



Introduction to TOS REST API

Nir Bar-el

API Project Manager

Tufin

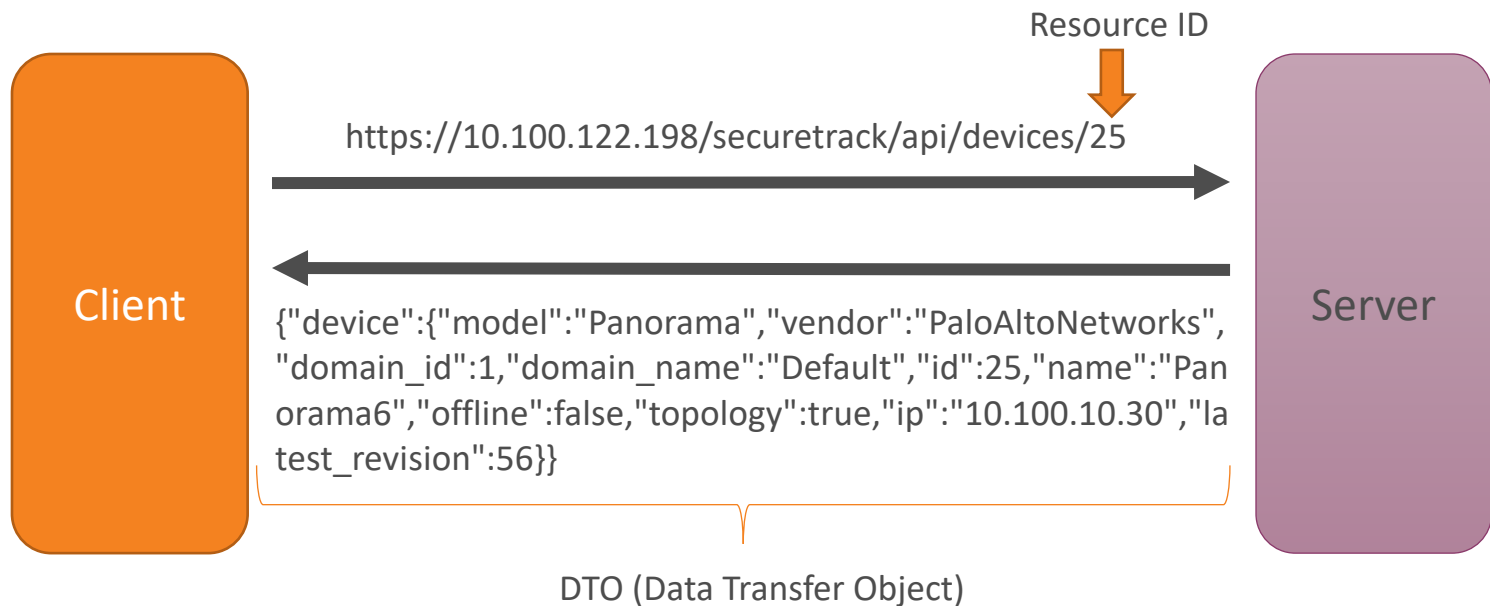
Last modified in June 2023

Introduction to REST API

- An API (Application Programmer Interface) lets programs utilize features from other programs
- Tufin's APIs lets clients automate their security policy tasks and embed these tasks into their custom applications and portals
- REST (REpresentational State Transfer) is an architectural style used for web services (also called **RESTful** Web Services)
- Components used by a REST are called resources
 - ⦿ SecureTrack resources can be devices, rules, network_objects, domains
 - ⦿ SecureChange resources can be tickets, workflow steps
 - ⦿ SecureApp resources can be applications, servers, customers

REST - Data Transfer

- REST uses HTTP (or HTTPS) for data transfer between the client and server
- The client sends an HTTP request with the relevant identifying information (for example a Resource ID)
- The server returns a Data Transfer Object (DTO) which includes the requested information



HTTP Methods used by REST

- **GET** - Provides read only access to a resource

GET	/security_policies/	Get unified security policies
GET	/security_policies/{id:[0-9]+}/export	Get unified security policy as CSV

- **POST** - Creates a new resource

POST	/security_policies/exceptions/	Create an exception
------	--------------------------------	---------------------

- **PUT** - Modifies an existing resource

PUT	/zones/{id:[0-9]+}	Modify a zone
-----	--------------------	---------------

- **PATCH** – Modifies some details of an existing resource

PATCH	/policies/{policyId}	Modify a tag policy
-------	----------------------	---------------------

- **DELETE** - Removes a resource

DELETE	/security_policies/{id:[0-9]+}	Delete unified security policy
--------	--------------------------------	--------------------------------

What is the difference between URI and URL

The terms URI and URL are commonly interchanged

- A **URI** (**U**niform **R**esource **I**dentifier) identifies a specific resource
- **URI** is a general name for two sub-types
 - ◉ **URL** (**U**niform **R**esource **L**ocator) – is a **URI** that includes the protocol
<https://10.100.122.198/securetrack/api/devices/25>
<ftp://ftp.download.com/public>
 - ◉ **URN** (**U**niform **R**esource **N**ame) – is a **URI** without a protocol
<urn:issn:1082-9873>
- Since all Tufin REST APIs start with <https://...> they can be referred to as URLs without making you feel guilty!

Tufin REST URL format

`https://192.168.1.1/securetrack/api/devices/20/rules/16373/documentation`

Prefix

Application

Resource

The Tufin URL is made up of 3 sections:

- Prefix - “**https://**” + the IP address of the TOS server
- Application –
 - SecureTrack – “**/securetrack/api/**”
 - SecureChange – “**/securechangeworkflow/api/securechange/**”
 - SecureApp – “**/securechangeworkflow/api/secureapp/**”
- Resource – Resource ID + any additional **path parameters** (included in path)

In the above example the path parameters are device id (20), and rule id (16373)

URL Query Parameters

The URL may also contain **query parameters**

- separated from the resource section by ‘?’
- separated from each other by ‘&’
- Each parameter has the format **parameter_name=parameter_value**

https://192.168.1.1/securetrack/api/network_objects/search?filter=text&name=host&exact_match=true

In the above network object search API, 3 parameters are provided

- ⦿ ‘filter’ parameter has the value ‘text’
- ⦿ ‘name’ parameter has the value ‘host’
- ⦿ ‘exact_match’ parameter has the value ‘true’

Pagination

- Some of the APIs may return a lot of elements
- These APIs usually support pagination
- Pagination allow the client to receive only part of the data
- Pagination uses two optional parameters, 'start' and 'count'
 - ◉ 'start' indicates the first returned element
 - ◉ 'start' is zero based – to get the second element use start=1
 - ◉ 'count' indicates the number of elements to return

<https://192.168.1.1/securetrack/api/devices/254/rules?start=50&count=10>

- In the above example, the returned data will include rules 51-60
- In most of the APIs, the returned DTO will include a 'total' field
- The client can use this field to figure how many partial calls will be needed

REST Summary

- REST is an architectural style for web services over http (or https)
- REST is used to get/create/change/delete resources like devices, rules or tickets
- REST uses two types of parameter
 - ⦿ Path parameters
https://192.168.1.1/securetrack/api/devices/20/rules/16373/documentation
 - ⦿ Query parameters – starts after ? Delimited by &
https://192.168.1.1/securetrack/api/devices/254/rules?start=50&count=10
- Some APIs support pagination – as specified in the documentation
- The same URL using a different HTTP methods has a different meaning

GET	/devices/{id:[0-9]+}/rules/{rule_id:[0-9]+}/documentation	Get specific rule documentation
PUT	/devices/{id:[0-9]+}/rules/{rule_id:[0-9]+}/documentation	Modify specific rule documentation
DELETE	/devices/{id:[0-9]+}/rules/{rule_id:[0-9]+}/documentation	Delete specific rule documentation

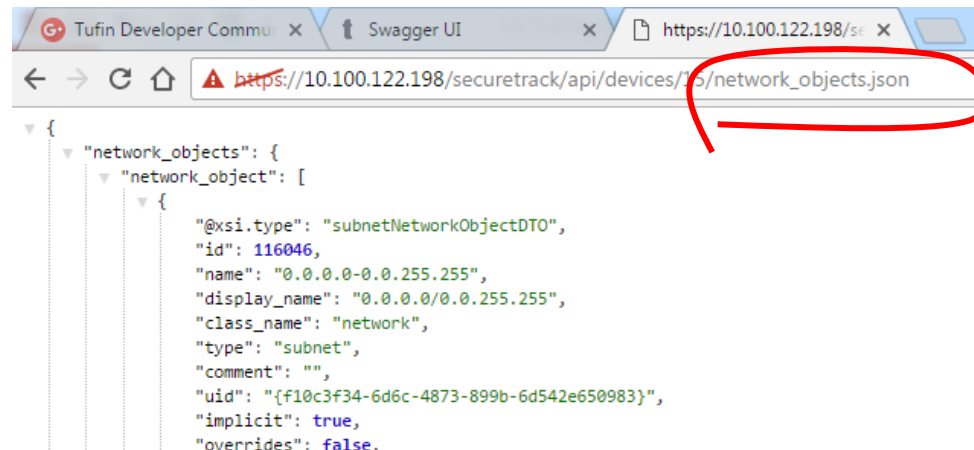
Executing GET REST API

- To execute a GET you can write the URL in the address bar of a browser
 - ◉ You will be prompted to provide a username and password



```
<network_objects>
  <network_object xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:type="subnetNetworkObjectDTO">
    <id>116046</id>
    <name>0.0.0.0-0.0.255.255</name>
    <display_name>0.0.0.0/0.0.255.255</display_name>
    <class_name>network</class_name>
    <type>subnet</type>
    <comment/>
    <uid>{f10c3f34-6d6c-4873-899b-6d542e650983}</uid>
    <implicit>true</implicit>
    <overrides>false</overrides>
  </network_object>
</network_objects>
```

- Add .json to the URL to get the response in JSON format, instead of the default XML
 - ◉ When using query parameters add .json before the ?



```
{
  "network_objects": {
    "network_object": [
      {
        "@xsi.type": "subnetNetworkObjectDTO",
        "id": 116046,
        "name": "0.0.0.0-0.0.255.255",
        "display_name": "0.0.0.0/0.0.255.255",
        "class_name": "network",
        "type": "subnet",
        "comment": "",
        "uid": "{f10c3f34-6d6c-4873-899b-6d542e650983}",
        "implicit": true,
        "overrides": false
      }
    ]
  }
}
```

Executing Other REST API Methods

- As explained REST APIs can be used also for adding, deleting or modifying resources
 - Use POST for adding a resource
 - Use PUT for modifying a resource
 - Use DEL for delete
- POST and PUT require data from the client, this data is called “body”
- The body contains the new value for the created/modified resource
- This data can be provided in either XML or JSON format

Executing Other REST API Methods (continued)

- For executing REST APIs with the other HTTP methods, you will need a REST client
- There are many free REST clients available
- If you choose to use Postman, you can use the Postman collections provided by Tufin
- A ZIP of the Tufin's API collection for Postman can be downloaded from https://forum.tufin.com/support/kc/rest-api/latest/postman-collection/tss_postman_collections.zip
- See [Using Tufin REST API collections in Postman.pdf](#) for more details

REST APIs HTTP status codes

- REST APIs use standard HTTP response status codes
- The response codes indicate whether the operation was successful
- When a REST API has successfully completed, the returned HTTP code will be one of:
 - **200 OK** (returned for GET operations) – The data was successfully retrieved
 - **201 Created** (returned for POST operations) – The resource was successfully added
 - **204 No Content** (returned for PUT and DEL operations) – The resource was successfully modified or deleted
- When a REST API call has failed, the returned HTTP code will usually be one of:
 - 4XX - indicates a client error
 - 5XX - indicates a server error
- Detailed descriptions of the error return codes for each API call can be found in the TOS REST API documentation

TOS REST API documentation

- Available locally on your TOS machine
 - ◉ SecureTrack - <https://192.168.1.1/securetrack/apidoc/>
 - ◉ SecureChange & SecureApp – <https://192.168.1.1/securechangeworkflow/apidoc/>
 - ◉ A link to API documentation is also available in the help menu



- Available online in the Tufin Knowledge Center
 - <https://forum.tufin.com/support/kc/rest-api/latest/securetrack/apidoc/>
 - <https://forum.tufin.com/support/kc/rest-api/latest/securechangeworkflow/apidoc/>
- API documentation for other TOS release are also available online
 - <https://forum.tufin.com/support/kc/rest-api/R22-2/securechangeworkflow/apidoc/>

TOS REST API Documentation (continued)

- To search the API documentation
 - use the three buttons below the main heading to set the expand level
 - Use the browser's VTRRL+F tp search the page

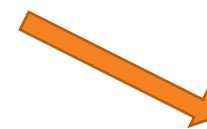
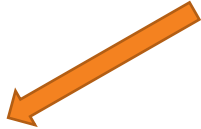
SecureChange/App REST API Documentation

Tufin Technologies

List all APIs

Expand all APIs

Collapse all APIs



SecureChange/App REST API Documentation

Tufin Technologies

List all APIs | Expand all APIs | Collapse all APIs

Access Portal

Method	URL	Description
GET	/repository/applications/(applicationId:[0-9]+)/application_access_requests/(requestId:[0-9]+)	Retrieve an existing application access request by ID
GET	/repository/applications/(applicationId:[0-9]+)/application_access_requests	Retrieve existing application access requests
POST	/repository/applications/(applicationId:[0-9]+)/application_access_requests	Create application access request
PUT	/repository/applications/(applicationId:[0-9]+)/application_access_requests/(requestId:[0-9]+)	Update an application access request
PUT	/repository/applications/(applicationId:[0-9]+)/application_access_requests	Update application access requests

Application Connections

ShowHide | List Operations | Expand Operations | Raw

SecureChange/App REST API Documentation

Tufin Technologies

List all APIs | Expand all APIs | Collapse all APIs

Access Portal

Method	URL	Description		
GET	/repository/applications/(applicationId:[0-9]+)/application_access_requests/(requestId:[0-9]+)	Retrieve an existing application access request by ID		
URL:	/securechangeworkflow/api/secureapp/repository/applications/(applicationId:[0-9]+)/application_access_requests/(requestId:[0-9]+)			
Parameters	Description	Allowed Values	Parameter Type	Data Type
*applicationId	The unique identifier of the application		path	integer
*requestId	The unique identifier of the application access request		path	integer
Usage Example	https://192.168.1.1/securechangeworkflow/api/secureapp/repository/applications/1/application_access_requests/1			
Response Messages	Reason	Response Model		
400	The application was already deleted.			
401	Access is denied.			

SecureChange/App REST API Documentation

Tufin Technologies

List all APIs | Expand all APIs | Collapse all APIs

Access Portal

Section	ShowHide	List Operations	Expand Operations	Raw
Application Connections	ShowHide	List Operations	Expand Operations	Raw
Application Identities	ShowHide	List Operations	Expand Operations	Raw
Application Interfaces	ShowHide	List Operations	Expand Operations	Raw
Application Migration	ShowHide	List Operations	Expand Operations	Raw
Application Packs	ShowHide	List Operations	Expand Operations	Raw
Application Pending Changes	ShowHide	List Operations	Expand Operations	Raw
Application Servers (across all applications)	ShowHide	List Operations	Expand Operations	Raw
Application Servers (by application)	ShowHide	List Operations	Expand Operations	Raw
Application Services (global)	ShowHide	List Operations	Expand Operations	Raw
Application Services (local)	ShowHide	List Operations	Expand Operations	Raw

Tufin's Developers' Community

- REST APIs enable Tufin users to extend TOS and integrate with 3rd parties
- Tufin encourages users to build on top of TOS
- The Tufin Developer Community lets TOS developers get assistance, share their projects, and learn from each other
- Tufin engineers are active on the community, and provide answers to posted questions
- We welcome all feedback!

Please be involved and contribute your ideas

[Tufin's developers' community](#)