properties that appear in the response to a GET request for a similar object. Any properties that you omit revert to the existing values, for a PUT request, or their default values, for a POST request. If you use a tool like curl, you can specify the JSON body in the command line. Several examples in this guide demonstrate the inclusion of a JSON body from the command line.

Creating a new resource with iControl

With the iControl® REST API, you can add a new resource to a BIG-IP® system by using the POST method on an iControl REST collection, and specifying the resource to create as a JSON body. When you create a resource, iControl REST sets all unspecified properties to their default values.

To add a new configuration object, specify the name of the resource as a JSON name/value pair and the path to the collection in the URI.

```
POST https://192.168.25.42/mgmt/tm/ltm/pool

{ "name":"tcb-pool-0" }
```

The response to the POST request shows a new configuration object.

```
{ "allowNat" : "yes",
  "allowSnat" : "yes",
  "fullPath" : "tcb-pool-0",
  "generation" : 5,
  "ignorePersistedWeight" : "disabled",
  "ipTosToClient" : "pass-through",
  "ipTosToServer" : "pass-through",
  "kind" : "tm:ltm:pool:poolstate",
  "linkQosToClient" : "pass-through",
  "linkQosToServer" : "pass-through",
  "loadBalancingMode" : "round-robin",
  "membersReference" : { "isSubcollection" : true,
      "link" : "https://localhost/mgmt/tm/ltm/pool/~Common~tcb-pool-0/
members?ver=11.6.0"
    },
  "minActiveMembers" : 0,
  "minUpMembers" : 0,
  "minUpMembersAction" : "failover",
  "minUpMembersChecking" : "disabled",
  "name" : "tcb-pool-0",
  "queueDepthLimit" : 0,
  "queueOnConnectionLimit" : "disabled",
  "queueTimeLimit" : 0,
  "reselectTries" : 0,
  "selfLink" : "https://localhost/mgmt/tm/ltm/pool/tcb-pool-0?ver=11.6.0",
  "serviceDownAction" : "none",
  "slowRampTime" : 10
}
```

After you create a new pool object by making a POST request, you can use the object.

Modifying a resource with PATCH

Using the PATCH method, you can modify properties of a resource without affecting any other properties.

To modify an object in the BIG-IP® system configuration, specify the resource in the URI. Do not specify a collection in the URI.

```
PATCH https://192.168.25.42/mgmt/tm/pool/~Common~tcb-pool2

{"member": [{:name":"192.168.25.32:80", "description":"Tertiary web
 server"}] }
```

The response to the PATCH request shows the changes to the resource.

```
{
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool2",
   "partition":"Common",
   "fullPath":"/Common/tcb-pool2",
   "generation":59,
   "selfLink":"https://../mgmt/tm/ltm/pool/~Common~tcb-pool2?ver=11.5.0",
   "allowNat":"yes",
   "allowSnat":"yes",
   "ignorePersistedWeight":"disabled",
   "ipTosToClient":"pass-through",
   "ipTosToServer":"pass-through",
   "linkQosToClient":"pass-through",
   "linkQosToServer":"pass-through",
   "loadBalancingMode":"round-robin",
   "minActiveMembers":0,
   "minUpMembers":0,
   "minUpMembersAction":"failover",
   "minUpMembersChecking":"disabled",
   "queueDepthLimit":0,
   "queueOnConnectionLimit":"disabled",
   "queueTimeLimit":0,
   "reselectTries":0,
   "slowRampTime":10,
   "membersReference":{
      "link":"https://../mgmt/tm/ltm/pool/~Common~tcb-pool2/members?
ver=11.5.0",
      "isSubcollection":true
   }
}
```

After completing the PATCH request, you can view the change to the individual resource.

About read only properties

If you specify a read only property with a PUT or POST method, iControl® REST accepts the request and generates an error response. If you specify other properties in addition to the read only property, a valid PUT or POST request will not generate an error, despite the inclusion of the read only property, .

For example, the following `curl` command specifies a read only property in an existing `cm device` object: `timeZone`. The response from iControl® REST indicates a missing property name. In this situation, iControl® REST ignores the read only property and generates the error message shown in the second block.

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X PUT -d \
    '{"time-zone":"EDT"}' \
    https://192.168.25.42/mgmt/tm/cm/device/bigip1
```

```
{
   "code":400,
   "message":"one or more properties must be specified",
```

```
   "errorStack":[

   ]
}
```

Adding or modifying in a specific partition

To add or modify a resource in an administrative partition, add the partition property to the JSON body to modify configuration objects. Use the query option on the command line, or include a `partition` property in the JSON body. Keep in mind that the $filter query parameter applies to GET requests only.

To modify a configuration object with a PUT method, identify the object's partition in the `partition` property.

This example uses the POST method to create a resource in a partition other than the `Common` partition. Specify the name of the resource, and the partition in which to create it, in the JSON body. The response to the request is shown in the third block.

```
POST https://192.168.25.42/mgmt/tm/ltm/pool
```

```
{ "name":"tcb-pool2", "partition":"~fw_objs" }
```

```
{
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool2",
   "partition":"fw_objs",
   "fullPath":"/fw_objs/tcb-pool2",
   "generation":7810,
   "selfLink":"https://localhost/mgmt/tm/ltm/pool/~fw_objs~tcb-pool2?
ver=11.5.0",
   "allowNat":"yes",
   "allowSnat":"yes",
   "ignorePersistedWeight":"disabled",
   "ipTosToClient":"pass-through",
   "ipTosToServer":"pass-through",
   "linkQosToClient":"pass-through",
   "linkQosToServer":"pass-through",
   "loadBalancingMode":"round-robin",
   "minActiveMembers":0,
   "minUpMembers":0,
   "minUpMembersAction":"failover",
   "minUpMembersChecking":"disabled",
   "queueDepthLimit":0,
   "queueOnConnectionLimit":"disabled",
   "queueTimeLimit":0,
   "reselectTries":0,
   "slowRampTime":10,
   "membersReference":{
      "link":"https://../mgmt/tm/ltm/pool/~fw_objs~tcb-pool2/members?
ver=11.5.0",
      "isSubcollection":true
   }
}
```

Following the creation of a new configuration object, this example modifies the member collection by using a PUT request. The URI includes the full path to the resource to modify. Specify the partition property, as well as any

properties you wish to modify. The partition property in the JSON body matches the folder name. The response to the request is shown in the third block.

```
PUT https://192.168.25.42/mgmt/tm/ltm/pool/~fw_objs~tcb-pool2
```

```
{ "name":"tcb-pool2", "partition":"/fw_objs",
"members": [ {"name":"192.168.25.32", "description":"Marketing server"} ] }
```

```
{
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool2",
   "partition":"fw_objs",
   "fullPath":"/fw_objs/tcb-pool2",
   "generation":7914,
   "selfLink":"https://localhost/mgmt/tm/ltm/pool/~fw_objs~tcb-pool2?
ver=11.5.0",
   "allowNat":"yes",
   "allowSnat":"yes",
   "description":"This pool exists in the fw_objs partition.",
   "ignorePersistedWeight":"disabled",
   "ipTosToClient":"pass-through",
   "ipTosToServer":"pass-through",
   "linkQosToClient":"pass-through",
   "linkQosToServer":"pass-through",
   "loadBalancingMode":"round-robin",
   "minActiveMembers":0,
   "minUpMembers":0,
   "minUpMembersAction":"failover",
   "minUpMembersChecking":"disabled",
   "queueDepthLimit":0,
   "queueOnConnectionLimit":"disabled",
   "queueTimeLimit":0,
   "reselectTries":0,
   "slowRampTime":10,
   "membersReference":{
      "link":"https://../mgmt/tm/ltm/pool/~fw_objs~tcb-pool2/members?
ver=11.5.0",
      "isSubcollection":true
   }
}
```

About relative partitions and folder names

If you use a relative folder path within a partition body, iControl® REST interprets the folder name relative to the parent partition. Set the parent partition by specifying the `$filter=partition eq` *`folder-name`* query parameter in the URI, or the `partition` property in the JSON body, depending on the type of request. The `$filter` query parameter applies to GET requests, whereas the `partition` property in a JSON body applies to PATCH, POST or PUT requests. For example, if the `$filter=partition` query option is set to `/eu` and the JSON body includes a reference to the `france` folder, iControl® REST interprets the folder path as `/eu/france`. To avoid ambiguity with partition and folder names, use absolute paths for all folders in JSON body, such as `/eu/france`.

The `$filter` query parameter differs from the OData query parameter in that it only supports filtering by partition names in iControl REST.

Deleting Access Policy Manager resources

Using iControl® REST, you can delete Access Policy Manager™ (APM™) resources.

To delete an Access Policy Manager (APM) resource, such as a `sample-log-setting` resource, make a DELETE request to a resource in the `/mgmt/tm/apm/log-setting` namespace.

```
DELETE https://192.168.25.42/mgmt/tm/apm/log-setting/sample-log-setting
```

iControl REST does not generate a response for a DELETE request but you can verify the deletion of the resource.

Partitions

About administrative partitions

Many types of BIG-IP® system objects, such as profiles and pools, reside in administrative partitions. Partitions are containers with administrative boundaries that you control with access permissions. Through restricted access to administrative partitions, the security model imposes greater control over the configuration objects,which reduces the likelihood of inadvertent changes to the system configuration.

The `Common` partition contains all default profiles, preconfigured monitors, default authentication iRules, the root and admin user accounts, and route domain 0, which is the default route domain. The `Common` partition is created by the BIG-IP® installation process. If there are no other administrative partitions on a system, all objects will be created in the `Common` partition. All administrators can access the `Common` partition. Administrators that have the `Administrator` or `Resource Administrator` role associated with their user account can create partitions.

When you create other partitions, you can associate a user account to that partition and grant permissions to administer that partition. In most circumstances, you either grant a user access to a single partition or universal access to all partitions, A user with access to a single partition can only create objects in that partition. If you grant a user universal access to all partitions, the user must select the partition in which to create an object by specifying the `sys/folder` namespace and the folder name in the request URI.

Every partition has a corresponding folder in the `sys/folder` namespace, including the `Common` partition, which has an associated `/Common` folder. You can specify a namespace in an iControl® REST URI when you create or delete a partition.

Important: You cannot remove the `Common` partition, regardless of your level of administrative access.

Creating folders

You can use iControl®REST methods and properties to create a folder for administrative purposes. There are three different approaches to creating a folder.

Important: You must make a separate request to iControl REST to assign user permissions on a partition.

1. You can create a root-level folder by specifying the path and folder name as the name of the resource.
   To create a root-level folder named `fw_objs`, make a POST request as shown:

   ```
   POST https://192.168.25.42/mgmt/tm/sys/folder

   {
      "name": "fw_objs",
      "partition": "/"
   }
   ```

   The resulting object will have the following properties:

   ```
   {
       "deviceGroup": "none",
       "fullPath": "/fw_objs",
   ```

```
    "generation": 393,
    "hidden": "false",
    "inheritedDevicegroup": "true",
    "inheritedTrafficGroup": "true",
    "kind": "tm:sys:folder:folderstate",
    "name": "fw_objs",
    "noRefCheck": "false",
    "selfLink": "https://localhost/mgmt/tm/sys/folder/~fw_objs?ver=...",
    "subPath": "/",
    "trafficGroup": "/Common/traffic-group-1",
    "trafficGroupReference": {
        "link": "https://localhost/mgmt/tm/cm/traffic-group/
~Common~traffic-group-1?ver=..."
    }
}
```

2. If you want to create a folder named `fw_objs` in the `/Common` folder, you can do so by specifying just the folder name in the `name` property.
   To create a folder named `fw_objs`, in the `/Common` folder, make a POST request as shown:

```
POST https://192.168.25.42/mgmt/tm/sys/folder

{
   "name": "fw_objs"
}
```

   In this step, the path is not specified and the folder is created in the `Common` partition, which is the default partition for iControl REST. If you compare the resulting object from the previous step with the resulting object from this step, you will notice that the current object does not include either the `subPath` property or the `partition` property. When you create a folder in the `Common` partition, iControl REST does not include the enclosing partition property in the response.

   The resulting object has the following properties:

```
{
    "deviceGroup": "none",
    "fullPath": "fw_objs",
    "generation": 403,
    "hidden": "false",
    "inheritedDevicegroup": "true",
    "inheritedTrafficGroup": "true",
    "kind": "tm:sys:folder:folderstate",
    "name": "fw_objs",
    "noRefCheck": "false",
    "selfLink": "https://localhost/mgmt/tm/sys/folder/fw_objs?ver=...",
    "trafficGroup": "/Common/traffic-group-1",
    "trafficGroupReference": {
        "link": "https://localhost/mgmt/tm/cm/traffic-group/
~Common~traffic-group-1?ver=..."
    }
}
```

3. Create a hierarchy of folders by specifying additional properties of the folder object.
   To create the folder `/fw_objs/fw_objs`, use a POST request and specify the `partition`, `subPath`, and `name`.

```
POST https://192.168.25.42/mgmt/tm/sys/folder

{
   "partition": "/",
   "subPath": "fw_objs",
   "name": "fw_objs"
```

```
}
```

You could have specified the partition property in this example as `/fw_objs`, instead of specifying the partition and the sub path separately. As a general approach, everything between the top-level slash ('/') and the partition name constitutes a sub path. Otherwise, a single name constitutes a partition name when preceded by a top-level slash ('/'), as shown in the first example.

The resulting object has the following properties:

```
{
    "deviceGroup": "none",
    "fullPath": "/fw_objs/fw_objs",
    "generation": 410,
    "hidden": "false",
    "inheritedDevicegroup": "true",
    "inheritedTrafficGroup": "true",
    "kind": "tm:sys:folder:folderstate",
    "name": "fw_objs",
    "noRefCheck": "false",
    "partition": "fw_objs",
    "selfLink": "https://localhost/mgmt/tm/sys/folder/~fw_objs~fw_objs?
ver=...",
    "trafficGroup": "/Common/traffic-group-1",
    "trafficGroupReference": {
        "link": "https://localhost/mgmt/tm/cm/traffic-group/
~Common~traffic-group-1?ver=..."
    }
}
```

Deleting an administrative partition

An administrative partition, other than Common, can be deleted with a `DELETE` request. In the URI, specify the folder name of the partition to delete, and submit the request without a JSON body. Because a folder name includes a forward slash, the folder name must be specified with a tilde character.

Important: You can only delete a partition if it is empty. Remove all objects in the partition before you attempt to delete the partition.

To delete a partitionspecify the `DELETE` method and the folder namespace `/mgmt/tm/sys/folder/` in the URI. Replace each forward slash (/) in the folder name with a tilde character (~).

In this example, the iControl® REST request deletes the `/fw_objs` partition from the system configuration. The response includes a response code to indicate success or failure, but the response does not produce a JSON body unless there is an error in the request.

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X DELETE \
    https://192.168.25.42/mgmt/tm/sys/folder/~fw_objs \
    |python -m json.tool
```

Transactions

About the iControl REST transaction model

Some administrative actions in the BIG-IP® system require multiple commands, and in some cases, those commands depend on the successful outcome of other commands. To accommodate complex processes like these, iControl® REST offers transactions, where a transaction is a sequence of individual commands performed as a single unit of work. Transactions work similarly to relational database systems. When handling a database transaction, a relational database system commits the changes if all of the SQL commands run successfully. If any of the SQL commands fail, the relational database system rolls back all of the changes. iControl REST supports a similar feature where a sequence of individual web service requests acts as a single unit of work.

The iControl REST methods you use to create, delete, modify, or query a resource make up the individual commands of a transaction. However, instead of processing each command on arrival, a transaction aggregates multiple commands into a single atomic operation. In this manner, an *atomic transaction* guarantees the all-or-none semantics of a transaction. A transaction completes successfully if all of the individual commands in the transaction complete successfully. Conversely, if any of the commands in a transaction fail, then the entire transaction fails. If the transaction fails, iControl REST rolls back any commands that completed prior to the operation that failed.

About iControl REST transaction phases

The life cycle of a transaction progresses through three phases:

Creation	This phase occurs when the transaction is created using a POST request.
Modification	This phase occurs when commands are added to the transaction, or changes are made to the sequence of commands in the transaction.
Commit	This phase occurs when iControl REST runs the transaction.

iControl REST reserves a namespace for transactions. All commands to create, delete, modify, or query resources within the framework of a transaction use the iControl REST transaction resource namespace `/mgmt/tm/transaction`. This namespace prevents a command from automatically being run by iControl REST when it receives a request. iControl REST creates a transaction in response to a POST request that includes an empty JSON body. In response, iControl REST generates an identifier for the transaction. When you create a transaction, the transaction resource associates three properties with that transaction:

- A read-only `transId` property that identifies a transaction for the life of the transaction.
- A writeable `state` property that indicates the state of the transaction. Values for this property are: STARTED, UPDATING, VALIDATING, COMPLETED, or FAILED. Other than when you commit a transaction, you never change the value of the writeable property `state`.
- A read-only `timeoutSeconds` property that specifies the time period during which to add commands to the transaction. iControl REST sets the value to 120 seconds.

In the modification phase, iControl REST adds a command to a transaction, if a request includes a valid transaction identifier. As with a request to create a transaction, a request to add a command is a POST method that specifies the transaction namespace. Aside from adding commands to a transaction, you can delete a command from a transaction or change the order of the commands in a transaction. Commands are added to a transaction in the order they are received. iControl REST assigns a command identifier to every command added to a transaction. Any changes to an existing transaction, such as a change to the order of the commands, must include a transaction identifier and a command identifier. Deletion of a command also requires a transaction identifier and a command identifier.

The final phase of a transaction is the commit phase. When you are ready to run a transaction, you make a PATCH request and specify the state of the transaction to indicate to iControl REST that it should run the transaction. You must specify the transaction identifier in your request.

Note: In iControl REST version 11.6.0, you can create multiple transactions per user.

About transaction validation

The iControl® REST API provides a property to validate a transaction without actually making any configuration changes to the BIG-IP® system. By using this property, iControl REST determines the likelihood of a successful transaction prior to any attempt to commit the transaction. To use this feature, create a transaction as you normally would and specify the `validateOnly` property in a JSON body when you commit the transaction with a PATCH request. iControl REST returns `HTTP 200 OK` if the transaction request does not generate any errors.

To validate a transaction request, specify `"validateOnly": true` in the JSON body of a PATCH request. The value of the property defaults to `false`. If you specify the property in any other phase than the commit phase, iControl REST ignores the property.

Additional transaction properties

Two noteworthy properties related to transactions include the `executionTimeout` and `asyncExecution` properties. The `executionTimeout` property is a read-only property that specifies the amount of time a transaction may run before the transaction times out. To prevent a transaction from running indefinitely, the property limits the transaction to 300 seconds. The `asyncExecution` property is a Boolean property that allows a transaction to run in the background. If you set `"asyncExecution" : true` in the submit request, the request returns a `202 Accepted` status to indicate that the transaction was accepted for processing. You must poll for the status of an asynchronous transaction. If you are unsure how to check the status of a request, see the topic on creating an iControl® REST transaction.

Creating an iControl REST transaction

Transactions allow you to run a sequence of commands as a single unit of work. Before you can populate a transaction, you must create a transaction by specifying the transaction endpoint.

1. To create a transaction, use the POST method with the `/tm/transaction` namespace. You must include an empty JSON body with the request.

```
POST https://192.168.25.42/mgmt/tm/transaction

{ }
```

   If the POST request is successful, the response contains the transaction identifier. You must include the transaction identifier in a request to indicate that an operation is part of a transaction. Note the three transactions properties in the response: `transId`, `state`, and `timeoutSeconds`.

```
{
   "transId":1389812351,
   "state":"STARTED",
   "timeoutSeconds":120,
   "kind":"tm:transactionstate",
   "selfLink":"https://localhost/mgmt/tm/transaction/1389812351?
ver=11.5.0"
}
```

2. To view the existing transactions, specify one of the transaction endpoints in a query request. To retrieve all transactions in a collection, specify the URI `https://<server name>/mgmt/tm/transaction`. To retrieve a specific transaction, specify the URI `https://<server name>/mgmt/tm/transaction/<transId>`, where *transId* is the identifier for the transaction. If you do not add a command to a transaction within one hundred and twenty (120) seconds, the transaction expires.

```
GET https://192.168.25.42/mgmt/tm/transaction
```

```
GET https://192.168.25.42/mgmt/tm/transaction/<transId>
```

Modifying a transaction

After you create a transaction, you can populate the transaction by adding commands. Individual commands comprise the operations that a transaction performs. Commands are added in the order they are received but you can delete commands or change the order of the commands in the transaction.

1. To add a command to a transaction, use the POST method and specify the `X-F5-REST-Coordination-Id` HTTP header with the transaction ID value from the example (`1389812351`). In the example, the request creates a new pool and adds a single member to the pool.

```
POST https://192.168.25.42/mgmt/tm/ltm/pool

X-F5-REST-Coordination-Id:1389812351

{
   "name":"tcb-xact-pool",
   "members": [ {"name":"192.168.25.32:80","description":"First pool for
 transactions"} ]
}
```

   The response indicates that iControl® REST added the operation to the transaction.

```
{
   "transId":1389812351,
   "state":"STARTED",
   "timeoutSeconds":120,
   "kind":"tm:transactionstate",
   "selfLink":"https://localhost/mgmt/tm/transaction/1389813931?
ver=11.5.0"
}
```

2. Optional: To query a single transaction, specify the URI `https://<server name>/mgmt/tm/transaction/transId`, where *transId* is the identifier of the transaction.

```
GET https://192.168.25.42/mgmt/tm/transaction/138912351
```

3. Optional: To obtain a list of commands in a transaction, specify the URI `https://<server name>/mgmt/tm/transaction/transId/commands`, where *transId* is the identifier of the transaction.

```
GET https://192.168.25.42/mgmt/tm/transaction/138912351/commands
```

4. Optional: To obtain the details of a single operation, specify the URI `https://<server name>/mgmt/tm/transaction/transId/commands/commandId`, where *transId* is the identifier of the transaction, and *commandId* is the identifier of the operation.

```
GET https://192.168.25.42/mgmt/tm/transaction/138912351/commands/1
```

5. Optional: To remove a command from a transaction, specify the URI `https://<server name>/mgmt/tm/transaction/transId/commands/commandId`, where *transId* is the identifier of the transaction, and *commandId* is the identifier of the command. iControl REST renumbers the remaining commands in the transaction.

```
DELETE https://192.168.25.42/mgmt/tm/transaction/138912351/commands/1
```

6. Optional: To change the evaluation order, specify the URI `https://<server name>/mgmt/tm/transaction/transId/commands/commandId`, where *transId* is the identifier for the transaction, and *commandId* is the identifier for the command. In the JSON message body, specify a key/value pair `"evalOrder":y`, where y represents a new evalOrder value. This action moves the command.

Committing an iControl REST transaction

After you finish adding commands to a transaction, and you are satisfied with the evaluation order of the commands, you can run the sequence of commands by committing the transaction. Each operation in the transaction must complete successfully. If an operation fails, the transaction rolls back any changes and returns an error. If you choose not to run the transaction at this point, you can delete the transaction.

1. To commit a transaction, use the PATCH method. In the JSON body, specify the state of the transaction as `VALIDATING`.

```
PATCH https://localhost/mgmt/tm/transaction/1389812351

{ "state":"VALIDATING" }
```

2. Optional: To delete a transaction, specify the URI `https://localhost/mgmt/tm/transaction/transId`, where *transId* is the transaction identifier. iControl® REST deletes all operations associated with this transaction.

```
DELETE https://localhost/mgmt/tm/transaction/1389812351
```

About iControl REST asynchronous tasks

iControl® REST requests run in a synchronous manner and complete within a short period of time, usually in a matter of seconds. A single iControl REST request may run for a longer period of time, and do so without providing any indication of the eventual success or failure of the request. In some situations, a request may time out prior to completion of the request.

iControl REST addresses the problems associated with a long-running request by allowing asynchronous tasks for some endpoints. A *long-running request* is a request that routinely takes more than 60 seconds to complete. If the endpoint you are targeting exists in the table of endpoints, you should consider making your request an asynchronous task. A POST request to an asynchronous task URI notifies iControl REST to create a task and then respond to additional requests for task state. As part of the initial response to the POST request, iControl REST returns a JSON body that includes a self link that you use to poll the task. To monitor an asynchronous task, you create a task and then poll the task by identifier to determine the state of the task. All asynchronous tasks are in one of the following states: `UPDATING`, `VALIDATING`, `COMPLETED`, or `FAILED`. iControl REST sets the initial state of a task to `UPDATING` and then returns an HTTP 200 status code to indicate the creation of the task.

When the asynchronous task completes, iControl REST changes the state of the task to `COMPLETED`. The response to a polling request for a completed task includes a JSON body with a self link to the task results. After you review the results, you should delete the results and then delete the task, in that order.

Asynchronous task endpoints

This table lists common iControl® REST API endpoints along with corresponding asynchronous task endpoints, organized by function.

Description	URI (synchronous)	URI (asynchronous)
Save/Load config	POST tm/sys/config	POST tm/task/sys/config
Save/Load UCS	POST tm/sys/ucs	POST tm/task/sys/ucs
Load IP geolocation data	POST tm/sys/geoip	POST tm/task/sys/geoip
Load classification signatures	POST tm/sys/classification-signature	POST tm/task/sys/classification-signature
Failover	POST tm/sys/failover	POST tm/task/sys/failover

Description	URI (synchronous)	URI (asynchronous)
Load DNS-Express® DB	POST tm/ltm/dns-express-db	POST tm/task/ltm/dns-express-db
Load URL DB feed list	POST tm/ltm/classification/urldb-feed-list	POST tm/task/ltm/classification/urldb-feed-list
Load classification signatures	POST tm/ltm/classification/signatures	POST tm/task/ltm/classification/signatures
Update signatures	POST tm/ltm/classification/update-signatures	POST tm/task/ltm/classification/update-signatures
Install EPSEC package	POST tm/apm/epsec/epsec-package	POST tm/task/apm/epsec/epsec-package
Create vCMP® guest	POST tm/vcmp/guest	POST tm/task/vcmp/guest
Run CLI scripts	POST tm/cli/script	POST tm/task/cli/script
Verify WOM configuration	POST tm/wom/verify-config	POST tm/task/wom/verify-config
Diagnose WOM connections	POST tm/wom/diagnose-conn	POST tm/task/wom/diagnose-conn
Load/Save/Publish WAM policy	POST tm/wam/policy	POST tm/task/wam/policy
Load firewall FQDN entity	POST tm/security/firewall/fqdn-entity	POST tm/task/security/firewall/fqdn-entity
Load IP intelligence feed list	POST tm/security/ip-intelligence/feed-list	POST tm/task/security/ip-intelligence/feed-list
Load/update anti-fraud signatures	POST tm/security/anti-fraud/signatures-update	POST tm/task/security/anti-fraud/signatures-update
Load/update anti-fraud engine update	POST tm/security/anti-fraud/engine-update	POST tm/task/security/anti-fraud/engine-update
Load PEM subscribers	POST tm/pem/subscribers	POST tm/task/pem/subscribers
Start/Stop/Restart ILX plug-in	POST tm/ilx/plugin	POST tm/task/ilx/plugin
Run config sync	POST tm/cm/config-sync	POST tm/task/cm/config-sync
Add device to trust domain	POST tm/cm/add-to-trust	POST tm/task/cm/add-to-trust
Remove device from trust domain	POST tm/cm/remove-from-trust	POST tm/task/cm/remove-from-trust

Using an asynchronous task

An asynchronous task provides an alternative to a long-running synchronous task.

1. To create an asynchronous task, locate the endpoint for the task in the asynchronous task endpoints table. For this example, identify the corresponding endpoint for `/tm/sys/ucs` (`/tm/task/sys/ucs`) and supply a JSON body.

```
POST https://192.168.25.42/mgmt/tm/task/sys/ucs

{
   "command": "save",
   "name": "myUcs"
}
```

In the response from the request, locate the reference endpoint (`selfLink`) to query for the task state. You will use the endpoint in the subsequent steps.

```
{
   "command":"save",
   "name":"myUcs",
   "selfLink":"https://localhost/mgmt/tm/task/sys/ucs/1234&ver=12.0.0",
   "_taskID":"1234",
   "_taskState":"UPDATING",
   "_taskTimeInStateMs":0,
   "_taskResultLink":"https://localhost/mgmt/tm/task/sys/ucs/1234/
result&ver=12.0.0",
   "_taskWaitTime":30000
}
```

2. To start the task, modify the state of the task in a PUT request and specify the `selfLink` as the endpoint. Specify `VALIDATING` as the value of the `_taskState` property. You may safely omit the version parameter in the URI.

   Note that if you do not modify the state of the task, the task will not run and eventually will be deleted.

```
PUT https://192.168.25.42/mgmt/tm/task/sys/ucs/1234

{
   "_taskState": "VALIDATING"
}
```

   If the request was successful, you should see a response similar to the following:

```
{
   "_code": 202,
   "errorStack": [],
   "message": "Task will execute asynchronously."
}
```

3. Verify the state of the asynchronous task.

   To monitor the progress of the task, you can periodically make a GET request to the reference endpoint to check the state of the task.

```
GET https://192.168.25.42/mgmt/tm/task/sys/ucs/1234
```

   The response at some point should indicate that the task has completed.

```
{
   "_taskId":1234,
   "_taskResultLink":"https://localhost/mgmt/tm/task/sys/ucs/1234/
result&ver=12.0.0",
   "_taskState":"COMPLETED",
   "_taskTimeInStateMs":0,
   "selfLink":"https://localhost/mgmt/tm/task/sys/ucs/1234&ver=12.0.0"
}
```

4. When the task completes, make a GET request to the result endpoint.

```
GET https://192.168.25.42/mgmt/tm/task/sys/ucs/1234/result
```

In this example, you submitted and started an asynchronous task, and viewed the results of the task.

After you view the results of the task, delete the results and then delete the initial task by URI.

Commands

About other tmsh global commands

Not all *Traffic Management Shell (tmsh) Reference* commands map directly to HTTP methods. For a `list` or `show` request of a resource, a GET request maps well to the requested operation, but the reference includes global commands that do not directly correspond to an HTTP method. iControl® REST implements the following set of `tmsh` commands:

- `cp`
- `generate`
- `install`
- `load`
- `mv`
- `publish`
- `reboot`
- `restart`
- `reset_stats`
- `run`
- `save`
- `send-mail`
- `start`
- `stop`

iControl REST supports these `tmsh` commands by mapping a command, as well as options, to JSON format.

The iControl REST format for `tmsh` commands follows this general approach:

- Use the POST method.
- Specify a namespace for the `tmsh` command in the URI.
- Specify the command and options as the values of the properties in the JSON body.

To run the command, use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/application/template`, along with the JSON body for the command. In each example, a relative URI is used in the request body.

Using the cp command

Utility commands do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/application/template`, along with a JSON body that specifies the name of the utility command.

To copy using the `cp` command, make an iControl® REST request with the POST method and specify the properties in a JSON body.

To copy a file using the `cp` command, make a POST request. In the JSON body, specify the command, file name, and target file name.

```
POST /mgmt/tm/sys/application/template
```

```
{
    "command":"cp",
    "name":"tempt1",
    "target":"tempt2",
```

```
}
```

Using the generate command

Global commands like generate do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/ltm/rule`, along with a JSON body that specifies the name of the command.

To generate signed scripts using the `generate` command, make an iControl® REST request with the POST method and specify the properties in a JSON body.

To generate a signed script using the generate command, make a POST request. In the JSON body, specify the command, script name, options, and a signing key. The signing key property name uses a hyphenated name instead of the came case naming convention of iControl® REST.

```
POST /mgmt/tm/ltm/rule
```

```
{
    "command":"generate",
    "name":"rule1",
    "options":[
        {
           "signature":true
        }
     ],
     "signing-key":"key1"
}
```

Using the install command

Global commands like `install` do not have a direct mapping to an HTTP method. So you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/software/image`, along with a JSON body that specifies the name of the command. This topic shows two examples of the `install` command.

1. To install and update components using the `install` command, make an iControl® REST request with the POST method and a JSON body.

   ```
   POST /mgmt/tm/sys/software/image
   ```

   ```
   {
       "command":"install",
       "name":"BIGIP-11.5.0.930.400.iso",
       "volume":"HD1.3"
   }
   ```

2. To perform the same task and take advantage of the options for the install command, follow the previous steps and specify the `create-volume` and `reboot` options in the JSON body. The create volume property name uses a hyphenated name instead of the camel-casing convention of iControl REST.

   ```
   POST /mgmt/tm/sys/software/image
   ```

   ```
   {
       "command":"install",
       "options":[
          {
             "create-volume":true
   ```

```
        },
        {
           "reboot":true
        }
    ],
    "name":"BIGIP-11.4.0.737.400.42.iso",
    "volume":"HD1.1"
}
```

Using iControl REST to create a key

Instead of using the `tmsh key` command to create a private key, you can make an iControl® REST request to the `key` endpoint.

To create a key, make an iControl REST request using the POST method and specify the name of the key in a JSON body.

```
POST https://192.168.25.42/mgmt/tm/sys/crypto/key
```

```
{
  "name": "key-no-part.key"
}
```

Note: You must specify the extension (`key`) in the name of the key to create. If you omit the extension, iControl REST generates an error response despite successfully creating the key.

```
{
  "kind": "tm:sys:crypto:key:keycollectionstate",
  "selfLink": "https://localhost/mgmt/tm/sys/crypto/key?ver\u003d13.1.0",
  "items": [
.
.
.
    {
      "kind": "tm:sys:crypto:key:keystate",
      "name": "/Common/key-no-part.key",
      "fullPath": "/Common/key-no-part.key",
      "generation": 44690,
      "selfLink": "https://localhost/mgmt/tm/sys/crypto/key/~Common~key-no-
part.key?ver\u003d13.1.0",
      "keySize": "2048",
      "keyType": "rsa-private",
      "securityType": "normal"
    },
.
.
.
  ]
}
```

Using the load command

Global commands like load do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/config`, along with a JSON body that specifies the name of the command.

Load BIG-IP® system configuration using the `load` command by making an iControl® REST request with the POST method and a JSON body.

To replace the running configuration using the load command, make a POST request. In the JSON body, specify the command.

```
POST /mgmt/tm/sys/config
```

```
{
    "command":"load",

    "name": "default"
}
```

Using the mv command

Global commands like `mv` do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/cm/device`, along with a JSON body that specifies the name of the command.

To copy using the `mv` command, make an iControl® REST request with the POST method and specify the properties in a JSON body.

To move or rename an object using the `mv` command, make a POST request. In the JSON body, specify the command, name, and target:

`POST /mgmt/tm/cm/device`

```
{
    "command":"mv",
    "name":"bigip1",
    "target":"selfdevice2",
}
```

Using the publish command

Global commands, such as publish, do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/asm/policy`, along with a JSON body that specifies the name of the command.

Publish changes in a policy by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

In the JSON body, specify the command, name of the policy, and the application service. The application service property name uses a hyphenated name instead of the camel case naming convention of iControl REST.

```
POST /mgmt/tm/asm/policy
```

```
{
    "command":"publish",
    "name":"testpolicy",
    "app-service":"service",
}
```

Using the reboot command

Global commands like reboot do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys`, along with a JSON body that specifies the name of the command.

Reboot a system, or boot a system into a different volume by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To reboot a system using the reboot command, make a POST request. In the JSON body, specify the command.

```
POST /mgmt/tm/sys
```

```
{
    "command":"reboot"
}
```

Using the restart command

Global commands like restart do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/service`, along with a JSON body that specifies the name of the command.

Restart a service by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To restart a service using the restart command, make a POST request. In the JSON body, specify the command and the name of the service to restart.

```
POST /mgmt/tm/sys/service
```

```
{
    "command":"restart",
    "name":"icrd"
}
```

Using the reset-stats command

Global commands like reset-stats do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/ltm/virtual`, along with a JSON body that specifies the name of the command.

Reset statistics for a component by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To reset statistics for a component using the reset-stats command, make a POST request. In the JSON body, specify the command and the name of the component.

```
POST /mgmt/tm/ltm/virtual
```

```
{
    "command":"reset-stats",
    "name":"http_vs1"
}
```

Using the run command

Global commands like run do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/util/ping`, along with a JSON body that specifies the name of the command.

Run a program by making an iControl® REST request with the POST method and specifying the properties in a JSON body. .

To run a command using the run command, make a POST request. In the JSON body, specify the command and the options for the command.

```
POST /mgmt/tm/util/ping
```

```
{
    "command":"run",
    "utilCmdArgs":"1.1.1.1 -c 1 -i 10"
}
```

Using the save command

Global commands like save do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/config`, along with a JSON body that specifies the name of the command.

Save the running configuration of a BIG-IP® system by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To save the running configuration using the save command, make a POST request. In the JSON body, specify the command.

```
POST /mgmt/tm/sys/config
```

```
{
    "command":"save"
}
```

To use the options available for the save command, specify the command and the options in a JSON body.

```
{
    "command":"save",
    "options":[
       {
          "file":"configfile.scf"
       }
    ]
}
```

Using the send-mail command

Global commands like send-mail do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/analytics/application-security/report`, along with a JSON body that specifies the name of the command.

Send an e-mail to recipients by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To send e-mail using the send-mail command, make a POST request. In the JSON body, specify the command. Specify the options, as well as the recipients, in the JSON body. Several of the property names use a hyphenated name instead of the camel case naming convention of iControl® REST.

```
POST /mgmt/tm/analytics/application-security/report
```

```
{
    "command":"send-mail",
    "view-by":"ip",
    "format":"pdf",
    "email-addresses":[
       "wchen@f5.com"
     ],
     "measures":[
        "illegal-transactions"
     ],
     "limit":20,
     "order-by":[
        {
           "measure":"illegal-transactions",
           "sort-type":"desc"
        }
     ],
     "smtp-config-override":"smtpserver"
}
```

Using the start command

Global commands like start do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/icall/handler/perpetual`, along with a JSON body that specifies the name of the command.

Start a service by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To start a service using the start command, make a POST request. In the JSON body, specify the command and the name of the service.

```
POST /mgmt/tm/sys/icall/handler/perpetual
```

```
{
    "command":"start",
    "name":"perphd1"
}
```

Using the stop command

Global commands like stop do not have a direct mapping to an HTTP method, so you must use the POST method and specify an absolute URI, such as `https://192.168.25.42/mgmt/tm/sys/icall/handler/perpetual`, along with a JSON body that specifies the name of the command.

Stop a service by making an iControl® REST request with the POST method and specifying the properties in a JSON body.

To stop a service using the stop command, make a POST request. In the JSON body, specify the command and the name of the service.

```
POST /mgmt/tm/sys/icall/handler/perpetual
```

```
{
    "command":"stop",
    "name":"perphd1"
}
```

Application Security Manager

Application Security Manager and iControl REST comparison

If you use Application Security Manager™ (ASM™), you should understand how ASM differs from iControl REST.

Application Security Manager™ (ASM™) shares much in common with iControl® REST. As with any organizing collection in iControl REST, ASM supports discovery of the API, common methods, as well as a set of query parameters. However, ASM offers some features that distinguish it from iControl REST, as outlined in the following list.

- ASM resource URIs include an MD5 hash that identifies the resource.
- ASM implements a larger set of Open Data Protocol (OData) query parameters, functions, and operators.
- ASM does not implement custom query parameters, like `expandSubcollections`.
- ASM does not support the `/stats` endpoint.
- ASM supports tasks, not transactions.

The following table lists the HTTP methods that ASM supports.

Method	Description
GET	For both collections and other resources, ASM supports the GET method to retrieve or search. The `$filter` query parameter support in ASM includes more options than iControl REST.
POST	For both collections and other resources, ASM supports the POST method to create an entity. A POST request must include a JSON body, although the JSON body may be empty.
DELETE	For most collections, ASM supports the DELETE method. ASM supports the deletion of subsets of collections that match a `$filter` query. For other resources, ASM supports the DELETE method. With the inclusion of a query parameter, ASM also supports the option to delete multiple entities.
PUT	For collections or elements, ASM does not support the PUT method.
PATCH	For collections, ASM supports the PATCH method. In ASM, PATCH can update multiple entities if you specify a query option in the URI. For other resources, ASM supports the PATCH method. The PATCH method

Method	Description
	updates specified properties; PATCH does not reset or overwrite properties that are not specified in the request.

ASM implements OData Version 4 and provides some support for OData Version 3 string functions. ASM supports the query options and functions, with restrictions, listed in the following table.

Parameter	Description
`$filter`	Specifies a filter for a retrieve, update, or delete operation. In ASM, `$filter` supports the `contains`, `endswith`, `startswith`, and `substringof`string functions. No math functions are supported.
`$select`	Specifies a subset of the properties to appear in the result set.
`$skip`	Specifies the number of rows to skip in the result set. The result set is chosen from the remaining rows.
`$top`	Specifies the first `n` rows of the result set.
`$expand`	Specifies the inclusion of related entities in the result set.
`$orderby`	Specifies the order in which to display items. The `$orderby` parameter cannot be applied to a subfield inside of an expanded field, such as `$orderby=requestPolicy/name` on `/tm/asm/events/requests`.

As with iControl REST, ASM also supports comparison and logical operators as described by the OData protocol. The following table lists the ASM operators.

Operator	Description
`eq`	Equal to operator.
`ne`	Not equal to operator.
`lt`	Less than operator.
`le`	Less than or equal to operator.
`gt`	Greater than operator.
`ge`	Greater than or equal to operator.
`and`	True if both operands are true operator. Supports grouping of fields within an element for $filter, such as `signatureOverrides/id eq 'IDx' AND signatureOverrides/isAllowed eq TRUE`.
`or`	True if either operand is true. In ASM, `$filter` supports the `or` operator for conditions that apply to one field, such as `(A eq 1 OR A eq 2)`.
`not`	Negation of operand operator.

ASM supports the aggregate OData functions SUM, AVG, MAX, and MIN.

The following table lists the ASM namespaces.

Namespace	Description
/tm/asm/attack-types	Collection, read-only.
/tm/asm/signatures	Collection that does not support update many or delete many requests.
/tm/asm/signature-statuses	Collection, read-only.
/tm/asm/signature-sets	Collection that does not support update many or delete many requests.
/tm/asm/signatures-update	Element
/tm/asm/signature-systems	Collection, read-only.
/tm/asm/policy-templates	Collection, read-only.
/tm/asm/policies	Collection that does not support update many or delete many requests. Collections within this namespace: • /tm/asm/policies/<MD5Hash>/methods • /tm/asm/policies/<MD5Hash>/filetypes • /tm/asm/policies/<MD5Hash>/cookies • /tm/asm/policies/<MD5Hash>/host-names • /tm/asm/policies/<MD5Hash>/blocking-settings/violations • /tm/asm/policies/<MD5Hash>/blocking-settings/evasions • /tm/asm/policies/<MD5Hash>/blocking-settings/http-protocols • /tm/asm/policies/<MD5Hash>/blocking-settings/web-services-securities • /tm/asm/policies/<MD5Hash>/urls • /tm/asm/policies/<MD5Hash>/parameters • /tm/asm/policies/<MD5Hash>/urls/<MD5Hash>/parameters • /tm/asm/policies/<MD5Hash>/whitelist-ips • /tm/asm/policies/<MD5Hash>/gwt-profiles • /tm/asm/policies/<MD5Hash>/json-profiles • /tm/asm/policies/<MD5Hash>/xml-profiles • /tm/asm/policies/<MD5Hash>/signatures • /tm/asm/policies/<MD5Hash>/signatures-sets

Retrieving Application Security Manager resources

Consistent with iControl® REST behavior, Application Security Manager™ (ASM™) supports querying of endpoints within the namespace `/mgmt/tm/asm`. As with any other organizing collection in iControl® REST, you can make a GET request to discover the resources of ASM.

1. Make a request to the endpoint `/mgmt/tm/asm` to query for ASM resources.
2. To discover the resources of ASM, make a GET request to the root namespace, (`/mgmt/tm/asm`), as shown in this example.

```
GET https://192.168.25.42/mgmt/tm/asm

{
   "selfLink":"https://localhost/mgmt/tm/asm",
```

```
   "kind":"tm:asm:asmcollectionstate",
   "items":[
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/tasks"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/signature-update"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/policies"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/policy-templates"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/signatures"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/signature-statuses"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/signature-sets"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/signature-systems"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/asm/attack-types"
         }
      }
   ]
}
```

3. To expand one of the links in the response, make another GET request, specifically for a resource.

   This example expands one of the links in the response from the previous request. Note that each URI contains a hash string as a resource identifier.

```
GET https://192.168.25.42/mgmt/tm/asm/policies

{
   "selfLink":"https://localhost/mgmt/tm/asm/policies",
   "kind":"tm:asm:policies:policycollectionstate",
   "items":[
      {
         "policyBuilderReference":{
```

```
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/policy-builder"
         },
         "blockingSettingReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/blocking-settings",
            "isSubCollection":true
         },
         "cookieReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/cookies",
            "isSubCollection":true
         },
         "hostNameReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/host-names",
            "isSubCollection":true
         },
         "selfLink":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A",
         "stagingSettings":{
            "signatureStaging":true,
            "enforcementReadinessPeriod":7
         },
         "versionDeviceName":"10000-1-E12U39.sh",
         "signatureReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/signatures",
            "isSubCollection":true
         },
         "createdDatetime":"2013-12-06T19:29:54Z",
         "filetypeReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/filetypes",
            "isSubCollection":true
         },
         "id":"MwavowFbOsSD-Fgt4trP6A",
         "modifierName":"admin",
         "versionDatetime":"2013-12-26T23:12:57Z",
         "subPath":"/Common",
         "versionLastChange":"Policy Attributes  [update]: Policy Builder
 determined that security policy \"/Common/my-VS\" is unstable.",
         "active":true,
         "caseInsensitive":false,
         "name":"my-VS",
         "description":"",
         "fullPath":"/Common/my-VS",
         "policyBuilderEnabled":true,
         "trustXff":false,
         "partition":"Common",
         "attributes":{
            "pathParameterHandling":"as-parameters",
            "triggerAsmIruleEvent":"disabled",
            "maskCreditCardNumbersInRequest":true,
            "inspectHttpUploads":false,
            "maximumHttpHeaderLength":2048,
            "maximumCookieHeaderLength":2048,
            "useDynamicSessionIdInUrl":false
         },
         "xmlProfileReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/xml-profiles",
            "isSubCollection":true
         },
```

```
         "methodReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/methods",
            "isSubCollection":true
         },
         "customXffHeaders":[

         ],
         "creatorName":"admin",
         "kind":"tm:asm:policies:policystate",
         "urlReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/urls",
            "isSubCollection":true
         },
         "virtualServers":[
            "/Common/my-VS"
         ],
         "headerReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/headers",
            "isSubCollection":true
         },
         "protocolIndependent":false,
         "lastUpdateMicros":1.386358822e+15,
         "signatureSetReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/signature-sets",
            "isSubCollection":true
         },
         "allowedResponseCodes":[
            400,
            401,
            404,
            407,
            417,
            503
         ],
         "parameterReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/parameters",
            "isSubCollection":true
         },
         "jsonProfileReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/json-profiles",
            "isSubCollection":true
         },
         "applicationLanguage":"utf-8",
         "enforcementMode":"transparent",
         "isModified":false,
         "gwtProfileReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/gwt-profiles",
            "isSubCollection":true
         },
         "whitelistIpReference":{
            "link":"https://../mgmt/tm/asm/policies/MwavowFbOsSD-
Fgt4trP6A/whitelist-ips",
            "isSubCollection":true
         },
         "versionPolicyName":"/Common/Dummy-VS"
      }
   ]
```

```
}
```

4. To search for properties of a resource, make a GET request and append a query string to the URI, as shown in this example.

```
GET https://192.168.25.42/mgmt/tm/asm/policies?$filter=name eq my-VS
```

Creating Application Security Manager resources

Consistent with iControl® REST behavior, Application Security Manager™ (ASM™) supports creation of resources within the namespace `/mgmt/tm/asm`. As with any other organizing collection in iControl® REST, you can make a POST request to create a resource in ASM.

To create a new resource, make a POST request using the namespace `/mgmt/tm/asm`.

```
POST https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/urls

{
   "name": "/login.php",
   "protocol": "http",
   "description": "A Login Page"
}
```

```
{
   "id": "<MD5HASH>",
   "name": "/login.php",
   "kind": "tm:asm:policies:urls:urlState",
   "selfLink": "https://localhost/mgmt/tm/asm/policies/<MD5HASH>/urls/
XPiqHHfl7UsVKku63zrd-g",
   "protocol": "http",
   "type": "explicit",
   "staging": true,
   "description": "A Login Page",
   "modifiedDatetime": "1990-12-31T23:59:60Z",
   "allowed": true,
   "checkFlow": false,
   "navigationParameters": false,
   "checkMetachars": true,
   "clickjackingProtection": false,
   "contentProfiles: [
      {
         "headerName": "*",
         "headerValue": "*",
         "headerOrder": "default",
         "type": "http",
         "inClassification": false
      }
   ]
   "parameterReference": {
      "link": "https://localhost/mgmt/tm/asm/policies/<MD5HASH>/urls/
XPiqHHfl7UsVKku63zrd-g/parameters"
   },
}
```

Updating Application Security Manager resources

Consistent with iControl® REST behavior, Application Security Manager™ (ASM™) supports updating of resources within the namespace `/mgmt/tm/asm`. As with any other resources in iControl® REST, you can update an ASM collection or other resource with a PATCH request.

1. To update a resource, make a PATCH request to a resource in the namespace `/mgmt/tm/asm` and include a JSON body.

```
PATCH https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/urls/

{
   "clickjackingProtection": true,
   "clickjackingtype":"Never"
}
```

2. To update multiple ASM entities with a single request, make a PATCH request and specify a query parameter in the URI.

```
PATCH https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/urls?
$filter=type eq explicit

{ "staging": false }
```

Deleting resources in Application Security Manager

Consistent with iControl® REST behavior, the namespace for Application Security Manager™ (ASM™) includes endpoints within the namespace `/mgmt/tm/asm/tasks/import-policy/`. As with any other resources in iControl REST, you can make a DELETE request to delete a resource in ASM.

1. To delete a resource, make a DELETE request and specify a resource in the namespace `/mgmt/tm/asm/tasks/import-policy/`.

```
DELETE https://192.168.25.42/mgmt/tm/asm/tasks/import-policy/
ZuJ5QPuFj9r_LwbrDgoPsg
```

```
{
   "isBase64":false,
   "status":"FAILURE",
   "name":"TCB policy",
   "lastUpdateMicros":1.389135008e+15,
   "kind":"tm:asm:tasks:import-policy:import-policy-taskstate",
   "selfLink":"https://../mgmt/tm/asm/tasks/import-policy/
ZuJ5QPuFj9r_LwbrDgoPsg",
   "filename":"tcbpolicy.xml",
   "id":"ZuJ5QPuFj9r_LwbrDgoPsg",
   "startTime":"2014-01-07T22:50:08Z",
   "result":{
      "message":"Exported policy file not found!."
   }
}
```

2. To delete multiple entities, make a DELETE request and specify a query parameter in the URI.

```
DELETE https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/urls/?
$filter=staging eq true
```

Application Security Manager policy

If you use Application Security Manager™ (ASM™) to import, export, or activate policy, you should understand how ASM differs from iControl REST.

iControl® REST supports the Application Security Manager™ (ASM™) features of importing, exporting, and activating policies. The individual task topics state all required properties for a request.

Property	Description
filename	Specifies the name of a local system file that contains the policy to import.
file	Specifies inline content in XML format to import. For import requests, the inline content is input. For export requests, the response contains the content inline.
isBase64	Indicates whether the inline content is Base64 encoded. Applies to both input and output content.
minimal	Indicates whether to export only custom settings.
name	Specifies the short name of a policy. Only applies to new policies.
fullPath	Specifies the fully qualified path and name of a policy.
policyReference	Specifies the link to a policy to activate, replace or create, or export.
policyTemplateReference	Specifies the template for a policy.

Importing a policy in Application Security Manager

iControl® REST supports the Application Security Manager™ (ASM™) task to import a policy from another ASM system. You can use the imported policy as a base policy on another system.

1. Optional: To upload a file from which to import the policy, use the POST method and specify the `/tm/asm/file-transfer/uploads` endpoint. You must specify the file name in the request.

```
POST https://192.168.25.42/mgmt/tm/asm/file-transfer/uploads/<filename>
```

   The following is an example request command:

```
curl -k -u admin:admin -H "Content-Type: application/json" "https://<big-
ip_man_ip>/mgmt/tm/asm/file-transfer/uploads/import_policy_API.xml" -X
 POST -H "Content-Range: 0-<lenght_in_bytes>/<length_in_bytes>" --data-
binary @/home/user-x/previous_exported_policy_API.xml
```

2. To import a policy, make a POST request to the `/mgmt/tm/asm/tasks/import-policy` namespace.
3. In the JSON body, specify a property that identifies the source of the import data.

   You must supply one property from the list:

   - `file`
   - `filename`
   - `policyReferenceTemplate`

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/import-policy

{
   "filename": "mypolicy.xml",
   "name": "NewPolicy"
}
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "name": "NewPolicy",
    "filename": "mypolicy.xml"
    "kind": "tm:asm:tasks:import-policy:importpolicytaskstate",
```

```
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

4. Make a GET request and specify the `id` property in the URI to determine the success of the policy import operation.

   The response shows the `result` and `status` properties that indicate the success of the request.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ

{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:import-policy:importpolicytaskstate",
    "name": "NewPolicy",
    "filename": "mypolicy.xml"
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "policyReference": {
            "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
        }
    }
}
```

Exporting a policy in Application Security Manager

iControl® REST supports the Application Security Manager™ (ASM™) task for exporting a policy to another server. You can use the exported policy as a base policy on another system.

1. To export a policy, make a POST request to the `/mgmt/tm/asm/tasks/export-policy` endpoint. You must specify either the `filename` property or the `inline` property in the request.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/export-policy

{
   "filename": "exported_file.xml",
   "minimal": true,
   "policyReference": {
            "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    }
}
```

   The response to the request contains the following data:

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "filename": "exported_file.xml",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "minimal": true,
```

```
    "kind": "tm:asm:tasks:export-policy:exportpolicytaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/export-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. Optional: To determine the status of the policy export operation, use the GET method and specify the `id` of the request.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/export-policy/
oqNah2PxtwwE4YyAHGekNQ
```

The response to the request contains the following data:

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "filename": "exported_file.xml",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "minimal": true,
    "kind": "tm:asm:tasks:export-policy:exportpolicytaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/export-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "filename": "exported_file.xml",
        "fileSize": 32045
    }
}
```

3. Optional: To download the file, use the GET method and specify the `/tm/asm/file-transfer/downloads` endpoint, along with the name of the exported file. You must specify the name of the file in the request.

```
GET https://192.168.25.42/mgmt/tm/asm/file-transfer/downloads/<filename>
```

Applying a policy in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) task to manually apply a policy that protects a web site.

1. To apply a policy, make a POST request with the `/tm/asm/tasks/apply-policy` namespace.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/apply-policy

{
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    }
}
```

The response to the request contains the following data:

```
{
```

```
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:apply-policy:applypolicytaskstate",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "lastUpdateMicros": 1370459678272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/apply-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To determine the status of the apply policy operation, make a GET request to the same namespace.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/apply-policy
```

The response to the request contains the following data:.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:apply-policy:applypolicytaskstate",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "lastUpdateMicros": 1370459678272126,
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/apply-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00"
}
```

Finding policy differences in Application Security Manager

You can determine the differences between two policies for the purpose of resolving and merging the differences. This aspect of the difference functionality is available only in the iControl® REST API.

1. To find the differences between policies, choose two policies to compare and denote the policies in a JSON body, as shown. You can compare any two policies that have these characteristics in common: encoding, case sensitivity, and protocol independence.

```
{
 "firstPolicyReference": { "link": "https://localhost/mgmt/tm/asm/
policies/example_1"},
 "secondPolicyReference": {"link": "https://localhost/mgmt/tm/asm/
policies/example_2"}
}
```

2. Make a POST request to the `/mgmt/tm/asm/tasks/policy-diff` endpoint and include the JSON body you created.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/policy-diff
```

The task returns an endpoint to a collection of differences between the policies, such as an entity that appears in one policy but not in the other.

```
{
 "firstPolicyReference": { "link": "https://localhost/mgmt/tm/asm/
policies/example_1"},
```

```
 "secondPolicyReference": {"link": "https://localhost/mgmt/tm/asm/
policies/example_2"},
 "differenceReference": { "link": "https://localhost/mgmt/tm/asm/policy-
diffs/8AcZwsnx7gvk34CV22hYrw/differences/example?ver=13.1.0"},
 "lastUpdateMicros": 0,
 "id": ""
}
```

You have created a collection of differences between two policies that you can use to merge and resolve the differences.

Merging policy differences in Application Security Manager

You can merge the differences between two policies by using the output of the policy differences task. Once you have a collection of differences, you can specify how to merge the differences into the policies.

To merge the differences between policy files, make a POST request to the `/mgmt/tm/asm/tasks/policy-merge` endpoint.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/policy-merge
```

```
{
  "policyDiffReference": {"link": "/mgmt/tm/asm/policy-diffs/example"},
  "addMissingEntitiesToFirst": true,
  "addMissingEntitiesToSecond": true,
  "handleCommonEntities": "ignore",
  "handleMissingEntitiesEnum": ["ignore", "accept-from-first", "accept-from-
second"],
  "itemFilter": ""
}
```

The `itemFilter` property lets you select a subset of differences to merge. You can think of this as the equivalent of the `$filter` query parameter applied to the collection of differences.

Application Security Manager signatures

If you use Application Security Manager™ (ASM™) to manage signatures, you should understand how ASM differs from iControl REST.

iControl® REST supports the Application Security Manager™ (ASM™) features to check, export, or update signatures.

Property	Description
`file`	Specifies inline imported or exported content in XML format.
`inline`	Indicates whether the exported signatures are contained inline in the response.
`isBase64`	Indicates whether the inline content is Base64 encoded, either input or output. If inline is set to TRUE, the exported signatures are Base64 encoded.
`filename`	Specifies the name of a local signature file.
`isUserDefined`	Indicates whether a signature is considered to be a user-defined signature.

Checking for signatures in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) task to check signatures for updates to the signature files.

1. To check for new signatures, make a POST request to the `/tm/asm/tasks/check-signatures` namespace, and include an empty JSON body (`{ }`).

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/check-signatures
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:check-signatures:check-signaturestaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/check-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To determine the status of the check for new signatures operation, make a GET request.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/check-signatures/
oqNah2PxtwwE4YyAHGekNQ
```

The response to the request contains the following data:

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:check-signatures:check-signaturestaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/check-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "updatesAvailable": false
    }
}
```

Updating signatures in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) task to update signatures.

1. Optional: To upload a file from which to update the signatures, use the POST method and specify the `/tm/asm/file-transfer/uploads` endpoint. You must specify the name of the file in the request.

```
POST https://192.168.25.42/mgmt/tm/asm/file-transfer/uploads/<filename>
```

2. To update signatures, make a POST request to the `/tm/asm/tasks/update-signatures` namespace and include an empty JSON body (`{ }`).

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/update-signatures

{}
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:update-signatures:update-signaturestaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/update-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
```

```
}
```

3. To determine the status of the update signatures operation, make a GET request.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/update-signatures/
oqNah2PxtwwE4YyAHGekNQ
```

The response contains the results of the task.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:update-signatures:update-signaturestaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/update-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "signatureStatusReference": {
            "link": "https://localhost/mgmt/tm/asm/signature_statuses/
vagoQLF6uOoBKvS8h3C19w"
        }
    }
}
```

Exporting signatures in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) task to export signatures for use on another ASM system.

1. To export signatures, make a POST request to the `/tm/asm/tasks/export-signatures` namespace, and specify the name of the output file in the JSON body.

```
POST https://192.168.25.42//mgmt/tm/asm/tasks/export-signatures

{
   "filename": "exported_file.xml",
}
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "filename": "exported_file.xml",
    "kind": "tm:asm:tasks:export-signatures:exportsignaturestaskstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/export-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. Optional: To determine the status of the export signatures operation, make a GET request.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/export-signatures/
oqNah2PxtwwE4YyAHGekNQ
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "filename": "exported_file.xml",
    "kind": "tm:asm:tasks:export-signatures:exportsignaturestaskstate",
```

```
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/export-signatures/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "filename": "exported_file.xml",
    }
}
```

3. Optional: To download the file, use the GET method and specify the `/tm/asm/file-transfer/downloads` endpoint, along with the name of the exported file. You must specify the name of the file in the request.

```
GET https://192.168.25.42/mgmt/tm/asm/file-tranfer/downloads/
exported_file.xml
```

Retrieving signature status information in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) feature to retrieve signature status information for a signature. Signature status includes information regarding additions and deletions to a signature file.

To retrieve signature status information, make a GET request to the `/tm/asm/signature-statuses` namespace.

```
GET https://192.168.25.42/mgmt/tm/asm/signature-statuses/<MD5HASH>
```

The items property shows the signature status.

```
{
    "selfLink": "https://localhost/mgmt/tm/asm/signature-statuses",
    "kind": "tm:asm:signature-statuses:signature-statuscollectionstate",
    "items": [
        {
            "sigsAdded": 0,
            "isUserDefined": false,
            "readme": "Attack Signature Database packaged with version
 11.5.0\n\n\ .... ",
            "sigsUpdatedMinor": 0,
            "sigsDeleted": 0,
            "modifiedSignatures": [],
            "loadTime": "2013-10-10T06:43:30Z",
            "sigsTotal": 0,
            "sigsUpdated": 0,
            "selfLink": "https://localhost/mgmt/tm/asm/signature-statuses/
cHzbviRdfEv6l_RRieAdqw",
            "kind": "tm:asm:signature-statuses:signature-statusstate",
            "timestamp": "2013-10-08T09:06:15Z",
            "sigsUpdatedMajor": 0,
            "id": "cHzbviRdfEv6l_RRieAdqw"
        }
    ]
}
```

Retrieving signature systems in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) feature to retrieve a signature system. You must supply the MD5 hash of a signature system to retrieve.

To retrieve signature system information, make a GET request with the `/tm/asm/signature-systems` namespace.

```
GET https://192.168.25.42/mgmt/tm/asm/signature-systems/MD5HASH
```

The response displays the signature system information, as a link to the resource.

```
{
    "selfLink": "https://localhost/mgmt/tm/asm/signature-systems/
EStDgGiP9nSPgKBhSlDyvQ",
    "kind": "tm:asm:signature-systems:signature-systemstate",
    "name": "General Database",
    "id": "EStDgGiP9nSPgKBhSlDyvQ"
}
```

Application Security Manager schema upload

If you use Application Security Manager™ (ASM™) to manage schemas, you should understand how iControl® REST supports schema upload tasks.

iControl® REST provides an endpoint for XML schema file uploads. Application Security Manager™ (ASM™) validates incoming data by using schema files that you upload and then associate to a policy.

Property	Description
`fileName`	Specifies the name of the XML schema file.
`contents`	Specifies the file contents as XML.

Uploading schema files in Application Security Manager

Associating an XML schema file to a profile necessitates the ability to upload XML schema files. After you upload the schema file, you can run a separate task to associate the validation file to the profile.

To upload the XML schema file, use the POST method and specify a policy within the `/tm/asm/policies` namespace.

```
POST https://192.168.25.42/mgmt/tm/asm/policies/xpqbO1mYOtgfv13j1khKeA/xml-
validation-files

{
            "fileName": "softwareupdate.wsdl",
            "contents": "<validation></validation>"
}
```

```
{
            "selfLink": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA/xml-validation-files/d7loGosItLc_ODXuPz83Uw",
            "kind": "tm:asm:policies:xml-validation-files:xml-validation-
filestate",
            "fileName": "softwareupdate.wsdl",
            "contents": "<begin></begin>",
            "lastUpdateMicros": 1393332020000000,
            "id": "d7loGosItLc_ODXuPz83Uw",
            "isReferenced": false
}
```

Application Security Manager policy restore

If you use Application Security Manager™ (ASM™) to restore policy, you should understand how iControl® REST implements ASM.

iControl® REST supports the Application Security Manager™ (ASM™) feature to restore policy based on policy history. When you restore a policy revision, you must include the `policyHistoryRevision` property in the body of a request, and specify the policy revision from which to restore. If you provide a `policyReference` property or name property in the body of the request, the task overwrites the policy. Otherwise, the task creates a new policy.

Property	Description
`policyHistoryRevision`	Specifies the link of the history revision to restore.

Restoring policy revisions in Application Security Manager

The `policyHistoryReference` property in Application Security Manager™ (ASM™) enables a task to restore a policy revision. The task overwrites the policy if the JSON body contains a `policyReference` or `name` property. Otherwise, the task creates a new policy.

1. To restore a policy revision, use the POST method with the `/tm/asm/task/import-policy` namespace.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/import-policy

{
   "policyHistoryReference": {
      "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/history-revisions/hGKdiXU7US4S4qtgexijUQ"
   },
   "policyReference": {
      "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
   }
}
```

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:import-policy:importpolicytaskstate",
    "policyHistoryReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/history-revisions/hGKdiXU7US4S4qtgexijUQ"
    },
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To check the status of the request, make a GET request with the `/tm/asm/task/import-policy` namespace and append the `id` property from the previous response.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ
```

The response displays the `status` property for the request.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind": "tm:asm:tasks:import-policy:importpolicytaskstate",
    "lastUpdateMicros": 1370459676272126,
```

```
    "policyHistoryReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/history-revisions/hGKdiXU7US4S4qtgexijUQ"
    },
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
    },
    "status": "COMPLETED",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-policy/
oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "policyReference": {
            "link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w"
        }
    }
}
```

Application Security Manager vulnerability import

If you use Application Security Manager™ (ASM™) to import vulnerability data, you should understand how iControl® REST implements ASM.

iControl® REST supports the Application Security Manager™ (ASM™) feature to import vulnerabilities from a file, or to download vulnerabilities from a scanner. You must include the `policyReference` property in the JSON body.

Property	Description
`policyReference`	Describes the path to the current policy, as a link.
`file`	Specifies the file contents, in XML format.
`filename`	Specifies the name of the file to read.
`isBase64`	Indicates whether the file consists of Base64-encoded data.
`scanId`	Specifies a scan ID. Required for Cenzic Hailstorm if you do not specify a `file` property.
`subscriptionId`	Specifies a subscription ID. Required for Cenzic Hailstorm if you do not specify a `file` property.
`onlyGetDomainNames`	Indicates whether the task parses the input file and then generates a count of all vulnerabilities without importing the vulnerabilities.
`importAllDomainNames`	Indicates whether the task parses the input file and imports all vulnerabilities.
`domainNames`	Specifies the domain names for which the task parses the input file and imports all vulnerabilities.

Importing vulnerabilities in Application Security Manager

iControl® REST supports the Application Security Manager™ (ASM™) feature to import vulnerability data from sources, such as files or scanners.

1. To import vulnerabilities, use the POST method with the `/tm/asm/tasks/import-vulnerabilities` namespace.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/import-vulnerabilities

{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
    "importAllDomainNames": false,
    "domainNames": [
        ""
    ],
    "subscriptionId": "4132",
    "scanId": "3883"
}
```

```
{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
    "isBase64": false,
    "importAllDomainNames": false,
    "status": "NEW",
    "lastUpdateMicros": 1395567859000000,
    "domainNames": [
        ""
    ],
    "subscriptionId": "4132",
    "scanId": "3883",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-
vulnerabilities/8PacFCQc0Umx45mheqdyew",
    "kind": "tm:asm:tasks:import-vulnerabilities:import-vulnerabilities-
taskstate",
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {}
}
```

2. To retrieve the status of the import vulnerability task, use the GET method.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/import-
vulnerabilities/8PacFCQc0Umx45mheqdyew
```

The response to the request contains the following data:

```
{
    "isBase64": false,
    "importAllDomainNames": false,
    "status": "COMPLETED",
    "lastUpdateMicros": 1395567859000000,
    "domainNames": [
        ""
    ],
    "onlyGetDomainNames": false,
    "subscriptionId": "4132",
    "scanId": "3883",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/import-
vulnerabilities/8PacFCQc0Umx45mheqdyew",
    "kind": "tm:asm:tasks:import-vulnerabilities:import-vulnerabilities-
taskstate",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA"
```

```
    },
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {
        "vulnerableHosts": [
            {
                "vulnerabilityCount": "4",
                "domainName": ""
            },
            {
                "vulnerabilityCount": "41",
                "domainName": "crackme.cenzic.com"
            }
        ]
    }
}
```

Querying vulnerability assessment subscriptions in Application Security Manager

Application Security Manager™ (ASM™) supports subscriptions to third-party scanners. You can query ASM for active vulnerability assessment subscriptions.

Note: ASM only supports subscriptions to Cenzic Hailstorm.

1. To determine the active vulnerability assessment subscriptions, use the POST method with the `/tm/asm/tasks/get-vulnerability-assessment-subscriptions` namespace and specify the `policyReference` property in the JSON body.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/get-vulnerability-assessment-
subscriptions

{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" }
}
```

   The response shows the request `status` property that indicates a new request and the `id` property that identifies the request for other operations.

```
{
    "kind": "tm:asm:tasks:get-vulnerability-assessment-subscriptions:get-
vulnerability-assessment-subscriptions-taskstate",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/get-vulnerability-
assessment-subscriptions/pCOSkFyRGWeAf6Kwcpj38w",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA"
    },
    "status": "New",
    "id": "pCOSkFyRGWeAf6Kwcpj38w",
    "startTime": "2014-03-24T09:35:57Z",
    "lastUpdateMicros": 1395653765000000,
    "result": { }
}
```

2. To obtain the output of this request, use the GET method with the `/tm/asm/tasks/get-vulnerability-assessment-subscriptions` namespace and append the `id` property to the URI.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/get-vulnerability-assessment-
subscriptions/pCOSkFyRGWeAf6Kwcpj38w
```

```
{
```

```
    "kind": "tm:asm:tasks:get-vulnerability-assessment-subscriptions:get-
vulnerability-assessment-subscriptions-taskstate",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/get-vulnerability-
assessment-subscriptions/pCOSkFyRGWeAf6Kwcpj38w",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA"
    },
    "status": "COMPLETED",
    "id": "pCOSkFyRGWeAf6Kwcpj38w",
    "startTime": "2014-03-24T09:35:57Z",
    "lastUpdateMicros": 1395653765000000,
    "result": {
        "subscriptions": [
            {
                "scans": [
                    {
                        "scanId": "3870",
                        "completeDateTime": "2013-04-03T08:33:27Z",
                        "status": "Complete"
                    },
                    {
                        "scanId": "3883",
                        "completeDateTime": "2013-04-09T08:55:50Z",
                        "status": "Complete"
                    }
                ],
                "url": "http://crackme.cenzic.com/Kelev/register/
register.php",
                "productId": "F5 Trial Scan",
                "subscriptionId": "4132"
            }
        ]
    }

}
```

Initiating vulnerability assessment in Application Security Manager

Vulnerability assessments provide access to third-party scanners, such as Cenzic Hailstorm. The `asm/tasks` namespace includes an endpoint to initiate a scan.

1. To initiate a vulnerability assessment, make a POST request with the `/tm/asm/tasks/initiate-vulnerability-assessment` namespace. Include the `policyReference` and `subscriptionId` properties in the JSON body.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/initiate-vulnerability-
assessment

{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
    "subscriptionId": "4132"
}
```

   The response shows the `status` and `id` properties of the request.

```
{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
    "status": "NEW",
    "lastUpdateMicros": 1395567859000000,
    "subscriptionId": "4132",
```

```
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/initiate-
vulnerability-assessment/8PacFCQc0Umx45mheqdyew",
    "kind": "tm:asm:tasks:initiate-vulnerability-assessment:initiate-
vulnerability-assessment-taskstate",
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {}
}
```

2. To retrieve the status of the initiate vulnerability assessment operation, use the GET method with the `/tm/asm/tasks/initiate-vulnerability-assessment` namespace and append the `id` property to the URI.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/initiate-vulnerability-
assessment/8PacFCQc0Umx45mheqdyew
```

The response shows the request `status` and `scanId` properties.

```
{
    "status": "COMPLETED",
    "lastUpdateMicros": 1395567859000000,
    "subscriptionId": "4132",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/initiate-
vulnerability-assessment/8PacFCQc0Umx45mheqdyew",
    "kind": "tm:asm:tasks:initiate-vulnerability-assessment:initiate-
vulnerability-assessment-taskstate",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA"
    },
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {
         "scanId": 4920
    }
}
```

Terminating vulnerability assessment in Application Security Manager

Vulnerability assessments provide access to third-party scanners, such as Cenzic Hailstorm. The `asm/tasks` namespace includes an endpoint to terminate a scan.

1. To terminate a vulnerability assessment, make a POST request with the `/tm/asm/tasks/terminate-vulnerability-assessment` namespace. Include a JSON body with the `policyReference` property.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/terminate-vulnerability-
assessment

{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
}
```

The response to the request includes the `id` that identifies the request for a query.

```
{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA" },
    "status": "NEW",
    "lastUpdateMicros": 1395567859000000,
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/terminate-
vulnerability-assessment/8PacFCQc0Umx45mheqdyew",
```

```
    "kind": "tm:asm:tasks:terminate-vulnerability-assessment:terminate-
vulnerability-assessment-taskstate",
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {}
}
```

2. To retrieve the status of the terminate vulnerability assessment operation, use the GET method with the `/tm/asm/tasks/terminate-vulnerability-assessment` namespace and append the `id` property to the URI.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/terminate-vulnerability-
assessment/8PacFCQc0Umx45mheqdyew
```

The response show the status of request.

```
{
    "status": "COMPLETED",
    "lastUpdateMicros": 1395567859000000,
    "subscriptionId": "4132",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/terminate-
vulnerability-assessment/8PacFCQc0Umx45mheqdyew",
    "kind": "tm:asm:tasks:terminate-vulnerability-assessment:terminate-
vulnerability-assessment-taskstate",
    "policyReference": {
        "link": "https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA"
    },
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {
    }
}
```

Application Security Manager vulnerability resolution

If you use Application Security Manager™ (ASM™) for vulnerability resolution, you should understand how iControl® REST implements ASM.

Application Security Manager™ (ASM™) supports options to resolve vulnerabilities, such as staging the suggested changes for a vulnerability.

Property	Description
`getPreResolveMessages`	Indicates that the task only displays the proposed changes for each vulnerability but does not implement the change.
`stageVulnerabilities`	Indicates that the changes made to a policy should be staged.
`vulnerabilities`	Specifies the reference to a vulnerability, as a collection of references.

Resolving vulnerabilities in Application Security Manager

When you resolve vulnerabilities, Application Security Manager™ (ASM™) configures the security policy to protect a web application against a vulnerability. If you choose, you can stage a vulnerability to allow more time to test the security policy. Otherwise, ASM applies the changes to the security policy immediately.

1. To resolve the vulnerabilities, use the POST method with the `/tm/asm/tasks/resolve-vulnerabilities` namespace, and specify the `vulnerabilities` property.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/resolve-vulnerabilities

{
"vulnerabilities": [
  {"link": "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/
vulnerabilities/abcdef1234567890"},
  {"link": "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/
vulnerabilities/qwertytrewqa1234"}
 ]
}
```

   The response includes the request `status` and `id` properties.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "vulnerabilities": [
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/abcdef1234567890"},
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/qwertytrewqa1234"}
     ],
       "kind": "tm:asm:tasks:resolve-
vulnerabilities:resolvevulnerabilitiesstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/resolve-
vulnerabilities/oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To determine the status of this operation, use the GET method with the `/tm/asm/tasks/resolve-vulnerabilities` namespace and append the `id` property to the URI.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/resolve-vulnerabilities/
oqNah2PxtwwE4YyAHGekNQ
```

   The response displays the `result` property.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "vulnerabilities": [
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/abcdef1234567890"},
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/qwertytrewqa1234"}
     ],
    "kind": "tm:asm:tasks:resolve-
vulnerabilities:resolvevulnerabilitiesstate",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/resolve-
vulnerabilities/oqNah2PxtwwE4YyAHGekNQ",
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
                  "message": "The system does not automatically mitigate
 the detection of an SQL injection vulnerability created as a result of a
 scanner payload that includes distractive meta characters.\nIn order to
 mitigate this vulnerability, manually add the disallowed meta characters
```

```
 to the vulnerable parameter.\nNote: Characters such as '\"< when injected
 may change the SQL query."
              }
}
```

Identifying vulnerabilities in Application Security Manager

iControl®REST supports the Application Security Manager™ (ASM™) task to resolve a vulnerability and obtain the messages that identify a vulnerability, without making changes to the security policy.

1. To retrieve the pre-resolve messages, use the POST method with the `/tm/asm/tasks/resolve-vulnerabilities` namespace, and specify the `vulnerabilities` and `getPreResolveMessages` properties.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/resolve-vulnerabilities

{
"vulnerabilities": [
  {"link": "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/
vulnerabilities/abcdef1234567890"},
  {"link": "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/
vulnerabilities/qwertytrewqa1234"}
 ],
 "getPreResolveMessages": true
}
```

   The response shows the request `status` and `id` properties.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "vulnerabilities": [
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/abcdef1234567890"},
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/qwertytrewqa1234"}
     ],
    "getPreResolveMessages": true
       "kind": "tm:asm:tasks:resolve-
vulnerabilities:resolvevulnerabilitiesstate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/resolve-
vulnerabilities/oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To determine the status of this operation, use the GET method with the `/tm/asm/tasks/resolve-vulnerabilities` namespace and append the `id` property to the URI.

```
GET https://192.168.25.42/mgmt/tm/asm/tasks/resolve-vulnerabilities/
oqNah2PxtwwE4YyAHGekNQ
```

   The response includes the `result` property and the text message data.

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "vulnerabilities": [
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/abcdef1234567890"},
        {"link": "https://localhost/mgmt/tm/asm/policies/
vagoQLF6uOoBKvS8h3C19w/vulnerabilities/qwertytrewqa1234"}
     ],
```

```
    "getPreResolveMessages": true
    "kind": "tm:asm:tasks:resolve-
vulnerabilities:resolvevulnerabilitiesstate",
    "selfLink": "https://localhost/mgmt/tm/asm/tasks/resolve-
vulnerabilities/oqNah2PxtwwE4YyAHGekNQ",
    "lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
                "message": "The following attack signature sets will be
 assigned to the security policy: Cross Site Scripting Signatures, SQL
 Injection Signatures\nStaging will be disabled for all signatures of
 Signature Set: Cross Site Scripting Signatures, SQL Injection Signatures"
              }
}
```

Exporting data protection in Application Security Manager

You can utilize the same cookie encryption seed with BIG-IP® systems that do not belong to a centralized management infrastructure (CMI) device group that is enabled for ASM®.

To export data protection, make a POST request to the `/mgmt/tm/asm/tasks/export-data-protection` endpoint.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/export-data-protection
```

```
{
  "filename": "example-export-keys"
}
```

```
{
  "endTime": "1970-01-01T00:00:00Z",
  "filename": "",
   "id": "",
  "lastUpdateMicros": 0,
  "result": {
    "message": "",
    "file": ""
  },
  "startTime": "1970-01-01T00:00:00Z",
  "status": "NEW",
  "statusEnums": [
    "NEW",
    "FAILURE",
    "COMPLETED",
    "STARTED"
  ]
}
```

Importing data protection in Application Security Manager

Following on the data protection export task, you can import the data protection keys on another BIG-IP® system in another data center to share traffic. By sharing the keys, you can share traffic for the same applications between data centers.

To import data protection, make a POST request to the `/mgmt/tm/asm/tasks/import-data-protection` endpoint.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/import-data-protection
```

```
{
  "filename": "example-export-keys"
}
```

```
{
  "endTime": "1970-01-01T00:00:00Z",
  "file": "",
  "filename": "",
  "graceAcceptingIntervalInMinutes": 2880,
  "graceSigningIntervalInMinutes": 30,
  "id": "",
  "lastUpdateMicros": 0,
  "result": {
    "DataProtectionReference": {
      "link": "https://localhost/mgmt/tm/asm/data-protection?ver=13.0.0"
    },
    "message": ""
  },
  "startTime": "1970-01-01T00:00:00Z",
  "status": "NEW",
  "statusEnums": [
    "NEW",
    "FAILURE",
    "COMPLETED",
    "STARTED"
  ]
}
```

Importing a certificate in Application Security Manager

Application Security Manager™ (ASM®) supports the task of importing SSL certificates. Once a certificate is imported, the certificate can be used for web services security (WSS) protection of an XML profile.

To import a certificate, make a POST request to the `/mgmt/tm/asm/tasks/import-certificate` endpoint.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/import-certificate
```

```
{
  "certificateName": "",
  "certificateType": "client",
  "certificateTypeEnums": ["client", "server"],
  "file": "",
  "filename": "",
  "saveExpiredOrUntrustedCertificate": false,
  "certificateReference": {"link": "https://localhost/mgmt/tm/asm/
certificates/example?ver=13.1.0"},
  ... [Other standard task fields]
}
```

The property `saveExpiredOrUntrustedCertificate` is a Boolean value that allows you to import untrusted or expired certificates.

Web Scraping Configuration settings

If you use Application Security Manager™ (ASM™) to manage web scraping configuration settings, you can use an iControl® REST API to retrieve or modify those settings.

iControl REST exposes properties to configure Session Transactions Anomaly settings. The values described here conform to the settings you are familiar with if you configure web scraping settings in the Traffic Management UI (TMUI). The string `webScrapingConfiguration` identifies the top-level member of this resource object.

Property	Description
`sessionTransactionAnomalyBlock`	Indicates whether the system blocks on a session transaction anomaly, as a Boolean.
`sessionOpeningAnomalyAlarm`	Indicates whether the system sends an alarm on a session opening anomaly, as a Boolean.
`suspiciousClientsAlarm`	Indicates whether the system sends an alarm on a suspicious client, as a Boolean.
`sessionTransactionAnomalyAlarm`	Indicates whether the system sends an alarm on a session transaction anomaly, as a Boolean.
`suspiciousClientsBlock`	Indicates whether the blocks on a suspicious client, as a Boolean.
`sessionOpeningAnomalyBlock`	Indicates whether the system blocks on a session opening anomaly, as a Boolean.
`usePersistentStorage`	Indicates whether the system uses persistent storage for client identification data, as a Boolean.
`botDetectionBlock`	Indicates whether the system blocks on bot detection, as a Boolean.
`useFingerprint`	Indicates whether the system uses fingerprinting to collect browser attributes, as a Boolean.
`botDetectionAlarm`	Indicates whether the system sends an alarm on bot detection, as a Boolean.

Session Transactions Anomaly settings

If you use Application Security Manager™ (ASM™) to manage web scraping configuration settings, you can use an iControl® REST API to retrieve and modify those settings.

iControl REST exposes properties to configure Session Transactions Anomaly settings. The values described in this topic conform to the settings you are familiar with if you configure web scraping settings in the Traffic Management UI (TMUI). The string `sessionTransactionsAnomaly` identifies the top-level member of this resource object.

Property	Description
`maximumSessionTransactionsPerSecond`	Specifies that the system considers traffic to be an attack if the number of transactions per session is equal to or greater than this number. The default value is 400.

Property	Description
`minimumSessionsTransactionsPerSecond`	Specifies that the system considers traffic to be an attack if the number of transactions per session is equal to or greater than this number, and either the session transactions increased by value or session transactions reached value was reached. If the number of transactions per session is less than this value, the system does not consider the traffic to be an attack, even if one of the session transactions increased by value or session transactions reached value was reached. The default value is 200.
`preventionDuration`	Specifies the length of time, in seconds, that the system prevents a session anomaly attack after the system detects and stops an attack, unless the system detects the end of the attack earlier. The system prevents attacks by blocking requests. The default value is 1800.
`sessionTransactionsPerSecondIncreasedBy`	Specifies that the system considers traffic to be an attack if the number of transactions in the session is greater than normal by this percent. Normal is defined as the average number of transactions per session for the whole site during the last hour. The default value is 500.

Bot Detection settings

If you use Application Security Manager™ (ASM™) to manage web scraping configuration settings, you can use an iControl® REST API to retrieve and modify those settings.

iControl® REST exposes properties to configure Bot Detection settings. The values described here conform to the settings you are familiar with if you configure web scraping settings in the Traffic Management UI (TMUI). The string `botDetection` identifies the top-level member of this resource object.

Property	Description
`rapidSurfingMaximumDistinctPages`	Specifies the maximum number of times that one page can be refreshed within a specified amount of time before the system considers the client source to be a bot. The default value is 120.
`rapidSurfingMaximumChangedPages`	Specifies the number of different pages that can be loaded within a specified amount of time before the system considers the client source to be a bot.
`checkEventsSequenceEnforcement`	Indicates whether the system performs event sequence enforcement. Configuring this setting protects your web application against bots by tracking the sequence of events that the browser triggers to detect irregular sequences. When an irregular sequence is detected, in order to prevent false positives, the client is not immediately marked as a bot. Instead, the client is prevented from being marked as human until the next web page is loaded.
`rapidSurfingMaximumTimeDuration`	Specifies the maximum amount of time that it takes either to refresh one web page, or to refresh a minimum number of pages once, in order for the system to suspect that a bot requested the page. The default value is 30.

Property	Description
`blockingPeriod`	Specifies the number of requests that the system considers unsafe, thus blocking them if the security policy is in blocking mode. The system did not detect a valid client during the grace interval, and automatically generates the Web Scraping Detected violation. In addition, the system no longer checks these requests for web scraping. After the client sends the number of requests specified in this setting, the system reactivates the grace interval. The default value is 500.
`graceThreshold`	Specifies the maximum number of requests the system reviews while trying to detect whether the client is human. As soon as the system makes that determination, it stops checking the requests. The default value is 100. Once the client determines that the client is valid, the system allows, and does not check, the next several requests, as specified by the safe interval setting. If the system does not detect a valid client during the grace interval, the system issues, and continues to issue, the Web Scraping Detected violation until it reaches the number of requests specified in the blocking period setting.
`safeIntervalThreshold`	Specifies the number of requests that the system considers safe. The system determined that these requests are sent by a human-backed client and therefore no longer checks these requests for web scraping. Once the number of requests sent by the client reaches the value specified in the setting, the system reactivates the grace interval. The default value is 2000.

Session Opening Anomaly settings

If you use Application Security Manager™ (ASM™) to manage web scraping settings, you can use an iControl® REST API to retrieve and modify those settings.

iControl® REST exposes properties to configure session opening anomaly settings. The properties described here conform to the settings you are familiar with if you configure session opening anomaly settings in the Traffic Management UI (TMUI). The string `sessionOpeningAnomaly` identifies the top-level member of this resource object.

Property	Description
`minimumSessionsOpenedPerSecond`	Specifies that the system considers traffic to be an attack if the number of sessions opened per second is equal to, or greater than, this number, and at least one of the sessions opened per second increased by or sessions opened per second reached numbers was reached. If the number of sessions opened per second is lower than this number, the system does not consider this traffic to be an attack even if one of the sessions opened per second increased by or sessions opened per second reached was reached. The default value is 25.
`checkSessionOpeningAnomaly`	Indicates whether the system detects session opening anomalies by IP address, as a Boolean value.
`clientSideIntegrityDefense`	Indicates whether the system determines if a client is a legal browser or an illegal script by sending a JavaScript challenge to each new session request from the detected IP address, and waiting for a response. The default value is false.

Property	Description
`rateLimiting`	Indicates whether the system drops sessions from suspicious IP addresses after the system determines that the client is an illegal script. The default value is false.
`maximumSessionsOpenedPerSecond`	Specifies that the system considers traffic to be an attack if the number of sessions opened per second is equal to, or greater than, this number. The default value is 50.
`dropIpAddressesWithBadReputation`	Indicates whether the system drops requests from IP addresses that have a bad reputation according to the system's IP address reputation database. Attacking IP addresses that do not have a bad reputation undergo rate limiting, as usual. The default value is disabled.
`sessionsOpenedPerSecondIncreaseRate`	Specifies that the system considers traffic to be an attack if the number of sessions opened per second increased by this number. The default value is 500.
`preventionDuration`	Specifies the length of time, in seconds, that the system prevents a session opening anomaly attack after the system detects and stops an attack, unless the system detects the end of the attack earlier. The default value is 1800.

Session Opening Threshold settings

If you use Application Security Manager™ (ASM™) to manage web scraping settings, you can use an iControl® REST API to retrieve and modify those settings.

iControl REST exposes properties to configure session opening threshold settings. The values described here conform to the settings you are familiar with if you configure session opening threshold settings in the Traffic Management UI (TMUI). The string `sessionOpeningThresholds` identifies the top-level member of the resource object.

Property	Description
`checkFingerprintResets`	Indicates whether the system uses fingerprinting to detect cookie deletion events. Fingerprinting assumes that each browser has a unique fingerprint, and therefore the system collects browser attributes to identify browsers and bots. The default value is false.
`openingPersistentStorageResetsDuration`	Specifies the length of time, in seconds, that the system has to detect a specified number of cookie deletion events before the system determines a request to be a web scraping attack and blocks the suspected illegal request.
`openingPersistentStorageInconsistencyEventsMaximum`	Specifies the maximum number of integrity fault events the system must detect to determine a web scraping attack. The default value is 3.
`preventionDuration`	Specifies the length of time, in seconds, that the system prevents a session opening threshold attack after the system detects and stops an attack, unless the system detects the end of the attack earlier. The system prevents attacks by rejecting requests from the attacking clients. The system identifies the attacking client based on a unique identification number that was stored in the attacking browser's persistent storage. The default value is 1800.

Property	Description
`checkStorageInconsistency`	Indicates whether the system blocks requests that it identifies as integrity fault events. The default value is false.
`checkStorageResets`	Indicates whether the system uses persistent device identification to detect cookie deletion events, The default value is false.
`openingPersistentStorageResetsMaximum`	Specifies the number of cookie deletion events that the system must detect in a specified time period before the system determines an attack to be a web scraping attack and blocks the suspected illegal request.
`fingerprintResetsTimeWindow`	Specifies the length of time, in seconds, that the system has to detect a specified number of cookie deletion events before the system determines a request to be a web scraping attack and blocks the suspected illegal request. The default value is 600.
`openingPersistentStorageInconsistencyEventsDuration`	Specifies the length of time, in seconds, that the system has to detect integrity fault events before the system determines an attack to be a web scraping attack. The default value is 600.
`fingerprintResetsThreshold`	Specifies the number of cookie deletion events the system must detect in a specified time period before the system determines that a request is a web scraping attack and blocks the suspected illegal request.

Suspicious Client settings

If you use Application Security Manager™ (ASM™) to manage web scraping settings, you can use an iControl® REST API to retrieve and modify those settings.

iControl REST exposes properties to configure suspicious client settings. The values described here conform to the settings you are familiar with if you configure suspicious client settings in the Traffic Management UI (TMUI). The string `suspiciousClients` identifies the top-level member of the resource object.

Property	Description
`detectBrowsersWithScrapingExtensions`	Indicates whether the system investigates browsers for web scraping plug-ins to determine if a client should be considered suspicious. The default value is false.
`scrapingExtensions`	Specifies an array of web scraping extensions that are considered illegal. If the system determines that a client is suspicious, it logs and blocks requests from this client.
`preventionDuration`	Specifies the length of time, in seconds, that the system prevents requests from a client after the system determines the client to be suspicious. The default value is 300.

iControl REST Web Scraping Settings

iControl® REST supports the programmability of Application Security Manager™ (ASM™) web scraping settings. The iControl REST interface provides a single endpoint that supports both query and modification requests. As a singleton resource, the web scraping resource supports a GET request to retrieve the current web scraping settings, and a PATCH request to modify resource properties. The PATCH method allows a partial representation of a resource as the request entity, which means that you only need to specify the properties you want to change and not the entire resource.

Retrieving web scraping settings

iControl ®REST supports Application Security Manager™ (ASM™) functionality by allowing retrieval of web scraping settings for a BIG-IP® system. You can automate the retrieval of settings from multiple BIG-IP systems by using the iControl REST API.

To retrieve the web scraping settings, make a GET request to the `/tm/asm/policies/<MDHASH>/web-scraping` endpoint.

```
GET https://192.168.25.42/mgmt/tm/asm/policies/<MDHASH>/web-scraping
```

iControl REST retrieves the web scraping settings for all traffic patterns.


```
{
   "suspiciousClients":{
      "detectBrowsersWithScrapingExtensions":false,
      "preventionDuration": 300,
      "scrapingExtensions":[]
   },
   "sessionOpeningThresholds":{
      "checkFingerprintResets":true,
      "checkStorageInconsistency":true,
      "checkStorageResets":true,
      "openingPersistentStorageResetsDuration": 707,
      "openingPersistentStorageResetsMaximum": 77,
      "fingerprintResetsTimeWindow": 607,
      "openingPersistentStorageInconsistencyEventsMaximum": 7,
      "persistentStorageMaxPreventionDuration": 1807,
      "openingPersistentStorageInconsistencyEventsDuration": 677,
      "fingerprintResetsThreshold": 17
   },
   "sessionOpeningAnomaly":{
      "minimumSessionsOpenedPerSeconds": 22,
      "checkSessionOpeningAnomaly":true,
      "PreventionDuration": 1802,
      "clientSideIntegrityDefense":true,
      "rateLimiting":true,
      "maximumSessionsOpenedPerSeconds": 52,
      "dropIpAddressesWithBadIpReputation":true,
      "sessionsOpenedPerSecondsIncressRate": 502
   },
   "botDetection":{
      "rapidSurfingMaximumDistinctPages": 301,
      "rapidSurfingMaximumChangedPages": 1201,
      "checkEventSequenceEnforcement":true,
      "rapidSurfingMaximumTimeDuration": 311,
      "unsafeIntervalTreshold": 10011,
      "graceTreshold": 1001,
      "safeIntervalTreshold": 20001
   },
   "sessionTransactionsAnomaly":{
      "maximumSessionTransactionsPerSecond": 403,
      "minimumSessionTransactionsPerSecond": 203,
      "maximumTransactionPreventionDuration": 1803,
      "sessionTransactionsPerSecondIncreaseRate": 503
   },
   "webScrapingConfiguration":{
      "alarmOnBotDetection":true,
      "blockOnSessionTransactionAnomaly":false,
      "alarmOnSessionOpeningAnomaly":true,
      "alarmOnSuspiciousClients":true,
      "alarmOnSessionTransactionAnomaly":true,
      "blockOnBotDetection":false,
      "blockOnSessionOpeningAnomaly":false,
      "usePersistentStorage":true,
```

```
      "useFingerprint":true,
      "blockOnSuspiciousClients":true,
      "persistentDataValidityPeriod": 126
   },
   "selfLink":"https://localhost/mgmt/tm/asm/policies/
xpqbO1mYOtgfv13j1khKeA/web-scraping?ver=12.0.0",
   "kind":"tm:asm:policies:web-scraping-settings:web-scraping-settingsstate"
}
```

Modifying web scraping settings

iControl® REST supports Application Security Manager™ (ASM™) functionality by enabling modifications to web scraping settings for a BIG-IP® system. You can automate the modification of settings from multiple BIG-IP systems by using the iControl REST API.

To modify the prevention duration property for session transaction anomalies, make a PATCH request to the `/tm/asm/policies/<MDHASH>/web-scraping` endpoint. Specify the top-level member for the traffic pattern you want to modify and the desired property change in the JSON body. To change more than one setting for session transaction anomalies, specify multiple properties in the resource object, separated by commas.

```
PATCH https://192.168.25.42/mgmt/tm/asm/policies/<MDHASH>/web-scraping
{
   "sessionTransactionsAnomaly": { "preventionDuration": 2400 }
}
```

The JSON body must include at least one top-level member that identifies a traffic pattern, such as `sessionTransactionsAnomaly`.

Learning Suggestion Object

If you use Policy Builder functionality in Application Security Manager™ (ASM™), the properties in this table appear as they would in a JSON body, in response to a GET request.

Property	Description
`id`	Specifies a unique identifier for a reference.
`creationDatetime`	Specifies the creation time for a suggestion, as a date-time value.
`lastOccurrenceDatetime`	Specifies the last time a matching request for a suggestion occurred.
`status`	Specifies the learning status of a suggestion. Possible values are: `pending`, `ignored`, `accept`, or `accept-and-stage`. The `accept-and-stage` status enables the staging flag for the target entity, if applicable, and implements the changes specified in the `entityChanges` field.
`alwaysManual`	Indicates that a suggestion will be learned manually, as a Boolean value. If `true`, a suggestion will never be learned automatically.
`comment`	Specifies the user's notes on the suggestion.
`isRead`	Indicates that a suggestion has been read by a user, as a Boolean.
`score`	Specifies an index based on R2 or R3 measurement that reflects the strength of a suggestion.

Property	Description
`triggerType`	Specifies the reason for the suggestion. Possible values are: `violation-mitigation` or `policy-refinement`.
`violationReference`	Specifies the type of violation that triggered a suggestion, if the trigger type is `violation`. This attribute is not required.
`entityChanges`	Specifies the changes to apply to an entity or entityReference if you accept the suggestion.
`entityKind`	Specifies the type of element for a suggestion. This attribute is not required.
`entityName`	Specifies the name of an item instance. This attribute is not required.
`action`	Specifies the suggested operation for an item. Possible values are: `delete`, `add-or-update`, `update-append`, or `update-remove`. This value is not required.
`entity`, `entityReference`	Specifies a reference to a policy if the entity exists in the policy; otherwise, specifies the details of the entity to create.
`parentEntityReference`	Specifies a reference to a parent policy entity that matched a wild card value.
`occurrenceCount`	Specifies the number of requests that triggered a suggestion.
`trustedIpCount`	Specifies the number of distinct client IP addresses, on a list of trusted clients, that triggered a suggestion.
`untrustedIpCount`	Specifies the number of distinct client IP addresses, not on a list of trusted clients, that triggered a suggestion.
`trustedSessionCount`	Specifies the number of distinct client sessions, from trusted IP addresses, that triggered a suggestion.
`untrustedSessionCount`	Specifies the number of distinct client sessions, not from trusted IP addresses, that triggered a suggestion.
`sampleRequests`	Specifies a collection of representative requests, from various IP addresses and sessions, that triggered a suggestion.
`description`	Specifies a description of the changes to implement.
`refinement`, `refinementReference`	Specifies a reason for a suggestion for when a type is `policy-refinement` rather than `violation-mitigation`.
`signatureReference`	Specifies a reference to an attack signature,either as an override on another object, or to effect a change on the signature itself, such as disabling the signature.
`metachar`	Specifies a reference to a metachar, either as an override on another object, or to modify the metachar itself, such as allowing the character itself.
`averageViolationRating`	Specifies the average violation rating for a suggestion, if applicable.
`violationRatingCounts`	Specifies the number of violation ratings for each request.

About using Policy Builder in iControl REST

Application Security Manager™ (ASM™) security policies undergo modification through a framework called unified learning and policy building. *Unified learning and policy building* supports both manual and automatic updates to a security policy. As an administrator, you can retrieve the policy builder suggestions and modify the policy suggestions using the iControl® REST API. Operations you can perform include ordering suggestions by scores or types, viewing more details about a suggestion, or viewing details about related suggestions. iControl REST supports three methods on the `/suggestions` endpoint: GET, DELETE, and PATCH. Other than GET requests to view the collection of suggestions, you will probably have cause to modify the individual suggestions to change the status of a suggestion, add a comment, or mark a suggestion as read. You can use the HTTP PATCH method to modify the `status`, `comment`, or `isRead` properties. As an aside, if you modify properties other than those mentioned, iControl REST ignores those properties in a request. Refer to the Learning Suggestion Object topic for descriptions of the policy builder object.

For more information about policy builder, see the BIG-IP® Application Security Manager (ASM) 12.0 documentation.

Retrieving Policy Builder suggestions

You can retrieve the suggestions for an Application Security Manager™ (ASM™) policy by making a GET request. By default, ASM retrieves the first 500 entities.

To retrieve the suggestions for an ASM policy, make a GET request to the `/suggestions` endpoint for a specific ASM policy.

```
GET https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/suggestions
```

The string `abcd1234` in the example represents a hypothetical MD5HASH value for a policy. An actual MD5 hash value would resemble the following string: `d57fb462a2364e494ed824d523acbfcd`.

The response includes the suggestions for the policy, up to 1000 entities.

```
{
   "selfLink":"https://localhost/mgmt/tm/asm/policies/abcd1234/suggestions",
   "kind":"tm:asm:policies:suggestions:suggestioncollectionstate",
   "items":[
      {
         "id":"123456",
         "selfLink":"https://localhost/mgmt/tm/asm/policies/abcd1234/
suggestions/123456"
         "kind":"tm:asm:policies:suggestions:suggestionstate",
         "creationDatetime":"2013-11-21T22:01:21Z",
         "lastOccurrenceDatetime":"2013-12-10T21:01:21Z",
         "status":"active",
         "alwaysManual":false,
         "comment":"",
         "isRead":false,
         "score":76,
         "occurrenceCount":378,
         "trustedClientIpCount":0,
         "trustedSessionCount":0,
         "untrustedClientIpCount":4,
         "untrustedSessionCount":3,
         "triggerType":"violation",
         "violationReference":{
            "link":"https://localhost/mgmt/tm/asm/violations/
ufg0smEkZrpmkoDHfSPGdQ"
         },
         "parentEntityReference":{
            "link":"https://localhost/......."
         }         "entityReference":{
            "link":"https://localhost/......."
```

```
         },

            "entity":{
            "kind":"tm:asm:policies:urls:parameterstate",
            "name":"foo",
            "level":"url",
            "url":{
               "name":"/foo.php",
               "protocol":"http",

            }
         },
         "entityChanges":{
            "signatureOverrides":[
               {
                  "signatureReference":{
                     "link":"https://localhost/mgmt/tm/asm/signatures/
N64gk_aRPRtaPA4Mt50_LQ"
                  },
                  "enabled":false,

               },

            },
            "requestReferences":[
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123000"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123001"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123002"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123003"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123004"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123005"
               },
               {
                  "link":"https://localhost/mgmt/tm/asm/events/
requests/123006"
               },

            ],

         }
      ]
   }
```

Modifying Policy Builder suggestions

You can modify a suggestion for an Application Security Manager™ (ASM™) policy by making a PATCH request. ASM limits the policy builder properties that you can change.

To modify a suggestion for an ASM policy, make a PATCH request to the `/suggestions/<id>` endpoint for a specific ASM policy. This example changes the status to `ignored`.

```
PATCH https://192.168.25.42/mgmt/tm/asm/policies/<MD5HASH>/
suggestions/465768
```

```
{
   "status": "ignored"
}
```

An MD5 hash is a one-way cryptographic hash function. An actual MD5 hash value would resemble the following string: `d57fb462a2364e494ed824d523acbfcd`.

About Device ID

A device identifier (Device ID) consists of an opaque string that identifies a client application. The stated purpose of the string is only to identify a client application to a virtual server. Application Security Manager™ (ASM™) features that take advantage of Device ID include brute force login, session awareness, and session hijacking prevention.

Device identification using fingerprinting

If you use Application Security Manager™ (ASM™) to manage device ID settings, you can use an iControl® REST API to retrieve and modify those settings.

The Application Security Manager™(ASM™) supports device identification using fingerprinting and exposes the configuration attributes listed in the table. Use the `/mgmt/tm/security/dos/profile/application` endpoint.

Attribute	Description
deviceIdClientSideDefense	Indicates whether to mitigate based on device ID with CS challenge.
deviceIdCaptchaChallenge	Indicates whether to mitigate based on device ID with CAPTCHA challenge.
deviceIdRateLimiting	Indicates whether to mitigate based on device ID with blocking requests.
deviceIdRequestBlockingMode	Specifies the mitigation when `deviceIdRateLimiting` is enabled, as `rate-limit` or `block-all`.
deviceIdMaximumTps	Specifies the maximum TPS per device ID to arouse suspicion.
deviceIdMinimumTps	Specifies the minimum TPS per device ID.
deviceIdTpsIncreaseRate	Specifies the percent rate of increase per device ID to arouse suspicion.

Application Security Manager (ASM) supports device identification using fingerprinting and exposes the enabling attributes listed in the table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/brute-force-attack-preventions` endpoint.

Attributes	Description
alarm	Indicate whether to send an alarm, as `true` or `false`. Defaults to `true`.

Attributes	Description
block	Indicates whether to block the request, as `true` or `false`. Defaults to `true`.
bruteForceProtectionForAllLoginPages	Indicates whether to apply measures to all login pages, as `true` or `false`. The property is available only for the default brute force protection item. Defaults to `false`.
useDeviceId	Indicates whether to count attempts based on device ID, as `true` or `false`. Defaults to `false`.
loginAttemptsFromTheSameClient	Specifies the number of login attempts before blocking. Defaults to 5.
preventionDuration	Specifies a value from an enum that sets the duration of brute force attack prevention, in seconds, or as the string `unlimited`.
measurementPeriod	Specifies a period of time during which to measure login attempts, as seconds. Defaults to 1.
id	Specifies an identifier, as a string.
reEnableLoginAfter	Specifies an interval, in seconds, to wait before re-enabling the login. Defaults to 600.
urlReference	Specifies a URL reference.
detectionCriteria	Specifies the detection criteria, as a JSON object, consisting of `failedLoginAttemptsIncreasePercent`, `failedLoginAttemptsRateReached`, and `minimumFailedLoginAttempts`. All values are integers, and default to 500, 100, 20, respectively.
preventionPolicy	Specifies the prevention policy, as a JSON object, consisting of `sourceIpBasedClientSideIntegrityDefense`, `sourceIpBasedRateLimiting`, `urlBasedClientSideIntegrityDefense`, and `urlBasedRateLimiting`. All values are Boolean, either `true` or `false`. Defaults to `false`, `true`, `false`, and `true`, respectively.
suspiciousCriteria	Specifies the suspicious criteria, as a JSON object, consisting of `failedLoginAtttemptsIncreasePercent`and `failedLoginAttemptsRateReached`. All values are integers. Default to 500 and 20, respectively.

Enforce method on a URL

If you use Application Security Manager™ (ASM™) to manage a per-URL list of allowed or disallowed methods, you can use an iControl® REST API to modify those settings.

The Application Security Manager™(ASM™ supports a mechanism to define a per-URL list of allowed or disallowed methods and exposes the configuration attributes listed in the table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/urls` endpoint.

Attribute	Description
methodsOverrideOnUrlCheck	Indicates whether the override is enabled, as `true` or `false`. Defaults to `false`.

Attribute	Description
methodOverrides	Specifies an array of key-value pairs, in JSON format. Defaults to `true` for the `allowed` property. You must specify a value for the `method` property.

Session awareness mechanisms using fingerprinting

If you use Application Security Manager™ (ASM™) to manage session awareness, you can use an iControl® REST API to retrieve and modify those settings.

The Application Security Manager™(ASM™ supports session awareness mechanisms using fingerprinting and exposes the configuration attributes listed in the table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/session-tracking` endpoint.

Attribute	Description
checkDeviceIdThreshold	Indicates whether to manage device ID scope, as `true` or `false`.
deviceIdThreshold	Specifies the number of violations per device ID scope when the device ID check is enabled.

Session hijacking prevention

If you use Application Security Manager™ (ASM™) to manage policy settings, you can use an iControl® REST API to retrieve and modify those settings.

The Application Security Manager™(ASM™) mitigates session hijacking by assigning a unique identifier to every client device. By maintaining device ID information for a session, ASM can determine if a session has been hijacked. Use the `/mgmt/tm/asm/policies/<MD5HASH>/session-tracking` endpoint.

Attribute	Description
sessionTrackingConfiguration/ enableTrackingSessionHijackingByDeviceId	Indicates whether session hijacking prevention is enabled in ASM by policy.

About WebSockets

The WebSocket protocol defines a bidirectional full-duplex communication channel between a client and a server within the context of an HTTP connection. A WebSocket connection initiates from an existing HTTP connection by sending an `upgrade` header with the value `websocket`. As part of the handshake between the client and the server, the server sends `101 Switching Protocols` in response. Application Security Manager™ (ASM™) supports a policy for WebSocket security, as a distinct protocol with configurable attributes. The specification for WebSockets can be found in *RFC 6455 - The WebSocket Protocol*.

WebSocket protocol

Application Security Manager™ (ASM™) supports security policy settings for the WebSocket protocol.

Application Security Manager™ (ASM™) supports the WebSocket protocol and exposes the attributes listed in the table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/websocket-urls` endpoint.

Attribute	Description
id	Specifies an identifier for the policy.
name	Specifies a name for the policy.
nameBase64Encoded	Indicates whether the name is encoded in Base64 format. Default is `false`.
type	Specifies a value from the WebSocket type enum, either `explicit` or `wildcard`.

Attribute	Description
description	Specifies an optional description for the WebSocket URL.
lastUpdateMicros	Specifies the last update time, in microseconds.
learnNewEntities	Specifies a value from the enum, either `always` or `never`.
protocol	Specifies a value from the WebSocket protocol enum, either `ws` or `wss`.
isAllowed	Indicates `true` for an allowed URL; `false` for a disallowed URL.
metaCharsOnWebsocketUrlCheck	Indicates whether to check meta-characters in the URL, as `true` or `false`. Defaults to false. Applies only to wild card URL types.
metacharOverrides	Specifies an array of `isAllowed` values and corresponding hexadecimal values that take precedence over the global URL meta-character settings. Defaults to `true` and `0x0`.
performStaging	Indicates whether staging is enabled, as `true` or `false`.
wildcardOrder	Specifies the matching order of wildcards, as an integer. Defaults to 0 (zero).
wildcardIncludesSlash	Indicates whether to match more than one segment of a URL for wildcard values, as `true` or `false`.
html5CrossOriginRequestsEnforcement	Specifies the CORS settings, as a JSON object. The object contains `crossDomainAllowedOrigin`, a JSON object, and the `enforcementMode` property. The `enforcementMode` enum values include `remove-all-headers`, `disabled`, and `enforce`.
extension	Specifies an action to take on handshake. Enum values include `ignore`, `block`, or `remove`. Defaults to `remove`.
checkPayload	Indicates whether to check the message payload, as `true` or `false`.
allowTextMessage	Indicates whether free formatted text is allowed in the message payload. Only set if `checkPayload` is `true`. Defaults to `true`.
allowJsonMessage	Indicates whether JSON is allowed in the message payload. Only set if `checkPayload` is `true`. Defaults to `false`.
allowBinaryMessage	Indicates whether binary content is allowed in the message payload. Only set if `checkPayload` is `true`. Defaults to `false`.
plainTextProfile	Specifies a link to a plain text profile for WebSocket messages. Only set if `allowTextMessage` is `true`.
jsonProfile	Specifies a link to a JSON profile for WebSocket messages. Only set if `allowJsonMessage` is `true`.
binaryMessageMaxSize	Specifies the maximum binary message size, as an integer. Defaults to 10000. Only set if `allowBinaryMessage` is `true`.
messageFrameMaxSize	Specifies the maximum size of a WebSocket frame, in bytes.

Attribute	Description
messageFrameMaxCount	Specifies the maximum number of message fragments per frame, as an integer. Defaults to 100.
checkMessageFrameMaxSize	Indicates whether to check the maximum specified value, as `true` or `false`. If `false`, allow any message size.
checkMessageFrameMaxCount	Indicates whether to check the maximum specified value, as `true` or `false`. If `false`, allow any message size.
checkBinaryMessageMaxSize	Indicates whether to check the maximum specified value, as `true` or `false`. If `false`, allow any message size.

The properties of the `crossDomainAllowedOrigin` object appear in the following table.

Attribute	Description
includeSubDomains	Indicates whether to include sub-domains, as `true` or `false`. Defaults to `false`.
originName	Specifies the origin, as a string.
originPort	Specifies the origin port number, as an integer. Defaults to `all` to specify all ports.
originProtocol	Specifies a value from an enum, `http`, `http/https`, or `https`. Defaults to `http/https`.

The properties of the `jsonProfile` object appear in the following table.

Attribute	Description
description	Specifies a description of the profile, as a string.
metacharElementCheck	Indicates whether to check for meta-characters, as `true` or `false`.
attackSignatureCheck	Indicates whether to check attack signatures, as `true` or `false`.
isReferenced	Indicates whether the profile is referenced, as `true` or `false`.
defenseAttributes	Specifies the defense attributes, as a JSON object.
sensitiveData	Specifies the sensitive data, as an array of `parameterName` strings.
lastUpdateMicros	Specifies the last update time, in microseconds.
metacharOverrides	Specifies metachar overrides, as an array of `isAllowed`, as `true` or `false`, and `metachar`, as a hexadecimal value.
name	Specifies a name for the profile, as a string.
signatureOverrides	Specifies signature overrides, as an array of `enabled`, as `true` or `false`, and `signatureReference`, an object.
id	Specifies an identifier for the profile, as a string.

The properties of the `defenseAttributes` object in the `jsonProfile` appear in the following table.

Attribute	Description
maximumTotalLengthOfJSONData	Specifies the length of JSON data, as an integer.
maximumValueLength	Specifies the length of a value, as an integer.
maximimStructureDepth	Specifies the depth of a structure, as an integer.
maximumArrayLength	Specifies the length of an array, as an integer.
tolerateJSONParsingWarnings	Indicates whether to ignore JSON parser warnings, as `true` or `false`.

The properties of the `plainTextProfile` object appear in the following table.

Attribute	Description
description	Specifies a description of the profile, as a string.
metacharElementCheck	Indicates whether to check for meta-characters, as `true` or `false`.
attackSignatureCheck	Indicates whether to check attack signatures, as `true` or `false`.
isReferenced	Indicates whether the profile is referenced, as `true` or `false`.
defenseAttributes	Specifies the defense attributes, as a JSON object.
lastUpdateMicros	Specifies the last update time, in microseconds.
metacharOverrides	Specifies metachar overrides, as an array of `isAllowed`, as `true` or `false`, and `metachar`, as a hexadecimal value.
name	Specifies a name for the profile, as a string.
signatureOverrides	Specifies signature overrides, as an array of `enabled`, as `true` or `false`, and `signatureReference`, an object.
id	Specifies an identifier for the profile, as a string.

The properties of the `defenseAttributes` object in the `plainTextProfile` appear in the following table.

Attribute	Description
maximumTotalLength	Specifies the length of data, as an integer.
maximumLineLength	Specifies the length of a line, as an integer.
performPercentDecoding	Indicates whether to do percent decoding, as `true` or `false`.

About AJAX/JSON Login

In addition to HTTP authentication and HTML forms authentication, modern web applications frameworks use AJAX authentication. A typical AJAX authentication request consists of a POST request of a login form, with a JSON response. Application Security Manager™ (ASM™) supports AJAX login pages.

AJAX/JSON Authentication

If you use Application Security Manager™ (ASM™) to manage AJAX/JSON authentication settings, you can use an iControl® REST API to retrieve and modify those settings.

Application Security Manager™ (ASM™) exposes the properties listed in the table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/login-pages` URI as the path to a specific login page resource.

Attribute	Description
urlReference	Specifies the URL path.
authenticationType	Specifies the authentication type for a request, as an enum. Allowed values include: `none`, `http-basic`, `ntlm`, `form`, `ajax-or-json-request`, or `http-digest`. For AJAX/JSON, specify `ajax-or-json-request` as the authentication type.
usernameParameterName	Specifies the name of the JSON element that corresponds to the user login.
usernameParameterNameBase64Encoded	Indicates whether the `usernameParameterName` attribute is specified in Base64 encoding.
passwordParameterName	Specifies the name of the JSON element the corresponds to the user password.
passwordParameterNameBase64Encoded	Indicates whether the `passwordParameterName` attribute is specified in Base64 encoding.
isReferenced	Indicates whether the login page is referenced elsewhere, as `true` or `false`.
id	Specifies an identifier for the login page.
accessValidation	Specifies the condition to use AJAX/JSON authentication, as a JSON object. The conditions are name-value pairs, where the name is: `cookieContains`, `headerContains`, `parameterContains`, `responseContains`, `responseHttpStatus`, or `responseOmits`.

The properties of the `accessValidation` object are listed in the following table.

Attribute	Description
cookieContains	Specifies a string contained in a cookie.
headerContains	Specifies a string included as a header.
parameterContains	Specifies a string included as a query parameter.
responseContains	Specifies a string contained in a response.
responseOmits	Specifies a string that is not contained in a response.
responseHttpStatus	Specifies a response status, as a string.
parameterContainsBase64Encoded	Indicates whether a query parameter is Base64 encoded.
responseContainsBase64Encoded	Indicates whether a response includes a Base64 encoded value.
responseOmitsBase64Encoded	Indicates whether a response does not include a Base64 encoded value.

Application Security Manager (ASM) also exposes the properties of logout pages listed in the following table. Use the `/mgmt/tm/asm/policies/<MD5HASH>/login-enforcement` URI as the path to a specific logout page resource.

Attribute	Description
expirationTimePeriod	Indicates whether the expiration setting is `enabled` or `disabled`. By default, a session expires after 600 seconds if no request is received.
authenticatedUrls	Specifies authenticated URLs, as an array.
logoutUrls	Specifies `requestContains`, as a string, `urlReference`, as a JSON object, and `requestOmits`, as a string.

Access Policy Manager

About Access Policy Manager

Access Policy Manager® (APM®) provides secure identity and access management for a BIG-IP® system. iControl® REST exposes the APM endpoints to enable programmatic access to APM resources and the benefits of automation.

APM adheres to the REST principles described previously in this guide:

- URI structure enables consistent access to collections and resources
- Links in resources, including self links, support discovery
- JSON encoding simplifies representation of resources
- HTTP transport provides methods to interact with resources, as well as security, authentication, caching, and content negotiation

Overview: URI format and structure

A principle of the REST architecture describes the identification of a resource by means of a Uniform Resource Identifier (URI). A URI identifies the name of a web resource; in this case, the URI also represents the tree structure of modules and components in `tmsh`. You can specify a URI with a web service request to create, read, update, or delete some component or module of a BIG-IP® system configuration. In the context of the REST architecture, the system configuration is synonymous with the representation of a resource, and web service requests read and write that representation using the iControl® REST API.

Tip: Use admin, the default administrative account, for requests to iControl REST. Once you are familiar with the API, you can create user accounts for iControl REST users with various permissions.

For the URI snippet shown here, the `management-ip` component of the URI is the fully qualified domain name (FQDN) or IP address of a BIG-IP device.

```
https://<management-ip>/mgmt/tm/...
```

In iControl REST, the URI structure for all requests includes the string `/mgmt/tm/` to identify the namespace for traffic management. Any identifiers that you append to that string specify collections.

```
https://<management-ip>/mgmt/tm/...
```

The ellipsis in the snippet indicates the location where you specify an *organizing collection*, which is a collection of links to other resources in iControl REST. Organizing collections are the functional equivalent of modules in `tmsh`.

In other words, the organizing collection `apm` in iControl REST is the `apm` module. In iControl REST, you can use the following URI to access all of the resources in the `apm` collection:

```
https://192.168.25.42/mgmt/tm/apm
```

Expanding on that approach, the URI in the following example designates all of the resources in the `report` collection. You can think of a collection as the equivalent of a `tmsh` sub-module. An iControl REST collection contains collections or resources.

```
https://192.168.25.42/mgmt/tm/apm/report
```

The URI in the following example designates a resource, which is a set of entities. In iControl REST, an *entity* is a property that you can configure, such as `"destAddrMax":2048`. A resource may also contain sub-collections. In the parlance of `tmsh`, a resource is the equivalent of a component.

```
https://192.168.25.42/mgmt/tm/apm/report/default-report
```

Important: iControl REST only supports secure access through HTTPS, so you must include credentials with each REST call. Use the same credentials you use for the BIG-IP device manager interface.

About resource formats

JavaScript Object Notation (JSON) defines the format for data interchange in iControl® REST. The JSON standard defines a human-readable format, based in part on the JavaScript programming language. Similar to the eXtensible Markup Language (XML) common to SOAP web services, JSON describes a structuring of data for exchange between clients and servers in REST web service requests. iControl REST processes a request body formatted as JavaScript Object Notation (JSON) format and generates a JSON body in a response. A response to a DELETE request typically does not include a JSON body.

JSON consists of two structures: name/value pairs (key/value pairs) organized as objects, and ordered lists of values organized as arrays. An object is contained within curly braces '{}' and an array is contained within square brackets '[]'. JSON objects can contain objects, strings, numbers, arrays, Boolean values (true or false), or null. For more information about JSON, see *RFC 7159 The JavaScript Object Notation (JSON) Data Interchange Format*.

About creating resources

Create new resources by using the HTTP POST method. iControl® REST supports the POST operation to create a resource in Access Policy Manager® (APM®). You must include a JSON body with a POST request, even if the JSON body is empty.

About retrieving resources

Retrieve resources by using the HTTP GET method. iControl® REST supports the GET operation to retrieve a resource, or a collection of resources, in Access Policy Manager®(APM®). Additionally, iControl REST supports the Open Data Protocol (OData) `$filter` query parameter to refine the result set.

About updating resources

Update resources by using either the HTTP PATCH or PUT methods. iControl® REST supports the HTTP PATCH operation to update a resource in Access Policy Manager® (APM®). Use PATCH to update specific properties and leave other properties unchanged. iControl REST also supports the HTTP PUT operation to update a resource, with the caveat that all unspecified properties are assigned default values.

About deleting resources

Delete resources by using the HTTP DELETE method. iControl® REST supports the DELETE operation in Access Policy Manager® (APM®). iControl REST returns an HTTP response code for a delete request but does not include a JSON body.

HTTP Response Codes

The tables list the common HTTP response codes that iControl® REST generates for every request.

Response code	Returned for	Description
200 OK	All HTTP methods	Indicates that a request completed successfully.
201 Created	POST	Indicates that a request created a resource, such as when you create an iControl REST transaction.

Response code	Returned for	Description
400 Bad Request	All HTTP methods	Indicates a malformed request, such as an incorrect name for a resource.
401 Unauthorized	All HTTP methods	Indicates an omitted HTTP `Authorization` header, or that you lack adequate permissions for the request to complete.
403 Forbidden	All HTTP methods	Indicates that the credentials supplied for an administrator lack adequate permissions for a request, or an attempt to perform an unsupported action, such as deleting a property.
404 Not Found	All HTTP methods	Indicates an attempt to access a resource that does not exist.
409 Conflict	POST, PUT	Indicates an attempt to create a resource that already exists. If you try to create a resource using the POST method, and the resource already exists, iControl REST generates this response.
415 Unsupported Media Type	POST, PUT	Indicates that the request includes a malformed JSON body in a request, or possibly that you specified an incorrect `Content-Type` header value.

Response code	Returned for	Description
500 Internal Server Error	All HTTP methods	Indicates that the iControl REST process is not available, such as when the process has not been started.
501 Not Implemented	POST	Indicates that an endpoint does not exist, or the corresponding `tmsh` request is unsupported.

Retrieving Access Policy Manager resources

Using iControl® REST, you can query Access Policy Manager® (APM®) resources.

1. To discover Access Policy Manager (APM) resources, make a GET request to the endpoint `/mgmt/tm/apm`.

```
GET https://192.168.25.42/mgmt/tm/apm
```

The response displays the structure of APM collection.

```
{
   "kind":"tm:apm:apmcollectionstate",
   "selfLink":"https://localhost/mgmt/tm/apm?ver=12.1.0",
   "items":[
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/aaa?ver=12.1.0"
         }
      },
```

```
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/configuration?
ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/epsec?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/ntlm?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/policy?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/profile?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/report?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/resource?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/saml?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/sso?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/acl?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/apm-avr-config?
ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/application?ver=12.1.0"
         }
      },
      {
         "reference":{
```

```
            "link":"https://localhost/mgmt/tm/apm/application-family?
ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/application-filter?
ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/log-setting?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/risk-class?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/session?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/swg-scheme?ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/url-filter?ver=12.1.0"
         }
      }
   ]
}
```

2. To discover one of the resources in APM, such as `/ntlm`, make a GET request to the endpoint `/mgmt/tm/apm/ntlm`.

```
GET https://192.168.25.42/mgmt/tm/apm/ntlm
```

The response displays the resources within the `/mgmt/tm/apm/ntlm` namespace.

```
{
   "kind":"tm:apm:ntlm:ntlmcollectionstate",
   "selfLink":"https://localhost/mgmt/tm/apm/ntlm?ver=12.1.0",
   "items":[
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/ntlm/machine-account?
ver=12.1.0"
         }
      },
      {
         "reference":{
            "link":"https://localhost/mgmt/tm/apm/ntlm/ntlm-auth?
ver=12.1.0"
         }
      }
   ]
```

```
}
```

Access Policy Manager endpoints

iControl® REST supports the Access Policy Manager® (APM®) endpoints listed here. All endpoints are relative to the traffic management namespace, `/mgmt/tm`.

Endpoint	Description
/apm/aaa	Configure authorization, authentication, and accounting (AAA) settings. You can configure APM to use various servers to provide user authentication, authorization to access resources, and accounting of user activities. APM supports RADIUS, RSA Native SecurID, and Windows Active Directory, among others.
/apm/acl	Restrict access to host and port combinations with Access Control Lists (ACLs).
/apm/apm-avr-config	Configure settings for Application Visibility and Reporting (AVR).
apm/application	Specify the web-based applications you can control by modifying the default allow or block action.
/apm/application-family	Specify categories of applications, such as instant messaging or e-mail.
/apm/application-filter	Specify the application filters you can use to allow or block access to the applications.
/apm/configuration	Specify settings for Secure Web Gateway (SWG) initialization.
/apm/epsec	Configure APM to enable client-side and server-side, endpoint security checks.
/apm/log-setting	Configure APM to log access policy events or audit events.
/apm/ntlm	Configure APM to use NTLM. You can create a machine account for APM to join a Windows domain. Authentication requests with a machine account create a secure channel to communicate with a domain controller.
/apm/policy	Configure policy for scheme assignment.
/apm/profile	Configure profile for traffic handling.
/apm/report	Configure settings for reporting.
/apm/resource	Specify network access and web access resource.
/apm/risk-class	Specify risk classes.
/apm/saml	Configure APM for Security Assertion Markup Language (SAML) framework for creating, requesting, and exchanging authentication and authorization data. You can configure APM as a native SAML 2.0 identity provider (IDP), or as a proxy to another SAML IDP, such as Active Directory Federation Services (ADFS).
/apm/session	Retrieve and manage user sessions.
/apm/sso	Configure APM to use the Single Sign-On (SSO) feature. You can define attributes for user name, password, and authentication methods for SSO, as well as a number of HTTP form-based SSO object attributes.
/apm/swg-scheme	Configure Secure Web Gateway (SWG) schemes to filter and categorize URLs. A scheme lets you group and schedule URL filters for specific days, or specific times during a day.

Endpoint	Description
/apm/url-filter	Configure APM to use a URL filter to specify one or more URL categories to allow or block. Using this endpoint, you can create multiple URL filters. With the exception of default URL filters, you can also delete URL filters.

Configuring LDAP settings in APM

The authentication, authorization, and auditing settings allow you to configure LDAP settings in Access Policy Manager® (APM®). LDAP is a lightweight implementation of the X.500 Directory Access Protocol (DAP) supported by a number of vendors. The iControl® REST API allows you to configure the LDAP server configuration but not the function of an LDAP server.

1. Before you attempt to add an LDAP account and configure it, make a GET request to the `/mgmt/mt/apm/aaa/ldap/example` endpoint to get the reference object.

```
GET https://192.168.25.42/mgmt/tm/apm/aaa/ldap/example
```

You can use the `/example` endpoint to get a representation of the APM resource, or any resource in iControl REST.


```
{
   "kind":"tm:apm:aaa:ldap:ldapcollectionstate",
   "selfLink":"https://localhost/mgmt/tm/apm/aaa/ldap/example?ver=12.1.0",
   "items":[
      {
         "propertyDescriptions":{
            "address":"",
            "adminDn":"",
            "adminEncryptedPassword":"",
            "appService":"",
            "baseDn":"",
            "cleanupCache":"",
            "description":"",
            "groupCacheTtl":"",
            "isLdaps":"",
            "locationSpecific":"",
            "pool":"",
            "port":"",
            "schemaAttr":{
               "groupMember":"",
               "groupMemberValue":"",
               "groupMemberof":"",
               "groupObjectClass":"",
               "userMemberof":"",
               "userObjectClass":""
            },
            "serversslProfile":"",
            "timeout":"",
            "usePool":""
         },
         "address":"any6",
         "adminDn":"",
         "adminEncryptedPassword":"",
         "appService":"",
         "baseDn":"",
         "cleanupCache":"none",
         "description":"",
         "groupCacheTtl":30,
         "isLdaps":"false",
```

```
         "locationSpecific":"true",
         "pool":"",
         "port":389,
         "schemaAttr":{
            "groupMember":"member",
            "groupMemberValue":"dn",
            "groupMemberof":"memberOf",
            "groupObjectClass":"group",
            "userMemberof":"memberOf",
            "userObjectClass":"user"
         },
         "serversslProfile":"",
         "timeout":15,
         "usePool":"enabled",
         "naturalKeyPropertyNames":[
            "name",
            "partition",
            "subPath"
         ]
      }
   ]
}
```

2. To configure LDAP server settings to use with APM, make a POST request to the `/mgmt/tm/apm/aaa/ldap` endpoint. Make sure that you specify `application/json` as the content type.

```
POST https://192.168.25.42/mgmt/tm/apm/aaa/ldap

{
  "name": "test_aaa_ldap",
  "address": "10.1.1.1",
  "adminDn": "\"CN=administrator, CN=users, DC=mydomain, DC=com\"",
  "adminEncryptedPassword": "p4s8w07d",
  "usePool": "disabled"
}
```

This example uses a small subset of properties found in the object. As shown in the JSON, you must escape the quotes ('\') in the JSON string to preserve the quotes. If you intend to use an LDAP server as an authentication or query server, you must use the visual policy editor and make the change manually.

The response includes a status code (`200 OK`) that indicates whether the request succeeded, but iControl REST also includes the newly created resource in the response.

```
{
   "kind":"tm:apm:aaa:ldap:ldapstate",
   "name":"test_aaa_ldap",
   "fullPath":"test_aaa_ldap",
   "generation":30,
   "selfLink":"https://localhost/mgmt/tm/apm/aaa/ldap/test_aaa_ldap?
ver=12.1.0",
   "address":"10.1.1.1",
   "adminDn":"CN=Administrator, CN=Users, DC=mydomain, DC=com",
   "adminEncryptedPassword":"$M$Uq$lXbiDrLRf0Ogq4zAX0pvYQ==",
   "cleanupCache":"none",
   "groupCacheTtl":30,
   "isLdaps":"false",
   "locationSpecific":"true",
   "port":389,
   "schemaAttr":{
      "groupMember":"member",
      "groupMemberValue":"dn",
      "groupMemberof":"memberOf",
      "groupObjectClass":"group",
```

```
      "userMemberof":"memberOf",
      "userObjectClass":"user"
   },
   "timeout":15,
   "usePool":"disabled"
}
```

3. To delete the LDAP settings, make a DELETE request and specify the LDAP server name (`test_aaa-ldap`) from the previous step.

   ```
   DELETE https://192.168.25.42/mgmt/tm/apm/aaa/ldap/test_aaa_ldap
   ```

   iControl REST deletes the resource and responds with an HTTP response. The response does not include a JSON body.

In this example, you configured LDAP server settings. Using the reference object as the starting point, you create a new LDAP server by specifying a small set of properties. After reviewing the new LDAP server, you then delete the server by specifying the resource name.

Creating a custom category in APM

On a BIG-IP® system, you have the option to use a default set of categories in a URL database or to define URL categories and filters. If you have a Secure Web Gateway (SWG) subscription, you can create custom URL categories to extend the URL database. If you do not have an SWG subscription, you can still create custom URL categories. Using the iControl® REST API, you can follow a two-step process to create a custom URL category and then attach the custom category to a URL filter.

1. To create a custom category, make a GET request to the `/sys/url-db/url-category` endpoint. Use the response to determine if a category exists, and if the category is allowed or blocked.

   ```
   GET https://192.168.25.42/mgmt/tm/sys/url-db/url-category
   ```

   ```
   {
      "kind":"tm:sys:url-db:url-category:url-categorycollectionstate",
      "selfLink":"https://localhost/mgmt/tm/sys/url-db/url-category?
   ver=12.1.0",
      "items":[
         {
            "kind":"tm:sys:url-db:url-category:url-categorystate",
            "name":"Entertainment",
            "partition":"Common",
            "fullPath":"/Common/Entertainment",
            "generation":1,
            "selfLink":"https://localhost/mgmt/tm/sys/url-db/url-category/
   ~Common~Entertainment?ver=12.1.0",
            "catNumber":10,
            "defaultAction":"allow",
            "description":"Sites with information about entertainment.",
            "displayName":"Entertainment",
            "isCustom":"false",
            "isRecategory":"false",
            "parentCatNumber":0,
            "severityLevel":0
         },
   ... (Truncated for readability)
         {
            "kind":"tm:sys:url-db:url-category:url-categorystate",
            "name":"Business",
            "partition":"Common",
   ```

```
         "fullPath":"/Common/Business",
         "generation":1,
         "selfLink":"https://localhost/mgmt/tm/sys/url-db/url-category/
~Common~Business?ver=12.1.0",
         "catNumber":1902,
         "defaultAction":"block",
         "displayName":"Business",
         "isCustom":"true",
         "isRecategory":"false",
         "parentCatNumber":0,
         "severityLevel":0,
         "urls":[
            {
               "name":"http://www.example.com/*",
               "type":"glob-match"
            },
            {
               "name":"http://www.example.com/?/",
               "type":"exact-match"
            }
         ]
      }
   ]
}
```

2. After you determine the custom category does not exist, create the custom category. As with other APM examples, append the `/example` endpoint to the URL from the previous step if you would like to see the sample representation of the object.

```
POST https://192.168.25.42/mgmt/tm/sys/url-db/url-category

{
  "displayName": "my-custom-category",
  "defaultAction": "block",
  "urls": [ ]
}
```

3. To attach the custom category to a URL filter, make a POST request to the `/tm/apm/url-filter` endpoint.

```
POST https://192.168.25.42/mgmt/tm/apm/url-filter

{
  "name": "my-url-filter",
  "allowedCategories": "my-custom-category"
}
```

   As with any iControl REST request, the response shows the result of the request.

In this example, you created a custom URL category and attached the custom category to a URL filter.

Managing user sessions in APM

Access Policy Manager® (APM®) tracks user sessions with session identifiers (session IDs). The `access-info` endpoint in APM enables you to make an iControl® REST request for a listing of all user sessions. The response contains the session ID, user login, and IP address for each session. As part of the session management process, you can make an iControl REST request to the `session` endpoint to delete a specific session.

1. To view the current user sessions in APM, make a GET request to the `/mgmt/tm/apm/access-info` endpoint.

   ```
   GET https://192.168.25.42/mgmt/tm/apm/access-info
   ```

   The response to this request includes the following data:

   ```
   {
     "apiRawValues": {
       "apiAnonymous": {
         "apm::access-info" "914c727f (login user=user1) client
    (IP=10.20.36.2)"
   ...(Truncated for readability)
     "kind": "tm:apm:access-info:access-infostats",
     "selfLink": "https://localhost/mgmt/tm/apm/access-info?ver=12.1.0"
   }
   ```

2. To view the sessions for a specific user name, make a GET request to the `mgmt/tm/apm/access-info` endpoint and use the `options` query parameter to specify the user name.

   ```
   GET https://192.168.25.42/mgmt/tm/apm/access-info?ver=12.0.0&options=apm-
   user
   ```

   In a similar manner, you can also specify an IP address to get a listing of all sessions for a specified IP address. Use the same query parameter (options) as in the example.

3. To delete a session, make a DELETE request and append the session identifier that identifies the resource to the `/mgmt/tm/apm/session` endpoint.

   ```
   DELETE https://192.168.25.42/mgmt/tm/apm/session/914c727f
   ```

   The affected user will no longer be able to access resources. The user must log in again.

   The response to the request, if successful, is `200 OK`.

In this example, you made an iControl REST request to APM to obtain a listing of all user sessions and made an additional request to delete a specific session.

Listing OAuth tokens

For all OAuth token examples, you must have configured Access Policy Manager® (APM®) on a BIG-IP® system to act as an authorization server (AS), or as a client to an external provider, such as Facebook or Google. Follow the steps outlined in the BIG-IP® Access Policy Manager®: Authentication and Single Sign-On, version 13.0 guide.

If you need the details of all OAuth tokens, you can query the default database. For an authorization server (AS) configured on a BIG-IP system, make a GET request to the AS.

To query the token database, make a GET request.

The response contains properties specific to each token in the database. You can use the output of the request to obtain the `oauthid` and `client-id` properties necessary to revoke a token.

```
GET https://192.168.25.42/mgmt/tm/apm/oauth/token-details
```

In this example, you requested a listing of all tokens in the database.

Getting a count of OAuth tokens

For all OAuth token examples, you must have configured Access Policy Manager® (APM®) on a BIG-IP® system to act as an authorization server (AS), or as a client to an external provider, such as Facebook or Google. Follow the steps outlined in the BIG-IP® Access Policy Manager®: Authentication and Single Sign-On, version 13.0 guide.

As part of the task of managing access to resources, you may need to query the OAuth database to get a count of the number of user tokens. This type of request uses the default database instance, according to the configuration you chose. Depending on the desired output, you can query for a count of all tokens or a count of tokens by application name.

1. To query for a count of tokens, make a GET request.

```
GET https://192.168.25.42/mgmt/tm/apm/oauth/token-details/stats?ver=13.0
```

   The response will contain JSON content, similar to the following output:

```
{
   "apiRawValues" : { "apiAnonymous" : "Total tokens : 7\n" },
   "kind" : "tm:apm:oauth:token-details:token-detailscollectionstats",
   "selfLink" : "https://localhost/mgmt/tm/apm/oauth/token-details/stats?
ver=12.1.0"
}
```

2. To query for a count of tokens associated with an application, make a GET request.

```
GET https://192.168.25.42/mgmt/tm/apm/oauth/token-details/stats?
ver=13.0&options="app-name", "application name"
```

   Supply the name of the application in double quotes (" ") in the request and not the string that appears in the example.

   The count by application response will contain JSON content, similar to the following output:

```
{
   "apiRawValues" : { "apiAnonymous" : "Total tokens : 5\n" },
   "kind" : "tm:apm:oauth:token-details:token-detailscollectionstats",
   "selfLink" : "https://localhost/mgmt/tm/apm/oauth/token-details/stats?
ver=12.1.0"
}
```

In this example, you requested a count of all tokens in the database, and then requested a count of all tokens for a specific application.

Revoking an OAuth token

For all OAuth token examples, you must have configured Access Policy Manager® (APM®) on a BIG-IP® system to act as an authorization server (AS), or as a client to an external provider, such as Facebook or Google. Follow the steps outlined in the BIG-IP® Access Policy Manager®: Authentication and Single Sign-On, version 13.0 guide.

If you need to revoke an OAuth token, make a REST call to handle the revocation. For an authorization server (AS) configured on a BIG-IP system, make a POST request to the AS.

To revoke a token, make a POST request. The `oauthid` and `client-id` properties can be found in the output of the token details listing. Both properties are string values that must be enclosed in quotes in the JSON body.

```
POST https://192.168.25.42/mgmt/tm/apm/oauth/token-details
```

```
{
   "command" : "revoke",
   "name" : "<oauthid>",
   "client-id" : "<clientid>",
   "db-instance" : "<database name>"
}
```

In this example, you revoked a token. You used the output of the previous example to find the associated properties that identify the token.

API Life Cycle

REST API life cycle policy

REST API life cycle policy describes an approach to manage the purpose or longevity of REST collections as well as `tmsh` resources. Life cycle management presents several use cases that affect resources and methods, and the REST API life cycle policy is designed to provide useful information about resources and properties that may be new or experimental, or being phased out. While deprecation of a resource or resource property represents the most likely use case, other use cases exist for early access features, as well as internal use or test. You should interpret the deprecation of a resource or property to mean that the use of a resource or property is discouraged, and not that a resource or resource property will be removed in the near term. The goal is to make you aware of changes before the changes happen.

The API life cycle policy introduces status values, with NO_STATUS being the default value. The NO_STATUS value indicates that no determination has been made about a resource or property, and that REST does not log usage of those resources. REST logs usage of just the resources and properties that match the status values that you configure. For deprecated and early access resources, a custom REST header (`X-F5-Api-Status`) indicates one of the following values:

- DEPRECATED_RESOURCE
- DEPRECATED PROPERTY
- EARLY_ACCESS_RESOURCE
- EARLY_ACCESS_PROPERTY

Using the REST API life cycle changes

The implementation of the REST API Life Cycle Policy provides an API status value and additional information as log entries for a request. The following examples demonstrate a request with various HTTP verbs and the corresponding headers (if any), and log entries.

Important: This feature works only with resource collections.

1. To generate the life cycle output for a query of a resource, make a GET request to the `/mgmt/tm/ltm/profile/ocsp-stapling-params` endpoint.

   ```
   GET https://192.168.25.42/mgmt/tm/ltm/profile/ocsp-stapling-params
   ```

   ```
   {
   ```

```
  "kind" : "tm:ltm:profile:ocsp-stapling-params:ocsp-stapling-
paramscollectionstate",
  "selfLink": "https://localhost/mgmt/tm/ltm/profile/ocsp-stapling-params?
ver=13.0.0"
}
```

2. To locate the detailed information for the resource, find the corresponding log entry in the `/var/log/icrd` log.

```
Dec 30 23:59:36 localhost notice icrd_child: 18826,18853,iControl REST
 Child
      Daemon,WARNING,[api-status-warning]: ltm/profile/ocsp-stapling-
params: deprecated
```

   A log message includes an identifier, such as `[api-status-warning]` to indicate log entries for REST API Life Cycle Management.

3. To generate the output for a query of a resource property, make a GET request to the `/mgmt/tm/ltm/profile/fastl4/fastL4` endpoint.

```
GET https://192.168.25.42/mgmt/tm/ltm/profile/fastl4/fastL4
```

```
{
....
  "serverTimestamp": "disabled",
  "softwareSynCookie": "disabled",
  "synCookieEnable": "enabled",
....
}
```

4. To locate the detailed information for the resource property, find the corresponding log entry in the `/var/log/icrd` log.

```
Dec 31 00:05:02 localhost notice icrd_child: 18826,18853,    iControl REST
 Child
      Daemon,WARNING,[api-status-warning]: ltm/profile/fastl4: no status;
 properties: deprecated:
      ltm/profile/fastl4/hardware-syn-cookie, ltm/profile/fastl4/software-
syn-cookie
```

5. To generate the life cycle output for the creation of a new resource, make a POST request to the `/mgmt/tm/ltm/profile/ocsp-stapling-params` endpoint. Specify the JSON body, as shown.

```
POST https://192.168.25.42/mgmt/tm/ltm/profile/ocsp-stapling-params
```

```
{
  "name": "myocsp",
  "dnsResolver": "dns-resolver-1"
}
```

   Note: The response includes the header and value `X-F5-Api-Status: DEPRECATED_RESOURCE`

6. To view the log information for the request, find the entry in the `/var/log/icrd` log.

```
Jan  5 01:14:08 localhost notice icrd_child[2562]:  2562, 2567,
 iControl REST Child
      Daemon,WARNING,[api-status-warning]: ltm/profile/ocsp-stapling-
params: deprecated
```

7. To generate the life cycle output for the creation of a new resource, with the deprecated API allowed setting set to false, make a POST request to the `/mgmt/tm/ltm/profile/ocsp-stapling-params` endpoint. Specify the JSON body, as shown.

```
POST https://192.168.25.42/mgmt/tm/ltm/profile/ocsp-stapling-params
```

```
{
  "name": "myocsp",
  "dnsResolver": "dns-resolver-1"
}
```

Note: The response includes the status message `HTTP/1.1 404 Not Found`.

8. To view the log information for the request, find the entry in the `/var/log/restjavad.0.log` log. Note that the entry appears in a different log file than previous examples.

```
[WARNING][157][05 Jan 2017 01:23:42 UTC][8100/mgmt/shared/resolver/groups
      ForwarderPassThroughWorker] [api-status-warning] The deprecate API
      /mgmt/tm/ltm/profile/ocsp-stapling-params/ is not available as per
 the
      /shared/settings/api-status/availability
```

9. To generate the output for the creation of a resource property, make a POST request to the `/mgmt/tm/ltm/profile/fastl4/fastL4` endpoint. Specify the JSON body, as shown.

```
POST https://192.168.25.42/mgmt/tm/ltm/profile/fastl4
```

```
{
  "name": "myfastl4",
  "softwareSynCookie": "enabled"
}
```

Note: The response includes the header and value `X-F5-Api-Status: DEPRECATED_PROPERTY`.

10. To view the log information for the request, find the entry in the `/var/log/icrd` log.

```
Jan  5 17:26:53 localhost notice icrd_child[2562]:  2562, 2568,
 iControl REST Child
      Daemon,WARNING,[api-status-warning]: ltm/profile/fastl4: no status;
 properties: deprecated:
      ltm/profile/fastl4/hardware-syn-cookie, ltm/profile/fastl4/software-
syn-cookie
```

11. To generate the life cycle output for the creation of a new resource property, with the deprecated API allowed setting set to false, make a POST request to the `/mgmt/tm/ltm/profile/fastl4` endpoint. Specify the JSON body, as shown.

```
POST https://192.168.25.42/mgmt/tm/ltm/profile/fastl4
```

```
{
  "name": "myfastl4",
  "softwareSynCookie": "enabled"
}
```

Note: The response includes the status message `HTTP/1.1 404 Not Found`, as well as the header and value `X-F5-Api-Status: DEPRECATED_PROPERTY`.

12. To view the log information for the request, find the entry in the `/var/log/icrd` log.

```
Jan  5 17:36:45 localhost notice icrd_child[2562]:  2562, 2567,
 iControl REST Child
```

```
      Daemon,WARNING,[api-status-warning]: ltm/profile/fastl4: no status;
 properties: deprecated:
      ltm/profile/fastl4/software-syn-cookie
```

Using the REST API life cycle changes with tmsh

The following examples show the `tmsh` equivalent of the previous REST examples.

1. To generate the API life cycle output for a resource, run a `tmsh` command to list the resource.
   `(tmos)# list ltm profile ocsp-stapling-params`

   ```
   api-status-warning] ltm/profile/ocsp-stapling-params is deprecated
   ltm profile ocsp-stapling-params ocsp-cur {
       dns-resolver dns-resolver-cur
   }
   ```

2. To locate the detailed information for the resource, find the corresponding log entry in the `/var/log/ltm` log.

   ```
   Dec 30 16:00:43 localhost warning tmsh[1409]: 01420013:4: [api-status-
   warning]
         ltm/profile/ocsp-stapling-params is deprecated
   ```

3. To generate the API life cycle output for a resource property, run a `tmsh` command as shown.
   `(tmos)# list ltm profile fastl4 fastL4 software-syn-cookie`

   ```
   [api-status-warning] ltm/profile/fastl4, properties : deprecated :
          software-syn-cookieltm profile fastl4 fastL4 {
   ...
      reassemble-fragments disabled
       reset-on-timeout enabled
       software-syn-cookie enabled
   }
   ```

4. To locate the detailed information for the resource property, find the corresponding log entry in the `/var/log/ltm` log.

   ```
   Dec 30 16:02:49 localhost warning tmsh[1731]: 01420013:4: [api-status-
   warning]
         ltm/profile/fastl4, properties : deprecated : software-syn-cookie
   ```

5. To generate the life cycle output for the creation of a new resource, run a `tmsh` command as shown. The output appears after the command.
   `(tmos)# create ltm profile ocsp-stapling-params myocsp dns-resolver dns-resolver-1`

   ```
   [api-status-warning] ltm/profile/ocsp-stapling-params is deprecated
   ```

6. To locate the detailed information for the resource, find the corresponding log entry in the `/var/log/ltm` log

   ```
   Jan  5 09:49:54 localhost warning tmsh[4542]: 01420013:4: [api-status-
   warning]
         ltm/profile/ocsp-stapling-params is deprecated
   ```

7. To generate the life cycle output for the creation of a new resource, with the deprecated API allowed setting set to false, run a `tmsh` command as shown. The output appears after the command.

```
(tmos)# create ltm profile ocsp-stapling-params myocsp dns-resolver dns-
resolver-1
```

```
[api-status-warning] ltm/profile/ocsp-stapling-params is deprecated. This
 command is not
      available or has properties which are not available.
```

8. To locate the detailed information for the resource, find the corresponding log entry in the `/var/log/ltm` log.

```
Jan  5 09:49:01 localhost warning tmsh[4493]: 01420013:4: [api-status-
warning]
      ltm/profile/ocsp-stapling-params is deprecated
```

9. To generate the output for the creation of a resource property, run a tmsh command as shown. The output appears after the command.

```
(tmos)# create ltm profile fastl4 myfastl4 software-syn-cookie enabled
```

```
[api-status-warning] ltm/profile/fastl4, properties : deprecated :
 software-syn-cookie
```

10. To locate the detailed information for the resource property, find the corresponding log entry in the `/var/log/ltm` log.

```
Jan  5 09:44:28 localhost warning tmsh[4426]: 01420013:4: [api-status-
warning]
      ltm/profile/fastl4, properties : deprecated : software-syn-cookie
```

11. To generate the life cycle output for the creation of a new resource, with the deprecated API allowed setting set to false, run a `tmsh` command as shown. The output appears after the command.

```
(tmos)# create ltm profile fastl4 myfastl4 software-syn-cookie enabled
```

```
[api-status-warning] ltm/profile/fastl4, properties : deprecated :
 software-syn-cookie
```

12. To locate the detailed information for the resource property, find the corresponding log entry in the `/var/log/ltm` log.

```
Jan  5 09:44:28 localhost warning tmsh[4426]: 01420013:4: [api-status-
warning]
      ltm/profile/fastl4, properties : deprecated : software-syn-cookie
```

Configuring the REST API life cycle settings

The REST API Life Cycle Policy supports configurable API states and log settings. By enabling specific settings, you enable the logging of information by REST and `tmsh`.

1. To view the settings for REST, make a GET request.

```
GET https://192.168.25.42/mgmt/shared/settings/api-status/availability/
```

The query results will be similar to the following snippet:

```
{
   "deprecatedApiAllowed": "true",
   "earlyAccessApiAllowed": "true",
   "testOnlyApiAllowed": "false"
}
```

Important: These settings affect the visibility of a resource. If you specify any of the states as disabled, you will not be able to view that resource in REST requests or in `tmsh`. A REST request will generate a 404 (Not Found) response code. In `tmsh`, tab completion will not expose these resources.

2. To change an `api-status` setting, make a PATCH or POST request. In the JSON body, specify the visibility settings to change. For example,

```
PATCH https://192.168.25.42/mgmt/shared/settings/api-status/availability/
```

```
{
   "earlyAccessApiAllowed": "false"
}
```

Confirm that the change was successful by making a query request.

3. To specify the resource settings that generate log entries, make a GET request.

```
GET https://192.168.25.42/mgmt/shared/settings/api-status/log/resource
```

The query results will again be similar to the following:

```
{
   "deprecatedApiAllowed": "true",
   "earlyAccessApiAllowed": "true",
   "testOnlyApiAllowed": "false"
}
```

4. To specify the property resource settings that generate log entries, make a GET request (results are omitted).

```
GET https://192.168.25.42/mgmt/shared/settings/api-status/log/resource-
property
```

For either endpoint, make a PATCH or POST request as shown to modify any of the settings.

In this topic, you queried and configured the visibility and logging settings for the API life cycle.

Configuring the REST API life cycle settings with tmsh

The REST API life cycle policy supports configurable API states and log settings. You can configure API states and log settings with `tmsh` commands.

1. To view the settings, type the following `tmsh` command.
   `(tmos)# list mgmt shared settings api-status availability`
2. To modify the `deprecated` setting, type the following `tmsh` command.
   `(tmos)# modify mgmt shared settings api-status availability`
   `{ deprecatedApiAllowed value false }`
   The command changes the state to disabled.
3. To view the settings for log resources, type the following `tmsh` command
   `(tmos)# list mgmt shared settings api-status log resource`
   You can view the log setting for resource properties by specifying `resource-property` instead of `resource` in the command.

```
mgmt shared settings api-status log resource
{
  deprecatedApiAllowed true
  testOnlyApiAllowed true
```

```
  earlyAccessApiAllowed true
}
```

In this topic, you used `tmsh` commands to view and modify API status and log settings.

Additional Features

About the example suffix

The inclusion of the `/example` suffix at the end of a URI prompts iControl® REST to generate a sample representation of a resource. The `/example` suffix may be used in a GET request to produce a representation that includes all properties, including null properties. The sample representation also includes the help text strings that describe each property and a list of *natural keys* for a resource. A natural key consists of one or more user-friendly properties that uniquely identify a resource, such as `area code/phone number`.

In iControl REST, a natural key is represented in JSON as a `naturalKeyPropertyNames` property, which is an array of `name`, `partition`, and `subPath`. The `name`, `partition`, and `subPath` form a full path to an object. For resources in the `Common` partition, iControl REST omits the partition name as a natural key. If an object is a singleton object, the `naturalKeysPropertyNames` array is empty.

BIG-IP® system components that support only the TMSH commands `SHOW`, `LIST`, `DELETE`, `LOAD`, `SAVE`, `INSTALL`, or `RUN` do not have default field values. For those components, an iControl REST request to the `/example` endpoint does not generate a response with default values.

The sample representation of Application Security Manager™ (ASM™) resources includes only default values and possible enumeration values. The sample representation of ASM resources does not include descriptions of the properties as help text.

The sample representation specifies a default value for a property, if a default value exists. If a property has no default value, the representation includes:

- An empty string ("") for a string property
- Zero (0) for a numerical property
- False for a Boolean property
- An empty JSON array or object

If a property takes a value from an enumeration, the representation displays the acceptable values as an array. iControl REST also appends the suffix `Enums` to the name of this array to identify the enumeration.

Tip: Copy the sample representation, make changes to the copy, and then paste the changes into the JSON body of a POST request.

About Access Policy Manager

Access Policy Manager® (APM®) provides secure identity and access management for a BIG-IP® system. iControl® REST exposes the APM endpoints to enable programmatic access to APM resources and the benefits of automation.

APM adheres to the REST principles described previously in this guide:

- URI structure enables consistent access to collections and resources
- Links in resources, including self links, support discovery
- JSON encoding simplifies representation of resources
- HTTP transport provides methods to interact with resources, as well as security, authentication, caching, and content negotiation

About HTTP response codes

Responses to all iControl® REST requests contain a response code, as listed here.

Success responses

Response code	Description
200 OK	Indicates success for all methods.

Error responses

Response code	HTTP methods	Description
400 Bad Request	all	Possible causes include: • malformed HTTP request • incorrect name for a resource in a request
401 Unauthorized	all	Possible causes include: • missing HTTP authorization header • insufficient permissions for the credentials supplied for an administrator
403 Forbidden	all	Possible causes include: • insufficient permissions for the credentials supplied for an administrator • attempt to perform an unsupported action, such as deleting a property
404 Not Found	all	Possible causes include: • attempting to access a resource that no longer exists in the database
409 Conflict	POST, PUT	Possible causes include: • attempting to create a resource that already exists Indicates a conflict between the requested state change and the current state of the resource. For example, this is the error response if you POST a resource that already exists.
415 Unsupported Media Type	POST, PUT	Possible causes include: • specifying an incorrect `Content-Type` header value

Response code	HTTP methods	Description
		• specifying a malformed JSON body with a POST or PUT request
500 Internal Server Error	all	Possible causes include: • attempting to access iControl REST when the process is not running
501 Not Implemented	POST	Possible causes include: • attempting to access a endpoint that does not exist • attempting to invoke an unsupported `tmsh` command through iControl REST

About log files

From the console or an SSH connection to your BIG-IP® device, you can find the following log files for iControl® REST:

- `/var/log/restjavad-audit.0.log` shows all authentications to the iControl REST service. This is an ordered list of every REST call.
- `/var/log/restjavad.0.log` contains information about connections to the iControl REST service, such as errors returned.
- `/var/log/icrd` shows the actions of the `icrd` process, which manages the threads for processing the REST calls.
- `/var/log/ltm` contains messages from `mcpd`, a process called by `icrd` that manages the system configuration.

Use standard `Unix` commands to work with these files, such as `tail`, `grep`, and `less`. In this example, the session logs in to a BIG-IP system through `ssh` and uses `tail -f` to monitor the `/var/log/restjavad-audit.0.log` log file:

```
juser@bench2:~/$ ssh root@192.168.25.42
Password: default
Last login: Fri Mar 29 09:03:25 2013 from 192.168.98.174
[root@localhost:Active:Standalone] config # tail -f /var/log/restjavad-
audit.0.log
[I][339][29 Mar 2013 16:04:06 UTC][ForwarderPassThroughWorker] \
  [run]{"user":"admin","method":"PUT",\
  "uri":"http://localhost:8100/mgmt/tm/ltm/pool/dns-pool2",\
  "status":"succeeded","from":"192.168.96.37"}
[I][340][29 Mar 2013 16:04:06 UTC][ForwarderPassThroughWorker] \
  [run]{"user":"admin","method":"GET",\
  "uri":"http://localhost:8100/mgmt/tm/ltm/pool","\
  status":"succeeded","from":"192.168.96.37"}
[I][341][29 Mar 2013 16:04:06 UTC][ForwarderPassThroughWorker] \
  [run]{"user":"admin","method":"DELETE",\
  "uri":"http://localhost:8100/mgmt/tm/ltm/pool/test-pool2",\
  "status":"succeeded","from":"192.168.96.37"}
[I][342][29 Mar 2013 16:04:07 UTC][ForwarderPassThroughWorker] \
  [run]{"user":"admin","method":"POST",\
  "uri":"http://localhost:8100/mgmt/tm/sys/folder",\
```

```
  "status":"succeeded","from":"192.168.96.37"}
[I][343][29 Mar 2013 16:04:07 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"admin","method":"DELETE",\
  "uri":"http://localhost:8100/mgmt/tm/sys/folder/~fw_objs",\
  "status":"succeeded","from":"192.168.96.37"}
[I][344][29 Mar 2013 16:04:07 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"admin","method":"DELETE",\
  "uri":"http://localhost:8100/mgmt/tm/sys/folder/~eu~east~romania",\
  "status":"succeeded","from":"192.168.96.37"}
[I][345][29 Mar 2013 16:04:07 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"admin","method":"POST",\
  "uri":"http://localhost:8100/mgmt/shared/authz",\
  "status":"succeeded","from":"192.168.96.37"}
[I][346][29 Mar 2013 16:04:07 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"admin","method":"GET",\
  "uri":"http://localhost:8100/mgmt/shared/authz",\
  "status":"succeeded","from":"192.168.96.37"}
[I][347][29 Mar 2013 16:04:10 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"dns_admin","method":"GET",\
  "uri":"http://localhost:8100/mgmt/tm/sys",\
  "status":"succeeded","from":"192.168.96.37"}
[I][350][29 Mar 2013 16:04:10 UTC][ForwarderPassThroughWorker]\
  [run] {"user":"admin","method":"GET",\
  "uri":"http://localhost:8100/mgmt/tm/ltm/pool/http-pool?$stats=true",\
  "status":"succeeded","from":"192.168.96.37"}
...
```

If you need to adjust the logging levels for `icrd`, contact F5® Networks Technical Support (`http://www.f5.com/support/`).

About public URIs

A URI is considered to be public if you can access it through an iControl® REST request. In general, all of the following are public:

- Traffic Management Shell (tmsh) modules
- Traffic Management Shell (tmsh) components
- Any component properties that are accessible through the `tmsh show` command.

To view the component properties, make a GET request of a parent component. By default, you cannot use a GET request to obtain them directly through a public URI.

The public URIs exist to provide direct access to some of those component properties. The iControl REST process allows these for convenience, for situations where a PUT request of the entire containing object (a component or collection) would be unwieldy.

In many cases, the second-to-last part of the path is the name of a component, and you need to provide a specific object name for that component before the final part of the path. For example, to access the public URI `/mgmt/tm/gtm/pool/members`, you must specify the DNS pool for which you want members, such as `/mgmt/tm/gtm/pool/pool5/members` for the members of `pool5`.

Legal Notices

Legal notices

Publication Date

This document was published on June 11, 2020.

Publication Number

MAN-0797-00

Copyright

Trademarks

For a current list of F5 trademarks and service marks, see http://www.f5.com/about/guidelines-policies/trademarks/.

All other product and company names herein may be trademarks of their respective owners.

Patents

This product may be protected by one or more patents indicated at: https://f5.com/about-us/policies/patents

Link Controller Availability

This product is not currently available in the U.S.

Export Regulation Notice

This product may include cryptographic software. Under the Export Administration Act, the United States government may consider it a criminal offense to export this product from the United States.

RF Interference Warning

This is a Class A product. In a domestic environment this product may cause radio interference, in which case the user may be required to take adequate measures.

FCC Compliance

This equipment has been tested and found to comply with the limits for a Class A digital device pursuant to Part 15 of FCC rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This unit generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Any modifications to this device, unless expressly approved by the manufacturer, can void the user's authority to operate this equipment under part 15 of the FCC rules.

Canadian Regulatory Compliance

This Class A digital apparatus complies with Canadian ICES-003.

Standards Compliance

This product conforms to the IEC, European Union, ANSI/UL and Canadian CSA standards applicable to Information Technology products at the time of manufacture.

Index

E

F

G

H

I

J

K

L

M

O

P

Q

R

S

T

U

V

W