iControl[®] REST User Guide

Version 11.6



Table of Contents

Legal Notices	7
Acknowledgments	9
Chapter 1: REST	13
What's new in iControl REST for version 11.6	14
About Representational State Transfer	14
About URI format	14
About reserved ASCII characters	15
About REST resource identifiers	15
About HTTP method semantics	16
About JSON format	16
About additional iControl REST properties	19
About null values and properties	20
About reserved property names	21
About property name format differences	21
About JSON formats and encodings	22
About API versions	22
About iControl and RBAC for user accounts	22
Associating a role with a user account	22
	05
Chapter 2: GET requests	
Discovering modules and components	20
About query parameters	30
About paging properties	
Paging through large collections	31
About sub-collection expansion	
Returning resources from an administrative partition	
Obtaining statistical output	37
Chapter 3: POST and PUT requests	41
About JSON format for POST and PUT	42
Adding and modifying resources	42
Modifying a configuration object	43
About read only properties	47
Adding or modifying in a specific partition	48
Deleting a configuration object	50
Chapter 4. Partitions	51

Adding an administrative partition	52
Deleting an administrative partition	53
Chapter 5. Transactions	55
About the iControl REST transaction model	56
Creating an iControl REST transaction	
Modifying a transaction	
Committing an iControl REST transaction	
Chapter 6: Commands	50
About other tmsh global commands	5 9 60
Lising the on command	
Using the generate command	
	61
Lising the load command	
Using the my command	
Lising the publish command	
Using the report command	
Using the restart command	03
Lising the reset-state command	
Using the run command	64
Using the save command	
Lising the send-mail command	
Using the start command	66
Using the stop command	
Chapter 7: Application Security Manager	60
Application Security Manager and iControl PEST comparison	09 70
Application Security Manager and IControl IXEOT comparison	
Creating Application Security Manager resources	
Lindating Application Security Manager resources	
Deleting resources in Application Security Manager	75
Application Security Manager policy	
Importing a policy in Application Security Manager	
Exporting a policy in Application Security Manager	78
Applying a policy in Application Security Manager	
Application Security Manager signatures	80
Checking for signatures in Application Security Manager	80
Updating signatures in Application Security Manager	
Exporting signatures in Application Security Manager	
Retrieving signature status information in Application Security Manage	3er 83
Retrieving signature systems in Application Security Manager	
Application Security Manager schema upload	
Uploading schema files in Application Security Manager	84
	-

Application Security Manager schemas and profiles	84
Associating schemas to profiles in Application Security Manager	85
Application Security Manager policy restore	86
Restoring policy revisions in Application Security Manager	87
Application Security Manager vulnerability import	88
Importing vulnerabilities in Application Security Manager	88
Querying vulnerability assessment subscriptions in Application Security	
Manager	90
Initiating vulnerability assessment in Application Security Manager	91
Terminating vulnerability assessment in Application Security Manager	92
Application Security Manager vulnerability resolution	93
Resolving vulnerabilities in Application Security Manager	93
Identifying vulnerabilities in Application Security Manager	95
About the example suffix	96

Chapter 8: Access Policy Manager	97
About Access Policy Manager	
Retrieving Access Policy Manager resources	99
Creating Access Policy Manager resources	100
Modifying Access Policy Manager resources	101
Deleting Access Policy Manager resources	101

Chapter 9: Additional features	103
About HTTP response codes	104
About log files	105
About public URIs	106
List of public URIs	106

Table of Contents

Legal Notices

Publication Date

This document was published on August 20, 2014.

Publication Number

MAN-0526-00

Copyright

Copyright © 2012-2014, F5 Networks, Inc. All rights reserved.

F5 Networks, Inc. (F5) believes the information it furnishes to be accurate and reliable. However, F5 assumes no responsibility for the use of this information, nor any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent, copyright, or other intellectual property right of F5 except as specifically described by applicable user licenses. F5 reserves the right to change specifications at any time without notice.

Trademarks

3DNS, Access Policy Manager, Acopia, Acopia Networks, Advanced Client Authentication, Advanced Routing, APM, Application Security Manager, ARX, AskF5, ASM, BIG-IP, Cloud Extender, CloudFucious, CMP, Data Manager, DevCentral, DevCentral [DESIGN], DNS Express, DSC, DSI, Edge Client, Edge Gateway, Edge Portal, EM, Enterprise Manager, F5, F5 [DESIGN], F5 Management Pack, F5 Networks, F5 World, Fast Application Proxy, Fast Cache, FirePass, Global Traffic Manager, GTM, IBR, Intelligent Browser Referencing, Intelligent Compression, IPv6 Gateway, iApps, iControl, iHealth, iQuery, iRules, iRules OnDemand, iSession, IT agility. Your way., L7 Rate Shaping, LC, Link Controller, Local Traffic Manager, LTM, Message Security Module, MSM, Netcelera, OneConnect, Packet Velocity, Protocol Security Module, PSM, Real Traffic Policy Builder, Scale^N, SSL Acceleration, StrongBox, SuperVIP, SYN Check, TCP Express, TDR, TMOS, Traffic Management Operating System, TrafficShield, Transparent Data Reduction, VIPRION, vCMP, WA, WAN Optimization Manager, WANJet, WebAccelerator, WOM, and ZoneRunner, are trademarks or service marks of F5 Networks, Inc., in the U.S. and other countries, and may not be used without F5's express written consent.

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: *http://www.f5.com/about/guidelines-policies/patents*

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.

Acknowledgments

This product includes software developed by Bill Paul. This product includes software developed by Jonathan Stone. This product includes software developed by Manuel Bouyer. This product includes software developed by Paul Richards. This product includes software developed by the NetBSD Foundation, Inc. and its contributors. This product includes software developed by the Politecnico di Torino, and its contributors. This product includes software developed by the Swedish Institute of Computer Science and its contributors. This product includes software developed by the University of California, Berkeley and its contributors. This product includes software developed by the Computer Systems Engineering Group at the Lawrence Berkeley Laboratory. This product includes software developed by Christopher G. Demetriou for the NetBSD Project. This product includes software developed by Adam Glass. This product includes software developed by Christian E. Hopps. This product includes software developed by Dean Huxley. This product includes software developed by John Kohl. This product includes software developed by Paul Kranenburg. This product includes software developed by Terrence R. Lambert. This product includes software developed by Philip A. Nelson. This product includes software developed by Herb Peyerl. This product includes software developed by Jochen Pohl for the NetBSD Project. This product includes software developed by Chris Provenzano. This product includes software developed by Theo de Raadt. This product includes software developed by David Muir Sharnoff. This product includes software developed by SigmaSoft, Th. Lockert. This product includes software developed for the NetBSD Project by Jason R. Thorpe. This product includes software developed by Jason R. Thorpe for And Communications, http://www.and.com. This product includes software developed for the NetBSD Project by Frank Van der Linden. This product includes software developed for the NetBSD Project by John M. Vinopal. This product includes software developed by Christos Zoulas. This product includes software developed by the University of Vermont and State Agricultural College and Garrett A. Wollman.

This product includes software developed by Balazs Scheidler (bazsi@balabit.hu), which is protected under the GNU Public License.

This product includes software developed by Niels Mueller (nisse@lysator.liu.se), which is protected under the GNU Public License.

In the following statement, *This software* refers to the Mitsumi CD-ROM driver: This software was developed by Holger Veit and Brian Moore for use with 386BSD and similar operating systems. *Similar operating systems* includes mainly non-profit oriented systems for research and education, including but not restricted to NetBSD, FreeBSD, Mach (by CMU).

This product includes software developed by the Apache Group for use in the Apache HTTP server project (http://www.apache.org/).

This product includes software licensed from Richard H. Porter under the GNU Library General Public License (© 1998, Red Hat Software), www.gnu.org/copyleft/lgpl.html.

This product includes the standard version of Perl software licensed under the Perl Artistic License (© 1997, 1998 Tom Christiansen and Nathan Torkington). All rights reserved. You may find the most current standard version of Perl at http://www.perl.com.

This product includes software developed by Jared Minch.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

This product contains software based on oprofile, which is protected under the GNU Public License.

This product includes RRDtool software developed by Tobi Oetiker (http://www.rrdtool.com/index.html) and licensed under the GNU General Public License.

This product contains software licensed from Dr. Brian Gladman under the GNU General Public License.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

This product includes Hypersonic SQL.

This product contains software developed by the Regents of the University of California, Sun Microsystems, Inc., Scriptics Corporation, and others.

This product includes software developed by the Internet Software Consortium.

This product includes software developed by Nominum, Inc. (http://www.nominum.com).

This product contains software developed by Broadcom Corporation, which is protected under the GNU Public License.

This product contains software developed by MaxMind LLC, and is protected under the GNU Lesser General Public License, as published by the Free Software Foundation.

This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory. Copyright ©1990-1994 Regents of the University of California. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- 2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- **3.** All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by the Computer Systems Engineering Group at Lawrence Berkeley Laboratory.
- 4. Neither the name of the University nor of the Laboratory may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes software developed by Sony Computer Science Laboratories Inc. Copyright $^{\odot}$ 1997-2003 Sony Computer Science Laboratories Inc. All rights reserved. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- 1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- **2.** Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY SONY CSL AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL SONY CSL OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 1

REST

- What's new in iControl REST for version 11.6
- About Representational State Transfer
- About URI format
- About HTTP method semantics
- About JSON format
- About API versions
- About iControl and RBAC for user accounts

What's new in iControl REST for version 11.6

This release of the iControl[®] REST User Guide covers new features of iControl REST version 11.6. New iControl REST general features in this version include:

• iControl REST supports multiple transactions per user.

New Application Security Manager[®] (ASM[®]) features in this version include:

- ASM supports the /example endpoint.
- ASM supports additional tasks to perform vulnerability assessment and resolution.
- · ASM supports the REST File Transfer Worker for file upload or download.

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.

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.

About URI format

The iControl[®] REST API enables the management of a BIG-IP[®] device by using web service requests. A principle of the REST architecture describes the identification of a resource by means of a Uniform Resource Identifier (URI). 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 REST architecture, the system configuration is the representation of a resource. 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.

In iControl REST, the URI structure for all requests includes the string /mgmt/tm/ to identify the namespace for traffic management. Any identifiers that follow the endpoint are considered to be resource collections.

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

https://management-ip/mgmt/tm/module

The URI in the previous example designates all of the tmsh subordinate modules and components in the specified *module*. iControl REST refers to this entity as an *organizing collection*. An organizing collection contains links to other resources. The *management-ip* component of the URI is the fully qualified domain name (FQDN) or IP address of a BIG-IP device.

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.

For example, use the following URI to access all of the components and subordinate modules in the ltm module:

https://192.168.25.42/mgmt/tm/ltm

The URI in the following example designates all of the subordinate modules and components in the specified *sub-module*. iControl REST refers to this entity as a collection; a collection contains resources.

https://management-ip/mgmt/tm/module/sub-module

The URI in the following example designates the details of the specified *component*. The *Traffic Management Shell (tmsh) Reference* documents the hierarchy of modules and components, and identifies details of each component. iControl REST refers to this entity as a resource. A resource may contains links to sub-collections.

https://management-ip/mgmt/tm/module[/sub-module]/component

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 transactions, iControl[®] REST does not define any additional 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 an operation.

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 custom query parameters. To distinguish them 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 operation. Also supports query strings.
POST	For both collections and resources, iControl REST supports the POST operation.
DELETE	For collections, iControl REST does not support the DELETE operation. For resources, iControl REST supports the DELETE operation.
PUT	For collections, iControl REST does not support the PUT operation. For resources, iControl REST partially supports the PUT operation.
PATCH	For collections, iControl REST does not support the PATCH operation. For resources, iControl REST supports the PATCH operation.

About JSON format

iControl[®] REST formats a response to a request in JavaScript Object Notation (JSON) format. When iControl REST processes a GET request, it generates a response code and a text body in JSON format. To indicate the format of the text body in a response, the HTTP Content-Type header indicates the content type as application/json. Likewise, an error response contains additional descriptive text in JSON format. A

response from iControl REST contains properties. Properties can 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. For a GET request, the properties consist of JSON objects or arrays, or both. JavaScript Object Notation (JSON) defines a data interchange format that facilitates communication between clients and servers. 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.

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" }

Both the name and value appear in double quotes (" "), and a colon (:) separates the name and the value in the string. 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 for a property is a JSON array, which is an ordered list of values, as shown:[{ "components":8, "security":"disabled" }]In JSON structure, square brackets ([]) enclose the objects in an array. Any objects between the square brackets 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

Many of the examples in this guide use curl, a common Unix utility for accessing URIs, as shown in the example to make an iControl REST call on a particular BIG-IP device. The curl utility uses the following general syntax:

curl -k -u username:password -X http-method uri

pairs can be thought of as property name and property value.

Where:

- *username* and *password* specify the credentials of a BIG-IP system administrator, such as admin:admin;
- http-method specifies a verb, such as GET, POST, PUT, or DELETE ; and
- *uri* identifies the BIG-IP system resource (for example, 192.168.25.42/mgmt/tm/ltm).

A response from iControl REST shows the properties and reference links as JSON objects and arrays.

```
"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.

```
"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.
      "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.

About iControl and RBAC for user accounts

iControl[®] REST supports role-based access control (RBAC). With RBAC, you can enable either broad access to resources or finely granular access to resources, depending on your needs. To simplify the management of a BIG-IP[®] user account created to access iControl REST resources, you can assign a role that grants permissions to all iControl REST resources.

Associating a role with a user account

Before you assign the iControl[®]REST API user role to an account, you must create a BIG-IP[®] user account. Use Traffic Management Shell (tmsh), the BIG-IP Configuration utility, or the BIG-IQTM system GUI to create a user account and assign the appropriate MCP role, such as manager.

As an administrator, you can create a user account for the specific purpose of accessing the iControl REST API. If you create a user account for this purpose, you must also assign a role to grant the account access to iControl REST resources.

Note: A user account must have administrative level access to the iControl[®] REST namespace to make iControl REST requests.

1. To create a user account, use tmsh as shown in the following example.

```
(tmos)# create iCR-user01 partition-access add { all-partitions { role manager } } password
p4ssw0r6
```

2. To get the properties of the user account, make a GET request for all users, as shown in the following example.

GET https://localhost/mgmt/shared/authz/users

3. To get the reference to add, locate the user account in the response data and copy the value of the selfLink property.

"selfLink" : "https://localhost/mgmt/shared/authz/users/iCR-user01"

4. To add the user account to the role, use the PATCH method and specify the userReferences property with the link to the user account in the JSON body.

PATCH https://localhost/mgmt/shared/authz/roles/iControl REST API User

{ "userReferences":[{"link":"https://localhost/mgmt/shared/authz/users/iCR-user01"}] }

The results of the previous operation show the changes to the userReferences array.

```
"name": "iControl REST API User",
  "userReferences": [
    {
      "link": "https://localhost/mgmt/shared/authz/users/iCR-user01"
    }
 ],
"resources": [
      "resourceMask": "/mgmt/tm/gtm/*/*/*",
      "restMethod": "GET"
    },
    {
      "resourceMask": "/mgmt/tm/auth/*/*/*",
      "restMethod": "PUT"
    },
. . .
],
 "generation": 3,
  "lastUpdateMicros": 1.4049301550701e+15,
  "kind": "shared:authz:roles:rolesworkerstate",
  "selfLink": "https://localhost/mgmt/shared/authz/roles/iControl REST API User"
```

After you verify the changes to the role, make sure that the account can successfully make iControl REST requests.

Chapter

GET requests

• Discovering modules and components

Discovering modules and components

The top-level modules in tmsh form the basis of discovery for components in tmsh. iControl[®] REST supports a subset of more than 600 components that exist in tmsh. The *Traffic Management Shell (tmsh) Reference* lists all tmsh modules and components, most of which appear under the same names in tmsh.

The tmsh root modules are:

- actions
- analytics
- apm
- asm
- auth
- cli
- cm
- gtm
- ltm
- net
- pem
- security
- sys
- transaction
- util
- vcmp
- wam
- wom

If you are familiar with command-line tools, use curl, or a similar utility, to make a request to iControl REST and specify a 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.

To discover the structure, make a request to iControl REST with the GET method and specify an organizing collection, as shown in this example. If you expand the links in the response to this request, you can discover the structure of iControl REST.

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

```
},
   "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"
```

ī

```
}
   },
   {
      "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"
```

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

This example expands one of the links in the response from the previous request by making another GET request to iControl REST for a collection.

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

```
},
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/dest-addr?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/global-settings?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/hash?ver=11.5.0"
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/msrdp?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/sip?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/source-addr?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/ssl?ver=11.5.0"
   }
},
{
   "reference":{
      "link":"https://../mgmt/tm/ltm/persistence/universal?ver=11.5.0"
   }
}
```

1

]

Note: Some JSON formatting tools employ different algorithms to sort names, which affects the order in which properties are displayed.

Make a GET request and specify one of the links in the response from the previous request. The response contains the properties of the sub-collection.

GET https://192.168.25.42/mgmt/tm/ltm/persistence/global-settings

```
"kind":"tm:ltm:persistence:global-settings:global-settingsstate",
"selfLink":"https://../mgmt/tm/ltm/persistence/global-settings?ver=11.5.0",
"destAddrLimitMode":"timeout",
"destAddrMax":2048,
"proxyGroup":"/Common/aol"
```

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
Şfilter	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.*

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

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

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 <code>\$skip</code> 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.

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 <code>\$top</code> query parameter to the URI. This query displays the first four items in the <code>sys</code> collection output. The response indicates the <code>nextLink</code> and <code>previousLink</code> 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" } }
 1,
 "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 <code>\$skip</code> query parameter to the URI. This example displays the next four items in the <code>sys</code> collection output. The response also indicates the <code>nextLink</code> and <code>previousLink</code> 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,
"tm:svs:sv
  "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 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",
"sourcePort":"not-a
   "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_17_policy__my-VS",
            "partition": "Common",
            "fullPath":"/Common/asm_auto_17_policy_my-VS",
            "generation":1,
"selfLink":"https://../~Common~my-VS/policies/~Common~asm auto 17 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"
             1,
      "link" : "https://../tm/ltm/pool/~Common~my-Pool/members?ver=11.5.0"
    },
  "minActiveMembers" : 0,
  "minUpMembers" : 0,
  "minUpMembersAction" : "failover",
  "minUpMembersChecking" : "disabled",
```



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
      }
   }
]
```
Obtaining statistical output

The response to a GET request contains the JSON representations that are equivalent to the output of the tmsh list command. iControl[®] REST includes a suffix for statistical information that produces statistical output equivalent to the tmsh show command.

- 1. Obtain statistical results for a resource by specifyi the resource of interest in the URI.
- 2. Append the endpoint stats to the URI of the resource to obtain statistical output.

To obtain statistics for a resource, this example queries the /Common/my-Pool object for current statistics. The response that contains the statistical output appears in the second block.

```
GET https://192.168.25.42/mgmt/tm/ltm/pool/~Common~my-Pool/stats
```

```
"kind":"tm:ltm:pool:poolstats",
"generation":1,
"selfLink":"https://../mgmt/tm/ltm/pool/~Common~my-Pool/stats?ver=11.5.0",
"entries":{
   "activeMemberCnt":{
      "value":0
   },
   "connqAll.ageEdm":{
      "value":0
   },
   "connqAll.ageEma":{
      "value":0
   "value":0
   },
   "connqAll.ageMax":{
      "value":0
   },
   "connqAll.depth":{
      "value":0
   },
   "connqAll.serviced":{
      "value":0
   },
   "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
   },
   "minActiveMembers":{
```

```
"value":0
      },
      "monitorRule":{
        "description": "none"
      },
      "tmName":{
        "description":"/Common/my-Pool"
      },
      "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
     }
   }
             For comparison, this is the response from the tmsh command to display statistical output.
root@(BIG-IP1)(...)(tmos)# show ltm pool my-Pool
                   _____
Ltm::Pool: my-Pool
_____
                         _____
Status
 Availability : unknown
 State : enabled
Reason : The children pool member(s) either don't have service checking enabled, or
service check results are not available yet
 Monitor : none
 Minimum Active Members : 0
 Current Active Members : 0
         Total Requests : 0
       Current Sessions : 0
Traffic
                                        ServerSide
 Bits In
                                                 0
                                                 0
 Bits Out
 Packets In
                                                 0
                                                 0
 Packets Out
Current Connections
                                                 0
```

1

}

Maximum Connections	0		
Total Connections	0		
	D 1		
Connection Queue	POOL	Pool and members	
Number of connections queued now	0	0	
Number of connections serviced	0	0	
Queue head entry age (ms)	0	0	
Maximum queue entry age ever (ms)	0	0	
Maximum queue entry age recently (ms)	0	0	
Average queue entry age (ms)	0	0	
root@(BIG-IP1)()(tmos)#			

Chapter

POST and PUT requests

- About JSON format for POST and PUT
- Adding and modifying resources

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.

Adding and modifying resources

The iControl[®] REST API enables you to add resources to a BIG-IP[®] system. To add a resource, use the POST method on an iControl[®] REST collection and specify the resource to create as a JSON body. When you create a resource, iControl[®] REST sets all unspecified properties to their default values.

- 1. To add a new configuration object to a collection by specifying the path to the collection in the URI.
- 2. Specify the name of the object to create, as a property name/value pair.
- 3. Add one or more resources to the parent sub-collection, as needed.

To demonstrate the use of POST, this example creates a new pool in the pool collection. The response to the request is shown in the second block. Note that iControl[®] REST sets all of the default properties for the configuration object.

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X POST -d \
    '{"name":"tcb-pool","members":[ \
    {"name":"192.168.25.32:80","description":"first member"} ]' \
    https://192.168.25.42/mgmt/tm/ltm/pool
```

The JSON body contains the name/value pairs for the name, partition, and pool members.

```
{"name":"tcb-pool2", "partition":"Common", "members":[{"name":"192.168.25.32:80",
"description":"Web server"}]}
```

The response to the request shows the properties for the tcb-pool2 resource, including a link to the sub-collection.

```
{
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool2",
   "partition":"Common",
"fullPath":"/Common/tcb-pool2",
   "generation":57,
   "selfLink": "https://localhost/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://localhost/mgmt/tm/ltm/pool/~Common~tcb-pool2/members?ver=11.5.0",
      "isSubcollection":true
   }
```

Modifying a configuration object

The PUT method allows modifications to properties of a configuration object without affecting any other properties.

- 1. To modify a configuration object by specifying the configuration object itself in the URI. Do not specify a collection.
- 2. Specify the properties to modify as name/value pairs in the JSON body.

To modify a resource, make a PUT request with a JSON body. The response that appears in the second block illustrates the changes to the properties.

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X PUT -d \
    '{"name":"tcb-pool","description":"backup web \
```

```
servers", "ignorePersistedWeight": "enabled"}' \
   https://192.168.25.42/mgmt/tm/ltm/pool/tcb-pool
ł
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool",
  "fullPath":"tcb-pool",
   "generation":2085,
   "selfLink":"https://../mgmt/tm/ltm/pool/tcb-pool?ver=11.5.0",
   "allowNat":"yes"
   "allowSnat":"yes",
   "description": "backup member",
   "ignorePersistedWeight":"enabled",
   "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-pool/members?ver=11.5.0",
      "isSubcollection":true
   }
```

To modify a resource, make a PATCH request with a JSON body. The response that appears in the second block illustrates the changes to the properties.

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

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X PUT -d \
    '{"name":"tcb-pool2","member": [ {:name":"192.168.25.32:80", "description":"Tertiary web
   server"}] }' \
    https://192.168.25.42/mgmt/tm/ltm/pool/tcb-pool2
```

```
"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
}
```

Modifying a collection

Configuration objects can contain collections, denoted by square brackets in JSON format. For example, an LTM[®] pool has a collection of member objects, and a rule-list (in the security firewall module) has a collection of rule objects. The actions of a PUT method depend greatly on the Request-URI. When you use the PUT method with a collection, only the properties that you specify in the JSON body will exist in the modified collection. In other words, the PUT method overwrites any existing properties of the collection. Therefore, if you use the PUT method to modify one member object of an LTM pool that originally had three members, the request reduces the collection to just the single member.

- 1. To modify a collection, specify the collection in the URI.
- 2. Specify the properties of the collection, as name/value pairs in the JSON body.

To make modifications to the sub-collection, make a PUT request to add a primary member and a backup member to a sub-collection. The semantics of a PUT request replace the member collection if you do not specify any existing members in the JSON body. If the member collection already exists, this request overwrites the properties of the sub-collection.

```
curl -k -u admin:admin -H "Content-Type: \
    application/json" -X PUT -d \
'{"name":"tcb-pool","members":[
     {"name":"192.168.25.32:80","description": "primary member"}, \
{"name":"192.168.25.33:80","description":"backup member}]}' \
    https://192.168.25.42/mgmt/tm/ltm/pool/tcb-pool
   "kind":"tm:ltm:pool:poolstate",
   "name":"tcb-pool",
   "fullPath":"tcb-pool",
   "generation":258,
   "selfLink":"https://localhost/mgmt/tm/ltm/pool/tcb-pool?ver=11.5.0",
   "allowNat":"yes"
   "allowSnat": "yes",
   "description":"# Paging server #"
   "ignorePersistedWeight":"disabled",
   "ipTosToClient":"pass-through",
   "ipTosToServer":"pass-through"
   "linkQosToClient":"pass-through",
"linkQosToServer":"pass-through",
```

"link":"https://../mgmt/tm/ltm/pool/~Common~tcb-pool/members?ver=11.5.0",

"loadBalancingMode": "round-robin",

"minUpMembersAction":"failover",
"minUpMembersChecking":"disabled",

"isSubcollection":true

"queueOnConnectionLimit":"disabled",

"minActiveMembers":0,
"minUpMembers":0,

"queueDepthLimit":0,

"queueTimeLimit":0,
"reselectTries":0,
"slowRampTime":10,
"membersReference":{

45

T } The changes to the members sub-collection appear as a link in the response in iControl® REST. To view the changes, either make a GET request and specify the member collection in the URI. GET https://192.168.25.42/mgmt/tm/ltm/pool/tcb-pool/members "kind":"tm:ltm:pool:members:memberscollectionstate", "selfLink":"https://localhost/mgmt/tm/ltm/pool/tcb-pool/members?ver=11.5.0", "items":["kind":"tm:ltm:pool:members:membersstate", "name":"192.168.25.32:80", "partition": "Common", "fullPath":"/Common/192.168.25.32:80", "generation":2133, "selfLink":"https://../pool/tcb-pool/members/~Common~192.168.25.32:80?ver=11.5.0", "address":"192.168.25.32", "connectionLimit":0, "description": "primary member", "dynamicRatio":1, "inheritProfile":"enabled", "logging":"disabled", "monitor":"default", "priorityGroup":0, "rateLimit":"disabled", "ratio":1, "session":"user-enabled", "state":"unchecked" }, { "kind":"tm:ltm:pool:members:membersstate", "name":"192.168.25.33:80", "partition": "Common", "fullPath":"/Common/192.168.25.33:80", "generation":2133, "selfLink":"https://../pool/tcb-pool/members/~Common~192.168.25.33:80?ver=11.5.0", "address":"192.168.25.33", "connectionLimit":0, "description": "backup member", "dynamicRatio":1, "inheritProfile":"enabled", "logging":"disabled", "monitor":"default", "priorityGroup":0, "rateLimit": "disabled", "ratio":1, "session":"user-enabled", "state":"unchecked" }]

Modifying a single object in a collection

A subcollection permits direct access to individual items, eliminating the need to specify all members of the collection when making a single change.

Specify the individual item, by name, to modify an item in a collection.



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":<sup>"</sup>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 with the 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 \$filterquery parameter applies to GET requests, whereas the partition property applies to 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 a configuration object

The HTTP DELETE method removes single resources from a BIG-IP[®] system configuration. The semantics of the DELETE method support the deletion of a resource, but iControl[®] REST does not support the use of DELETE to remove collections.

To determine the resource to delete, use the GET method to confirm that the resource you want to delete currently exists.

This example uses the GET method to get the configuration from a BIG-IP[®] system. In this case, the resource exists in the $/fw_objs$ folder. The response to the request is shown in the second block.

```
GET https://192.168.25.42/mgmt/tm/ltm/pool/~fw objs~tcb-pool2
```

```
"kind":"tm:ltm:pool:poolstate",
"name":"tcb-pool2",
"partition":"fw objs"
"fullPath":"/fw_objs/tcb-pool2",
"generation":1,
"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
}
```

This example uses the DELETE method to delete the resource. iControl[®] REST generates a response code but no other output for a successful delete request.

DELETE https://192.168.25.42/mmt/tm/ltm/pool/~fw objs~tcb-pool2

Chapter 4

Partitions

• About administrative 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.

Adding an administrative partition

To add an administrative partition, you must have the Administrator or Resource Administrator role associated with your user account. Every administrative partition has an associated folder, and the sys/folder namespace indicates a partition to iControl[®] REST.

Important: You must assign user permissions to the partition through a separate request to iControl[®] REST.

Use the POST operation to create to a new partition in the sys/folder module. Specify the folder name, including the leading slash (/) character, as the value of the name property in the JSON body.

This iControl[®] REST request adds the fw_objs partition to a BIG-IP[®] system configuration. The second code block contains the response to the request.

```
"generation":2014,
"selfLink":"https://localhost/mgmt/tm/sys/folder/~fw_objs?ver=11.5.0",
"deviceGroup":"none",
"hidden":"false",
"inheritedDevicegroup":"true",
"inheritedTrafficGroup":"true",
"noRefCheck":"false",
"trafficGroup":"/Common/traffic-group-1"
}
```

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
```

Chapter **5**

Transactions

• About the iControl REST transaction model

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.

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":30,
"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. If name>/mgmt/tm/transaction/<transId>, where transId is the identifier for the transaction. If you do not add a command to a transaction within thirty (30) 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":30,
"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 and specify the X-F5-REST-Coordination-ID HTTP header with the transaction ID value. 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

Chapter 6

Commands

• About other tmsh global 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
```

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, along with a JSON body that specifies the name of the command.

Install and update components using the install command by making an $iControl^{$ [®] REST request with the POST method and a JSON body.

To install and update components using the install command, make a POST request. In the JSON body, specify the command, image, and volume.

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

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

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 came case naming convention of iControl[®] REST.

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

```
{
    "command":"install",
    "options":[
        {
            "create-volume":true
        },
        {
            "reboot":true
        }
    ],
    "name":"BIGIP-11.4.0.737.400.42.iso",
    "volume":"HD1.1"
}
```

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"

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"
    }
 1,
 "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"

{

}

Chapter

Application Security Manager

- Application Security Manager and iControl REST comparison
- Application Security Manager policy
- Application Security Manager signatures
- Application Security Manager schema upload
- Application Security Manager schemas and profiles
- Application Security Manager policy restore
- Application Security Manager vulnerability import
- Application Security Manager vulnerability resolution
- About the example suffix

Application Security Manager and iControl REST comparison

If you use Application Security Manager^{$^{\text{TM}}$} (ASM^{$^{\text{TM}}$}), you should understand how ASM differs from iControl REST.

Application Security Manager^{$^{\text{TM}}$} (ASM^{$^{\text{TM}}$}) 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 <i>sfilter</i> 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 <i>filter</i> 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.
РАТСН	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 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
Şfilter	Specifies a filter for a retrieve, update, or delete operation. In ASM, <code>\$filter</code> supports the contains, endswith, startswith, and substringofstring 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.

Parameter	Description
Şorderby	Specifies the order in which to display items. The <code>Sorderby</code> parameter cannot be applied to a subfield inside of an expanded field, such as
	$\verb+ Sorderby=requestPolicy/name on /tm/asm/events/requests.$

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.
or	True if either operand is true. In ASM, Sfilter 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.

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</md5hash>	
	 /tm/asm/policies/<md5hash>/filetypes</md5hash> 	
	 /tm/asm/policies/<md5hash>/cookies</md5hash> 	
	 /tm/asm/policies/<md5hash>/host-names</md5hash> 	
	 /tm/asm/policies/<md5hash>/blocking-settings/violations</md5hash> 	
	 /tm/asm/policies/<md5hash>/blocking-settings/evasions</md5hash> 	
	 /tm/asm/policies/<md5hash>/blocking-settings/http-protocols</md5hash> 	
	• /tm/asm/policies/ <md5hash>/blocking-settings/web-services-securities</md5hash>	
	 /tm/asm/policies/<md5hash>/urls</md5hash> 	
	• /tm/asm/policies/ <md5hash>/parameters</md5hash>	
	• /tm/asm/policies/ <md5hash>/urls/<md5hash>/parameters</md5hash></md5hash>	
	• /tm/asm/policies/ <md5hash>/whitelist-ips</md5hash>	

Namespace Descri	Description		
• /tm/	asm/policies/ <md5hash>/gwt-profiles</md5hash>		
• /tm/	asm/policies/ <md5hash>/json-profiles</md5hash>		
• /tm/	asm/policies/ <md5hash>/xml-profiles</md5hash>		
• /tm/	asm/policies/ <md5hash>/signatures</md5hash>		
• /tm/	asm/policies/ <md5hash>/signatures-sets</md5hash>		

Retrieving Application Security Manager resources

Consistent with iControl[®] REST behavior, Application Security Manager^M (ASM^M) 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"
```
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-Fqt4trP6A/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-Fqt4trP6A",
         "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"
   1,
   "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-Fqt4trP6A/qwt-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^M (ASM^M) 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
      }
   1
   "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^M (ASM^M) 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^M (ASM^M) 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.

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.

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

Importing a policy in Application Security Manager

iControl[®] REST supports the Application Security Manager^M (ASM^M) 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>

- 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.

Exporting a policy in Application Security Manager

iControl[®] REST supports the Application Security Manager^M (ASM^M) 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",
   "endTime": "2013-06-05T15:14:36-04:00",
   "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 ManagerTM (ASMTM) 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": {
    "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"
}
```

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^M (ASM^M) 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 ManagerTM (ASMTM) 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^{$^{\text{M}}$} (ASM^{$^{\text{M}}$}) 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",
    "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^{$^{\text{M}}$} (ASM^{$^{\text{M}}$}) 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^M (ASM^M) 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 ManagerTM (ASMTM) 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^M (ASM^M) 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": 139333202000000,
"id": "d7loGosItLc_ODXuPz83Uw",
"isReferenced": false
```

Application Security Manager schemas and profiles

If you use Application Security Manager[™] (ASM[™]) to manage profiles, you should understand how iControl[®] REST implements ASM.

iControl[®] REST provides an endpoint to associate the XML schema files that you upload to a security profile.

{

Property	Description
primaryXmlValidationFileReference	Specifies the primary validation file to associate with the profile.
xmlValidationFileReferences	Specifies the supporting XML schema files to associate with the profile.
xmlProfileReference	Specifies the profile for which to import the schema files.
followSchemaLinks	Indicates whether ASM will retrieve external schema resources to process the validation files.

Associating schemas to profiles in Application Security Manager

You must upload the validation files before you can associate schemas to profiles in Application Security Manager^M (ASMTM).

iControl[®]REST supports the Application Security ManagerTM (ASMTM) task to associate a schema with a policy to validate an input document. ASM can validate an input document based on an XML schema.

1. To associate the XML schema file to a profile, use the POST method with the /tm/asm/tasks/associate-xml-validation-files-to-xml-profile namespace.

```
POST https://192.168.25.42/mgmt/tm/asm/tasks/associate-xml-validation-files-to-xml-profile
```

```
"primaryXmlValidationFileReference": {
```

```
"link":
```

],

"kind":

"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/1111111111111111"

```
},
"xmlValidationFileReferences": [
    {"link":
```

"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}

```
"xmlProfileReference": {
    "link":
```

"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-profile/abcdef1234567890" },

```
"followSchemaLinks": true
```

```
' "id": "oqNah2PxtwwE4YyAHGekNQ",
    "primaryXmlValidationFileReference": {
    "link":
    "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/111111111111
    },
    "xmlValidationFileReferences": [
    {"link":
    "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "xmlProfileReference": {
        "link":
    "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "xmlProfileReference": {
        "link":
    "https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-profile/abcdef1234567890"
    ],
    "state="text-align: center;">"tttps://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "state="text-align: center;">"tttps://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "state="text-align: center;">"tttps://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "state="text-align: center;">"tttps://state="text-align: center;">"tttps://state="text-align: center;" text-align: cente
```

```
"tm:asm:tasks:associate-xml-validation-files-to-xml-profile:associatexmlvalidationfilestoxmlprofilestate",
    "lastUpdateMicros": 1370459676272126,
    "status": "NEW",
    "selfLink":
    "https://localhost/mgmt/tm/asm/tasks/associate-xml-validation-files-to-xml-profile/oqNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00"
}
```

2. To retrieve the status of the profile association operation, use the GET method and specify the id from the previous result.

GET https://192.168.25.42/mgmt/tm/asm/tasks/associate-xml-validation-files-to-xml-profile/oqNah2PxtwwE4YyAHGekNQ

```
{
    "id": "oqNah2PxtwwE4YyAHGekNQ",
    "kind":
"tm:asm:tasks:associate-xml-validation-files-to-xml-profile:associatexmlvalidationfilestoxmlprofilestate",
    "primaryXmlValidationFileReference": {
   "link":
"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/1111111111111111"
     },
    "xmlValidationFileReferences": [
   {"link":
"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-validation-files/abcdef1234567890"}
    ],
    "xmlProfileReference": {
        "link":
"https://localhost/mgmt/tm/asm/policies/vagoQLF6uOoBKvS8h3C19w/xml-profile/abcdef1234567890"
    ł,
    "followSchemaLinks": true,
"lastUpdateMicros": 1370459676272126,
    "status": "COMPLETED",
    "selfLink":
"https://localhost/mgmt/tm/asm/tasks/associate-xml-validation-files-to-xml-profile/ogNah2PxtwwE4YyAHGekNQ",
    "startTime": "2013-06-05T15:14:36-04:00",
    "endTime": "2013-06-05T15:14:56-04:00",
    "result": {
        "missingFiles": []
```

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 ManagerTM (ASMTM) 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/hGKdiXU7US4S4gtgexijUQ"
    },
    "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^{$^{\text{TM}}$} (ASM^{$^{\text{TM}}$}) 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/xpqb01mY0tgfv13j1khKeA"
},
```

```
"importAllDomainNames": false,
    "domainNames": [
        .....
    ],
    "subscriptionId": "4132",
    "scanId": "3883"
{
   "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/xpqb01mY0tgfv13j1khKeA"
},
   "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/xpqbOlmYOtgfv13j1khKeA"
    },
    "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 ManagerTM (ASMTM) 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/xpqb01mY0tgfv13j1khKeA"
}
```

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/xpqb01mY0tgfv13j1khKeA"
},
"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",

```
"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.

 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/xpqb01mY0tgfv13j1khKeA"
    "subscriptionId": "4132"
}
```

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

```
{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/xpqbOlmYOtgfv13j1khKeA"
    },
        "status": "NEW",
        "lastUpdateMicros": 139556785900000,
        "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/xpqb01mY0tgfv13j1khKeA"
    },
    "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.

 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/xpqb01mY0tgfv13j1khKeA"
 },
}

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

```
{
    "policyReference": { "link": "https://localhost/mgmt/tm/asm/policies/xpqbOlmYOtgfv13j1khKeA"
},
    "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": 139556785900000,
    "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/xpqb01mY0tgfv13j1khKeA"
    },
    "id": "8PacFCQc0Umx45mheqdyew",
    "startTime": "2014-03-23T09:44:15Z",
    "result": {
     }
}
```

Application Security Manager vulnerability resolution

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

Application Security Manager^M (ASM^M) 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^M (ASM^M) 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.

 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"}
    ]
}
```



```
"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

 $\label{eq:limit} $$/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": "ogNah2PxtwwE4YyAHGekNO",
    "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^{$^{\text{TM}}$} (ASM^{$^{\text{TM}}$}) task to resolve a vulnerability and obtain the messages that identify a vulnerability, without making changes to the security policy.

 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",
```

About the example suffix

The inclusion of the /example suffix in a URI prompts iControl[®] REST to generate a sample representation of a resource. The /example suffix may be used with resources to produce a representation that includes all properties, including null properties.

The response for Application Security Manager^{TM} (ASM^{TM}) resources contains only default values and possible enumeration values. The response does not contain descriptions of the properties as help text.

The sample representation includes 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.

Tip: Copy the sample representation, make changes to the copy, and then paste the changes into the JSON body of a POST request.

Access Policy Manager

• About Access Policy Manager

About Access Policy Manager

Access Policy Manager^{$^{\text{TM}}$} (APM^{$^{\text{TM}}$}) provides secure identity and access management for a BIG-IP[®] system. iControl[®] REST exposes the APM namespace to enable programmatic access to APM resources and the benefits of automation.

Method	Description
GET	For both collections and resources, APM supports the GET operation to retrieve or search. The <i>ffilter</i> query parameter support in APM follows the general iControl [®] REST support.
POST	For both collections and resources, APM supports the POST operation to create an entity. A POST request must include a JSON body, even if the body is empty.
DELETE	For collections, APM does not support the DELETE operation. For resources, APM supports the DELETE operation.
PUT	For collections, iControl REST does not support the PUT operation. For resources, iControl REST partially supports the PUT operation.
РАТСН	For collections, APM does not support the PATCH operation. For resources, APM supports the PATCH operation. The PATCH method updates specified properties but does not reset or overwrite other properties of a resource.

Endpoint	Description
/tm/apm/aaa	Configure authorization, authentication, and accounting.
/tm/apm/configuration	Specify settings for Secure Web Gateway initialization.
/tm/apm/epsec	Enable client-side and server-side endpoint security checks.
/tm/apm/ntlm	Use NTLM authentication through external servers.
/tm/apm/policy	Configure policy for scheme assignment.
/tm/apm/profile	Configure profile for traffic handling.
/tm/apm/report	Configure settings for reporting.
/tm/apm/resource	Network access and web access resource.
/tm/apm/sso	Configure user name, password, and authentication method for single sign-on (SSO).
/tm/apm/saml	Configure Security Assertion Markup Language (SAML) settings.
/tm/apm/acl	Restrict access to host and port combinations with Access Control Lists (ACLs).
/tm/apm/apm-avr-config	Configure settings for application visibility and reporting.
/tm/apm/log-setting	Specify access policy events and audit logging levels.
/tm/apm/risk-class	Specify risk classes.
/tm/apm/swg-scheme	Configure Secure Web Gateway schemes to filter and categorize URLs.
/tm/apm/url-filter	Configure URL filtering to allow or block URLs.

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=11.5.1",
"items":[
   {
"reference":{
    "link":"https://localhost/mgmt/tm/apm/aaa?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/configuration?ver=11.5.1"
 }
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/epsec?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/ntlm?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/policy?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/profile?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/report?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/resource?ver=11.5.1"
}
   1,
"reference":{
    "link":"https://localhost/mgmt/tm/apm/sso?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/acl?ver=11.5.1"
}
   },
```

```
"reference":{
    "link":"https://localhost/mgmt/tm/apm/apm-avr-config?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/log-setting?ver=11.5.1"
}
   },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/risk-class?ver=11.5.1"
}
  },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/swg-scheme?ver=11.5.1"
}
  },
"reference":{
    "link":"https://localhost/mgmt/tm/apm/url-filter?ver=11.5.1"
}
   }
]
```

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=11.6.0",
  "items": [
    {
        "reference": {
            "link": "https://localhost/mgmt/tm/apm/ntlm/machine-account?ver=11.6.0"
        }
    },
    {
        "reference": {
            "link": "https://localhost/mgmt/tm/apm/ntlm/ntlm-auth?ver=11.6.0"
        }
    }
    ]
```

Creating Access Policy Manager resources

Using iControl[®] REST, you can create Access Policy Manager[®] (APM[®]) resources.

To create an Access Policy Manager (APM) resource, make a POST request and specify the endpoint in which to create the resource.

POST https://192.168.25.42/mgmt/tm/apm/ntlm/machine-account

```
"name": "new-machine-acct",
```

{

```
"domainFqdn": "example.org",
   "administratorName": "administrator",
   "administratorPassword": "pAssword",
   "machineAccountName": "server01",
   "machineAccountPassword": "pAssword"
```

The request creates a new machine account and joins the machine to the domain. Note that this request attempts to contact a domain controller to validate the machine account.

Modifying Access Policy Manager resources

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

To modify a machine account resource, make a PUT request to an endpoint in the /mgmt/tm/apm/ntlm/machine-account namespace.

PUT https://192.168.25.42/mgmt/tm/apm/ntlm/machine-account

```
{
    "name": "new-machine-acct",
    "domainFqdn": "example.org",
    "administratorName": "administrator",
    "administratorPassword": "pAssword",
    "machineAccountName": "server-01",
    "machineAccountPassword": "pAssword"
}
```

This request modifies the machine account resource, and changes the machineAccountName property to server-01.

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.

Chapter

Additional features

- About HTTP response codes
- About log files
- About public URIs
- List of public URIs

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	POST, PUT	Possible causes include:
Туре		 specifying an incorrect Content-Type header value specifying a malformed JSON body with a POST or PUT request
500 Internal Server	all	Possible causes include:
Error		• attempting to access iControl REST when the process is not running

Response code	HTTP methods	Description
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/tr/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 GTM[™] pool for which you want members, such as /mgmt/tm/gtm/pool/pool5/members for the members of pool5.

List of public URIs

iControl[®] REST contains these public URIs:

- /mgmt/tm/apm/aaa/oam/accessgates
- /mgmt/tm/apm/aaa/securid/config-files
- /mgmt/tm/apm/policy/agent/endpoint-check-software/items
- /mgmt/tm/apm/policy/customization-group/templates
- /mgmt/tm/apm/profile/access/domain-groups
- /mgmt/tm/apm/profile/connectivity/client-policy
- /mgmt/tm/apm/resource/app-tunnel/apps
- /mgmt/tm/apm/resource/client-traffic-classifier/entries
- /mgmt/tm/apm/resource/network-access/optimized-app
- /mgmt/tm/apm/resource/portal-access/items

- /mgmt/tm/apm/resource/sandbox/files
- /mgmt/tm/apm/sso/form-basedv2/forms
- /mgmt/tm/auth/remote-role/role-info
- /mgmt/tm/gtm/pool/members
- /mgmt/tm/gtm/server/virtual-servers
- /mgmt/tm/ltm/dns/dnssec/key/generation
- /mgmt/tm/ltm/policy-strategy/operands
- /mgmt/tm/ltm/policy/rules
- /mgmt/tm/ltm/policy/rules/actions
- /mgmt/tm/ltm/policy/rules/conditions
- /mgmt/tm/ltm/pool/members
- /mgmt/tm/ltm/profile/analytics/alerts
- /mgmt/tm/ltm/profile/analytics/traffic-capture
- /mgmt/tm/ltm/profile/rewrite/uri-rules
- /mgmt/tm/ltm/virtual/fw-rules
- /mgmt/tm/net/route-domain/fw-rules
- /mgmt/tm/net/router-advertisement/prefixes
- /mgmt/tm/net/self/fw-rules
- /mgmt/tm/net/wccp/services
- /mgmt/tm/pem/policy/rules
- /mgmt/tm/pem/policy/rules/flow-info-filters
- /mgmt/tm/security/dos/profile/application
- /mgmt/tm/security/dos/profile/protocol-dns
- /mgmt/tm/security/dos/profile/protocol-sip
- /mgmt/tm/security/firewall/global-rules/rules
- /mgmt/tm/security/firewall/management-ip-rules/rules
- /mgmt/tm/security/firewall/policy/rules
- /mgmt/tm/security/firewall/rule-list/rules
- /mgmt/tm/security/log/profile/application
- /mgmt/tm/security/log/profile/network
- /mgmt/tm/security/log/profile/protocol-dns
- /mgmt/tm/security/log/profile/protocol-sip
- /mgmt/tm/sys/application/template/actions
- /mgmt/tm/sys/file/apache-ssl-cert/bundle-certificates
- /mgmt/tm/sys/file/ssl-cert/bundle-certificates
- /mgmt/tm/sys/file/system-ssl-cert/bundle-certificates
- /mgmt/tm/sys/icall/handler/perpetual/subscriptions
- /mgmt/tm/sys/icall/handler/triggered/subscriptions
- /mgmt/tm/sys/ltcfg-class/fields
- /mgmt/tm/sys/ltcfg-instance/fields
- /mgmt/tm/sys/ntp/restrict
- /mgmt/tm/sys/snmp/communities
- /mgmt/tm/sys/snmp/traps
- /mgmt/tm/sys/snmp/users
- /mgmt/tm/wam/policy/nodes
- /mgmt/tm/wam/policy/nodes/invalidations
- /mgmt/tm/wam/policy/nodes/invalidations/cache-content
- /mgmt/tm/wam/policy/nodes/invalidations/request

- /mgmt/tm/wam/policy/nodes/matching
- /mgmt/tm/wam/policy/nodes/proxy
- /mgmt/tm/wam/policy/nodes/proxy-override
- /mgmt/tm/wam/policy/nodes/substitutions
- /mgmt/tm/wam/policy/nodes/variation
Index

/example endpoint 96

A

administrative partition about 52 API versions URI about 22 APM Access Policy Manager about 98 **Application Security Manager** differences 70 policy 76, 86 schema 84 signatures 80 vulnerability 88 vulnerability resolution 93 ASM Application Security Manager deleting 76 POST 75 retrieving with GET 72 updating with PATCH 75 ASM policies exporting 78 ASM policy applying 79 importing 77 ASM policy revisions restoring 87 ASM profile associating 85 ASM schema uploading 84 ASM signatures exporting 82 updating 81 ASM vulnerabilities importing 88 resolving 93 ASM vulnerability initiating 91 terminating 92

С

camel case for JSON properties in iControl REST 21 check ASM signatures 80 cp command using 60 creating Access Policy Manager APM 100

D

DELETE with iControl REST 50 deleting Access Policy Manager APM 101 deleting configuration objects with iControl REST 50

Ε

Error codes in iControl REST responses 104 Expanding an iControl REST component limits 33 Expanding an iControl-REST component 34

F

features new 14 filter partition OData 49 Folder See also Partition relative to parent's partition 49 See also Partition format for JSON properties in iControl REST 21

G

generate POST commands 61

Н

HTTP semantics 16

I

iControl about user account roles 22 iControl null values and REST flags about 20 iControl public URIs 106 iControl REST discovering modules and components 26 log files 105 modifying configuration through 43 POST 42 iControl REST properties about 19 icrd log files 105 install POST commands 61

J

JSON getting show content instead of list content 37 JSON format about 16 JSON format POST and PUT about 42

L

load POST commands 62 Logging levels contact Support to change 105 Logs for iControl REST 105

Μ

Modify collection within a config object JSON 45 modifying Access Policy Manager APM 101 Modify single object collection PUT 46 mv command using 62

0

OData pagination 31

Ρ

Paging 31 Partition accessing 36 adding 52 adding or modifying in 48 deleting 53 PATCH, See PUT POST to BIG-IP config, with iControl REST 42 public URIs 106 publish POST commands using 63 PUT in iControl REST 43

Q

query parameters about 30

R

Read-only properties silently ignored in PUT and POST operations 47 reboot POST commands 63 **Representational State Transfer** about 14 reserved ASCII characters about 15 reset-stats POST commands 64 Response codes in iControl REST responses 104 restart POST commands 64 **REST** resource identifiers about 15 retrieving Access Policy Manager APM 99 role iControl REST API user 22 run POST commands 64

S

signatures retrieving 83 signature systems retrieving 83 start POST commands 66 string encoding standards about 22

Т

tmsh global commands, GET about 60 tmsh property names about 21 transaction committing 58 creating 56 modifying 57 transaction atomic requests about 56

U

URI about 15 URI format about 14

V

vulnerabilities resolving 95 vulnerability assessment subscriptions querying 90