



BeyondTrust

Remote Support 24.1 API Programmer's Guide

Table of Contents

Remote Support API Programmer's Guide	6
Version 1.24.1 (for BeyondTrust 24.1.x)	6
CLI Tool	6
Authenticate to the Remote Support API	7
Create a Token	7
Request an API Resource	8
Authentication Errors	8
Request Rate Limits	8
API Use Cases	9
AWS Registration	9
AWS Cleanup	10
Scripting a New Setup	14
Integrate an external Chatbot with Remote Support	17
Configuration API	19
View the Configuration API Documentation in /login	19
Access the YAML file via API	20
Download the YAML file	20
Remote Support Command API	21
API Command: get_logged_in_reps	23
API Command: set_rep_status	25
API Command: get_support_teams	26
API Command: generate_session_key	28
API Command: join_session	31
API Command: transfer_session	32
API Command: send_chat_message	33
API Command: create_virtual_customer	35
API Command: leave_session	37
API Command: set_session_attributes	38
API Command: get_session_attributes	39
API Command: terminate_session	40
API Command: get_appliances	41

API Command: get_connected_client_list	42
API Command: get_connected_clients	44
API Command: check_health	49
Alternative Method - HTTP Status Check	50
API Command: set_failover_role	51
API Command: import_jump_shortcut	53
API Command: get_api_info	61
Representative Console Scripting and Client Scripting API	63
The BeyondTrust Representative Console Script File	63
New and Deprecated Command Line Parameters for the Representative Console	64
The BeyondTrust Client Scripting API	64
Parameters for Client Scripting API	65
API Script Command: login	66
API Script Command: generate_session_key	67
API Script Command: push_and_start_local	68
API Script Command: push_and_start_remote	69
API Script Command: start_jump_item_session	70
API Script Command: start_rdp_session	72
API Script Command: start_vnc_session	74
API Script Command: start_shell_jump_session	75
API Script Command: start_vpro_session	77
Session Generation API	78
Optional Parameters	78
Query Examples	81
Start Sessions with Session Key Acceptance	82
Use JavaScript to Start Click-to-Chat or Full Client Sessions	83
Start Sessions with External Keys (TicketID)	90
Start Sessions with an Embedded Support Button	91
Reporting API	92
Download Reports with SupportSession	94
Download Reports with SupportSessionListing	103
Download Reports with SupportSessionSummary	105
Download Reports with SupportSessionRecording	107

Download Reports with ShowMyScreenRecording	108
Download Reports with CommandShellRecording	109
Download Reports with PresentationSession	110
Download Reports with PresentationSessionListing	116
Download Reports with PresentationSessionRecording	118
Download Survey Reports with SupportCustExitSurvey and SupportRepExitSurvey	119
Download Reports with SupportTeam	124
Download Syslog Report	128
Download Reports with ArchiveListing	129
Download Reports with Archive	131
Download Reports with LicenseUsage	149
Download Reports with VaultAccountActivity	150
Real-Time State API	153
Protocol of the Real-Time State API	154
System State Model of the Real-Time API	158
JavaScript Library for the Real-Time State API	164
Basic Use of the JavaScript Library	165
Detailed Use of the JavaScript Library	167
Advanced Use of the JavaScript Library	170
Working Demonstration of the JavaScript Library	171
Backup API	175
Query Example	175
Test Scenario	176
API Change Log	177
API Version 1.24.1 for RS 24.1.x	177
API Version 1.23.1 for RS 23.3.x	177
API Version 1.22.2 for RS 22.1.x	177
API Version 1.22.2 for RS 22.3.x	177
API Version 1.22.2 for RS 22.2.x	177
API Version 1.22.1 for RS 22.1.x	177
API Version 1.21.1 for RS 21.3.x	178
API Version 1.19.2 for RS 19.2.x and 20.1.x	178
API Version 1.19.0 for RS 19.1.x	178

API Version 1.18.0 for RS 18.2.x	178
API Version 1.16.0 for RS 17.1.x	178
API Version 1.15.1 for RS 16.2.x	178
API Version 1.15.0 for RS 16.1.x	179
API Version 1.13.1 for RS 15.2.x	179
API Version 1.13.0 for RS 15.1.x	179
API Version 1.12.0 for RS 14.2.x and 14.3.x	180
API Version 1.11.0 for RS 14.1.x	180
API Version Reference	181
Disclaimers, Licensing Restrictions, and Tech Support	183

Remote Support API Programmer's Guide

Version 1.24.1 (for BeyondTrust 24.1.x)

Front-end integration of the BeyondTrust API enables customers to integrate Remote Support support sessions with third-party or in-house developed applications to pull report data, issue commands, or automatically save backups of the B Series Appliance's software configuration.

One common use case of an API integration is linking a help desk ticketing system to Remote Support sessions to track issue resolution.

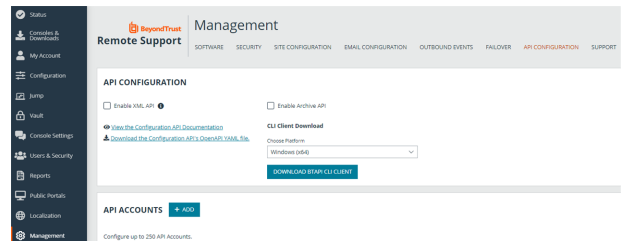
You could also add a feature to an application so that a representative can generate a session from within that application instead of the representative console.

To use the BeyondTrust API, ensure that the **Enable XML API** option is checked on the **Management > API Configuration** page of the **/login** administrative interface.

For some examples in this guide, URLs such as **support.example.com** are used. Please replace this URL with your B Series Appliance's public site URL.

The command and reporting APIs return XML responses that declare a namespace. If you are parsing these responses with a namespace-aware parser, you must set the namespace appropriately or ignore the namespace while parsing the XML.

- Reporting API: <https://www.beyondtrust.com/namespaces/API/reporting>
- Command API: <https://www.beyondtrust.com/namespaces/API/command>



Note: The above namespaces are returned XML data and are not functional URLs.

CLI Tool

A Command Line Interface (CLI) tool can be downloaded from the administrative interface. The CLI tool makes it easier to use and configure APIs and automation scripts, and integrate them with your BeyondTrust Remote Support installation.

i For more information on downloading and installing the CLI Tool, please see [API Configuration](#) in the *BeyondTrust Remote Support Admin Guide* at <https://www.beyondtrust.com/docs/remote-support/getting-started/admin/api-configuration.htm>.

Authenticate to the Remote Support API

API requests are executed by sending an HTTP request to the B Series Appliance. Send the request using any HTTPS-capable socket library or scripting language module, URL fetcher such as cURL, or an OAuth library specific to your platform. BeyondTrust's Remote Support web APIs use OAuth as the authentication method.

To authenticate to the API, you must create an API account on the [/login > Management > API Configuration](#) page. The account must have permission to access the necessary APIs. API requests require a token to be created and then submitted with each API request.



For more information, please see the following:

- For creating an API account, [API Configuration: Enable the XML API and Configure Custom Fields at `www.beyondtrust.com/docs/remote-support/getting-started/admin/api-configuration.htm`](#)
- Example API request at ["Test Scenario" on page 176](#)

Create a Token

Create a token by POSTing to the URL of your Remote Support site followed by `/oauth2/token`:

```
https://support.example.com/oauth2/token
```

The OAuth client ID and client secret associated with the API account should be Base64 encoded and included in an HTTP basic authorization header:

```
Authorization: Basic <base64-encoded "client_id:secret">
```

Include the following POST body in the request:

```
grant_type=client_credentials
```

If the request is processed without error, you will receive an access token JSON response:

```
{
  "access_token": "<token>"
  "token_type": "Bearer"
  "expires_in": 3600
}
```



Note: This token expires after one hour. Any calls to the API past that point must have a new token. Each API account can have a maximum of 30 valid tokens. If an API account attempts to generate more than 30 tokens, then the oldest token is invalidated before a new one is generated.



Note: The client secret cannot be modified, but it can be regenerated on the `/login > Management > API Configuration` page. Regenerating a client secret and then saving the account immediately invalidates any OAuth tokens associated with the account. Any API calls using those tokens will be unable to access the API. A new token must be generated using the new client secret.

Request an API Resource

Now that you have an access token, you can make GET/POST requests via HTTPS to the web API:

```
https://support.example.com/api/command
```

The obtained token is used for HTTP authentication and must be included in an HTTP authorization header with each request:

```
Authorization: Bearer <token>
```

If the token is valid, you gain access to the requested URL.

Authentication Errors

Requests made to the web API with expired or invalid tokens result in a JSON error response:

```
{
  "error": "access_denied"
  "message": "The resource owner or authorization server denied the request."
}
```



IMPORTANT!

When making consecutive API calls, you must close the connection after each API call.

Request Rate Limits

Requests are limited to 20 per second and 15,000 per hour.

This limit applies to all API endpoints and is per API account.

Responses include headers with the rate limit information:



Example:

```
X-RateLimit-Limit      15000
X-RateLimit-Remaining  14996
```


API Use Cases

AWS Registration

Registration of an asset is performed in a user data script. We provide an example script that works with the standard AWS Linux AMI (though it should work for any Linux AMI).

Setup in /login

We configure the endpoints that come online so that all go into the same Jump Group and are accessed via the same Jumpoint. For this example, we use Jumpoint with ID 1 and a shared Jump Group with ID 1. These are referenced in the script below as JUMPOINT_ID and JUMP_GROUP_ID. Configure access to this Jumpoint and Jump Group as needed.

Generate an API account for your AWS scripts to use, and note the CLIENT_ID and CLIENT_SECRET for use in the script below.

The API Account created does not need access to Vault in this example.

Setup SSH Credentials in Vault

If you already have a key pair in AWS you want to use, make sure you have the private key available. If not, open the EC2 section and navigate to **Network and Security > Key Pairs** in the AWS console. Generate a new key pair and save the private key.

In /login, navigate to **Vault > Accounts** and add a new generic account. Set the type to **SSH** and add the username you are using on the AMI (AWS defaults this to **ec2-user**) as well as the private key. This username is the TARGET_USER in the script below.

At the bottom of the account configuration, associate this account with the Jump Group from above by selecting **Jump Items Matching Criteria** and selecting the desired Jump Group.

Save the new account.

Once the account is saved, configure a Group Policy to grant users permission to inject it.

Deploy the Instances in EC2

EC2 instance initialization is performed with user data scripts. The script below registers a Linux AMI as a Shell Jump with the Jumpoint and Jump Group configured.

Prepare and deploy a Linux AMI in EC2. In the user data field, paste this script:

```
#!/bin/bash

# SRA API Credentials
export BT_CLIENT_ID=XXX
export BT_CLIENT_SECRET=XXX
export BT_API_HOST=XXX

# The Jump Group and Jumpoint to use for the Jump Item we create
JUMP_GROUP_ID=1
JUMP_GROUP_TYPE=shared
JUMPOINT_ID=1
```

```
TARGET_USER=ec2-user
# Query the AWS Meta-data service for information about this instance to use
# when creating the Jump Item
INSTANCE_ID=`curl http://169.254.169.254/latest/meta-data/instance-id`
INSTANCE_IP=`curl http://169.254.169.254/latest/meta-data/public-ipv4`
INSTANCE_NAME=$INSTANCE_IP
http_response=$(curl -s -o name.txt -w "%{http_code}" http://169.254.169.254/latest/meta-
data/tags/instance/Name)
if [ "$http_response" == "200" ]; then
    INSTANCE_NAME=$(cat name.txt)
fi

apt update
apt install -y unzip
curl -o btapi.zip -L https://$BT_API_HOST/api/config/v1/cli/linux
unzip btapi.zip

echo "
name=\"${INSTANCE_NAME:-$INSTANCE_IP}\"
hostname=$INSTANCE_IP
jump_group_id=$JUMP_GROUP_ID
jump_group_type=$JUMP_GROUP_TYPE
username=$TARGET_USER
protocol=ssh
port=22
terminal=xterm
jumpoint_id=$JUMPOINT_ID
tag=$INSTANCE_ID
" | ./btapi -k add jump-item/shell-jump

rm name.txt
rm btapi
rm btapi.zip
```

- Add the client credentials as `BT_CLIENT_ID` and `BT_CLIENT_SECRET`.
- Add the site's hostname as `BT_API_HOST` (just the hostname, no HTTPS).
- Make sure that `TARGET_USER`, `JUMPOINT_ID`, and `JUMP_GROUP_ID` (and type) are the values configured above.

This script downloads the `btapi` command line tool and pipes the instance's data to create a new Shell Jump item. The Jump Item is available for immediate use once the instance shows online.

This script uses the `InstanceId` as the item's tag so that you may easily filter it later when performing cleanup. It also attempts to read the instance's `Name` tag to use as the Jump Item's name field for easy identification later. In order for this to work, you must set **Allow tags in metadata** to **Enable** when launching the instance in AWS. If the `Name` is not available, the instance's IP address is used instead.

AWS Cleanup

Cleaning up terminated AWS Jump Items may be automated in multiple ways, depending on the desired behavior. Here, we show two different methods: a script that may be run on-demand to clean up terminated instances, and an AWS Lambda function and EventBridge rule that is triggered automatically.

On-Demand Script

If you want to clean up Jump Items on demand, the following script can be run as needed or scheduled to run as needed with a tool like **chron**.

```
#!/bin/bash

export BT_CLIENT_ID=XXX
export BT_CLIENT_SECRET=XXX
export BT_API_HOST=XXX

export AWS_ACCESS_KEY_ID=XXX
export AWS_SECRET_ACCESS_KEY=XXX

# Note this requires the AWS CLI tool to be installed
INSTANCE_IDS=$(aws ec2 describe-instances --query 'Reservations[*].Instances[*].[InstanceId]' --filters 'Name=instance-state-name,Values=[terminated]' --output text)

if [[ -z "$INSTANCE_IDS" ]]; then
    exit
fi

for inst in "${INSTANCE_IDS[@]"; do
    ID=$(echo "tag=$inst" | btapi --env-file=~/.config/aws-api -kK list jump-item/shell-jump | perl -ne '/^0__id=(\d+)/ && print $1')
    btapi --env-file=~/.config/aws-api delete jump-item/shell-jump $ID
done
```

AWS Hooks

Setting up the hooks in AWS requires two pieces in AWS:

- A Lambda function to do the cleanup
- An EventBridge rule to call the Lambda function

The following example is one way to configure these pieces

Create the Lambda

This example uses Python, but you can use the same logic for any language you prefer.

This example makes use of the `requests`, `requests_oauthlib`, and `oauthlib` python libraries. To use these, you must create and upload a layer with these dependencies to attach to the lambda. This may be performed from a local Linux machine with the same python version installed that the lambda uses, or you may use the AWS Cloud9 service to spin up a compatible environment.

To create the layer, use the following commands:

```
mkdir tmp
cd tmp
virtualenv v-env
source ./v-env/bin/activate
```

```
pip install requests oauthlib requests_oauthlib
deactivate

mkdir python
# Using Python 3.9
cp -r ./v-env/lib64/python3.9/site-packages/* python/.
zip -r requests_oauthlib_layer.zip python

# Or manually upload the zip under AWS Lambda > Layers
aws lambda publish-layer-version --layer-name requests_oauthlib --zip-file fileb://requests_oauthlib_layer.zip --compatible-runtimes python3.9
```

With the layer added, navigate to AWS Lambda and create a new function. Select **Python** as the runtime with the same version used above. The function requires **Describe*** permissions for EC2 as well as the general AWS Lambda role.

Once the function is created, replace the contents of the generated **lambda_function.py** file with this script:

```
import boto3
import os
from oauthlib.oauth2 import BackendApplicationClient
from requests_oauthlib import OAuth2Session

ec2 = boto3.client('ec2', region_name=os.environ.get('AWS_REGION'))

BT_CLIENT_ID = os.environ.get('BT_CLIENT_ID')
BT_CLIENT_SECRET = os.environ.get('BT_CLIENT_SECRET')
BT_API_HOST = os.environ.get('BT_API_HOST')

class API:
    def __init__(self) -> None:
        self.client = BackendApplicationClient(client_id=BT_CLIENT_ID)
        self.oauth = OAuth2Session(client=self.client)
        self.token = 'bad'

    def call(self, method, url, headers=None, data=None, **kwargs):
        def reload_token(r, *args, **kwargs):
            if r.status_code == 401:
                self.refreshToken()
                return self.call(method, url, headers=headers, data=data, **kwargs)
            elif r.status_code > 400:
                r.raise_for_status()

        d = data if method != 'get' else None
        p = data if method == 'get' else None
        resp = self.oauth.request(
            method,
            f"https://{BT_API_HOST}/api/config/v1/{url}",
            headers=headers, json=d, params=p, hooks={'response': reload_token}, **kwargs)

        resp.raise_for_status()

        return resp

    def refreshToken(self) -> None:
```

```
self.token = self.oauth.fetch_token(
    token_url=f"https://{BT_API_HOST}/oauth2/token",
    client_id=BT_CLIENT_ID,
    client_secret=BT_CLIENT_SECRET
)

client = API()

def lambda_handler(event, context):
    instances = ec2.describe_instances(
        Filters=[
            {'Name': 'instance-state-name', 'Values': ['terminated']}
        ]
    )
    data = []

    for r in instances['Reservations']:
        for inst in r['Instances']:
            print(inst)
            d = {
                'id': inst['InstanceId'],
                'state': inst['State'],
                'ip': inst.get('PublicIpAddress'),
                'name': [x['Value'] for x in inst['Tags'] if x['Key'] == 'Name'],
            }
            response = client.call('get', 'jump-item/shell-jump', data={'tag': inst
['InstanceId']})
            items = response.json()
            if len(items) > 0:
                item = items[0]
                d['data'] = item
                client.call('delete', f'jump-item/shell-jump/{item["id"]}')
            data.append(d)

    return {
        'statusCode': 200,
        'body': data
    }
```

Next, scroll to the bottom of the page to the **Layers** panel. Click **Add a layer** and select the layer that was created above.

This script is designed to read the BT API information from the environment. You must add the BT_API_HOST, BT_CLIENT_ID, and BT_CLIENT_SECRET configuration variables under **Configuration** -> **Environment** variables.

Configuring EventBridge

Navigate to **Amazon EventBridge > Rules** and click **Create rule**. Name the rule, ensure it is enabled, select **Rule with an event pattern**, and click **Next**.

To build the event pattern, choose the **AWS Events or EventBridge partner events** option in the **Event source** panel, and then scroll down to the **Event pattern** panel. Select the **Custom patterns (JSON Editor)** option, paste the following pattern, and click **Next**.

```
{
  "source": ["aws.ec2"],
  "detail-type": ["EC2 Instance State-change Notification"],
  "detail": {
    "state": ["terminated"]
  }
}
```

For the event target, select **AWS Service**, then pick **Lambda function** from the dropdown. For **function**, select the name of the Lambda created above. Finish creating the rule definition.

Finished

Once the rule and lambda are in place, the lambda is invoked when any EC2 instance moves or is moving to **terminated** status and is removed from the Jump Item list.

Scripting a New Setup

The script below runs through a more complicated automated process. This script sets up the given instance to be a Jumpoint for a VPC and creates a new Jump Group and SSH key in Vault for the VPC. It then grants access to these new resources to a given Group Policy.

This script assumes an Ubuntu Server instance.



Note: Amazon Linux AMIs are not supported as Jumpoint hosts. Jumpoint hosts require GLIBC 2.27 and the Amazon Linux AMIs support only 2.26.

```
#!/bin/bash
set -euo pipefail
set -x

# SRA API Credentials
export BT_CLIENT_ID=XXX
export BT_CLIENT_SECRET=XXX
export BT_API_HOST=XXX

# Set to the ID of the Group Policy to tie everything together
GROUP_POLICY_ID=XXX

# Set this to the user account for this instance
TARGET_USER=ubuntu
# Query AWS metadata for this instance to data needed when creating items later
INSTANCE_IP=`curl http://169.254.169.254/latest/meta-data/public-ipv4`
macid=$(curl http://169.254.169.254/latest/meta-data/network/interfaces/macs/)
# Using the VPC ID as the base for all our names
NAME_BASE=$(curl http://169.254.169.254/latest/meta-data/network/interfaces/macs/${macid}/vpc-id)

HOME=${HOME:=/home/${TARGET_USER}}

# For running as a user
JUMPOINT_BASE_DIR="$HOME/.beyondtrust/jumpoint"
SYSTEMD_DIR="$HOME/.config/systemd/user"
```

```
SYSTEMD_ARGS=--user
JUMPOINT_USER=""

if [ "$(whoami)" == "root" ]; then
    # For running as root
    JUMPOINT_BASE_DIR="/opt/beyondtrust/jumpoint"
    SYSTEMD_DIR="/etc/systemd/system"
    SYSTEMD_ARGS=""
    JUMPOINT_USER="--user $TARGET_USER"
fi

# Make the command calls a bit easier to write
ORIG_PATH=$PATH
pwd=$(pwd)
export PATH=$pwd:$PATH

# Ubuntu server does not have unzip by default
sudo apt update
sudo apt install -y unzip

# Download jq into the current directory for ease of parsing JSON responses
curl -L https://github.com/stedolan/jq/releases/download/jq-1.6/jq-linux64 -o jq
chmod +x jq
curl -o btapi.zip -L https://$BT_API_HOST/api/config/v1/cli/linux
unzip btapi.zip

# Create a Jumpoint for this VPC
jpc=$(echo "
name=$NAME_BASE
platform=linux-x86
shell_jump_enabled=True
" | btapi -k add jumpoint)

jpid=$(echo "$jpc" | jq '.id')

echo "Created Jumpoint with id [$jpid]"

# Download and run the Jumpoint installer
installer=$(btapi download "jumpoint/$jpid/installer" | jq -r '.file')
chmod +x "$installer"
# Make sure the base install directory exists
mkdir -p "$JUMPOINT_BASE_DIR"

# IMPORTANT: Make sure your linux distro has all the packages needed to install
# the Jumpoint. Ubuntu server 22 needs these two
sudo apt install -y libxkbcommon0 fontconfig
sh "$installer" --install-dir "$JUMPOINT_BASE_DIR/$BT_API_HOST" $JUMPOINT_USER

# Make sure the systemd service directory exists (mostly for the user mode directory)
mkdir -p "$SYSTEMD_DIR"

# Create the systemd service file
echo "[Unit]
Description=BeyondTrust Jumpoint Service
Wants=network.target"
```

```
After=network.target

[Service]
Type=forking
ExecStart=$JUMPOINT_BASE_DIR/$BT_API_HOST/init-script start" > "$SYSTEMD_DIR/jumpoint.$BT_API_
HOST.service"

if [ "$(whoami)" != "$TARGET_USER" ]; then
    echo "User=$TARGET_USER" >> "$SYSTEMD_DIR/jumpoint.$BT_API_HOST.service"
fi

echo "
Restart=no
WorkingDirectory=$JUMPOINT_BASE_DIR/$BT_API_HOST

[Install]
WantedBy=default.target
" >> "$SYSTEMD_DIR/jumpoint.$BT_API_HOST.service"

# Load the Jumpoint service and start it
systemctl $SYSTEMD_ARGS daemon-reload
systemctl $SYSTEMD_ARGS start "jumpoint.$BT_API_HOST.service"

# Cleanup the installer file
rm -f "$installer"

# Create a Jump Group for this VPC
jg=$(echo "
name=\"$NAME_BASE Jump Group\"
" | btapi -k add jump-group)

jgid=$(echo "$jg" | jq '.id')

# Create an SSH Key for this VPC and add the private key to Vault
# NOTE, you will need to manually associate this credential to the
# Jump Group for this VPC in /login
ssh-keygen -f "./key" -P "" -q -t ed25519
touch "$HOME/.ssh/authorized_keys"
cat ./key.pub >> "/home/$TARGET_USER/.ssh/authorized_keys"
priv=$(cat ./key)

vk=$(echo "
type=ssh
name=\"$NAME_BASE SSH\"
username=$TARGET_USER
private_key=\"$priv\"
" | btapi -k add vault/account)

vkid=$(echo "$vk" | jq '.id')

# Cleanup the key
rm -f ./key
rm -f ./key.pub

# Create an SSH Jump item back to this instance
```



```
echo "  
name=\"\$NAME_BASE Jumpoint\  
hostname=\"\$INSTANCE_IP  
jump_group_id=\"\$jgid  
jump_group_type=shared  
username=\"\$TARGET_USER  
protocol=ssh  
port=22  
terminal=xterm  
jumpoint_id=\"\$jpid  
" | btapi -k add jump-item/shell-jump  
  
# Modify the Group Policy to grant access to the Jumpoint, Jump Group and Vault Account  
echo "jumpoint_id=\"\$jpid" | btapi -k add group-policy/\$GROUP_POLICY_ID/jumpoint  
echo "jump_group_id=\"\$jgid" | btapi -k add group-policy/\$GROUP_POLICY_ID/jump-group  
echo "  
account_id=\"\$vkid  
role=inject  
" | btapi -k add group-policy/\$GROUP_POLICY_ID/vault-account  
  
# Cleanup the tools downloaded at the top of this script  
rm -f jq  
rm -f btapi  
rm -f btapi.zip  
  
# Reset PATH  
export PATH=\"\$ORIG_PATH
```

Integrate an external Chatbot with Remote Support

In this use case, a company wants to have a chatbot provide the initial support to their users but enable it to elevate the session to a representative in Remote Support, where a representative can take over the call. This use case illustrates how various APIs can work together, and is designed to work in three phases, outlined below.

Phase 1: User interacts with the Chatbot

The user is receiving support directly from the Chatbot. Remote Support and RS APIs are not involved in this phase. When the chatbot determines that an agent should be involved, the process moves to Phase 2.

Phase 2: The Chatbot begins a session in Remote Support

In this phase, the customer is connected to a Representative in Remote Support by the Chatbot. Some of the steps in this phase are optional, and can be customized based on how the interaction best fits into the desired workflow. The interaction should move to Phase 3 when the user needs to interact with Remote Support directly, and the chatbot should no longer be involved.

1. Start a session in Remote Support by calling the *create_virtual_customer* action of the Command API.
2. (Optional) Place the relevant chat history from the chatbot interaction into the session for the Representative to review. There are two ways to do this:

- Call *send_chat_message* action of the Command API for each chat message to add it to the session's chat history.
 - Or, add the entire transcript to a custom attribute on the session using the *set_session_attributes* action of the Command API.
3. (Optional) Proxy messages between the customer and the representative. This allows the customer to stay in the same context, but chat with the Representative instead of the chatbot.
- To send messages from the user to the Representative, the bot should use the *send_chat_message* action of the Command API.
 - To display chat messages from the Representative to the user, the chatbot should be configured to receive the *Someone Sends a Chat Message* outbound event and look for inbound messages for the user's session.

Phase 3: The User interacts with the Representative in Remote Support

To move into this phase, the chatbot should direct the user to the URL given as part of the *create_virtual_customer* response. Once the user has been directed to that URL, the session is fully in Remote Support and the chatbot is no longer involved. Any remaining chatbot UI can be closed at this point.

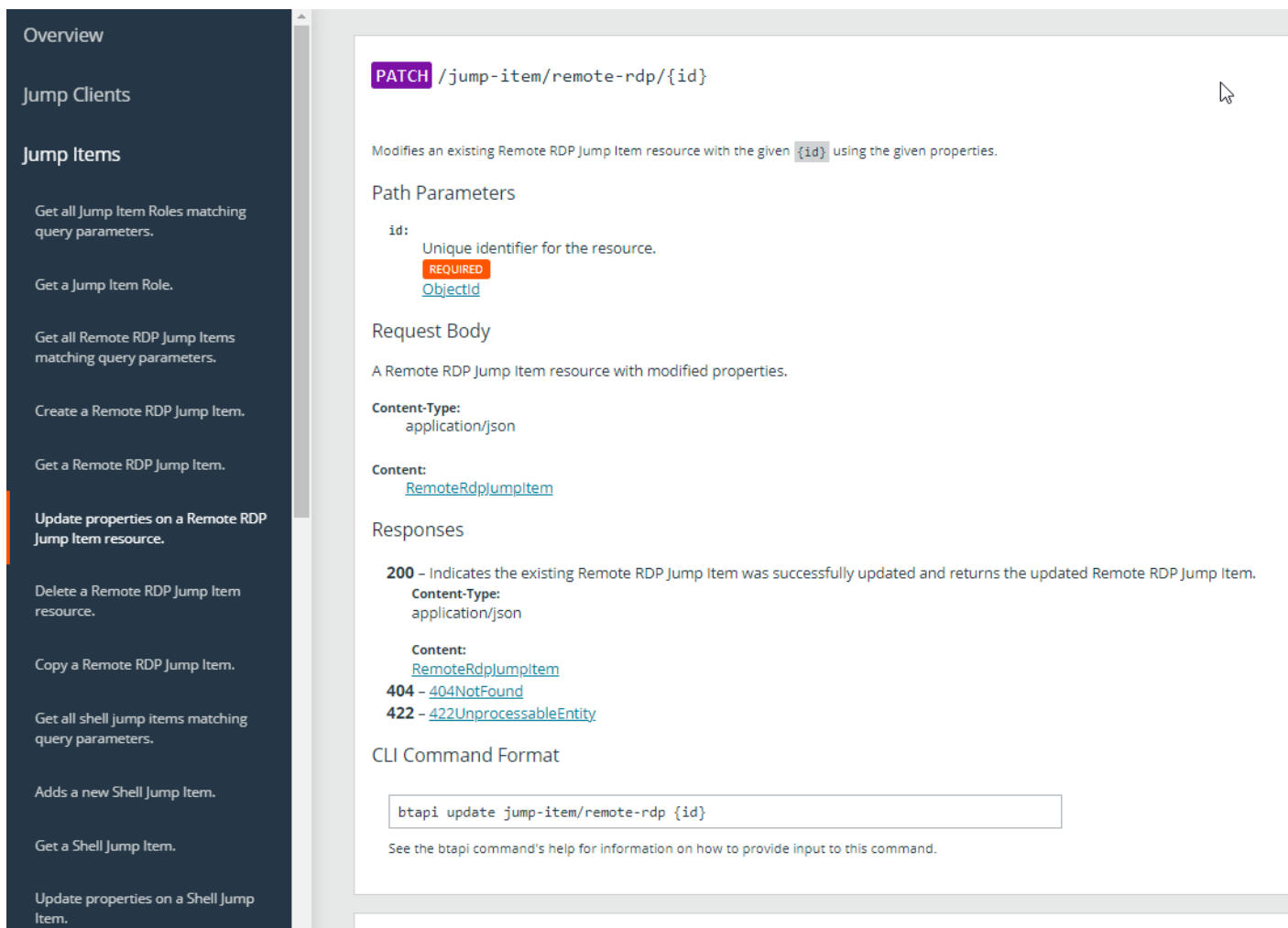
If the chatbot interface is web based, the *create_virtual_customer* call can optionally return a click-to-chat URL. In this case, the chatbot can simply redirect to that URL within the same window and the user will transition to Remote Support's click-to-chat interface.

Configuration API

The Configuration API is written according to OpenAPI standards, and enables end users to view documentation for the API using their preferred OpenAPI tool, such as Swagger, Postman, or RediDoc. You can either view the Configuration API documentation directly in the product (/login), or download the YAML file and use a tool of your choice to view the documentation.

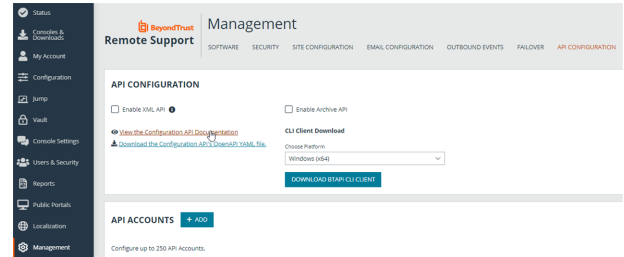
View the Configuration API Documentation in /login

Log in to your site (for example, <https://example.com/login/apidocs.html>) and enter your credentials. There are lists, descriptions, and examples for all available configuration APIs.



The screenshot displays the API documentation interface. On the left is a dark sidebar with navigation links: Overview, Jump Clients, Jump Items, and a list of actions like 'Get all Jump Item Roles matching query parameters', 'Create a Remote RDP Jump Item', etc. The main content area shows details for the **PATCH /jump-item/remote-rdp/{id}** endpoint. It includes a description: 'Modifies an existing Remote RDP Jump Item resource with the given {id} using the given properties.' Below this are sections for 'Path Parameters' (with a required 'id' parameter), 'Request Body' (described as a Remote RDP Jump Item resource), 'Content-Type' (application/json), 'Content' (RemoteRdpJumpItem), and 'Responses' (200 success, 404 Not Found, 422 UnprocessableEntity). At the bottom, there is a 'CLI Command Format' section with a code box containing `btapi update jump-item/remote-rdp {id}` and a note to see the command's help for more information.

if you are already logged in to the administrative interface, you can click **Management** on the left menu, then click the **API Configuration** tab. Click **View the Configuration API Documentation**.



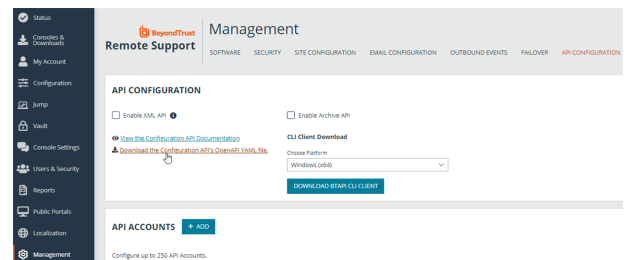
Access the YAML file via API

By following the steps below and referring to the documentation for the OpenAPI tool of your choice, you can view the API documentation and even *try out* features of the API using an intuitive browser user interface.

1. Go to `/login > Management > API Configuration`.
2. Under **API Accounts**, click **Add**.
3. Enter a name to identify your new API account.
4. Make sure the **Configuration API > Allow Access** box is checked.
5. Click **Save**.
6. Download and install your favorite software for running API calls. Please refer to the documentation for your selected software before proceeding, if needed.
7. In `/login > Management > API Configuration`, select the new API account you just created and click the edit icon.
8. Copy the **OAuth Client ID** and paste it into your selected software.
9. Back in `/login`, click **Generate New Client Secret**, copy it, and paste it into you selected software.
10. Click **Save** to save your API account.
11. Import the **OpenAPI.yaml** file from your site, using your preferred OpenAPI tool. The **OpenAPI.yaml** file can be accessed by creating a new **GET** request with the URL format <https://example.com/api/config/v1/openapi.yaml>. Once imported, the documentation for the Configuration APIs will be automatically generated. Follow the instructions in your API call software in order to complete these steps.

Download the YAML file

Alternatively, you can download the YAML file by clicking the **Download the Configuration API's OpenAPI YAML file**



Remote Support Command API

The command API is designed to send commands to your BeyondTrust site from an outside application. Commands can start or transfer a support session, get a list of logged-in representatives, or obtain a list of support teams and issues. You can also check the health of your B Series Appliance, change a B Series Appliance's failover role, or get information about your BeyondTrust API version.

The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7.

Commands are executed by sending an HTTP request to the B Series Appliance. Send the request using any HTTPS-capable socket library, scripting language module, or URL fetcher such as **cURL** or **wget**. Use either **GET** or **POST** as the request method.



Note: POST requests must include a "Content-Type: application/x-www-form-urlencoded" HTTP header when supplying parameters in the request body, and the parameters must be url-encoded. Multipart POST requests are not supported.



IMPORTANT!

When making consecutive API calls, you must close the connection after each API call.

The command API URL is <https://support.example.com/api/command>.

An XML schema describing the command API response format is available at <https://support.example.com/api/command.xsd>.

Required Parameters for Command API

action=[string]	The type of action to perform. Can be any of the following: <table style="width: 100%; border: none;"> <tr> <td style="padding: 2px;">get_logged_in_reps</td> <td style="padding: 2px;">get_appliances</td> </tr> <tr> <td style="padding: 2px;">get_support_teams</td> <td style="padding: 2px;">get_connected_client_list</td> </tr> <tr> <td style="padding: 2px;">generate_session_key</td> <td style="padding: 2px;">get_connected_clients</td> </tr> <tr> <td style="padding: 2px;">join_session</td> <td style="padding: 2px;">check_health</td> </tr> <tr> <td style="padding: 2px;">transfer_session</td> <td style="padding: 2px;">set_failover_role</td> </tr> <tr> <td style="padding: 2px;">set_session_attributes</td> <td style="padding: 2px;">import_jump_shortcuts</td> </tr> <tr> <td style="padding: 2px;">get_session_attributes</td> <td style="padding: 2px;">get_api_info</td> </tr> <tr> <td style="padding: 2px;">terminate_session</td> <td></td> </tr> </table>	get_logged_in_reps	get_appliances	get_support_teams	get_connected_client_list	generate_session_key	get_connected_clients	join_session	check_health	transfer_session	set_failover_role	set_session_attributes	import_jump_shortcuts	get_session_attributes	get_api_info	terminate_session	
get_logged_in_reps	get_appliances																
get_support_teams	get_connected_client_list																
generate_session_key	get_connected_clients																
join_session	check_health																
transfer_session	set_failover_role																
set_session_attributes	import_jump_shortcuts																
get_session_attributes	get_api_info																
terminate_session																	



IMPORTANT!

If you experience a high volume of support requests, repeatedly calling a command such as **get_logged_in_reps** or **get_support_teams** might bottleneck your system. Therefore, a best practice is to not request a list of representatives or teams with each support request. Instead, if making the same API call in succession, consider caching the results for a period of time and reusing them. New sessions requests should reference the cached list instead of calling for the list each time.

The command API returns XML responses that declare a namespace. If you are parsing these responses with a namespace-aware parser, you need to set the namespace appropriately or ignore the namespace while parsing the XML:

Command API: <https://support.example.com/namespaces/API/command>



Note: The above namespace is returned XML data and is not a functional URL.

API Command: get_logged_in_reps

The `get_logged_in_reps` request returns XML data about all logged-in representatives. It requires no additional parameters.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#). The API account must have read-only or full access to the command API.

XML Response for get_logged_in_reps Query

`<logged_in_reps>`

Returns a `<rep>` element for each logged-in representative. If no representatives are logged in, this element will contain no `<rep>` elements. If an error occurs, it will contain an `<error>` element describing the problem.

Element Names and Attributes

/logged_in_reps/rep

<code>id</code> (attribute)	Unique ID assigned to the representative.
<code><display_name></code>	This element is deprecated as of API version 1.10.0 but still exists for backwards compatibility. Its value is the same as that of <code><public_display_name></code> .
<code><public_display_name></code>	The public display name currently assigned to the representative.
<code><private_display_name></code>	The private display name currently assigned to the representative.
<code><type></code>	The type of rep logged in. Types include Normal and Invited .
<code><direct_link></code>	An HTML anchor tag containing the URL that customers can use to download the customer client to connect directly to the representative.
<code><logged_in_since></code>	The date and time at which the representative logged in.
<code><presentation_count></code>	The number of active presentations the representative is currently running.
<code><support_session_count></code>	The number of active sessions the representative is currently running.
<code><showing_on_rep_list></code>	Integer value (1 or 0) indicating if the rep has permission to show on the public site and has the Showing On Representative List option checked in the representative console.

Query Example: get_logged_in_reps

`get_logged_in_reps`

`https://support.example.com/api/command?action=get_logged_in_reps`

**IMPORTANT!**

If you experience a high volume of support requests, repeatedly calling a command such as `get_logged_in_reps` or `get_support_teams` might bottleneck your system. Therefore, a best practice is to not request a list of representatives or teams with each support request. Instead, if making the same API call in succession, consider caching the results for a period of time and reusing them. New sessions requests should reference the cached list instead of calling for the list each time.

API Command: set_rep_status

The `set_rep_status` command sets the status for representatives logged into the representative console.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account must have read-only or full access to the command API.

Required Parameter for set_rep_status

<code>rep_id=[integer]</code>	The user ID of the representative whose status gets updated.
<code>code_name=[string]</code>	The code name of the status to set.

XML Response for set_rep_status Query

<code><success></code>	Present if the status was set successfully.
<code><error></code>	Returns an error message if the status was not set successfully.

Query Example: set_rep_status Query

Set the status of the representative with ID 5 to the status with the code name <code>busy</code> .	<code>https://support.example.com/api/command?action=set_rep_status&rep_id=5&code_name=busy</code>
---	--

API Command: `get_support_teams`

The `get_support_teams` request returns XML data containing all configured support teams and all the issues configured for each team.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account must have read-only or full access to the command API.

Optional Parameter for `get_support_teams`

`showmembers=1`

Causes the output to also list all the representatives who are members of each team. Depending on team configuration, showing all members could add a significant amount of data to the output and should be used sparingly.

XML Response for `get_support_teams` Query

`<support_teams>`

Contains a `<support_team>` element for each support team. If no support teams have been created, this element will contain no `<support_team>` elements. If an error occurs, it will contain an `<error>` element describing the problem.

Element Names and Attributes

/support_teams/support_team

<code>id</code> (attribute)	Unique ID assigned to the support team.
<code><name></code>	The name of the support team.
<code><issues></code>	Contains an <code><issue></code> element for each issue associated with this support team, as described below. If no issues have been configured for this team, the <code><issue></code> element will be blank.
<code><support_session_count></code>	The number of sessions waiting in this team queue.
<code><members></code>	Displayed only if the <code>showmembers</code> parameter has been included in the request. Contains a <code><representative></code> element for each member of this team. If no representatives have been assigned to this team, the <code><members></code> element will be blank.

/support_teams/support_team/issues/issue

<code>id</code> (attribute)	Unique ID assigned to this issue.
<code><title></code>	The title of the issue.

/support_teams/support_team/members/representative

<code>id</code> (attribute)	Unique ID assigned to the representative.
<code><username></code>	The username assigned to the representative.

<code><display_name></code>	This element is deprecated as of API version 1.10.0 but still exists for backwards compatibility. Its value is the same as that of <code><public_display_name></code> .
<code><public_display_name></code>	The public display name currently assigned to the representative.
<code><private_display_name></code>	The private display name currently assigned to the representative.

Query Examples: `get_support_teams`

Show names and issues	<code>https://support.example.com/api/command?action=get_support_teams</code>
Show names, issues, and members	<code>https://support.example.com/api/command?action=get_support_teams&showmembers=1</code>



IMPORTANT!

If you experience a high volume of support requests, repeatedly calling a command such as `get_logged_in_reps` or `get_support_teams` might bottleneck your system. Therefore, a best practice is to not request a list of representatives or teams with each support request. Instead, if making the same API call in succession, consider caching the results for a period of time and reusing them. New sessions requests should reference the cached list instead of calling for the list each time.

API Command: generate_session_key

The **generate_session_key** command creates a new session key to be used in starting a support session. Note that if your B Series Appliance has multiple public sites, the session key created may be associated with any of these sites, depending on the method used to download the customer client.

For example, Site A has a hostname of support.example.com, and Site B has a hostname of remote.example.com. When a **generate_session_key** request is made to support.example.com with a **url_hostname** of support.example.com, both a session key code and a unique session key URL will be generated.

If the customer goes to the generated URL to download the customer client, then the session will be associated with Site B, because the session key URL points to the hostname designated by the **url_hostname** parameter.

However, the customer could also download the customer client by submitting the session key code on either site. Therefore, if the customer goes to Site A to submit the code, then the session will be associated with Site A, while if they go to Site B, the session will be associated with Site B.

The API account must have full access to the command API.


Required Parameters for generate_session_key

<code>type=[string]</code>	The type of session for which you would like to generate a session key. Currently, the only supported value is support .
<code>queue_id=[string]</code>	<p>The queue in which the session should be placed. Can be one of general, rep:[id], or team:[id], where [id] is the numeric ID for the representative, or team queue in which you wish to place this session.</p> <p>Can also be rep_username:[username]. This call will work only if a single user with the given username exists; otherwise, an error message will be returned.</p> <p>Can also be issue:[issue_code_name].</p> <p>To get a representative's ID, see "API Command: get_logged_in_reps" on page 23. To get a team's ID, see "API Command: get_support_teams" on page 26.</p>

Optional Parameters for generate_session_key

<code>session.custom.external_key=[string]</code>	An arbitrary string that can link this session to an identifier on an external system, such as a help desk ticket ID. This has a maximum length of 1024 characters.
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>
<code>session.priority=[integer]</code>	The priority of the session, from 1 to 3 . 1 = high, 2 = medium, and 3 = low.
<code>session.skills=[string]</code>	A comma-separated list of the code names of skills to assign to a session.
<code>ttl=[integer]</code>	Time in seconds for which this key should be valid. If omitted, the maximum session

	key timeout set in the administrative interface will be used.
url_hostname=[string]	Hostname to use in the URL generated for the session key. Defaults to the primary hostname for your BeyondTrust Appliance B Series.

 **Note:** If a parameter is set via `generate_session_key` and is then overwritten via another API (e.g., `start_session`) then the second attribute will take precedence.

XML Response for generate_session_key Query

<type>	The type of session for which this key was generated. Currently, the only supported value is support .
<ttl>	Time in seconds for which this key is valid.
<expires>	The timestamp at which this session key expires.
<queue>	The queue in which this session will be placed. This value will be rep or team . Also contains an available attribute. For a session key targeting a representative, the value is 1 if that representative is logged in. For a session key targeting a team, the value is 1 if at least one representative is logged in for that queue or the queue is persistent. If no representative is available for the targeted queue, the value is 0 .
<queue_id>	The numeric ID of the queue.
<external_key>	A string that links this session to an identifier on an external system, such as a help desk ticket ID.
<short_key>	The seven-character string that the customer can enter on your public site to start a session.
<key_url>	The session key url to which the customer can go to start a session.
<mail_subject>	The subject line of the session key email invitation. This is static text defined by BeyondTrust and is not configurable.
<mail_body>	The body of the session key email invitation. This is static text defined by BeyondTrust and is not configurable.

Query Examples: generate_session_key

Specific representative by ID	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=rep:1</code>
Specific team	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=team:1</code>
Specific representative by username	<code>https://support.example.com/api/command?action=generate_session_key&type=support&</code>

	<code>queue_id=rep_username:admin</code>
Specific issue	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=issue:other</code>
Specific team, 1 hour time to live	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=team:1&ttl=3600</code>
Specific team, external key and custom field	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=team:1&session.custom.external_key=ABC1234&session.custom.custom_field1=Custom%20Value</code>
Specific team, specific hostname	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=team:1&url_hostname=support.example.com</code>
Specific team, skills and priority set	<code>https://support.example.com/api/command?action=generate_session_key&type=support&queue_id=team:1&session.priority=1&session.skills=codename1,codename2</code>

API Command: join_session

The `join_session` command adds a logged in representative to an existing support session.

The API account must have full access to the command API.

Required Parameters for join_session

<code>lsid=[string]</code>	The ID of the session to which the representative will be added.
<code>rep_id=[string]</code> or <code>rep_username=[string]</code>	Unique ID or username assigned to the representative. To get a representative's ID, see "API Command: get_logged_in_reps" on page 23 .

XML Response for join_session Query

<code><success></code>	Returns a message of Representative successfully added if successful.
<code><error></code>	Returns an error message if not successful.

Query Examples: join_session

Add representative to session <code>c69a8e10bea9428f816cfababe9815fe</code>	<code>https://support.example.com/api/command?action=join_session&lsid=c69a8e10bea9428f816cfababe9815fe&rep_id=151</code>
--	---

API Command: transfer_session

The **transfer_session** command transfers an active session from one queue to another.

The API account must have full access to the command API.

Required Parameters for transfer_session

lsid=[string]	The ID of the session you wish to transfer.
queue_id=[string]	The queue to which this session should be transferred. Can be one of rep:[id] , team:[id] , or rep_username:[string] where [id] is the numeric ID for the representative or team queue and [string] is the name of the representative to which you wish to transfer this session. To get a representative's ID or name, please see API Command: get_logged_in_reps . To get a team's ID, see API Command: get_support_teams .

XML Response for transfer_session Query

<success>	Returns a message of Successfully transferred if the transfer was successful.
<error>	Returns an error message if the transfer was not successful.

Query Examples: transfer_session

Session c69a8e10bea9428f816cfababe9815fe to specific representative	https://support.example.com/api/command? action=transfer_session& lsid=c69a8e10bea9428f816cfababe9815fe&queue_id=rep:1
Session c69a8e10bea9428f816cfababe9815fe to specific team	https://support.example.com/api/command? action=transfer_session& lsid=c69a8e10bea9428f816cfababe9815fe&queue_id=team:1

API Command: send_chat_message

The **send_chat_message** operation sends a chat message to an existing session. All members of the session see the chat message. The chat message can be sent on behalf of an existing member of the session, such as a customer or a representative, or it can be sent as a system message. The chat bot uses this API to send the end user's chat messages to the representative.

i *The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#). The API account must have read-only or full access to the command API.*

Required Parameters for send_chat_message

Name	Type	Required?	Description
team_id	integer	required	This is the unique id of the support team in which the chat message will be sent. This parameter cannot be used with Isid at the same time.
to_rep	integer	optional	The value should be a valid <user_id> for a representative . The team chat message will display as a private message sent from the representative specified in impersonate to the representative with the given <user_id>. (This parameter should work with team_id and impersonate).
to_rep_username	string	optional	The value should be a valid <username> for a representative . The team chat message will display as a private message sent from the representative specified in impersonate to the representative with the given <username>. (This parameter should work with team_id and impersonate).
Isid	GUID	required	This is the unique <i>Logging Session ID</i> (LSID) of the session in which the chat message will be sent.
body	string	required	The content of the chat message. Text formatting can be specified using BBCode (see below).
impersonate	string	optional	<p>This is a string identifying the user in the session from which the message appears to be sent. Valid values are:</p> <ul style="list-style-type: none"> customer: The message will appear to come from the customer who is in the session. This value can only work with parameter Isid. rep:<user_id>: The message appears to come from the representative with the given user_id. rep_username:<username>: The message appears to come from the representative with the given username. <p>If this parameter is not provided, the chat message is rendered as a system chat message.</p>
timestamp	integer	optional	The UNIX timestamp at which the chat message was originally sent. The timestamp must be in the past according to the B Series Appliance's clock. If not specified, chat messages are sent with the current time.

Text Formatting

The chat message's body can be formatted using the following BBCode:

BBCode	Result
[b]bolded text[/b]	bolded text
[color=blue]blue text[/color]	blue text
OR	OR
[color=#0000FF]blue text[/color]	blue text
[i]italicized text[/i]	<i>italicized text</i>
[u]underlined text[/u]	<u>underlined text</u>
[u][li]list element 1[/li][li]list element 2[/li][/u]	<ul style="list-style-type: none"> list element 1 list element 2
[url]https://www.beyondtrust.com/[/url]	https://www.beyondtrust.com/

Logging

Chat messages sent via this Command API include a special attribute in reports indicating which API Account originally sent the message. End users and representatives do not know the chat message came from an API Account unless they are in the session and see that the chat message is sent on their own behalf.

Real-time Chat Translations

Chat messages sent with this Command API operation are not affected by the Real-Time Chat Translations feature. All chat messages sent by this operation will be rendered to both customers and representatives exactly as they were specified in the API call. No real-time translation is performed.

Query Examples: send_chat_message

```
send_chat_message
```

```
https://support.example.com/api/command?  
action=: send_chat_message
```

API Command: create_virtual_customer

The `create_virtual_customer` request creates a virtual customer in Remote Support and places that customer in a new session. This operation is necessary so that Remote Support can accurately display and log chat messages forwarded by the chat bot on behalf of the end user. Before calling this API, integrations must use the `generate_session_key` operation to get a 7 digit short key to start a session.

i *The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#). The API account must have read-only or full access to the command API.*

Required Parameters for create_virtual_customer

<code>short_key</code>	The short_key returned from the <code>generate_session_key</code> operation.
<code>name</code>	The public display name of the customer.
<code>ip_address</code>	The public IP address of the customer.
<code>operating_system</code>	The operating system of the customer.
<code>computer_name</code>	The computer name to be shown with the customer.

Optional Parameter for create_virtual_customer

<code>locale=[string]</code>	<p>The locale code of the language to be used for this chat bot. The language must be installed on your B Series Appliance. To see which languages are installed, go to <code>/login > Localization > Languages</code>.</p> <p>Available language packages can include English (en-us), German (de), Latin American Spanish (es), EU Spanish (es-sp), Finnish (fi), EU French (fr), Italian (it), Dutch (nl), Brazilian Portuguese (pt-br), EU Portuguese (pt-pt), Swedish (sv-se), Turkish (tr), Japanese (ja), Simplified Chinese (zh-hans), Traditional Chinese (zh), and Russian (ru).</p>
------------------------------	--

XML Response for create_virtual_customer Query

<code>download_url</code>	The URL from which the native customer client can be downloaded to elevate from a virtual chat-only customer to a non-virtual full remote support session.
<code>chat_url</code>	If this parameter is set to <code>1</code> , then the <code>download_url</code> returned redirects to a click-to-chat (c2c) session. If it is omitted or given any other value, then <code>download_url</code> returns a URL to download the full client (this is the current behavior).

Query Examples: create_virtual_customer

<code>create_virtual_customer</code>	<code>https://support.example.com/api/command?action=: create_virtual_customer</code>
--------------------------------------	---

**IMPORTANT!**

If you experience a high volume of support requests, repeatedly calling a command such as `get_logged_in_reps` or `get_support_teams` might bottleneck your system. Therefore, a best practice is to not request a list of representatives or teams with each support request. Instead, if making the same API call in succession, consider caching the results for a period of time and reusing them. New sessions requests should reference the cached list instead of calling for the list each time.

API Command: leave_session

The **leave_session** operation removes a customer or representative from a session. It is the opposite of the **join_session** operation, except it also works on customers. This operation is necessary to remove the end user from the Remote Support session, when the end user disconnects from the chat bot at any time.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account must have read-only or full access to the command API.

Parameters for leave_session

lsid	This is the unique Logging Session ID (LSID) of the session in which the chat message will be sent.
rep_id (optional, see note)	The unique ID of the representative to remove from the session.
rep_username (optional, see note)	The username of the representative to remove from the session.
customer (optional, see note)	Must be have a value of 1. If 1, the customer will be removed from the session.



Note: You must provide one of the optional parameters listed (**rep_id**, **rep_username**, **customer**). If none of these parameters is provided, an error is returned.

Query Examples: leave_session

leave_session	<code>https://support.example.com/api/command?action=leave_session</code>
---------------	---



IMPORTANT!

If you experience a high volume of support requests, repeatedly calling a command such as **get_logged_in_reps** or **get_support_teams** might bottleneck your system. Therefore, a best practice is to not request a list of representatives or teams with each support request. Instead, if making the same API call in succession, consider caching the results for a period of time and reusing them. New sessions requests should reference the cached list instead of calling for the list each time.

API Command: set_session_attributes

The `set_session_attributes` command sets the external key and other custom attributes for an active support session.

The API account must have full access to the command API.

Required Parameter for set_session_attributes

`lsid=[string]`

The ID of the session whose attributes you wish to set. The session must currently be active.

Optional Parameters for set_session_attributes

`session.custom.external_key=[string]`

An arbitrary string that can link this session to an identifier on an external system, such as a help desk ticket ID. This has a maximum length of 1024 characters.

`session.custom.[custom field]=[string]`

The code name and value of any custom fields. These fields must first be configured in **/login > Management > API Configuration**.

Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.



Note: If an attribute is not listed in the URL, it will keep its existing value. To clear an attribute, you must set the attribute to an empty string.

XML Response for set_session_attributes Query

`<success>`

Returns a message of **Session attributes were set** if the attributes were set successfully.

`<error>`

Returns an error message if the attributes were not set successfully.

Query Examples: set_session_attributes

Set external key for session
 c69a8e10bea9428f816cfababe9815fe

`https://support.example.com/api/command?action=set_session_attributes&lsid=c69a8e10bea9428f816cfababe9815fe&session.custom.external_key=ABC123`

Set a custom value for session
 c69a8e10bea9428f816cfababe9815fe

`https://support.example.com/api/command?action=set_session_attributes&lsid=c69a8e10bea9428f816cfababe9815fe&session.custom.custom_field1=Custom%20Value`

API Command: `get_session_attributes`

The `get_session_attributes` command returns attributes set for an active support session.

i *The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API"](#) on page 7. The API account must have read-only or full access to the command API.*

Required Parameter for `get_session_attributes`

<code>Isid=[string]</code>	The ID of the session whose attributes you wish to get. The session must currently be active.
----------------------------	---

XML Response for `get_session_attributes` Query

<code><custom_attributes></code>	Contains a <code><custom_attribute></code> element for each custom attribute set for the session.
<code><error></code>	Returns an error message if the attributes were not retrieved successfully.

Element Names and Attributes

/custom_attributes/custom_attribute

<code>display_name (attribute)</code>	The display name assigned to the custom attribute.
<code>code_name (attribute)</code>	The code name assigned to the custom attribute.

Query Example: `get_session_attributes`

Get custom attributes for session <code>c69a8e10bea9428f816cfababe9815fe</code>	<code>https://support.example.com/api/command?action=get_session_attributes&Isid=c69a8e10bea9428f816cfababe9815fe</code>
--	--

API Command: terminate_session

The **terminate_session** command terminates a support session that is in progress.

The API account must have full access to the command API.

Required Parameter for terminate_session

Isid=[string]

The unique ID representing the session you wish to terminate.

XML Response for terminate_session Query

<success>

Returns a message of **Successfully terminated** if the termination was successful.

<error>

Returns an error message if the termination was not successful.

Query Examples: terminate_session

Session
da4b510978a541d49398e88c66e28475
terminated

[https://support.example.com/api/command?
action=terminate_session&
Isid=da4b510978a541d49398e88c66e28475](https://support.example.com/api/command?action=terminate_session&Isid=da4b510978a541d49398e88c66e28475)

API Command: get_appliances

The `get_appliances` command returns XML data containing all configured B Series Appliances in a failover relationship or cluster.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#). The API account must have read-only or full access to the command API.

XML Response for get_appliances

`<appliances>`

Contains an `<appliance>` element for each connected B Series Appliance. If an error occurs, it will contain an `<error>` element describing the problem.

Element Names and Attributes

/appliances/appliance

<code>id</code> (attribute)	The B Series Appliance's GUID.
<code><name></code>	The name of the B Series Appliance.
<code><public_hostname></code>	The public hostname of the B Series Appliance.
<code><cluster_role></code>	The role the B Series Appliance plays in a cluster. Can be one of single (if not part of a cluster), primary , or traffic .
<code><failover_role></code>	The role the B Series Appliance plays in a failover relationship. Can be one of none (if failover is not configured), primary , or backup .
<code><accepting_connections></code>	Integer value (1 or 0) indicating if this B Series Appliance accepts new client connections. If not part of a cluster, this will always be 1 .
<code><timezone_offset></code>	The number of seconds away from UTC.

Query Example: get_appliances

`get_appliances`


https://support.example.com/api/command?action=get_appliances

API Command: `get_connected_client_list`

The `get_connected_client_list` command returns XML data containing a summary or list of all connected BeyondTrust clients.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account must have read-only or full access to the command API.

Optional Parameters for `get_connected_client_list`

<code>type=[string]</code>	The types of clients to return in the results. Can be a comma-separated list of values. Supported values are all (default), representative , support_customer , presentation_attendee , and push_agent .
<p> Note: Currently, pinned_client is not a possible value. If the count of pinned Jump Clients is needed in the summary, then all must be specified.</p>	
<code>summary_only=[boolean]</code>	To return only a summary, set this to 1.

XML Response for `get_connected_client_list`

<code><connected_client_list></code>	Contains a <code><connected_client_summary></code> element with a summary of the data. Also contains a <code><connected_client></code> element for each client currently connected to the B Series Appliance. If an error occurs, it will contain an <code><error></code> element describing the problem.
--	---

Element Names and Attributes

<i>/connected_client_list/connected_client_summary</i>	
<code><appliance_summary></code>	An <code><appliance_summary></code> element is created for each connected B Series Appliance.
<i>/connected_client_list/connected_client_summary/appliance_summary</i>	
<code>id</code> (attribute)	The B Series Appliance's GUID.
<code><count></code>	A <code><count></code> element is created for each type of client connected to this B Series Appliance.
<i>/connected_client_list/connected_client_summary/appliance_summary/count</i>	
<code>type</code> (attribute)	The type of client connected to the B Series Appliance. Can be one of representative , support_customer , presentation_attendee , push_agent , or pinned_client .
<i>/connected_client_list/connected_client</i>	
<code>type</code> (attribute)	The type of client connected to one of the clustered B Series Appliances. Can be one of representative , support_customer , presentation_attendee , or push_agent .

id (attribute)

A unique identifier which remains valid only while the client is connected.

Query Examples: `get_connected_client_list`

Get a list of all connected clients	<code>https://support.example.com/api/command?action=get_connected_client_list</code>
Get a list of all connected representatives	<code>https://support.example.com/api/command?action=get_connected_client_list&type=representative</code>
Get a list of all connected representatives and support customers	<code>https://support.example.com/api/command?action=get_connected_client_list&type=representative,support_customer</code>
Get a summary of all connected clients	<code>https://support.example.com/api/command?action=get_connected_client_list&summary_only=1</code>
Get a summary of all connected representatives	<code>https://support.example.com/api/command?action=get_connected_client_list&summary_only=1&type=representative</code>
Get a summary of all connected representatives and support customers	<code>https://support.example.com/api/command?action=get_connected_client_list&summary_only=1&type=representative,support_customer</code>

API Command: `get_connected_clients`

The `get_connected_clients` command returns XML data containing details of all connected BeyondTrust clients.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#). The API account must have read-only or full access to the command API.

Required Parameters for `get_connected_clients`

<code>type=[string]</code>	The types of clients to return in the results. Can be a comma-separated list of values. Supported values are all (default), representative , support_customer , presentation_attendee , and push_agent .
<code>id=[string]</code>	The ID of the client. To get client IDs, see "API Command: get_connected_client_list" on page 42 . Can be a comma-separated list of values. A maximum of 100 IDs is supported. This ID is a unique identifier which remains valid only while the client is connected.
<code>include_connections=[boolean]</code>	If this is set to 1 , then the client's list of connections to B Series Appliances and an event log about those connections will be included in the results.

XML Response for `get_connected_clients`

<code><connected_clients></code>	Contains a child element for each connected client, including <code><connected_representative></code> , <code><connected_support_customer></code> , <code><connected_presentation_attendee></code> , and <code><connected_push_agent></code> .
--	--

Element Names and Attributes

`/connected_clients/connected_representative`

<code>id</code> (attribute)	A unique identifier which remains valid only while the client is connected.
<code><client_connections></code>	Contains a <code><client_connections></code> element and an <code><event_log></code> element. This element is returned only if the query specifies include_connections .
<code><hostname></code>	The hostname of the representative's computer.
<code><platform></code>	The operating system of the representative's computer. Also contains an id attribute that briefly notes the selected platform for the client.
<code><timezone_offset></code>	The number of seconds away from UTC.
<code><connected_since></code>	The date and time at which this connection was made. Data is returned in ISO 8601 format. Also contains a ts attribute which displays the connection start time as a UNIX timestamp (UTC). This element is returned only if the query specifies include_connections .

<user_id>	Unique ID assigned to the representative.
<type>	The type of account the representative is using. Can be one of Normal or Invited .
<username>	The username assigned to the representative.
<public_display_name>	The public display name assigned to the representative. Note that this field contains the public display name's value at the time of the conference, which may not match the current value if the public_display_name has subsequently been changed.
<private_display_name>	The private display name assigned to the representative. Note that this field contains the private display name's value at the time of the conference, which may not match the current value if the private_display_name has subsequently been changed.
<start_session_url>	A URL that can be sent to a customer to start a support session with the representative.
<presentation_count>	The number of presentations the representative is performing. Can be either 0 or 1 .
<support_session_count>	The number of sessions the representative is participating in.
<showing_on_rep_list>	Integer value (1 or 0) indicating if the representative appears in the representative list on the public site.
<routing_idle>	Integer value (1 or 0) indicating if the representative has a status of idle.
<routing_busy>	Integer value (1 or 0) indicating if the representative has a status of busy.
<routing_enabled>	Integer value (1 or 0) indicating if the representative has automatic session assignment enabled or disabled.
<routing_available>	Integer value (1 or 0) indicating if the representative is available to have sessions automatically assigned.
<support_license>	The type of license used by the representative.
<support_session_isids>	Contains an <lsid> element for each session in which the representative is participating. This field corresponds with the <lsid> field of the <connected_support_customer> element.

/connected_clients/connected_support_customer

id (attribute)	A unique identifier which remains valid only while the client is connected.
<client_connections>	Contains a <client_connections> element and an <event_log> element. This element is returned only if the query specifies include_connections .
<hostname>	The hostname of the customer's computer.
<platform>	The operating system of the customer's computer. Also contains an id attribute that briefly notes the selected platform for the client.
<timezone_offset>	The number of seconds away from UTC.
<connected_since>	The date and time at which this connection was made. Data is returned in ISO 8601 format. Also contains a ts attribute which displays the connection start time as a UNIX timestamp (UTC). This element is returned only if the query specifies include_connections .

<code><name></code>	The name which the customer entered in the Your Name field of the front-end survey or which was assigned programmatically.
<code><non_interactive></code>	Indicates if the session is a remote desktop protocol (RDP) session or a Shell Jump session. Can be either rdp or shelljump . If neither, this element is not returned.
<code><lsid></code>	A string which uniquely identifies this session. This field corresponds with the <code><lsid></code> field of the <code><connected_representative></code> element.

/connected_clients/connected_presentation_attendee

<code>id</code> (attribute)	A unique identifier which remains valid only while the client is connected.
<code><client_connections></code>	Contains a <code><client_connections></code> element and an <code><event_log></code> element. This element is returned only if the query specifies include_connections .
<code><hostname></code>	The hostname of the attendee's computer.
<code><platform></code>	The operating system of the attendee's computer. Also contains an <code>id</code> attribute that briefly notes the selected platform for the client.
<code><timezone_offset></code>	The number of seconds away from UTC.
<code><connected_since></code>	The date and time at which this connection was made. Data is returned in ISO 8601 format. Also contains a ts attribute which displays the connection start time as a UNIX timestamp (UTC). This element is returned only if the query specifies include_connections .
<code><name></code>	The name which the attendee entered when joining the presentation or which was assigned programmatically.

/connected_clients/connected_push_agent

<code>id</code> (attribute)	A unique identifier which remains valid only while the client is connected.
<code><client_connections></code>	Contains a <code><client_connection></code> element and an <code><event_log></code> element. This element is returned only if the query specifies include_connections .
<code><hostname></code>	The hostname of the Jumpoint's host computer.
<code><platform></code>	The operating system of the Jumpoint's host computer. Also contains an <code>id</code> attribute that briefly notes the selected platform for the client.
<code><timezone_offset></code>	The number of seconds away from UTC.
<code><connected_since></code>	The date and time at which this connection was made. Data is returned in ISO 8601 format. Also contains a ts attribute which displays the connection start time as a UNIX timestamp (UTC). This element is returned only if the query specifies include_connections .
<code><name></code>	The Jumpoint's name.

/client_connection

<appliance_id>	The GUID of the B Series Appliance to which the client is connected.
<purpose>	The reason the representative is connected to this B Series Appliance. Can be either primary or traffic . If not part of a cluster, this will always be primary .
<receive_traffic_node>	Integer value (1 or 0) indicating whether this is the client's default traffic node or not. If not part of a cluster, this will always be 0 .
<connected_since>	The date and time at which the client connected. Data is returned in ISO 8601 format. Also contains a ts attribute which displays the connection start time as a UNIX timestamp (UTC).
<private_ip>	The client's private IP address that was used to connect to the B Series Appliance.

/event_log

<event>	<p>An <event> element is created for each event that took place during this connection. Up to the last 20 events are returned.</p> <p>Events detail when and why a client connected to a B Series Appliance. Events also include failures to connect to nodes and normal disconnects.</p> <p>Includes a ts attribute which displays the timestamp of the event.</p>
---------	---

Query Examples: `get_connected_clients`

Get a detailed list of all connected clients	<code>https://support.example.com/api/command?action=get_connected_clients</code>
Get a detailed list of all connected representatives	<code>https://support.example.com/api/command?action=get_connected_clients&type=representative</code>
Get a detailed list of all connected representatives and support customers	<code>https://support.example.com/api/command?action=get_connected_clients&type=representative,support_customer</code>
Get a detailed list of all clients with IDs 101, 102, and 103	<code>https://support.example.com/api/command?action=get_connected_clients&id=101,102,103</code>
Get a detailed list of all clients with IDs 101, 102, and 103 AND whose type is representative or customer	<code>https://support.example.com/api/command?action=get_connected_clients&id=101,102,103&type=representative,support_customer</code>
Get a detailed list, with connection information, of all connected clients	<code>https://support.example.com/api/command?action=get_connected_clients&include_connections=1</code>
Get a detailed list, with connection information, of all connected representatives	<code>https://support.example.com/api/command?action=get_connected_clients&type=representative&include_connections=1</code>
Get a detailed list, with connection information, of all connected representatives	<code>https://support.example.com/api/command?action=get_connected_clients&</code>

and support customers	<code>type=representative,support_customer&include_connections=1</code>
Get a detailed list, with connection information, of all clients with IDs 101, 102, and 103	<code>https://support.example.com/api/command?action=get_connected_clients&id=101,102,103&include_connections=1</code>
Get a detailed list, with connection information, of all clients with IDs 101, 102, and 103 AND whose type is representative or customer	<code>https://support.example.com/api/command?action=get_connected_clients&id=101,102,103&type=representative,support_customer&include_connections=1</code>

API Command: check_health

The `check_health` command returns XML data containing information about the BeyondTrust Appliance B Series, specifically including information needed for failover purposes.

i The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account must have read-only or full access to the command API.

XML Response for check_health Query

<code><appliance></code>	The hostname of the B Series Appliance. Also contains an <code>id</code> attribute that contains the B Series Appliance's GUID.
<code><version></code>	The version number and build number of the BeyondTrust software running on the B Series Appliance.
<code><success></code>	Integer value (1 or 0) indicating if the health check of the B Series Appliance was successful.
<code><error_message></code>	Returns an error message if a problem is found. If no error is found, this element will not be returned.
<code><failover_role></code>	The role the B Series Appliance plays in the failover relationship. Can be one of none (if failover is not configured), primary , or backup .
<code><enabled_shared_ips></code>	Contains an <code><ip></code> element for each IP address which is shared between the primary and backup B Series Appliances. If no shared IP addresses are enabled, this element will not be returned.
<code><last_data_sync_time></code>	The date and time at which the last data sync occurred between the primary and backup B Series Appliances. Data is returned in ISO 8601 format. Also contains a <code>ts</code> attribute which displays the data sync time as a UNIX timestamp (UTC).
<code><last_data_sync_status></code>	Contains a string showing the status of the last data sync.

Query Example: check_health

<code>check_health</code>	<code>https://support.example.com/api/command?action=check_health</code>
---------------------------	--

Alternative Method - HTTP Status Check

In addition to or alternative to using the API command above, you can use https://support.example.com/check_health to check the health of a B Series Appliance. This returns an HTTP status of 200 if the probe is successful and 500 (Server Error) if not. While you will see a simple human-readable message showing success or failure, no other data is exposed.

API Command: set_failover_role

The `set_failover_role` command sets the failover role of a B Series Appliance to either primary or backup.

The API account must have full access to the command API.

Required Parameter for set_failover_role

<code>role=[string]</code>	The role to assign to this B Series Appliance. Can be either primary or backup .
----------------------------	--

Optional Parameters for set_failover_role

<code>data_sync_first=[boolean]</code>	To perform a data sync with the peer B Series Appliance before failing over, set this to 1 . All users on the existing primary B Series Appliance will be disconnected during the data sync, and no other operations will be available until the swap is complete. To fail over without a final data sync, set this to 0 .
<code>force=[boolean]</code>	This option is only applicable when contacting the primary B Series Appliance and attempting to set its role to backup. If this is set to 1 , then this B Series Appliance will become the backup even if the peer B Series Appliance cannot be contacted.

XML Response for set_failover_role Query

<code><success></code>	If a data sync is being performed first, returns a message of Successfully started data sync. Role change will occur upon successful completion . Otherwise, returns a message of Successfully changed role .
<code><error></code>	Returns an error message if the role was not set successfully.

Query Examples: set_failover_role

Set failover role to primary	<code>https://support.example.com/api/command?action=set_failover_role&role=primary</code>
Set failover role to backup	<code>https://support.example.com/api/command?action=set_failover_role&role=backup</code>
Set failover role to primary and perform a data sync	<code>https://support.example.com/api/command?action=set_failover_role&role=primary&data_sync_first=1</code>
Set failover role to backup and perform a data sync	<code>https://support.example.com/api/command?action=set_failover_role&role=backup&data_sync_first=1</code>
Set failover role to backup even if the primary	<code>https://support.example.com/api/command?</code>

B Series Appliance cannot be contacted

action=**set_failover_role**&role=**backup**&force=**1**

Set failover role to backup even if the primary B Series Appliance cannot be contacted, and perform a data sync


<https://support.example.com/api/command?>
action=**set_failover_role**&role=**backup**&data_sync_first=**1**&
force=**1**

API Command: import_jump_shortcut

The `import_jump_shortcut` command creates a Jump shortcut. When dealing with a large number of Jump shortcuts, it may be easier to import them programmatically than to add them one by one in the representative console.

The API account must have full access to the command API.

Required Parameters for import_jump_shortcut - Local Jump

<code>name=[string]</code>	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
<code>local_jump_hostname=[string]</code>	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
<code>group=[string]</code>	The code name of the Jump Group with which this Jump Item should be associated.
	 Note: When using the <code>import</code> method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcut - Local Jump

<code>tag=[string]</code>	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
<code>comments=[string]</code>	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
<code>jump_policy=[string]</code>	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
<code>session_policy=[string]</code>	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.
<code>public_portal=[string]</code>	The public portal through which this Jump Item should connect.

Required Parameters for import_jump_shortcut - Remote Jump

<code>name=[string]</code>	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
<code>remote_jump_hostname=[string]</code>	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
<code>jumpoint=[string]</code>	The code name of the Jumpoint through which the endpoint is accessed.
<code>group=[string]</code>	The code name of the Jump Group with which this Jump Item should be associated.



Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcut - Remote Jump

tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
session_policy=[string]	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.

Required Parameters for import_jump_shortcut - Remote VNC

remote_vnc_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
jumpoint=[string]	The code name of the Jumpoint through which the endpoint is accessed.
name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
group=[string]	The code name of the Jump Group with which this Jump Item should be associated.


Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcut - Remote VNC

port=[integer]	A valid port number from 100 to 65535 . Defaults to 5900 .
tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to

	this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.


Required Parameters for import_jump_shortcut - Local VNC

name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
group=[string]	The code name of the Jump Group with which this Jump Item should be associated. <div style="border: 1px solid black; background-color: #e0f0ff; padding: 5px; margin-top: 10px;">  Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items. </div>
local_vnc_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.

Optional Parameters for import_jump_shortcut - Local VNC

tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.

Required Parameters for import_jump_shortcut - Remote Desktop Protocol


name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
remote_rdp_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
jumpoint=[string]	The code name of the Jumpoint through which the endpoint is accessed.
group=[string]	The code name of the Jump Group with which this Jump Item should be associated. <div style="border: 1px solid black; background-color: #e0f0ff; padding: 5px; margin-top: 10px;">  Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items. </div>

Optional Parameters for import_jump_shortcut - Remote Desktop Protocol

rdp_username=[string]	The username to sign in as.
domain=[string]	The domain the endpoint is on.
display_size=[string]	The resolution at which to view the remote system. Can be primary (default - the size of your primary monitor), all (the size of all of your monitors combined), or XxY (where X and Y are a supported width and height combination - e.g., 640x480).
quality=[string]	The quality at which to view the remote system. Can be black_white (black and white for lowest bandwidth consumption), few_colors (8-bit color quality for fast performance), more_colors (16-bit color for medium color quality image and performance), full_colors (32-bit for true color reproduction), or video_opt (VP9 codec for more fluid video). This cannot be changed during the remote desktop protocol (RDP) session.
console=[boolean]	1 : Starts a console session. 0 : Starts a new session (default).
ignore_untrusted=[boolean]	1 : Ignores certificate warnings. 0 : Shows a warning if the server's certificate cannot be verified.
tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.
session_policy=[string]	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.

Required Parameters for import_jump_shortcut - Local Remote Desktop Protocol

name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
rdp_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
group=[string]	The code name of the Jump Group with which this Jump Item should be associated.



Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcuts - Local Remote Desktop Protocol

rdp_username=[string]	The username to sign in as.
domain=[string]	The domain the endpoint is on.
display_size=[string]	The resolution at which to view the remote system. Can be primary (default - the size of your primary monitor), all (the size of all of your monitors combined), or XxY (where X and Y are a supported width and height combination - e.g., 640x480).
quality=[string]	The quality at which to view the remote system. Can be black_white (black and white for lowest bandwidth consumption), few_colors (8-bit color quality for fast performance), more_colors (16-bit color for medium color quality image and performance), full_colors (32-bit for true color reproduction), or video_opt (VP9 codec for more fluid video). This cannot be changed during the remote desktop protocol (RDP) session.
console=[boolean]	1 : Starts a console session. 0 : Starts a new session (default).
ignore_untrusted=[boolean]	1 : Ignores certificate warnings. 0 : Shows a warning if the server's certificate cannot be verified.
tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.
session_policy=[string]	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.

Required Parameters for import_jump_shortcut - Shell Jump Shortcut

name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
shelljump_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
jumpoint=[string]	The code name of the Jumpoint through which the endpoint is accessed.
protocol=[string]	Can be either ssh or telnet .
group=[string]	The code name of the Jump Group with which this Jump Item should be associated.



Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcut - Shell Jump Shortcut

shelljump_username=[string]	The username to sign in as.
port=[integer]	A valid port number from 1 to 65535. Defaults to 22 if the protocol is ssh or 23 if the protocol is telnet.
terminal=[string]	Can be either xterm (default) or VT100.
keep_alive=[integer]	The number of seconds between each packet sent to keep an idle session from ending. Can be any number from 0 to 300. 0 disables keep-alive (default).
tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.
session_policy=[string]	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.

Required Parameters for import_jump_shortcut - Intel vPro Shortcut

name=[string]	The name of the endpoint to be accessed by this Jump Item. This name identifies the item in the session tabs. This string has a maximum of 128 characters.
vpro_hostname=[string]	The hostname of the endpoint to be accessed by this Jump Item. This string has a maximum of 128 characters.
jumpoint=[string]	The code name of the Jumpoint through which the endpoint is accessed.
group=[string]	The code name of the Jump Group with which this Jump Item should be associated.

Note: When using the import method, a Jump Item cannot be associated with a personal list of Jump Items.

Optional Parameters for import_jump_shortcuts - Intel vPro Shortcut

tag=[string]	You can organize your Jump Items into categories by adding a tag. This string has a maximum of 1024 characters.
comments=[string]	You can add comments to your Jump Items. This string has a maximum of 1024 characters.
jump_policy=[string]	The code name of a Jump Policy. You can specify a Jump Policy to manage access to this Jump Item.
public_portal=[string]	The public portal through which this Jump Item should connect.
session_policy=[string]	The code name of a session policy. You can specify a session policy to manage the permissions available on this Jump Item.

XML Response for import_jump_shortcut Query

<success>	Returns a message of Successfully imported Jump Item shortcut if the import succeeded.
<error>	Returns an error message if the import failed.

Query Examples: import_jump_shortcut

Import a Local Jump shortcut to the endpoint with hostname "ABCDEF02", pinning it to team "remote_access"	<code>https://support.example.com/api/command?action=import_jump_shortcut&name=Endpoint&local_jump_hostname=ABCDEF02&group=remote_access</code>
Import a Local Jump shortcut to the endpoint with hostname "ABCDEF02", pinning it to team "remote_access" and specifying its tag, comments, Jump Policy, and Session Policy	<code>https://support.example.com/api/command?action=import_jump_shortcut&name=Endpoint&local_jump_hostname=ABCDEF02&group=remote_access&tag=Frequent%20Access&comments=Web%20server&jump_policy=Notify&session_policy=Servers</code>
Import a Remote Jump shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London", and pinning it to team "remote_access"	<code>https://support.example.com/api/command?action=import_jump_shortcut&remote_jump_hostname=ABCDEF02&jumpoint=London&group=remote_access</code>
Import a Remote Desktop Protocol shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London", and pinning it to team "remote_access"	<code>https://support.example.com/api/command?action=import_jump_shortcut&rdp_hostname=ABCDEF02&jumpoint=London&group=remote_access</code>
Import a Remote Desktop Protocol shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London", and	<code>https://support.example.com/api/command?action=import_jump_shortcut&rdp_hostname=ABCDEF02&jumpoint=London&group=remote_access&rdp_username=admin&</code>

pinning it to team "remote_access". Set the username, domain, display size, and quality. Make it a console session, and ignore untrusted certificates.

```
domain=example&display_size=1280x720&
quality=more_colors&console=1&ignore_untrusted=1
```

Import a Shell Jump shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London" on SSH, and pinning it to team "remote_access"

```
https://support.example.com/api/command?action
=import_jump_shortcut&shelljump_hostname=ABCDEF02&
jumpoint=London&protocol=ssh&group=remote_access
```

Import a Shell Jump shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London" on SSH, and pinning it to team "remote_access". Set the username, port, and terminal type, and set the keep-alive time to two minutes.

```
https://support.example.com/api/command?action
=import_jump_shortcut&shelljump_hostname=ABCDEF02&
jumpoint=London&protocol=ssh&group=remote_access&
shelljump_username=admin&port=25&terminal=vt100&keep_alive=120
```

Import a Intel® vPro shortcut to the endpoint with hostname "ABCDEF02", accessed through Jumpoint "London", and pinning it to team "remote_access"

```
https://support.example.com/api/command?action
=import_jump_shortcut&vpro_hostname=ABCDEF02&
jumpoint=London&group=remote_access
```

API Command: get_api_info

The `get_api_info` request returns XML data containing the current API version information.

i *The command API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API"](#) on page 7. The API account must have read-only or full access to the command API.*

XML Response for get_api_info Query

<code><api_version></code>	The software version of the current BeyondTrust API.
<code><company_name></code>	The Company API Name listed on the Status > Information page of the /login administrative interface.
<code><timestamp></code>	The server's current timestamp at the time this report was pulled.
<code><permissions></code>	The permissions of the API account used to issue this command. The permissions shown are detailed below.

Element Names and Attributes

/get_api_info/permissions/permission

<code>perm_backup</code>	Integer value (1 or 0) indicating if the API account has permission to use the backup API.
<code>perm_command</code>	String indicating if the API account has full access to the command API, read_only access, or no access (deny).
<code>perm_configuration</code>	Integer value (1 or 0) indicating if the API account has permission to use the Configuration API.
<code>perm_configuration_vault_account</code>	Integer value (1 or 0) indicating if the API account has permission to manage Vault accounts via the Configuration API.
<code>perm_ecm</code>	Integer value (1 or 0) indicating if the API account can be used by an Endpoint Credential Manager client to connect to the appliance.
<code>perm_real_time_state</code>	Integer value (1 or 0) indicating if the API account has permission to use the Real-Time State API.
<code>perm_reporting_archive</code>	Integer value (1 or 0) indicating if the API account has permission to use the archive reporting API.
<code>perm_reporting_license</code>	Integer value (1 or 0) indicating if the API account has permission to use the license usage reporting API.
<code>perm_reporting_presentation</code>	Integer value (1 or 0) indicating if the API account has permission to use the presentation session reporting API.
<code>perm_reporting_support</code>	Integer value (1 or 0) indicating if the API account has permission to use the support

	session reporting API.
perm_reporting_vault	Integer value (1 or 0) indicating if the API account has permission to run Vault Activity reports.
perm_vault_backup	Integer value (1 or 0) indicating if the API account has permission to backup the Vault encryption key, the key used to encrypt and decrypt all the Vault credentials stored on the appliance.

Query Example: get_api_info

get_api_info	https://support.example.com/api/command?action=get_api_info
--------------	---

Representative Console Scripting and Client Scripting API

The BeyondTrust Representative Console Scripting feature is composed of three parts:

1. The BeyondTrust Representative Console Script file format
2. New and deprecated command line parameters for the representative console
3. The BeyondTrust client scripting API

The BeyondTrust Representative Console Script File

A BeyondTrust Representative Console Script (BRCS) is a file that contains a sequence of commands to be executed by the BeyondTrust representative console. The file extension is in the format "brcs-*<companySiteName>*" (Company Site Name is the name used to access your support site). During installation the BeyondTrust representative console will use the OS to associate the representative console with the BRCS file type. Therefore users can double-click a BRCS file and have it automatically executed by the BeyondTrust representative console.

BRCS files have the following format:

```
BRCS1.0
<command>
<command>
...
```

This is more formally expressed as:

```
brcs_file = header , newline , commands ;
header = "BRCS" , version ;
version = digit , "." , digit ;
commands = command { newline , command } ;
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
newline = "\n" | "\r\n" ;
```



Note: Note that script files can have a maximum of 10 commands.

Each command consists of a set of key-value pairs separated by "&". The key in each pair is separated from the value by "=". Keys and values use the percent-encoding algorithm described in [RFC3986 section 2.1](#). This is commonly referred to as url-encoding or url-escaping. It is commonly seen in the address bar of web browsers to represent the parameters passed to a web server. Commands have the following format:

```
action=<action>&parameter1=value1&parameter2=value2...
```

This is more formally expressed as:

```
command = "action=", value, [ parameters ] ;
parameters = "&", parameter, [ parameters ] ;
parameter = url_encoded_string, "=", url_encoded_string ;
url_encoded_string = { * see RFC 3986 * } ;
```

New and Deprecated Command Line Parameters for the Representative Console

Two command line parameters have been added to the representative console to support BRCS:

```
run-script <BRCS command>
run-script-file <path to BRCS file>
```

These command line parameters allow customers to implement BRCS login via the command line. These two new parameters overlap with two existing parameters. Therefore, the "-jump" and "-push" command line parameters are now deprecated and will be removed in a future release.

Example

Old Command Line	New Command Line
push<hostname>	run-script "action=push_and_start_local&hostname=<hostname>"
jump<search string>	run-script "action=start_pinned_client_session&search_string=<search_string>"

Different behaviors can be seen when running a script from the command line depending on the state of the representative console:

- If the representative console is not running, then attempting to run a script from the command line causes the representative console to start the login dialog. After the representative successfully logs in, the script is run.
- If the representative console is already running, but the representative is not logged in, then the login dialog is shown. After the representative logs in, the script is run.
- If the representative console is already running and the representative is already logged in, then attempting to run a script from the command line causes the existing instance of the representative console to run the script.

Representative console exit status:

- If an invalid script is given on the command line, then the representative console will terminate with an exit status > 0.
- If a valid script is given on the command line, then the representative console will terminate with an exit status of 0.

Examples:

```
bomgar-rep.exe --run-script "action=generate_session_key&session.custom.external_key=123456789"
bomgar-rep.exe --run-script-file my_script_file.brsc-beta60
```

The BeyondTrust Client Scripting API

The client scripting API enables you to generate a BeyondTrust Representative Console Scripting (BRCS) file which allows you to send commands to the BeyondTrust representative console from external applications.

Customers can use the client scripting API to generate BRCS files that can start a support session with a specific Jump Client, push and start a session with a Windows system within a local network, prompt representatives to generate a session key, start a vPro session with a specified system, or to simply log in to the representative console.



The client scripting API URL is https://support.example.com/api/client_script.

This API accepts a client type (**rep**), an operation to perform (**generate**), a command to put in the script file, and a set of parameters to pass to the command. Here is an example of a valid Client Scripting API request:


```
https://support.example.com/api/client_script?type=rep&operation=generate&action=start_pinned_client_session&search_string=ABCDEFG02
```

The above request will prompt the user to download a BeyondTrust representative console script file. After downloading the script file, the user can run it using the representative console. In this case, the script file will contain commands to start a session with the Jump Client whose hostname, comments, public IP, or private IP matches the search string "ABCDEFG02".

Parameters for Client Scripting API

type=rep type=web_console	Currently the API only supports rep or web_console as the client type.										
operation=generate operation=execute	The operation to perform. Currently the API only supports generate and execute as the operation. <div style="border: 1px solid #0070c0; background-color: #e6f2ff; padding: 5px; margin-top: 10px;">  Note: If the type is rep, the operation should be generate. If the type is web_console, the operation should be execute. </div>										
action=<command> or action=<command>¶meter=[value]	The name of the command to run. Most commands have required and/or optional parameters. Available actions include: <table style="width: 100%; border: none; margin-top: 10px;"> <tr> <td><u>login</u></td> <td><u>start_rdp_session</u></td> </tr> <tr> <td><u>generate_session_key</u></td> <td><u>start_shell_jump_session</u></td> </tr> <tr> <td><u>push_and_start_local</u></td> <td><u>start_vpro_session</u></td> </tr> <tr> <td><u>push_and_start_remote</u></td> <td><u>start_vnc_session</u></td> </tr> <tr> <td><u>start_jump_item_session</u></td> <td><u>delete_script_file</u></td> </tr> </table> <p>Two actions are automatically added to the BRCS file: login and delete_script_file. The delete_script_file action has no parameters.</p> <div style="border: 1px solid #0070c0; background-color: #e6f2ff; padding: 5px; margin-top: 10px;">  Note: The type web_console supports only the generate_session_key and start_jump_item_session actions. </div>	<u>login</u>	<u>start_rdp_session</u>	<u>generate_session_key</u>	<u>start_shell_jump_session</u>	<u>push_and_start_local</u>	<u>start_vpro_session</u>	<u>push_and_start_remote</u>	<u>start_vnc_session</u>	<u>start_jump_item_session</u>	<u>delete_script_file</u>
<u>login</u>	<u>start_rdp_session</u>										
<u>generate_session_key</u>	<u>start_shell_jump_session</u>										
<u>push_and_start_local</u>	<u>start_vpro_session</u>										
<u>push_and_start_remote</u>	<u>start_vnc_session</u>										
<u>start_jump_item_session</u>	<u>delete_script_file</u>										

API Script Command: login

When generating any BeyondTrust Representative Console Script, the **login** command is automatically added as the first command in the script file. It does not need to be specified in the URL used to generate the script file.

By default, this command opens the representative console and attempts to log in using the credentials saved locally in the representative console. If no credentials are saved, the command simply opens the representative console login prompt. Once the representative has correctly authenticated, the script continues running.

The **login** command has no effect if a representative is already logged into the representative console.

If you wish to specify the credentials to be used, you can create a separate script specifically to be used for logging in. The **login** command passes the login mechanism along with a username and password. Both username and password parameters are sent in plain text, unencrypted.



IMPORTANT!

*You cannot specify multiple commands in the URL used to generate a script. For example, you cannot specify **login** and an action such as **generate_session_key** in the same URL. Each command must be generated as a separate script.*

*However, a skilled developer may edit the **.brcs** script file once it has been generated in order to modify the login credentials and then run another command. BeyondTrust does not support scripts modified in this manner.*

Optional Parameters for login

mechanism=[string]	The mechanism to use for authentication. Currently, only username_password is supported. If this parameter is supplied, both other parameters must also be supplied.
username=[string]	The username of the account with which to log in. If this parameter is supplied, both other parameters must also be supplied.
password=[string]	The password of the account with which to log in. If this parameter is supplied, both other parameters must also be supplied.

Query Examples: login

Log in to the representative console, specifying the username and password	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=login&mechanism=username_password&username=username&password=password</code>
--	---

API Script Command: generate_session_key

In the context of the client scripting API, the **generate_session_key** command causes the representative console to show the Generate Session Key dialog. Parameters can be passed to the command to customize the behavior.

Optional Parameters for the generate_session_key Command

<code>public_portal_hostname=[string]</code>	The hostname of the public portal that should be selected by default when the Generate Session Key dialog is shown. The representative will still have the option to change the public portal on the dialog. This field has a maximum length of 255 characters.
<code>session.custom.[custom field]=[string]</code>	The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration . Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.



Note: Parameters are optional for the **generate_session_key** command. Omitting them will simply cause the representative console to show the **Generate Session Key** dialog.

Query Examples: generate_session_key

Show the Generate Session Key dialog	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=generate_session_key</code>
Show the Generate Session Key dialog with the public portal hostname "support.example.com" selected	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=generate_session_key&public_portal_hostname=support.example.com</code>
Show the Generate Session Key dialog and associate custom attributes with any support sessions started using the session key or URL shown on the dialog	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=generate_session_key&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123</code>

API Script Command: `push_and_start_local`

The `push_and_start_local` command attempts to push the customer client to a computer on the local network to start a support session. This can also be described as a local Jump.

Required Parameter for `push_and_start_local`

`hostname=[string]`

The hostname of the computer that is the target of the push and start operation. This field has a maximum length of 255 characters.

Optional Parameter for `push_and_start_local`

`session.custom.[custom field]=[string]`

The code name and value of any custom fields. These fields must first be configured in **/login > Management > API Configuration**.

Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.

Query Examples: `push_and_start_local`

Jump to the local network computer "ABCDEF02"

`https://support.example.com/api/client_script?type=rep&operation=generate&action=push_and_start_local&hostname=ABCDEF02`

Jump to the local network computer "ABCDEF02" and associate custom attributes with the session

`https://support.example.com/api/client_script?type=rep&operation=generate&action=push_and_start_local&hostname=ABCDEF02&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123`

API Script Command: `push_and_start_remote`

The `push_and_start_remote` command attempts to push the customer client to a computer on a remote network through a Jumpoint in order to start a support session. This can also be described as a remote Jump.

Required Parameter for `push_and_start_remote`

<code>target=[string]</code>	The hostname or IP address of the target machine.
------------------------------	---

Optional Parameters for `push_and_start_remote`

<code>jumpoint=[string]</code>	<p>The Jumpoint through which to start the session. This Jumpoint must be on the same subnet as the target computer.</p> <p>If not specified and the user has access to only one Jumpoint, then that Jumpoint is used automatically. If not specified and the user has access to more than one Jumpoint, then a dialog will open from which the user must select a Jumpoint.</p>
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>


Query Examples: `push_and_start_local`

Jump to the remote computer "ABCDEF02" through the Jumpoint "Network01"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=push_and_start_remote&target=ABCDEF02&jumpoint=Network01</code>
Jump to the remote computer "ABCDEF02" through the Jumpoint "Network01" and associate custom attributes with the session	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=push_and_start_remote&target=ABCDEF02&jumpoint=Network01&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123</code>

API Script Command: `start_jump_item_session`

The `start_jump_item_session` command attempts to start a session with a BeyondTrust Jump Item. Users may run this command for all Jump Items they are permitted to access via the Jump management interface in the representative console.

Optional Parameters for the `start_jump_item_session` Command

<code>jump.method</code>	If specified, only Jump Items using the designated Jump method are included in the results. Acceptable values for this field are push (remote push), local_push , pinned (Jump Client), rdp , vnc , and shelljump .
<code>credential_id</code>	If specified, only a Jump Item with that specific credential ID associated is returned. This field has a maximum length of 255 characters.
<code>search_string</code>	Identifies the search criteria used to select and return specific Jump Items as results. <div style="border: 1px solid black; background-color: #e0f0ff; padding: 5px; margin-top: 10px;">  Note: This parameter is required only if no of the client fields below are specified. </div>
<code>client.comments</code>	If specified, only Jump Items with the given comments are included in the results. This field has a maximum length of 255 characters. Search is partial and case-insensitive.
<code>client.hostname</code>	If specified, only Jump Items with the given hostname are included in the results. This field has a maximum length of 255 characters. Search is partial and case-insensitive.
<code>client.private_ip</code>	If specified, only Jump Clients with the given private IP address are included in the results. This search field applies only to Jump Clients. This field has a maximum length of 255 characters. Search is partial and case-insensitive.
<code>client.public_ip</code>	If specified, only Jump Clients with the given public IP address are included in the results. This search field applies only to Jump Clients. This field has a maximum length of 255 characters. Search is partial and case-insensitive.
<code>client.tag</code>	If specified, only Jump Items with the given tag are included in the results. This field has a maximum length of 255 characters. Search is partial and case-insensitive.
<code>session.custom.[custom field]=[string]</code>	The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration . Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.


IMPORTANT!

At least one **client.*** parameter must be specified. If multiple **client.*** parameters are specified, then only clients matching all criteria are returned.

Query Examples: start_jump_item_session

Start a session with a Jump Item whose hostname contains "ABCDEF02"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_jump_item_session&client.hostname=ABCDEF02</code>
Start a session with a Jump Item whose comments contain "maintenance" and whose tag contains "server"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_jump_item_session&client.comments=maintenance&client.tag=server</code>
Start a session with a pinned Jump Client whose private IP address begins with "10.10.24" and associate custom attributes with the session	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_jump_item_session&client.private_ip=10.10.24&jump.method=pinned&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123</code>



Note: If more than one Jump Item matches the search criteria, then a dialog opens, giving the user the option to select the appropriate Jump Item.

API Script Command: start_rdp_session

The `start_rdp_session` command initiates a Microsoft Remote Desktop Protocol session on the target machine using a Jumpoint.

Required Parameter for the start_rdp_session Command

<code>target=[string]</code>	The hostname or IP address of the machine targeted for an RDP session.
------------------------------	--

Optional Parameters for the start_rdp_session Command

<code>jumpoint=[string]</code>	<p>The Jumpoint through which to start the RDP session. This Jumpoint must be on the same subnet as the target computer.</p> <p>If not specified and the user has access to only one Jumpoint, then that Jumpoint is used automatically. If not specified and the user has access to more than one Jumpoint, then a dialog will open from which the user must select a Jumpoint.</p>
<code>username=[string]</code>	The username to use when authenticating. If not specified, the representative must enter the username.
<code>display_size=[string]</code>	May be one of primary_display (default), all_displays , or <width>x<height> , where <width> and <height> are integers representing a standard resolution.
<code>quality=[string]</code>	May be one of low , performance (default), performance_quality , or quality .
<code>console=[string]</code>	Either no (default) or yes . If yes, the RDP session connects to the physical console.
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>

Query Examples: start_rdp_session

Start an RDP session with the computer "ABCDEF02"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_rdp_session&target=ABCDEF02</code>
Start an RDP session with the computer "ABCDEF02" through the Jumpoint "Network01"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_rdp_session&target=ABCDEF02&jumpoint=Network01</code>
Start an RDP session with the computer "ABCDEF02" through the Jumpoint "Network01". Authenticate with "jsmith", set the resolution to 1024x768, set the quality to "quality", and create a console session	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_rdp_session&target=ABCDEF02&jumpoint=Network01&username=jsmith&display_size=1024x768&quality=quality&console=yes</code>
Start an RDP session with the computer	<code>https://support.example.com/api/client_script?type=rep&operation=generate&</code>

"ABCDEF02" and associate custom attributes with the session

```
action=start_rdp_session&target=ABCDEF02&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123
```

API Script Command: `start_vnc_session`

The `start_vnc_session` command initiates a VNC session on the target machine using a Jumpoint.

Required Parameter for the `start_vnc_session` Command

<code>target=[string]</code>	The hostname or IP address of the machine targeted for a VNC session.
------------------------------	---

Optional Parameters for the `start_vnc_session` Command

<code>jumpoint=[string]</code>	<p>The Jumpoint through which to start the VNC session. This Jumpoint must be on the same subnet as the target computer.</p> <p>If not specified and the user has access to only one Jumpoint, then that Jumpoint is used automatically. If not specified and the user has access to more than one Jumpoint, then a dialog opens from which the user must select a Jumpoint.</p>
<code>port=[integer]</code>	The port number on which to connect. Defaults to 5900.
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>

Query Examples: `start_vnc_session`

Start a VNC session with the computer "ABCDEF02"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_vnc_session&target=ABCDEF02</code>
Start a VNC session with the computer "ABCDEF02" through the Jumpoint "Network01"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_vnc_session&target=ABCDEF02&jumpoint=Network01</code>
Start a VNC session with the computer "ABCDEF02" and associate custom attributes with the session	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_vnc_session&target=ABCDEF02&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123</code>

API Script Command: `start_shell_jump_session`

The `start_shell_jump_session` command initiates a Shell Jump session, creating an SSH or Telnet connection to a remote network device.

Required Parameter for the `start_shell_jump_session` Command

<code>target=[string]</code>	The hostname or IP address of the machine targeted for a Shell Jump session.
------------------------------	--

Optional Parameters for the `start_shell_jump_session` Command

<code>jumpoint=[string]</code>	<p>The Jumpoint through which to start the Shell Jump session. This Jumpoint must be on the same subnet as the target computer.</p> <p>If not specified and the user has access to only one Jumpoint, then that Jumpoint is used automatically. If not specified and the user has access to more than one Jumpoint, then a dialog will open from which the user must select a Jumpoint.</p>
<code>username=[string]</code>	The username to use when authenticating. If not specified, the representative must enter the username.
<code>protocol=[string]</code>	The network protocol to use. May be one of ssh (default) or telnet .
<code>port=[integer]</code>	The port number on which to connect. Defaults to 22.
<code>terminal</code>	The terminal type to use. May be one of xterm (default) or vt100 .
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>

Query Examples: `start_shell_jump_session`

Start a Shell Jump session with the computer "ABCDEF02"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_shell_jump_session&target=ABCDEF02</code>
Start a Shell Jump session with the computer "ABCDEF02" through the Jumpoint "Network01"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_shell_jump_session&target=ABCDEF02&jumpoint=Network01</code>
Start a Shell Jump session with the computer "ABCDEF02" through the Jumpoint "Network01". Authenticate with "jsmith", and use a Telnet protocol through port 40 with terminal type vt100	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_shell_jump_session&target=ABCDEF02&jumpoint=Network01&username=jsmith&protocol=telnet&port=40&terminal=vt100</code>
Start a Shell Jump session with the computer	<code>https://support.example.com/api/client_script?type=rep&operation=generate&</code>

"ABCDEF02" and associate custom attributes with the session

```
action=start_shell_jump_session&target=ABCDEF02&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123
```

API Script Command: start_vpro_session

The `start_vpro_session` command initiates a vPro session on the target machine using the specified Jumpoint.

Required Parameters for the start_vpro_session Command

<code>target=[string]</code>	The hostname or IP address of the machine targeted for a vPro session.
<code>jumpoint=[string]</code>	The Jumpoint through which to start the vPro session. This Jumpoint must be on the same subnet as the target computer and must be configured for vPro support.



Note: To initiate a vPro session using BeyondTrust Client Scripting, you must specify the target machine's hostname or private IP. (If Kerberos is used for vPro authentication, then the fully qualified domain name must be specified.) The Jumpoint name must also be specified. See the query examples below.

Optional Parameter for the start_vpro_session Command

<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>
---	---

Query Examples: start_vpro_session

Start a vPro session with the computer "ABCDEF02" through the Jumpoint "Network01"	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_vpro_session&target=ABCDEF02&jumpoint=Network01</code>
Start a vPro session with the computer "ABCDEF02" through the Jumpoint "Network01" and associate custom attributes with the session	<code>https://support.example.com/api/client_script?type=rep&operation=generate&action=start_vpro_session&target=ABCDEF02&jumpoint=Network01&session.custom.custom_field1=Custom%20Value&session.custom.custom_field2=123</code>

Session Generation API

The public site is a collection of forms and links which generates an HTTP request each time a session is requested, resulting in the customer client's being downloaded to the remote computer. The request generated depends on the configuration of the public site and the method used by the customer to request support:

- By selecting a name from a list of logged-in representatives
- By entering a unique session key
- By submitting an issue

The session generation API URL is https://support.example.com/api/start_session

The queue in which to place the customer after connecting can be specified in one of three ways:

Session Generation by Representative Name and ID

<code>id=[integer]</code>	The numeric ID for the representative with whom to start the session. To get a representative's ID, see "API Command: get_logged_in_reps" on page 23 .
<code>name=[string]</code>	The public display name for this same representative.

Session Generation by Session Key

<code>short_key=[string]</code>	The seven-character string used to start a session.
---------------------------------	---

Session Generation by Issue Submission Survey

<code>issue_menu=[integer]</code>	Must be set to 1 .
<code>codeName=[string]</code>	The code name of the selected issue. Use either codeName or id , not both.
<code>id=[integer]</code>	The numeric ID for the selected menu item. If Display Reps in Issues Menu is enabled from the Public Site Configuration page of the /login web interface, this will be the unique ID for a representative in the list. Otherwise, it will be the unique ID for an issue found in the issues list. To get a representative's ID, see "API Command: get_logged_in_reps" on page 23 . To get a list of valid issue IDs, see "API Command: get_support_teams" on page 26 . Use either codeName or id , not both.

You may also add optional parameters to pass additional information to the session.

Optional Parameters

<code>c2cjs=[integer]</code>	If set to 1, causes the session to start as a click-to-chat session. This method of starting a click-to-chat session is less preferred to using JavaScript. (See "Use JavaScript to Start Click-to-Chat or Full Client Sessions" on page 83 .)
<code>customer.name=[string]</code>	Customer's name (maximum of 100 characters).

<code>customer.company=[string]</code>	Customer's company name (maximum of 100 characters).
<code>customer.company_code=[string]</code>	Customer's company code (maximum of 100 characters).
<code>customer.details=[string]</code>	Customer's problem description (maximum of 1024 bytes). The number of characters varies depending on the language and character set used for the public portal. If in doubt, you can use one of the available online character byte count tools to help you determine your character limit.
<code>locale=[string]</code>	<p>The locale code of the language to be used for this customer client. The language must be installed on your B Series Appliance. To see which languages are installed, go to /login > Localization > Languages.</p> <p>Available language packages can include English (en-us), German (de), Latin American Spanish (es), EU Spanish (es-sp), Finnish (fi), EU French (fr), Italian (it), Dutch (nl), Brazilian Portuguese (pt-br), EU Portuguese (pt-pt), Swedish (sv-se), Turkish (tr), Japanese (ja), Simplified Chinese (zh-hans), Traditional Chinese (zh), and Russian (ru).</p>
<code>chat_locale</code>	The chat_locale parameter can be used to set the chat locale to a language that is supported by GeoFluent but is not supported by BeyondTrust. Example: https://support.example.com/api/start_session?short_key=2962035&locale=en-us&chat_locale=cs-cs .
<code>platform=[string]</code>	The specific Windows® platform (such as Windows 32-bit or 64-bit) for which to download the customer client. Use the correct parameters for the desired platform, winNT-32 or winNT-64 .
<code>session.custom.external_key=[string]</code>	A key to an external help desk ticket system (maximum of 1024 characters).
<code>session.custom.[custom field]=[string]</code>	<p>The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration.</p> <p>Each attribute must be specified as a different parameter. Each custom field has a maximum length of 1024 characters. The maximum total size of all combined custom fields, including the external key, must be limited to 10KB.</p>
<code>session.priority=[integer]</code>	The priority of the session, from 1 to 3 . 1 = high, 2 = medium, and 3 = low. The priority set by the API overrides the priority already set for the selected issue. If this parameter is not set by the API, the priority assigned to the selected issue will be used.
<code>session.skills=[string]</code>	A comma-separated list of the code names of skills to assign to a session. The skills set by the API override the skills already set for the selected issue. If this parameter is not set by the API, the skills assigned to the selected issue will be used.
<code>timezone_offset=[integer]</code>	<p>The difference in minutes between the client's local time and UTC. Positive values are east of the Prime Meridian, and negative values are west of the Prime Meridian.</p> <p>This affects sessions routed through an Atlas cluster traffic node, where the cluster is configured to use the timezone offset method for choosing traffic nodes. The traffic node must have Accepting New Client Connections enabled. The primary node redirects the client to the traffic node that is closest to the client's timezone. If multiple traffic nodes are equally close, then one is chosen at random. Clock skew between the primary node and the traffic node must be less than two minutes.</p> <p>This parameter is ignored if the B Series Appliance is not configured as above.</p>

Note that if your B Series Appliance has multiple public sites, the session created will be associated with the public site whose domain name matches the request's domain name. For example, if Site A has a hostname of support.example.com and Site B has a hostname of support.example.com, a session generation request made to support.example.com will create a session associated with Site A.

Also note the session key behavior if the option to prompt customers before downloading the customer client is enabled from **/login > Public Portals > Public Sites**. If prompting is enabled, then only the external key can be passed along with the session key. If the prompt is disabled, you can include other parameters with the session key.

Query Examples

Specific representative	<code>https://support.example.com/api/start_session?id=1&name=Admin</code>
Session key	<code>https://support.example.com/api/start_session?short_key=1234567&session.custom.external_key=1234</code>
Issue submission survey using issue ID	<code>https://support.example.com/api/start_session?issue_menu=1&customer.name=John%20Doe&customer.company=Company%20Name&customer.company_code=1234&customer.details=I%20need%20support&id=1&session.custom.external_key=1234&session.custom.custom_field1=Custom%20Value</code>
Issue submission survey using issue code name	<code>https://support.example.com/api/start_session?issue_menu=1&customer.name=John%20Doe&customer.company=Company%20Name&customer.company_code=1234&customer.details=I%20need%20support&codeName=issue_code_name&session.custom.external_key=1234&session.custom.custom_field1=Custom%20Value</code>
Issue submission survey using ID, with skills and priority	<code>https://support.example.com/api/start_session?issue_menu=1&customer.name=John%20Doe&customer.company=Company%20Name&customer.details=I%20need%20support&id=1&session.priority=1&session.skills=codename1,codename2</code>
Locale	<code>https://support.example.com/api/start_session?id=1&name=Admin&locale=en-us</code>
Platform	<code>https://support.example.com/api/start_session?id=1&name=Admin&platform=winNT-32</code>
Timezone Offset	<code>https://support.example.com/api/start_session?id=1&name=Admin&timezone_offset=-300</code>

Start Sessions with Session Key Acceptance

An alternative method of starting a session is to create a web form where your customers can enter short session key strings to start sessions with you.

To create a session key entry form, create a web form with the action of https://support.example.com/api/start_session and a method of either **GET** or **POST**. You must also use a text box with the name of **short_key**, as shown in the example below.

```
<form action="https://support.example.com/api/start_session" method="get">
  Session Key: <input type="text" name="short_key" /><br />
  <input type="submit" value="Submit" />
</form>
```

You may also include an external key to start a session.

```
<form action="https://support.example.com/api/start_session" method="get">
  Session Key: <input type="text" name="short_key" /><br />
  External Key: <input type="text" name="session.custom.external_key" /><br />
  <input type="submit" value="Submit" />
</form>
```

Using this form, your customer can enter a generated seven-character session key and an optional external key to start a session with you.

Use JavaScript to Start Click-to-Chat or Full Client Sessions

To start a session using JavaScript, you must first reference an external JavaScript file that is included on your BeyondTrust Appliance B Series. You must then tell the API the hostname from which the JavaScript files and other resources should be lazily loaded. This hostname should not include the protocol (e.g., support.example.com). Both of these elements should be included in the head of your web page, as shown in the example below.

```
<head>
  <script type="text/javascript" src="https://support.example.com/api/start_
  session.js"></script>
  <script type="text/javascript">BG.setSite("support.example.com");</script>
</head>
```

Then, within the body, you must include an event attribute to trigger a session start. In most cases, this will be an **onclick** event attribute on an anchor or button element. With this event, call the **BG.start** method, using the arguments to start the session with session key submission, representative selection, or issue submission.

```
BG.start(startType, options)
```

The **startType** can be either **BG.START_TYPE.CHAT** or **BG.START_TYPE.FULL**. This determines which type of session is launched. The **options** parameter expects a generic JavaScript object which contains set properties.

IMPORTANT!

Your HTML page must have a valid standards-compliant Doctype. View recommended Doctype declarations at <http://www.w3.org/QA/2002/04/valid-dtd-list.html>.



Note: *start_session.js* will not work with public portals that require SAML authentication. This is due to issues with Cross-Origin Resource Sharing (CORS).



For more information, please see "[options Properties](#)" on page 87.

Script Examples

Several common examples are listed below, though more are possible. Each of the sessions below can be called either of the session types by changing the start type to **BG.START_TYPE.CHAT** or **BG.START_TYPE.FULL**.

Starting a Full Client Session with a Session Key

```
BG.start(BG.START_TYPE.FULL, {
  sessionKey: '1728331'
});
```

Starting a Full Client Session with Representative Information

```
BG.start(BG.START_TYPE.FULL, {
  rep: {
    id: 30,
    name: 'Admin'
  }
});
```

Starting a Full Client Session with a Session Key and Custom Fields

```
BG.start(BG.START_TYPE.FULL, {
  sessionKey: '8679485',
  attributes: {
    external_key: 'abc123',
    custom_field1: 'Custom Value',
    custom_field2: 123
  }
});
```

Starting a Full Client Session with a Session Key and a Specified Language

```
BG.start(BG.START_TYPE.FULL, {
  sessionKey: '8679485',
  locale: 'en-us'
});
```

Starting a Full Client Session with an Issue Object (by ID) and Customer Information

```
BG.start(BG.START_TYPE.FULL, {
  issue: {
    id: 12
  },
  customer: {
    name: 'Customer Name',
    company: 'Company Name',
    company_code: 'Company Code',
    details: 'Issue Details'
  }
});
```

Starting a Full Client Session with an Issue Object (by Code Name) and Skills

```
BG.start(BG.START_TYPE.FULL, {  
  issue: {  
    codeName: 'Other'  
  }  
  skills: ["skill11", "skill12"]  
});
```

Starting a Full Client Session with an Issue Form Element

```
BG.start(BG.START_TYPE.FULL, {  
  issue: document.getElementById('formID')  
});
```

Example Issue Form

```
<form id="id">  
  <input type="hidden" name="codeName" value="Other" />  
  <input type="hidden" name="skills" value="skill11,skill12" />  
  <input type="hidden" name="customer.name" value="Customer Name" />  
  <input type="hidden" name="customer.company" value="Company Name" />  
  <input type="hidden" name="customer.company_code" value="Company Code" />  
  <input type="hidden" name="customer.details" value="Issue Details" />  
  <input type="hidden" name="session.custom.external_key" value="abc123" />  
  <input type="hidden" name="session.custom.custom_field1" value="Custom Value" />  
  <input type="hidden" name="session.custom.custom_field2" value="123" />  
</form>
```

Starting a Click-to-Chat Session with a Session Key

```
BG.start(BG.START_TYPE.CHAT, {  
  sessionKey: '8347482'  
});
```



Note: Click-to-chat sessions may have an additional *uiOptions* object.

Starting a Click-to-Chat Session with a Session Key and `fallbackToFullWindow`

```
BG.start(BG.START_TYPE.CHAT, {  
  sessionKey: '7683902',  
  uiOptions: {  
    fallbackToFullWindow: false  
  }  
});
```

options Properties

The **options** parameter accepts the following possible arguments:

Name	Type	Required or Exclusive?	Description
sessionKey	String	Exclusive - Start Method	The numeric session key.
rep	Object	Exclusive - Start Method	An object identifying the representative with whom to start the session. See the table below.
issue	DOM Element	Exclusive - Start Method	A DOM element specifying the support issue. The DOM element can be retrieved using document.getElementById('id') . See the table below.
issue	Object	Exclusive - Start Method	An object specifying the support issue. See the table below.
skills	Array	No	The skills to apply to this session for the purpose of routing. Can be combined with any start method.
customer	Object	No	An object providing information about the customer. Can be combined with any start method. See the table below.
locale	String	No	The locale code of the language to be used for this customer client. The language must be installed on your BeyondTrust Appliance B Series. To see which languages are installed, go to /login > Localization > Languages . Available language packages can include English (en-us), German (de), Latin American Spanish (es), EU Spanish (es-sp), Finnish (fi), EU French (fr), Italian (it), Dutch (nl), Brazilian Portuguese (pt-br), EU Portuguese (pt-pt), Swedish (sv-se), Turkish (tr), Japanese (ja), Simplified Chinese (zh-hans), Traditional Chinese (zh), and Russian (ru).
attributes	Object	No	Session attributes. Can be combined with any start method. See the table below.
uiOptions	Object	No	User interface options. Available only for Click-to-Chat sessions. See the table below.



Note: Only one of the properties marked as **Exclusive - Start Method** should be specified.

rep Properties

The **rep** object must have the following properties:

Name	Type	Required or Exclusive?	Description
id	Integer	Required	The representative's unique ID number. To get a representative's ID, see " API Command: get_logged_in_reps " on page 23.
name	String	Required	The representative's public display name.

issue Properties

The **issue** object may be a JavaScript object with defined properties. Alternatively, it may be a DOM element, which should be a form. This DOM element must have one or more child inputs with defined names. If unnecessary elements are within the form, they will be ignored by the server. In either case, the accepted properties or input names are:

Name	Type	Required or Exclusive?	Description
id	Integer	Exclusive - Issue Identifier	The support issue's unique ID number. To get a list of issue IDs, see " API Command: get_support_teams " on page 26.
codeName	String	Exclusive - Issue Identifier	The support issue's unique code name.



Note: Only one of the properties marked as **Exclusive - Issue Identifier** should be specified.

customer Properties

The **customer** object has the following properties:

Name	Type	Required or Exclusive?	Description
name	String	No	The customer's name.
company	String	No	The customer's company name.
company_code	String	No	The customer's company code.
details	String	No	A description of the customer's problem.

attributes Properties

The **attributes** object has the following properties:

Name	Type	Required or Exclusive?	Description
external_key	String	No	The external key to associate with the session.
[custom field]	String	No	The code name and value of any custom fields. These fields must first be configured in /login > Management > API Configuration .

uiOptions Properties

The **uiOptions** object has the following property:

Name	Type	Required or Exclusive?	Description
fallbackToFullWindow	Boolean	No	Only used with click-to-chat sessions. If true , then a full-screen browser window will be used to render the click-to-chat client if a pop-up window cannot be created.

Full Client Functionality

Starting a session using the API methods above opens a dialog box in the customer's browser, determines the optimal download method for this client, and initiates the download. The experience varies depending on which download mechanism JavaScript has determined will work best. One of five different displays is shown:

- **ClickOnce** - A session started in Internet Explorer uses the ClickOnce technology to attempt to download and run the BeyondTrust customer client.
- **JavaScript launch** - If the user does not have ClickOnce support or if the user cancels the ClickOnce prompt, the launch method falls back to JavaScript.
 - JavaScript tells the browser to download the BeyondTrust customer client, which pops up a browser download dialog along with instructions for completing the download.
 - If the user cancels the JavaScript download dialog, the launch method falls back to the manual launch.
- **Manual launch** - If the user cancels all dialogs, they can click a link to re-trigger the download.
- **Mobile display** - Behavior varies on the type of session being requested.
 - For a full-client session, the device is searched for the BeyondTrust customer client app.
 - If the app is already installed, the app opens, and the session automatically begins.
 - If the app is not installed, a browser dialog provides a link to install the BeyondTrust customer client app from the device's app store. Once the app is installed, the second link in the browser dialog provides a method to start the session.

Click-to-Chat Without Using JavaScript

If you need to start a session with click-to-chat from an external site without writing any JavaScript, you may add the parameter **c2cjs=1** to any of the documented **start_session** API URLs. This will cause the request to respond with a click-to-chat page instead of the full customer client download, regardless of the settings for the public site.

For example, to redirect the current page to start a click-to-chat session with a specific representative:

```
<a href="https://support.example.com/api/start_session?id=12&name=John&c2cjs=1">Chat with John</a>
```

To open click-to-chat for a specific representative in a new browser tab - not a new window - in most browsers:

```
<a href="https://support.example.com/api/start_session?id=12&name=John&c2cjs=1" target="_blank" rel="noopener noreferrer">Chat with John</a>
```

Please note that if you wish to open click-to-chat in a new, smaller browser window instead of the current window or a new browser tab, you must either use **start_session.js** or write your own JavaScript to create and correctly size the new window.



Note: For the sake of appearance, opening click-to-chat in an appropriately sized window is the recommended method. A window that is not properly sized will function correctly but will result in disproportionate margins.

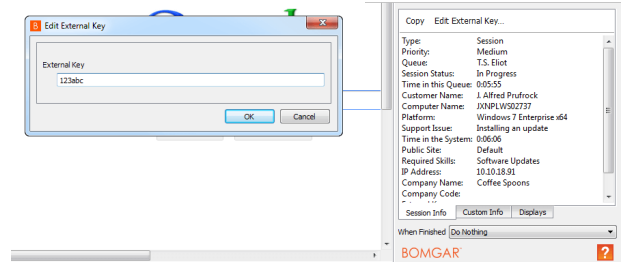
Start Sessions with External Keys (TicketID)

The external key is a text string sent to the B Series Appliance to be logged as a property of a particular support session. This key typically originates from software separate from the BeyondTrust software. It does not need to be a unique value but usually is. The external key can be specified in one of two ways.

Manual Entry

The support representative who has ownership of a BeyondTrust session can manually assign a key value from within the representative console.

The summary pane of a session displays that session's external key. Click the **Edit External Key** button to modify this value. You can edit the external key from either the **Session Info** tab or the **Custom Info** tab. Editing the external key in one location will change it in both.



Programmatic Assignment

The second, more useful way of designating an external key is from within the URL sent to the B Series Appliance by the customer client. The external key can be specified with any of the session start methods documented above, as well as with the **generate_session_key** command detailed below.

The issue submission survey generates an HTTP request similar in format to the following example:

```
https://support.example.com/api/start_session?issue_
menu=1&customer.name=John%20Doe&customer.company=Company%20Name&customer.company_
code=1234&customer.details=I%20need%20support&id=1&session.custom.external_key=1234
```

Note the **session.custom.external_key** parameter specified in the sample request provided. If an external key is specified in this manner, the representative console will automatically populate its external key field with the given value.

The API allows creation of a custom web site or application that can be used instead of the public site to establish a support session. Code within this custom software must generate HTTP requests in the format displayed in the example above to initiate the session and pre-populate the external key within the representative console.

Using the External Key

Once a support session has an external key associated with it, you can use the reporting API to generate XML session data containing the external key. This provides the means for middleware to be developed to provide a relationship between the reporting data used by BeyondTrust and the correlating ticketing system's ticket numbers.

Start Sessions with an Embedded Support Button

Define custom links within your in-house developed applications to call a pre-installed Support Button. An embedded Support Button gives support providers the ability to streamline the support path for specific applications.

The command line options can leverage the same session initiation methods as configured in the Support Button profile, or they can bypass the user interface altogether. You can configure this embedded Support Button to point to a specific issue so that when a customer clicks the button, a session will immediately start with the team best suited to handle that type of problem. The command also can assign an external key to sessions started from this embedded Support Button.

To create an embedded Support Button, a Support Button must first be deployed on the remote system. You may wish to define the Support Button profile so that neither the desktop shortcut nor the menu shortcut is created. Because a Support Button must be pre-installed, using an embedded Support Button prevents users from having to download the customer client each time they request support. Embedded Support Buttons are a Windows-only feature.

To start an embedded Support Button programmatically, first open the registry editor and locate the Support Button registry entry. Then copy the value data. The registry entry can be found in one of two places:

- **Single User Install:** HKEY_CURRENT_USER\Software\Bomgar\CallbackButton\- **All Users Install:** HKEY_LOCAL_MACHINE\Software\Bomgar\CallbackButton\

After pasting the value data into a text editor or the command prompt, add optional arguments.

Embedded Support Button Command Line Options

<code>--session-value <variable_name> <value></code>	Currently, the only supported variable is session.custom.external_key .
<code>--issue-code-name</code>	This selects the issue for the support session. If --issue-code-name is provided without --present-front-end-survey , a session started from this Support Button will start immediately without offering the customer the issue submission form or any additional session start methods.
<code>--present-front-end-survey</code>	If provided with --issue-code-name , a session started from this Support Button will take the customer to the issue submission form with the issue pre-selected. The customer will not be given any additional session start methods.

Finally, run the command line.

Example:

```
C:\Users\admin\AppData\Local\bomgar-scc-cb\support.example.com\bomgar-cb-w0dc30ji8hz65yji8hz65yji8hz65yji8hz65yc40ic90.exe --cbdir C:\Users\admin\AppData\Local\bomgar-scc-cb\support.example.com\ --session-value session.custom.external_key "abc123" --issue-code-name install_update --present-front-end-survey
```



Note: The command line option **--cbdir <support_button_install_directory>** is required.




Note: This feature is not supported for ARM-based Windows systems.


Reporting API

The reporting API is designed to enable you to pull reporting data in XML format, suitable for importing into external databases and applications. The data presented is the same as in the session and exit survey reports of the **/login** administrative interface.

The reporting API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7.

XML data is pulled by sending a simple HTTP request to the BeyondTrust Appliance B Series. The request can be sent using any HTTPS-capable socket library, scripting language module, or a URL fetcher such as **cURL** or **wget**. Either **GET** or **POST** may be used as the request method.

 **Note:** Even if your B Series Appliance has multiple public sites, all reports return data associated with all public sites unless the request contains a specific parameter to limit the sites pulled.

 **Note:** POST requests must include a "Content-Type: application/x-www-form-urlencoded" HTTP header when supplying parameters in the request body, and the parameters must be url-encoded. Multipart POST requests are not supported.

IMPORTANT!

When making consecutive API calls, you must close the connection after each API call.

The reporting API URL is <https://support.example.com/api/reporting>.

An XML schema which formally describes the format of the returned reporting data is available at <https://support.example.com/api/reporting.xsd>.

IMPORTANT!

To run **ArchiveListing** or **Archive** reports, ensure that the **Enable Archive API** option is checked on the **Management > API Configuration** page of the **/login** administrative interface. The state archive API can be enabled independently of other APIs.

Required Parameters for Reporting API

generate_report=[string]

The type of report to be generated. Report types can be any of the following:

SupportSession	PresentationSessionRecording
SupportSessionListing	SupportCustExitSurvey
SupportSessionSummary	SupportRepExitSurvey
SupportSessionRecording	SupportTeam
ShowMyScreenRecording	ArchiveListing

CommandShellRecording	Archive
PresentationSession	LicenseUsage
PresentationSessionListing	EndpointLicenseUsage

The reporting API returns XML responses that declare a namespace. If you are parsing these responses with a namespace-aware parser, you will need to set the namespace appropriately or ignore the namespace while parsing the XML.

Reporting API: <https://www.beyondtrust.com/namespaces/API/reporting>



Note: The above namespace is returned XML data and is not a functional URL.

Download Reports with SupportSession

The **SupportSession** query returns full information for all sessions which match given search parameters. You may use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**
- **Isid**
- **Isids**

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.


Parameters for SupportSession

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return all sessions, even those still in progress, that began on or after this date and that are within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report should return all sessions, even those still in progress, that began at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return only closed sessions that ended on or after this date and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return only closed sessions that ended at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.
<code>Isid=[string]</code>	The ID of the session for which you wish to see details.
<code>Isids=[comma-separated strings]</code>	A comma-delimited list of the IDs of sessions for which you wish to see details.

Optional Parameter for SupportSession

<code>limit=[string]</code>	The category by which to filter results. Can be one of the following:	
	all	Returns all results.
	rep:[id]	Returns sessions owned by a representative, specified by user ID. To get a representative's ID, see "API Command: get_logged_in_reps" on page 23 .

team:[all]	Returns sessions owned by any team.
team:[id]	Returns sessions owned by a team specified by team ID. To get a team's ID, see "API Command: get_support_teams" on page 26 .
members:[id]	Returns sessions owned by members of a team specified by team ID. To get a team's ID, see "API Command: get_support_teams" on page 26 .
site:[id]	Returns sessions run through a public site specified by site ID. The default public site always has an ID of 1.




Note: The *limit* parameter cannot be used in conjunction with either *Isid* or *Isids*. If it is used with either of these parameters, the *limit* parameter will be ignored.

XML Response for SupportSession Query

<code><session_list></code>	Contains a <session> element for each session that matches the given criteria. If no sessions are returned, this element will contain no <session> elements. If an error occurs during the search, it will contain an <error> element describing the problem.
-----------------------------------	--

Element Names and Attributes

<i>/session_list/session</i>	
<code>Isid</code> (attribute)	A string which uniquely identifies this session.
<code><session_type></code>	Indicates the type of session for which the report was run. The value will always be support in the current BeyondTrust API version.
<code><lseq></code>	An incrementing number used to represent support sessions in a non-string format. <div style="border: 1px solid black; padding: 10px; margin-top: 10px; background-color: #e6f2ff;">  <p>Note: The LSEQ element is not guaranteed to be unique or strictly sequential.</p> </div>
<code><start_time></code>	The date and time the session was begun either by the customer's running the customer client or by the representative's initiating a Jump session. Data is returned in ISO 8601 format. Also contains a timestamp attribute which displays the start time as a UNIX timestamp (UTC).
<code><end_time></code>	The date and time the session was ended either by the customer's closing the customer client or by the representative's closing the session. Data is returned in ISO 8601 format. Also contains a timestamp attribute which displays the end time in UNIX timestamp (UTC). This element will be empty for sessions which are still in progress when the report was run or which closed abnormally.

<duration>	Session length in HH:MM:SS format.
<public_site>	The name of the public site associated with the session. Also contains an id attribute, which displays the unique ID assigned to the public site.
<jumpoint>	The name of the Jumpoint through which this session was initiated, if any. Also contains an id attribute, which displays the unique ID assigned to the Jumpoint.
<external_key>	An arbitrary string that can link this session to an identifier on an external system, such as a help desk ticket ID. This can be input from within the representative console or defined programmatically.
<custom_attributes>	Contains a <custom_attribute> element for each custom field assigned to a session. This element displays only if custom fields have been defined. The format of each <custom_attribute> element is described below.
<session_chat_view_url>	The URL at which this session's chat transcript can be viewed in a web browser. This element is displayed only for sessions that have successfully ended.
<session_chat_download_url>	The URL at which this session's chat transcript can be downloaded. This element is displayed only for sessions that have successfully ended.
<session_recording_view_url>	The URL at which the video of the session may be viewed in a web browser. This element is displayed only if screen sharing recording was enabled at the time of the session. It is available only for sessions that have successfully ended and only if the requesting user has permission to view session recordings.
<session_recording_download_url>	The URL at which the video of the session may be downloaded. This element is displayed only if screen sharing recording was enabled at the time of the session and only if the rep initiated screen sharing during the session. It is available only for sessions that have successfully ended and only if the requesting user has permission to view session recordings.
<show_my_screen_recordings>	Contains a <show_my_screen_recording> element for each Show My Screen session that was initiated during the session. This element is displayed only if the representative shared their screen during the session, if Show My Screen recording was enabled at the time of the session, and only if the requesting user has permission to view Show My Screen recordings. Each <show_my_screen_recording> element contains the child elements <download_url> and <view_url> as described below.
<command_shell_recordings>	Contains a <command_shell_recording> element for each command shell that was initiated during the session. This element is displayed only if the representative opened a remote command shell during the session, if command shell recording was enabled at the time of the session, and if the requesting user has permission to view session recordings. Each <command_shell_recording> element contains the child elements <download_url> and <view_url> as described below.
<file_transfer_count>	The number of file transfers which occurred during the session.
<file_move_count>	The number of files renamed via the File Transfer interface during the session.
<file_delete_count>	The number of files deleted via the File Transfer interface during the session.
<primary_customer>	Lists the gsnumber as an attribute and as an element the name of the customer who initiated the session or, for a Jump session, the computer name of the remote system accessed by the representative.

<code><primary_rep></code>	Lists the gsnumber and id as attributes, and as an element the name of the final representative to own the session. If the session closed before it was transferred to a representative, this element will not be displayed.
<code><primary_team></code>	Lists the team ID and name of the final team to which this session was transferred. If the session was never transferred to a team, this element will not be displayed.
<code><customer_list></code>	A list of all customers who participated in the session. There should always be exactly one customer per session in the current BeyondTrust API version. The format of each <customer> element is described below.
<code><rep_list></code>	A list of all representatives who participated in the session, whether as session owners or conference members. The format of each <representative> element is described below. If the customer closed the session before it was transferred to a representative, this element will be empty.
<code><team_list></code>	A list of all teams to which the session belonged, whether by the session being initiated in a team queue, by a representative's explicitly transferring the session to a team, or by a session falling back into a team queue after a lost connection. This element may be empty, or it may contain one or more <team> elements as described below.
<code><cust_survey_list></code>	Contains a <cust_exit_survey> element if a customer exit survey was completed. This element is displayed only for sessions that have successfully ended and only if the customer submitted the survey. This element contains several child elements.
<code><rep_survey_list></code>	Contains a <rep_exit_survey> element if a representative survey was completed. This element is displayed only for sessions that have successfully ended and only if the representative submitted the survey. This element contains several child elements.
<code><session_details></code>	Contains a chronological list of all events which occurred during the session. This element contains one or more child <event> elements.

/session_list/session/custom_attributes/custom_attribute

<code>display_name (attribute)</code>	The display name assigned to the custom attribute.
<code>code_name (attribute)</code>	The code name assigned to the custom attribute.

/session_list/session/show_my_screen_recordings/show_my_screen_recording

<code>instance (attribute)</code>	The instance of the Show My Screen session, starting with 0 .
<code><download_url></code>	The URL at which the video of the Show My Screen session may be downloaded.
<code><view_url></code>	The URL at which the video of the Show My Screen session may be viewed in a web browser.

/session_list/session/command_shell_recordings/command_shell_recording

<code>instance (attribute)</code>	The instance of the command shell session, starting with 0 .
<code><download_url></code>	The URL at which the video of the command shell session may be downloaded.
<code><view_url></code>	The URL at which the video of the command shell session may be viewed in a web browser.

browser.

/session_list/session/customer_list/customer

gsnumber (attribute)	Uniquely identifies the customer in regards to their current connection to the BeyondTrust Appliance B Series. A gsnumber may be recycled, so while two people connected at the same time will never have the same gsnumber, one person may have a gsnumber that was assigned to another person in the past. Can be used to correlate a <customer> element with a <primary_cust> or with an event's <performed_by> or <destination> element.
<username>	The name used to identify the customer during the session. The method used to obtain this name varies depending on how the session was started.
<public_ip>	The customer's public IP address.
<private_ip>	The customer's private IP address.
<hostname>	The hostname of the customer's computer.
<os>	The operating system of the customer's computer.
<primary_cust>	Integer value (1 or 0) indicating if this customer was the first customer of the session. In the current version of the BeyondTrust API, this value is always 1.
<info>	Contains detailed information about the customer as either entered in the front-end survey or designated programmatically. This field contains several child elements as described below.

/session_list/session/customer_list/customer/info

<name>	The name which the customer entered in the Your Name field of the front-end survey or which was assigned programmatically.
<company>	The company name which the customer entered in the Company field on the front-end survey or which was assigned programmatically.
<company_code>	The code which the customer entered in the Company Code field on the front-end survey or which was assigned programmatically.
<issue>	The numeric ID of the issue or the representative which the customer selected from the dropdown of the front-end survey or which was designated programmatically.
<details>	The description of the problem as entered by the customer in the Describe Your Issue text area field of the front-end survey or as programmatically assigned.

/session_list/session/rep_list/representative

gsnumber (attribute)	<p>Uniquely identifies the representative in regards to their current connection to the BeyondTrust Appliance B Series. A gsnumber is assigned on a per-connection basis, so if a representative leaves a session and then rejoins without logging out of the B Series Appliance, their gsnumber will remain the same.</p> <p>However, if the representative's connection is terminated for any reason, when that</p>
----------------------	---


	<p>representative logs back into the B Series Appliance, they will be assigned a new gsnumber and will also appear multiple times in the <code><rep_list></code> element.</p> <p>A gsnumber may be recycled, so while two people connected at the same time will never have the same gsnumber, one person may have a gsnumber that was assigned to another person in the past. Can be used to correlate a <code><representative></code> element with a <code><primary_rep></code> or with an event's <code><performed_by></code> or <code><destination></code> element.</p>
id (attribute)	Unique ID assigned to the representative.
<code><username></code>	The username assigned to the representative.
<code><display_name></code>	This element is deprecated as of API version 1.10.0 but still exists for backwards compatibility. Its value is the same as that of <code><private_display_name></code> .
<code><public_display_name></code>	The public display name assigned to the representative. Note that this field contains the public display name's value at the time of the conference, which may not match the current value if the public_display_name has subsequently been changed.
<code><private_display_name></code>	The private display name assigned to the representative. Note that this field contains the private display name's value at the time of the conference, which may not match the current value if the private_display_name has subsequently been changed.
<code><display_number></code>	The display number assigned to the representative. Like <code><display_name></code> , this is the display number at the time of the session and may not match the current value.
<code><public_ip></code>	The representative's public IP address.
<code><private_ip></code>	The representative's private IP address.
<code><hostname></code>	The hostname of the representative's computer.
<code><os></code>	The operating system of the representative's computer.
<code><session_owner></code>	Integer value (1 or 0) indicating whether the representative was an actual owner of the session or was merely a conference member.
<code><primary_rep></code>	Integer value (1 or 0) indicating if the representative was the final representative to own the session.
<code><seconds_involved></code>	Integer value indicating the number of seconds the representative was involved in this session.
<code><invited></code>	Integer value (1) present only if the representative is an invited user.

/session_list/session/team_list/team

[value]	The display name of the support team. Note that this field contains the team name as it currently appears, which may not match the value at the time of the session if the team name has been subsequently changed.
id (attribute)	Integer value representing the team's unique ID.
primary_team (attribute)	Integer value (1 or 0) indicating if this team was the last team to which the session was transferred.

/session_list/session/session_details/event

timestamp (attribute)	The system time at which the event occurred.																																		
event_type (attribute)	<p>The type of event which occurred. Event types include the following:</p> <table border="1"> <tr><td>Callback Button Deployed</td><td>Files Shared</td></tr> <tr><td>Callback Button Removed</td><td>Legal Agreement Response</td></tr> <tr><td>Chat Message</td><td>Registry Exported</td></tr> <tr><td>Command Shell Session Started*</td><td>Registry Imported</td></tr> <tr><td>Conference Member Added</td><td>Registry Key Added</td></tr> <tr><td>Conference Member Departed</td><td>Registry Key Deleted</td></tr> <tr><td>Conference Member State Changed</td><td>Registry Key Renamed</td></tr> <tr><td>Conference Owner Changed</td><td>Representative Exit Survey</td></tr> <tr><td>Customer Exit Survey</td><td>Service Access Allowed</td></tr> <tr><td>Directory Created</td><td>Session Assigned</td></tr> <tr><td>External Key</td><td>Session Assignment Response</td></tr> <tr><td>File Deleted</td><td>Session End</td></tr> <tr><td>File Download</td><td>Session Foreground Window Changed</td></tr> <tr><td>File Download Failed</td><td>Session Note Added</td></tr> <tr><td>File Moved</td><td>Session Start</td></tr> <tr><td>File Upload</td><td>Show My Screen Recording</td></tr> <tr><td>File Upload Failed</td><td>System Information Retrieved</td></tr> </table> <p>*Will only appear if recording is enabled for this session.</p>	Callback Button Deployed	Files Shared	Callback Button Removed	Legal Agreement Response	Chat Message	Registry Exported	Command Shell Session Started*	Registry Imported	Conference Member Added	Registry Key Added	Conference Member Departed	Registry Key Deleted	Conference Member State Changed	Registry Key Renamed	Conference Owner Changed	Representative Exit Survey	Customer Exit Survey	Service Access Allowed	Directory Created	Session Assigned	External Key	Session Assignment Response	File Deleted	Session End	File Download	Session Foreground Window Changed	File Download Failed	Session Note Added	File Moved	Session Start	File Upload	Show My Screen Recording	File Upload Failed	System Information Retrieved
Callback Button Deployed	Files Shared																																		
Callback Button Removed	Legal Agreement Response																																		
Chat Message	Registry Exported																																		
Command Shell Session Started*	Registry Imported																																		
Conference Member Added	Registry Key Added																																		
Conference Member Departed	Registry Key Deleted																																		
Conference Member State Changed	Registry Key Renamed																																		
Conference Owner Changed	Representative Exit Survey																																		
Customer Exit Survey	Service Access Allowed																																		
Directory Created	Session Assigned																																		
External Key	Session Assignment Response																																		
File Deleted	Session End																																		
File Download	Session Foreground Window Changed																																		
File Download Failed	Session Note Added																																		
File Moved	Session Start																																		
File Upload	Show My Screen Recording																																		
File Upload Failed	System Information Retrieved																																		
<performed_by>	The entity that performed the action. Indicates the entity's gsnumber and also its type , indicating whether this action was performed by the system , a customer , or a representative .																																		
<destination>	The entity to which the event was directed. Indicates the entity's gsnumber and also its type , indicating whether this action was directed to the system , a customer , or a representative .																																		
<body>	The text of the message as displayed in the chat log area.																																		
<encoded_body>	Can be shown in place of the < body > element above. Contains the base64 (RFC 2045 section 6.8) encoded value of what would have been shown in the < body > element, and is shown ONLY if the < body > text contains characters that are invalid according to XML specification. These characters are typically the result of binary data being sent through chat messages.																																		
<filename>	The name of the transferred file.																																		
<filesize>	An integer indicating the size of the transferred file.																																		
<system_information>	Applies only to System Information Retrieved events wherein the system information is pulled automatically upon session start. This element contains multiple < category >																																		

	child elements as described below.
	 Note: System information is logged only when pulled automatically at the beginning of the session and not when specifically requested by the representative. This is to prevent overload with the large amount of dynamic data that can be retrieved from the remote system.
<files>	If this event involved the transferring of files, then this element will contain a <file> element for every file transferred.
<data>	Contains an arbitrary number of <value name="_" value="_" /> elements. The name and number of these elements varies based on event_type . For example, when a representative joins the session, a Conference Member Added event would contain <value> elements for the representative's name, username, private_ip, public_ip, hostname, os, support_teams, and user_id .

/session_list/session/session_details/event/system_information/category

<description>	Contains multiple <field> elements, each of which contains a descriptor for the specific data field. For example, the Drives category would have <field> elements Drive, Type, Percent Used , etc. These <field> elements can be compared to table header cells.
<data>	Contains multiple <row> elements, each of which contains multiple <field> elements that correspond to the <field> elements above. For example, the Drives category would have a separate <row> for each drive on the remote computer. An example <row> might contain <field> elements C:\, Local Disk, 60% , etc. These <row> elements can be compared to table rows, with each <field> element a table cell.

Query Examples for SupportSession

Sessions started October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0</code>
Sessions started the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=31</code>
Sessions started 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_time=1475308800&duration=0</code>
Sessions started 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_time=1475308800&duration=36000</code>
Sessions ended October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSession&end_date=2016-10-01&duration=0</code>

Sessions ended the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSession&end_date=2016-10-01&duration=31</code>
Sessions ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSession&end_time=1475308800&duration=36000</code>
Session c69a8e10bea9428f816cfababe9815fe	<code>https://support.example.com/api/reporting?generate_report=SupportSession&lsid=c69a8e10bea9428f816cfababe9815fe</code>
Sessions c69a8e10bea9428f816cfababe9815fe, a5eaa58591047b88556f944804227b0, 5bf07601298b495b87310da9ce571e22	<code>https://support.example.com/api/reporting?generate_report=SupportSession&lsids=c69a8e10bea9428f816cfababe9815fe,a5eaa58591047b88556f944804227b0,5bf07601298b495b87310da9ce571e22</code>
Sessions started October 1 2016 to present for all sessions	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=all</code>
Sessions started October 1 2016 to present for a specific rep	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=rep:1</code>
Sessions started October 1 2016 to present for all teams	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=team:all</code>
Sessions started October 1 2016 to present for a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=team:1</code>
Sessions started October 1 2016 to present for members of a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=members:1</code>
Sessions started October 1 2016 to present for a specific public site	<code>https://support.example.com/api/reporting?generate_report=SupportSession&start_date=2016-10-01&duration=0&limit=site:1</code>

Download Reports with SupportSessionListing

The **SupportSessionListing** query returns a list of session IDs, external keys, and availability of a recording for sessions which match given search parameters. You may use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for SupportSessionListing

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return all sessions, even those still in progress, that began on or after this date and that are within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report should return all sessions, even those still in progress, that began at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return only closed sessions that ended on or after this date and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return only closed sessions that ended at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.


XML Response for SupportSessionListing Query

<code><session_summary_list></code>	Contains a <session_summary> element for each session that matches the given criteria. If no sessions are returned, this element will contain no <session_summary> elements. If an error occurs during the search, it will contain an <error> element describing the problem.
---	--

Element Names and Attributes

/session_summary_list/session_summary

<code>Isid (attribute)</code>	The session ID for the given support session.
<code><Isid></code>	An incrementing number used to represent support sessions in a non-string format.

	 Note: The LSEQ element is not guaranteed to be unique or strictly sequential.
has_recording (attribute)	Integer (1 or 0) indicating if the given session has a session recording.
external_key (attribute)	An arbitrary string that can link this session to an identifier on an external system, such as a help desk ticket ID. This can be input from within the representative console or defined programmatically. This element will be displayed only if an external key has been defined.

Query Examples for SupportSessionListing

Sessions started October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&start_date=2016-10-01&duration=0</code>
Sessions started the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&start_date=2016-10-01&duration=31</code>
Sessions started 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&start_time=1475308800&duration=0</code>
Sessions started 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&start_time=1475308800&duration=36000</code>
Sessions ended October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&end_date=2016-10-01&duration=0</code>
Sessions ended the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&end_date=2016-10-01&duration=31</code>
Sessions ended 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&end_time=1475308800&duration=0</code>
Sessions ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportSessionListing&end_time=1475308800&duration=36000</code>

Download Reports with SupportSessionSummary

The **SupportSessionSummary** query returns an overview of support session statistics for representatives, teams or sites. You may use any of the following sets of parameters to generate reports:

- **start_date**, **duration**, and **report_type**
- **start_time**, **duration**, and **report_type**
- **end_date**, **duration**, and **report_type**
- **end_time**, **duration**, and **report_type**

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for SupportSessionSummary

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return all sessions, even those still in progress, that began on or after this date and that are within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report should return all sessions, even those still in progress, that began at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return only closed sessions that ended on or after this date and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return only closed sessions that ended at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.
<code>report_type=[string]</code>	Accepted values are rep (to show representative summary statistics), team (to show team summary statistics), or site (to show public site summary statistics).

XML Response for SupportSessionSummary Query

<code><summary_list></code>	Contains a <summary> element for each record that matches the given criteria. If no sessions are returned, this element will contain no <summary> elements. If an error occurs during the search, it will contain an <error> element describing the problem.
-----------------------------------	---

Element Names and Attributes

<i>/summary_list/summary</i>	
<code>id</code> (attribute)	Returns the representative's, team's, or site's unique ID.
<code>type</code> (attribute)	Specifies the report type being generated: rep , team , or site .

<code><display_name></code>	The display name of the team or site, or the private display name of the representative. Note that since summary reports represent an aggregation of sessions over a period of time, the display name used is the current value for the representative, team, or site, which may have been edited since the time of the first returned session.
<code><total_sessions></code>	The total number of sessions run by the rep, team, or site in the time specified.
<code><avg_sessions_per_weekday></code>	The average number of sessions conducted on Monday through Friday by the representative, team, or site, expressed as a decimal rounded to the nearest point.
<code><avg_duration></code>	The average length of each session, expressed as HH:II:SS.

Query Examples

Sessions started October 1 2016 to present, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_date=2016-10-01&duration=0&report_type=rep</code>
Sessions started October 1 2016 to present, by team	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_date=2016-10-01&duration=0&report_type=team</code>
Sessions started October 1 2016 to present, by site	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_date=2016-10-01&duration=0&report_type=site</code>
Sessions started the month of October 2016, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_date=2016-10-01&duration=31&report_type=rep</code>
Sessions started 8:00 AM October 1 2016 to present, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_time=1475308800&duration=0&report_type=rep</code>
Sessions started 8:00 AM October 1 2016 to 6:00 PM October 1 2016, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&start_time=1475308800&duration=36000&report_type=rep</code>
Sessions ended October 1 2016 to present, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&end_date=2016-10-01&duration=0&report_type=rep</code>
Sessions ended the month of October 2016, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&end_date=2016-10-01&duration=31&report_type=rep</code>
Sessions ended 8:00 AM October 1 2016 to present, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&end_time=1475308800&duration=0&report_type=rep</code>
Sessions ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportSessionSummary&end_time=1475308800&duration=36000&report_type=rep</code>

Download Reports with SupportSessionRecording

The **SupportSessionRecording** query returns the requested support session recording file. Depending on your browser, this query will either immediately begin download or prompt you to open or save the file.

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameter for SupportSessionRecording

Isid=[string]

The session ID for which you wish to download the video recording of the support session.

Query Example for SupportSessionRecording

SupportSessionRecording: Session
c69a8e10bea9428f816cfababe9815fe

[https://support.example.com/api/reporting?
generate_report=SupportSessionRecording&
Isid=c69a8e10bea9428f816cfababe9815fe](https://support.example.com/api/reporting?generate_report=SupportSessionRecording&Isid=c69a8e10bea9428f816cfababe9815fe)

Download Reports with ShowMyScreenRecording

The **ShowMyScreenRecording** query returns the requested Show My Screen recording. Depending on your browser, this query will either immediately begin download or prompt you to open or save the file.

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for ShowMyScreenRecording

<code>lsid=[string]</code>	The session ID for which you wish to download the video recording of the Show My Screen session.
<code>instance=[integer]</code>	The instance number of the Show My Screen recording you wish to download. Instances are enumerated starting with 0 . The instance number can be obtained from the SupportSession report.

Query Examples for ShowMyScreenRecording

ShowMyScreenRecording: First instance of session c69a8e10bea9428f816cfababe9815fe	<code>https://support.example.com/api/reporting?generate_report=ShowMyScreenRecording&lsid=c69a8e10bea9428f816cfababe9815fe&instance=0</code>
ShowMyScreenRecording: Third instance of session c69a8e10bea9428f816cfababe9815fe	<code>https://support.example.com/api/reporting?generate_report=ShowMyScreenRecording&lsid=c69a8e10bea9428f816cfababe9815fe&instance=2</code>

Download Reports with CommandShellRecording

The **CommandShellRecording** query returns the requested command shell recording. Depending on your browser, this query will either immediately begin download or prompt you to open or save the file.

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for CommandShellRecording

<code>lsid=[string]</code>	The session ID for which you wish to download the video recording of the command shell.
<code>instance=[integer]</code>	The instance number of the command shell recording you wish to download. Instances are enumerated starting with 0 . The instance number can be obtained from the SupportSession report. Unlike the API, the SupportSession report enumerates instances starting at 1. The instance 1 in a SupportSession report should be referenced as instance 0 in the API.

Query Examples for CommandShellRecording

CommandShellRecording: First shell instance of session c69a8e10bea9428f816cfababe9815fe	https://support.example.com/api/reporting? generate_report=CommandShellRecording& lsid=c69a8e10bea9428f816cfababe9815fe&instance=0
CommandShellRecording: Third shell instance of session c69a8e10bea9428f816cfababe9815fe	https://support.example.com/api/reporting? generate_report=CommandShellRecording& lsid=c69a8e10bea9428f816cfababe9815fe&instance=2

Download Reports with PresentationSession

The **PresentationSession** query returns full information for all presentations that match given search parameters. You may use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**
- **Isid**
- **Isids**

The API account must have the permission **Allow Access to Presentation Session Reports and Recordings**.

Permissions for Viewing PresentationSession Reports

View Only His/Her Sessions	View presentations if they are the user who created the presentation.
View His/Her Teams' Sessions	View presentations if they are on the same team as the user who created the presentation.
View All Sessions	View all available presentations.


Parameters for PresentationSession

start_date=[YYYY-MM-DD]	Specifies that the report should return all presentations, even those still in progress, that began on or after this date and that are within the duration specified below.
start_time=[timestamp]	Specifies that the report should return all presentations, even those still in progress, that began on or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
end_date=[YYYY-MM-DD]	Specifies that the report should return only completed presentations that ended on or after this date and that are within the duration specified below.
end_time=[timestamp]	Specifies that the report should return only completed presentations that ended on or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
duration=[integer]	Specifies the date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.
Isid=[string]	The ID of the presentation for which you wish to see details.
Isids=[comma-separated strings]	A comma-delimited list of the IDs of presentations for which you wish to see details.

XML Response for PresentationSession Report

<code><session_list></code>	Contains a <code><session></code> element for each presentation that matches the given criteria. If no presentations are returned, this element will contain no <code><session></code> elements. If an error occurs during the search, it will contain an <code><error></code> element describing the problem.
-----------------------------------	--

Element Names and Attributes

<i>/session_list/session</i>	
<code>Isid</code> (attribute)	A string that uniquely identifies this presentation.
<code><session_type></code>	Indicates the type of session for which the report was run. The value will always be presentation for the PresentationSession report.
<code><lseq></code>	An incrementing number used to represent presentations in a non-string format. <div data-bbox="609 884 1511 995" style="border: 1px solid black; padding: 5px; margin-top: 10px;">  Note: <i>The LSEQ element is not guaranteed to be unique or strictly sequential.</i> </div>
<code><start_time></code>	The date and time the presentation was begun. Data is returned in ISO 8601 format. Also contains a timestamp attribute which displays the start time as a UNIX timestamp (UTC).
<code><end_time></code>	The date and time the presentation was ended. Data is returned in ISO 8601 format. Also contains a timestamp attribute which displays the end time as a UNIX timestamp (UTC). This element will be empty for presentations that are still in progress when the report was run, or that closed abnormally.
<code><duration></code>	Session length in HH:MM:SS format.
<code><session_chat_view_url></code>	The URL at which the presentation's chat transcript can be viewed in a web browser. This element is displayed only for presentations that have successfully completed.
<code><session_chat_download_url></code>	The URL at which this presentation's chat transcript can be downloaded. This element is displayed only for presentations that have successfully completed.
<code><session_recording_view_url></code>	The URL at which the video of the presentation may be viewed in a web browser. This element is displayed only if screen sharing recording was enabled at the time of the presentation. It is available only for presentations that have successfully completed, and only if the requesting user has permission to view presentation recordings.
<code><session_recording_download_url></code>	The URL at which the video of the presentation may be downloaded. This element is displayed only if screen sharing recording was enabled at the time of the presentation and only if the presenter showed their screen during the presentation. It is available only for presentations that have successfully completed, and only if the requesting user has permission to view presentation recordings.
<code><attendee_list></code>	A list of all attendees who participated in the presentation. The format of each <code><attendee></code> element is described below.

<rep_list>	A list of all presenters who participated in the presentation; however, if no presentations have successfully ended or if only the requesting user has permission to view presentation recordings. The format of each <representative> element is described below
<session_details>	Contains a chronological list of all events that occurred during the presentation. This element contains one or more child <event> elements, described below.

/session_list/session/attendee_list/attendee

gsnumber (attribute)	Uniquely identifies the attendee in regards to their current connection to the B Series Appliance. A gsnumber may be recycled, so while two people connected at the same time will never have the same gsnumber, one person may have a gsnumber that was assigned to another person in the past. Can be used to correlate an <attendee> element with an event's <performed by> or <destination> element.
<username>	The name used to identify the attendee during the session.
<public_ip>	The attendee's public IP address.

/session_list/session/rep_list/representative

gsnumber (attribute)	Uniquely identifies the representative in regards to his or her current connection to the B Series Appliance. A gsnumber is assigned on a per-connection basis, so if a representative leaves a presentation and then rejoins without logging out of the B Series Appliance, his or her gsnumber will remain the same. However, if the representative's connection is terminated for any reason, when that representative logs back into the B Series Appliance, he or she will be assigned a new gsnumber and will also appear multiple times in the <rep_list> element.
id (attribute)	Unique ID assigned to the representative.
<username>	The username assigned to the representative.
<public_display_name>	The public display name assigned to the representative. Note that this field contains the public display name's value at the time of the presentation, which may not match the current value if the public display name has changed.
<private_display_name>	The private display name assigned to the representative. Note that this field contains the private display name's value at the time of the presentation, which may not match the current value if the private display name has changed.
<public_ip>	The representative's public IP address.
<private_ip>	The representative's private IP address.
<hostname>	The hostname of the representative's computer.
<os>	The operating system of the representative's computer.
<seconds_involved>	Integer value indicating the number of seconds the representative was involved in this session.

/session_list/session/session_list/event

timestamp (attribute)	The system time at which the event occurred.
<event_type> (attribute)	The type of event that occurred.
<performed_by>	The entity that performed the action. Indicates the entity's gsnumber and also its type, indicating whether this action was performed by the system, an attendee, or a representative.
<destination>	The entity to which the event was directed. Indicates the entity's gsnumber and also its type, indicating whether this action was directed to the system, an attendee, or a representative.
<body>	The text of the message as displayed in the chat log area.
<encoded_body>	Can be shown in place of the <body> element above. Contains the base64 (RFC 2045 section 6.8) encoded value of what would have been shown in the <body> element, and is shown ONLY if the <body> text contains characters that are invalid according to XML specification. These characters are typically the result of binary data being sent through chat messages.
<data>	Contains an arbitrary number of <value name="_" value="_" /> elements. The name and number of these elements varies based on event type.
<name>	The name which the customer entered in the Your Name field of the front-end survey or which was assigned programmatically.
<company>	The company name which the customer entered in the Company field on the front-end survey or which was assigned programmatically.
<company_code>	The code which the customer entered in the Company Code field on the front-end survey or which was assigned programmatically.
<issue>	The numeric ID of the issue or the representative which the customer selected from the dropdown of the front-end survey or which was designated programmatically.
<details>	The description of the problem as entered by the customer in the Describe Your Issue text area field of the front-end survey or as programmatically assigned.
gsnumber (attribute)	<p>Uniquely identifies the representative in regards to their current connection to the B Series Appliance. A gsnumber is assigned on a per-connection basis, so if a representative leaves a session and then rejoins without logging out of the B Series Appliance, their gsnumber will remain the same.</p> <p>However, if the representative's connection is terminated for any reason, when that representative logs back into the B Series Appliance, they will be assigned a new gsnumber and will also appear multiple times in the <rep_list> element.</p> <p>A gsnumber may be recycled, so while two people connected at the same time will never have the same gsnumber, one person may have a gsnumber that was assigned to another person in the past. Can be used to correlate a <representative> element with a <primary_rep> or with an event's <performed_by> or <destination> element.</p>
id (attribute)	Unique ID assigned to the representative.

<username>	The username assigned to the representative.
<display_name>	This element is deprecated as of API version 1.10.0 but still exists for backwards compatibility. Its value is the same as that of <private_display_name>.
<public_display_name>	The public display name assigned to the representative. Note that this field contains the public display name's value at the time of the conference, which may not match the current value if the public_display_name has subsequently been changed.
<private_display_name>	The private display name assigned to the representative. Note that this field contains the private display name's value at the time of the conference, which may not match the current value if the private_display_name has subsequently been changed.
<display_number>	The display number assigned to the representative. Like <display_name>, this is the display number at the time of the session and may not match the current value.
<public_ip>	The representative's public IP address.
<private_ip>	The representative's private IP address.
<hostname>	The hostname of the representative's computer.
<os>	The operating system of the representative's computer.
<session_owner>	Integer value (1 or 0) indicating whether the representative was an actual owner of the session or was merely a conference member.
<primary_rep>	Integer value (1 or 0) indicating if the representative was the final representative to own the session.
<seconds_involved>	Integer value indicating the number of seconds the representative was involved in this session.
<invited>	Integer value (1) present only if the representative is an invited user.

Query Examples for PresentationSession

Presentations started October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=PresentationSession&start_date=2016-10-01&duration=0</code>
Presentations started the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=PresentationSession&start_date=2016-10-01&duration=31</code>
Presentations started 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=PresentationSession&start_time=1475308800&duration=0</code>
Presentations started 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=PresentationSession&start_time=1475308800&duration=36000</code>
Presentations ended October 1 2016 to	<code>https://support.example.com/api/reporting?</code>

present	generate_report= PresentationSession &end_date= 2016-10-01 &duration= 0
Presentations ended the month of October 2016	https://support.example.com/api/reporting?generate_report= PresentationSession &end_date= 2016-10-01 &duration= 31
Presentations ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016	https://support.example.com/api/reporting?generate_report= PresentationSession &end_time= 1475308800 &duration= 36000
Presentation c69a8e10bea9428f816cfababe9815fe	https://support.example.com/api/reporting?generate_report= PresentationSession &lsid= c69a8e10bea9428f816cfababe9815fe
Presentations c69a8e10bea9428f816cfababe9815fe, a5eaaa58591047b88556f944804227b0, 5bf07601298b495b87310da9ce571e22	https://support.example.com/api/reporting?generate_report= PresentationSession &lsids= c69a8e10bea9428f816cfababe9815fe, a5eaaa58591047b88556f944804227b0, 5bf07601298b495b87310da9ce571e22
Presentations started October 1 2016 to present for all sessions	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= all
Presentations started October 1 2016 to present for a specific rep	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= rep:1
Presentations started October 1 2016 to present for all teams	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= team:all
Presentations started October 1 2016 to present for a specific team	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= team:1
Presentations started October 1 2016 to present for members of a specific team	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= members:1
Presentations started October 1 2016 to present for a specific public site	https://support.example.com/api/reporting?generate_report= PresentationSession &start_date= 2016-10-01 &duration= 0 &limit= site:1

Download Reports with PresentationSessionListing

The **PresentationSessionListing** query returns a list of presentations that are ready to be archived. You may use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**

The API account must have the permission **Allow Access to Presentation Session Reports and Recordings**.

Parameters for PresentationSessionListing

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return all presentations, even those still in progress, that began on or after this date, and that are within the duration specified below.
<code>start_time=[UNIX timestamp]</code>	Specifies that the report should return all presentations, even those still in progress, that began on or after this time and that are within the duration specified below. The time must be a UNIX timestamp, (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return only completed presentations that ended on or after this date and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return only completed presentations that ended on or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.

XML Response for PresentationSessionListing Query

<code><presentation_summary_list></code>	Contains a <presentation_summary> element for each presentation that matches the given criteria. If no sessions are returned, this element will contain no <presentation_summary> elements. If an error occurs during the search, it will contain an <error> element describing the problem.
--	---

XML Output Example

This report outputs XML similar to the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<presentation_summary_list xmlns="http://support.example.com/namespaces/API/command">
  <presentation_summary_lsid="6d547436638c41ad8f945388d2d465d2" has_recording="1" />
</presentation_summary_list>
```

```
<presentation_summary lsid="abc12345678901adff9d5388d2123456" has_recording="0" />
</presentation_summary_list>
```



Note: The XML elements need to be specified in the reporting.xsd.

Query Examples for PresentationSessionListing

Presentations started October 1 2016 to present	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&start_date=2016-10-01&duration=0
Presentations started the month of October 2016	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&start_date=2016-10-01&duration=31
Presentations started 8:00 AM October 1 2016 to present	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&start_time=1475308800&duration=0
Presentations started 8:00 AM October 1 2016 to 6:00 PM October 1 2016	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&start_time=1475308800&duration=36000
Presentations ended October 1 2016 to present	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&end_date=2016-10-01&duration=0
Presentations ended the month of October 2016	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&end_date=2016-10-01&duration=31
Presentations ended 8:00 AM October 1 2016 to present	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&end_time=1475308800&duration=0
Presentations ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016	https://support.example.com/api/reporting?generate_report=PresentationSessionListing&end_time=1475308800&duration=36000

Download Reports with PresentationSessionRecording

The **PresentationSessionRecording** query returns the requested presentation recording. Depending on your browser, this query will either immediately begin download or prompt you to open or save the file.

The API account must have the permission **Allow Access to Presentation Session Reports and Recordings**.

Parameter for PresentationSessionRecording

Isid=[string]

The session ID for which you wish to download the video recording of the presentation session.

Query Example for PresentationSessionRecording

PresentationSessionRecording: Session
c69a8e10bea9428f816cfababe9815fe

[https://support.example.com/api/reporting?
generate_report=PresentationSessionRecording&
Isid=c69a8e10bea9428f816cfababe9815fe](https://support.example.com/api/reporting?generate_report=PresentationSessionRecording&Isid=c69a8e10bea9428f816cfababe9815fe)



Note: As of 16.1, **PresentationRecording** has been deprecated in favor of **PresentationSessionRecording**. **PresentationRecording** is still available for backward compatibility.

Download Survey Reports with SupportCustExitSurvey and SupportRepExitSurvey

The **SupportCustExitSurvey** and **SupportRepExitSurvey** queries return the questions and answers to the customer or representative survey. You may use any of the following sets of parameters to generate reports:

- **start_date**, **duration**, **report_type**, and **id**
- **start_time**, **duration**, **report_type**, and **id**
- **end_date**, **duration**, **report_type**, and **id**
- **end_time**, **duration**, **report_type**, and **id**
- **Isid** and **report_type**
- **Isids** and **report_type**

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for SupportCustExitSurvey and SupportRepExitSurvey

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return all sessions, even those still in progress, that began on or after this date and that are within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report should return all sessions, even those still in progress, that began at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return only closed sessions that ended on or after this date and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return only closed sessions that ended at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.
<code>report_type=[string]</code>	Enter rep to filter results according to the representative who last owned the session or team to filter according to team.
<code>id=[integer]</code>	May be the numeric ID of the representative or team that you wish to view or "all" to display data for all representatives or teams. To get a representative's ID, see " API Command: get_logged_in_reps " on page 23. To get a team's ID, see " API Command: get_support_teams " on page 26.
<code>Isid=[string]</code>	The session ID for which you wish to see the survey report.
<code>Isids=[comma-separated strings]</code>	A comma-delimited list of the session IDs for which you wish to see survey reports.

Optional Parameter

site_id=[integer]

The numeric ID of the public site by which to filter results. Only surveys whose support sessions are associated with the given public site are returned. If this parameter is not specified, results from all public sites are returned. The default public site always has an ID of 1.

XML Response for SupportCustExitSurvey and SupportRepExitSurvey Queries

<exit_survey_list>

Contains an <exit_survey> element for each session that matches the given criteria. If no sessions are returned, this element will contain no <exit_survey> elements. If an error occurs during the search, it will contain an <error> element describing the problem.

Element Names and Attributes

/exit_survey_list/exit_survey

Isid (attribute)	The unique ID of the session for which this survey was submitted.
ts (attribute)	The start time of the session for which this survey was submitted.
<session_type>	Indicates the type of session for which the report was submitted. This value will always be support in the current BeyondTrust API version.
<public_site>	The name of the public site associated with the session. Also contains an id attribute, which displays the unique ID assigned to the public site..
<submitted_by>	The name of the customer or private display name of the representative who submitted the survey. This element also has a type attribute with the value of cust or rep , indicating whether this survey was submitted by a customer or a representative.
<primary_customer>	The display name of the customer who initiated the session. This element also has an id attribute, the value of which is always 0 .
<primary_rep>	The private display name of the final representative to own the session, as it appeared at the time of the session. This element also has an id attribute, which is the representative's unique ID. This element will be absent if the customer closed the session before it was accepted by a representative.
<primary_team>	The display name of the last team to which the session was transferred. This element also has an id attribute, which is the team's unique ID. This element will be absent if the session was never transferred to a team.
<customer_list>	Listing of all customers who participated in this session. For full details, see the descriptions of the <customer_list> and <customer> elements in the SupportSession section.
<rep_list>	Listing of all representatives who participated in this session. For full details, see the descriptions of the <rep_list> and <representative> elements in the SupportSession section.

<code><team_list></code>	Listing of all teams to which the session was transferred. For full details, see the descriptions of the <code><team_list></code> and <code><team></code> elements in the SupportSession section.
<code><rep_resolved></code>	This element is present for backwards compatibility. In the BeyondTrust API versions 1.0.0 and above, this value will always be 0 .
<code><question_list></code>	Contains a <code><question></code> element for each question in this survey. This element contains several child elements as described below. Note that the <code><question></code> elements and their child <code><answer></code> elements are displayed as they are currently configured in the administrative interface. If a question was edited since the time of the first returned survey, the answers may not appear exactly as they were submitted.

/exit_survey_list/exit_survey/question_list/question

<code>id</code> (attribute)	The unique ID of this question.
<code><name></code>	The name of the question as used to identify it within the web interface.
<code><type></code>	The type of question, which can be radio , checkbox , select , text or textarea .
<code><label></code>	The question text as displayed to the user taking the survey.
<code><report_header></code>	The value used to identify this question in the report.
<code><answer_list></code>	Listing of <code><answer></code> elements entered by the user. Radio , text , and textarea questions have a maximum of one <code><answer></code> . Checkbox and select questions may have more than one <code><answer></code> if multiple selection is enabled.

/exit_survey_list/exit_survey/question_list/question/answer_list

<code><answer></code>	The answer entered by the user. For radio , checkbox and select questions, this is the logged value for the selected options. For text and textarea types, it is the text typed by the user. If the question is unanswered, it will be blank.
-----------------------------	--

Query Examples for SupportCustExitSurvey and SupportRepExitSurvey

Customer surveys for sessions started October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=all</code>
Customer surveys for sessions started October 1 2016 to present for all teams, by team	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=0&report_type=team&id=all</code>
Customer surveys for sessions started October 1 2016 to present for a specific rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=1</code>
Customer surveys for sessions started October 1 2016 to present for a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=0&report_type=team&id=1</code>
Customer surveys for session started the month of October 2016 for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=31&report_type=rep&id=all</code>

Customer surveys for sessions started 8:00 AM October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_time=1475308800&duration=0&report_type=rep&id=all</code>
Customer surveys for session started 8:00 AM October 1 2016 to 6.00 PM October 1 2016 for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_time=1475308800&duration=36000&report_type=rep&id=all</code>
Customer surveys for sessions ended October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&end_date=2016-10-01&duration=0&report_type=rep&id=all</code>
Customer surveys for session ended the month of October 2016 for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&end_date=2016-10-01&duration=31&report_type=rep&id=all</code>
Customer surveys for sessions ended 8:00 AM October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&end_time=1475308800&duration=0&report_type=rep&id=all</code>
Customer surveys for session ended 8:00 AM October 1 2016 to 6.00 PM October 1 2016 for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&end_time=1475308800&duration=36000&report_type=rep&id=all</code>
Customer surveys for sessions started October 1 2016 to present for all reps, by rep, for a specific site	<code>https://support.example.com/api/reporting?generate_report=SupportCustExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=all&site_id=1</code>
Representative surveys for sessions started October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=all</code>
Representative surveys for sessions started October 1 2016 to present for all teams, by team	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=0&report_type=team&id=all</code>
Representative surveys for sessions started October 1 2016 to present for a specific rep	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=1</code>
Representative surveys for sessions started October 1 2016 to present for a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=0&report_type=team&id=1</code>
Representative surveys for session started the month of October 2016 for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=31&report_type=rep&id=all</code>
Representative surveys for sessions started 8:00 AM October 1 2016 to present for all reps, by rep	<code>https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_time=1475308800&duration=0&report_type=rep&id=all</code>
Representative surveys for session started 8:00 AM October 1 2016 to 6.00 PM October	<code>https://support.example.com/api/reporting?</code>

1 2016 for all reps, by rep	generate_report=SupportRepExitSurvey&start_time=1475308800&duration=36000&report_type=rep&id=all
Representative surveys for sessions ended October 1 2016 to present for all reps, by rep	https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&end_date=2016-10-01&duration=0&report_type=rep&id=all
Representative surveys for session ended the month of October 2016 for all reps, by rep	https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&end_date=2016-10-01&duration=31&report_type=rep&id=all
Representative surveys for sessions ended 8:00 AM October 1 2016 to present for all reps, by rep	https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&end_time=1475308800&duration=0&report_type=rep&id=all
Representative surveys for session ended 8:00 AM October 1 2016 to 6.00 PM October 1 2016 for all reps, by rep	https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&end_time=1475308800&duration=36000&report_type=rep&id=all
Representative surveys for sessions started October 1 2016 to present for all reps, by rep, for a specific site	https://support.example.com/api/reporting?generate_report=SupportRepExitSurvey&start_date=2016-10-01&duration=0&report_type=rep&id=all&site_id=1

Download Reports with SupportTeam

The **SupportTeam** query returns information about activity within a support team. You may use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**

The API account must have the permission **Allow Access to Support Session Reports and Recordings**.

Parameters for SupportTeam

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return team activity that began on or after this date and that is within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report should return team activity that began at or after this time and that is within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report should return team activity that ended on or after this date and that is within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report should return team activity that ended at or after this time and that is within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration will represent days; if start_time or end_time is specified, duration will represent seconds.

Optional Parameter for SupportTeam

<code>team_id=[integer]</code>	The numeric ID of the team by which to filter results. Only the activity within the specified team will be returned. If this parameter is not specified, results from all teams will be returned. To get a team's ID, see "API Command: get_support_teams" on page 26 .
--------------------------------	---

XML Response for SupportTeam Query

<code><team_activity_list></code>	<p>Contains a <team_activity> element for each team with any activity within the given parameters. If no teams are returned, this element will contain no <team_activity> elements. If an error occurs during the search, it will contain an <error> element describing the problem.</p> <p>Also contains <start_time> and <end_time> elements displaying the time parameters in the system time and with a timestamp attribute in UTC.</p>
---	---

Element Names and Attributes

/team_activity_list/team_activity


id (attribute)	Integer representing the team's unique ID.
name (attribute)	The display name of the support team. Note that this field contains the team name as it currently appears, which may not match the value at the time of the conference if the team name has been subsequently changed.
<logged_in_representatives>	Contains a <representative> element for each representative in that team who was logged into the representative console before the first event in the report occurred. If no representatives were logged in at the start time, this element will be empty.
<events>	Contains an <event> element for each event that occurred within this team.

/team_activity_list/team_activity/logged_in_representatives/representative

gsnumber (attribute)	<p>Uniquely identifies the representative in regards to their current connection to the BeyondTrust Appliance B Series. A gsnumber is assigned on a per-connection basis, so if a representative leaves a session and then rejoins without logging out of the B Series Appliance, their gsnumber will remain the same.</p> <p>However, if the representative's connection is terminated for any reason, when that representative logs back into the B Series Appliance, they will be assigned a new gsnumber.</p> <p>A gsnumber may be recycled, so while two people connected at the same time will never have the same gsnumber, one person may have a gsnumber that was assigned to another person in the past. Can be used to correlate a <representative> element with an event's <performed_by> or <destination> element.</p>
id (attribute)	Unique ID assigned to the representative.
<display_name>	This element is deprecated as of API version 1.10.0 but still exists for backwards compatibility. Its value is the same as that of <private_display_name> .
<public_display_name>	The public display name assigned to the representative. Note that this field contains the public display name's value at the time of the conference, which may not match the current value if the public_display_name has subsequently been changed.
<private_display_name>	The private display name assigned to the representative. Note that this field contains the private display name's value at the time of the conference, which may not match the current value if the private_display_name has subsequently been changed.
<public_ip>	The representative's public IP address.
<private_ip>	The representative's private IP address.

/team_activity_list/team_activity/events/event

timestamp (attribute)	The system time at which the event occurred.
event_type (attribute)	The type of event which occurred. Event types include the following:

	<table border="1"> <tr><td>Chat Message</td><td>Pinned Session Moved Away from Queue</td></tr> <tr><td>Conference Member Added</td><td>Pinned Session Moved to Queue</td></tr> <tr><td>Conference Member Departed</td><td>Pinned Session Password Modified</td></tr> <tr><td>Conference Member State Changed</td><td>Representative Monitoring Started</td></tr> <tr><td>Conference Owner Changed</td><td>Representative Monitoring Stopped</td></tr> <tr><td>File Download</td><td>Session Pinned to Queue</td></tr> <tr><td>File Download Failed</td><td>Session Transferred Away from Queue</td></tr> <tr><td>File Upload</td><td>Session Transferred to Queue</td></tr> <tr><td>File Upload Failed</td><td>Session Unpinned from Queue</td></tr> <tr><td>Files Shared</td><td></td></tr> </table>	Chat Message	Pinned Session Moved Away from Queue	Conference Member Added	Pinned Session Moved to Queue	Conference Member Departed	Pinned Session Password Modified	Conference Member State Changed	Representative Monitoring Started	Conference Owner Changed	Representative Monitoring Stopped	File Download	Session Pinned to Queue	File Download Failed	Session Transferred Away from Queue	File Upload	Session Transferred to Queue	File Upload Failed	Session Unpinned from Queue	Files Shared	
Chat Message	Pinned Session Moved Away from Queue																				
Conference Member Added	Pinned Session Moved to Queue																				
Conference Member Departed	Pinned Session Password Modified																				
Conference Member State Changed	Representative Monitoring Started																				
Conference Owner Changed	Representative Monitoring Stopped																				
File Download	Session Pinned to Queue																				
File Download Failed	Session Transferred Away from Queue																				
File Upload	Session Transferred to Queue																				
File Upload Failed	Session Unpinned from Queue																				
Files Shared																					
<performed_by>	The entity that performed the action. Indicates the entity's gsnumber and also its type , indicating whether this entity was the system or a representative.																				
<destinations>	If this event was targeted to one or more specific representatives, it will contain one or more <destination> elements as described below.																				
<files>	If this event involved the transferring of files, then this element will contain a <file> element for every file transferred.																				
<data>	<p>Contains an arbitrary number of <value name="_" value="_" /> elements. The name and number of these elements varies based on the event_type. For example, when a representative logs into the representative console, a Conference Member Added event would contain <value> elements for the hostname, name, os, private_ip, public_ip, support_teams and user_id.</p> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;">  <p>Note: The XML data returned for the value of the element named support_teams includes an extra ; delimiter at the beginning of the data stream.</p> </div>																				
<body>	The text of the chat message as displayed in the chat log area.																				
<encoded_body>	Can be shown in place of the <body> element above. Contains the base64 (RFC 2045 section 6.8) encoded value of what would have been shown in the <body> element, and is shown ONLY if the <body> text contains characters that are invalid according to XML specification. These characters are typically the result of binary data being sent through chat messages.																				

/team_activity_list/team_activity/events/event/destinations/destination

gsnumber (attribute)	Indicates the gsnumber of the entity to which the event was destined.
[value]	The name of the entity to which the event was destined.

/team_activity_list/team_activity/events/event/files/file

name (attribute)	The name of the transferred file.
size (attribute)	An integer indicating the size of the transferred file.

Query Examples for SupportTeam

Activity started October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&start_date=2016-10-01&duration=0</code>
Activity started the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&start_date=2016-10-01&duration=31</code>
Activity started 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&start_time=1475308800&duration=0</code>
Activity started 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&start_time=1475308800&duration=36000</code>
Activity started October 1 2016 to present for a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&start_date=2016-10-01&duration=0&team_id=1</code>
Activity ended October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&end_date=2016-10-01&duration=0</code>
Activity ended the month of October 2016	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&end_date=2016-10-01&duration=31</code>
Activity ended 8:00 AM October 1 2016 to present	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&end_time=1475308800&duration=0</code>
Activity ended 8:00 AM October 1 2016 to 6:00 PM October 1 2016	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&end_time=1475308800&duration=36000</code>
Activity ended October 1 2016 to present for a specific team	<code>https://support.example.com/api/reporting?generate_report=SupportTeam&end_date=2016-10-01&duration=0&team_id=1</code>

Download Syslog Report

The **Syslog** query downloads a ZIP file containing all Syslog files available on the appliance. Syslog files include all changes made on the /login administrative interface within the last 30 days.

Query Example for Syslog

Syslog

https://access.example.com/api/reporting?generate_report=Syslog

Download Reports with ArchiveListing

The **ArchiveListing** query returns a list of available state archives and event archives for a given date. History is stored for the past seven active days (days that the B Series Appliance's processes have been running).

State archives are created automatically every thirty minutes, and event archives contain events spanning an interval of ten minutes. Each listing represents an archive that is available for download.

Archives are not available immediately after the time they represent. It may take a few minutes for a state or event archive to be created. It is helpful to query **ArchiveListing** before querying **Archive** to be sure that the archive has been created.

The API account must have the permission **Allow Access to Archive Reports**.



IMPORTANT!

To run **ArchiveListing** or **Archive** reports, ensure that the **Enable Archive API** option is checked on the **Management > API Configuration** page of the `/login` administrative interface. The state archive API can be enabled independently of other APIs.

Parameters for ArchiveListing

`date=[YYYY-MM-DD]`

Specifies the date for which to return a list of archives.

XML Response for ArchiveListing Query

`<archives>`

Contains zero or more `<state_archive>` and/or `<event_archive>` elements. A state archive contains a snapshot of the system state at a given time. An event archive contains an ordered list of events that occurred between two points in time.

Element Names and Attributes

`/archives/state_archive`

`<time>`

The time at which the snapshot of the system's state was taken, returned in ISO 8601 format. Also contains a **timestamp** attribute which displays the time as a UNIX timestamp (UTC).

`<index>`

The state archive's unique index for the given day. The first state archive of the day always starts at 1. This corresponds with the index of the state archive report.

`<event_archive_index>`

The index of the event archive that starts immediately after the snapshot was taken.

`/archives/event_archive`

`<time>`

The time of the first possible event in the archive, returned in ISO 8601 format. Also contains a **timestamp** attribute which displays the start time as a UNIX timestamp (UTC).

`<end_time>`

The time of the last possible event in the archive, returned in ISO 8601 format. Also

	contains a timestamp attribute which displays the end time as a UNIX timestamp (UTC).
<index>	The archive's unique index for the given day. The first event archive of the day always starts at 1. This corresponds with the index of the event archive report.

Query Example for ArchiveListing

Archive listings for the date of October 1 2016	https://support.example.com/api/reporting? generate_report=ArchiveListing&date=2016-10-01
---	--

Download Reports with Archive

The **Archive** query returns a gzip compressed file with the system state or an event log for a given date and time. The file must be decompressed before it can be parsed. The decompressed file never exceeds 4 GB. History is stored for the past seven active days (days that the B Series Appliance's processes have been running).

State archives are created automatically every thirty minutes, and event archives contain events spanning an interval of ten minutes. Each archive corresponds with an archive listing. Integrations must choose the appropriate archive and replay events from that archive to reconstruct the state at a given point in time.

Archives are not available immediately after the time they represent. It may take a few minutes for a state or event archive to be created. It is helpful to query **ArchiveListing** before querying **Archive** to be sure that the archive has been created.

The API account must have the permission **Allow Access to Archive Reports**.



IMPORTANT!

To run **ArchiveListing** or **Archive** reports, ensure that the **Enable Archive API** option is checked on the **Management > API Configuration** page of the **/login** administrative interface. The state archive API can be enabled independently of other APIs.

Parameters for Archive

<code>type=[string]</code>	The type of archive to download. May be either state or event .
<code>date=[YYYY-MM-DD]</code>	Specifies the date for which to return the archive.
<code>index=[integer]</code>	The index of the archive to download. This index corresponds with the index in ArchiveListing .

JSON Response for Archive Query

The state archive file consists of a single JSON (JavaScript Object Notation) object. Multiple tables and IDs may be specified. If a table does not have any data, its value is null. Possible tables and fields are detailed below.

The event archive file consists of a JSON object for each line in the decompressed file. Events are ordered from oldest to newest in the order in which they actually occurred. Possible tables and fields are detailed below. The following event types are supported:

<code>model_insert</code>	Creates a table with all fields included, even if blank.
<code>model_update</code>	Adds or modifies one or more rows in a table.
<code>model_delete</code>	Removes a table.
<code>truncate_model</code>	Signifies that the B Series Appliance's processes have restarted and that all events logged so far are no longer valid. If any integrations are building data or tables based on the event archive file, they should clean all rows from all tables when this event is received.
<code>session_event</code>	Creates a table with data about events that occurred within the session.

JSON Tables and Fields for `model_insert` and `model_update`

customer_client

Stores all customer clients connected to the B Series Appliance. Note that a **support_session** can exist without a **customer_client**. This typically occurs when the customer closes the customer client and the representative does not immediately terminate the session in the representative console.

<code>(id)</code>	An integer that identifies this table row. This ID is unique for the lifetime of the model.
<code>"client_type":["string"]</code>	The type of customer client running on the endpoint or communicating with the endpoint. May be one of desktop , mobile , rdp , vnc , vpro , shell , or web_chat .
<code>"connected":["boolean"]</code>	1 : The customer client is connected to the B Series Appliance. 0 : The customer client is not connected. The endpoint system may be rebooting or in the process of elevating.
<code>"created_timestamp":["timestamp"]</code>	A UNIX timestamp (UTC) representing the time at which this table row was created.
<code>"elevated":["boolean"]</code>	1 : The customer client is running in an elevated context. 0 : The customer client is running in a user context.
<code>"hostname":["string"]</code>	The endpoint's hostname.
<code>"operating_system":["string"]</code>	The endpoint's operating system.
<code>"private_ip":["string"]</code>	The endpoint's private IP address.
<code>"public_ip":["string"]</code>	The endpoint's public IP address.
<code>"support_session_id":["integer"]</code>	The foreign key that links this table row to other table rows that reference this session.



queue

Stores all active support session queues. A queue is active if one or more of its members are logged in or if it is a persistent queue.

<code>(id)</code>	An integer that identifies this table row. This ID is unique for the lifetime of the model.
<code>"code_name":["string"]</code>	The code name assigned to the support team. Personal queues do not have code names.
<code>"created_timestamp":["timestamp"]</code>	A UNIX timestamp (UTC) representing the time at which this table row was created.
<code>"name":["string"]</code>	The support team's display name.
<code>"support_team_id":["string"]</code>	The support team's unique identifier. The ID is not shown for personal queues.
<code>"type":["string"]</code>	The type of queue. May be one of prewait , team , or personal .

representative

Stores all representatives logged into a representative console

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"connected":[boolean]	<p>1: The representative is connected to the B Series Appliance. 0: The representative is not connected to the B Series Appliance.</p> <div style="border: 1px solid black; padding: 5px; background-color: #e0f0ff;">  <p>Note: A representative may be considered logged in even if the representative console is not connected to the B Series Appliance. This could occur if the representative console has lost its connection but the reconnect timeout has not been reached.</p> </div>
"console_in_focus":[boolean]	<p>1: The representative console is in focus on the representative's computer. 0: The representative console is not in focus.</p>
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"hostname":[string]	The hostname of the representative's system.
"operating_system":[string]	The operating system of the representative's system.
"private_display_name":[string]	The representative's private display name.
"private_ip":[string]	The private IP address of the representative's system.
"public_display_name":[string]	The representative's public display name.
"public_ip":[string]	The public IP address of the representative's system.
"queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue.
"routing_available":[boolean]	<p>1: The representative is available to handle new support sessions. 0: The representative is unavailable to handle new support sessions.</p>
"routing_busy":[boolean]	<p>1: The representative has reached their maximum number of support sessions for session assignment. 0: The representative is not marked as busy.</p>
"routing_enabled":[boolean]	<p>1: The representative has automatic session assignment enabled. 0: The representative has automatic session assignment disabled.</p>
"routing_idle":[boolean]	<p>1: The representative is idle. 0: The representative is active.</p>
"selected_support_session_id":[integer]	<p>The ID of the support session currently active in the representative console.</p> <div style="border: 1px solid black; padding: 5px; background-color: #e0f0ff;">  <p>Note: The active support session is recorded once every five seconds. If the representative switches sessions very frequently, some of those session selection change events may not be recorded.</p> </div>

"skill_code_names": "[string]"	A comma-separated list of skill code names assigned to this representative.
"type": "[string]"	The type of representative connection. May be one of normal or invited .
"user_id": [integer]	The representative's unique ID which identifies them in BeyondTrust.
"username": "[string]"	The username this representative uses to authenticate to BeyondTrust.

representative_queue

Stores logged-in representatives in relation to their support queues. While team leads and managers have access to their team members' personal queues, that access is not represented in this table.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp": [timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"queue_id": [integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue.
"representative_id": [integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.
"role": [string]	The user's role in the team. May be one of member , lead , or manager .

representative_support_session

Stores representatives participating in support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp": [timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"representative_id": [integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.
"selected_support_session_service_tool": [string]"	The selected support session service tab. May be one of screen_sharing , file_transfer , system_information , remote_shell , or registry .
"support_session_id": [integer]	The foreign key that links this table row to other table rows that reference this session.

representative_support_session_tool

Stores a list of support tools in use by representatives in sessions. Rows are added when the representative starts using the tool and are removed when the representative stops using the tool.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"representative_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.
"support_session_id":[integer]	The foreign key that links this table row to other table rows that reference this session.
"tool":[string]	The selected support session service tab. May be one of screen_sharing , file_transfer , system_information , remote_shell , or registry .

support_session

Stores all active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"customer_company":[string]	The end user's company name.
"customer_company_code":[string]	The end user's company code.
"customer_description":[string]	The issue description as provided by the customer.
"customer_name":[string]	The end user's name.
"estimated_pickup_timestamp":[timestamp]	The time at which the system expects this session to be accepted or transferred. This could be a time in the past or future.
"issue_code_name":[string]	The code name of the issue selected by the customer on the front-end survey.
"issue_description":[string]	The description of the issue selected by the customer on the front-end survey.
"lsid":[string]	The logging session ID that uniquely identifies this session. This LSID can be used in /login > Reports , with the reporting API, in the integration client, etc.
"primary_team_queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue. This field stores the last team queue that owned the session before the session ended.
"priority":[integer]	The session's priority. May be one of 1 (high), 2 (medium), or 3 (low).
"public_site_id":[integer]	The ID of the public site associated with the session.
"public_site_name":[string]	The name of the public site associated with the session.
"queue_entry_timestamp":[timestamp]	The time at which this session entered its current queue.

"queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue. This field stores the queue in which the session currently resides.
"start_method":[string]	The method with which the session was started. May be one of session_key (the seven-digit code or the URL), rep_list , issue_submission , local_jump , remote_jump , shell_jump , local_rdp , remote_rdp , local_vnc , remote_vnc , or vpro .

support_session_attribute

Stores custom session attributes assigned to active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"code_name":[string]	The code name assigned to the support session attribute.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"support_session_id":[integer]	The foreign key that links this table row to other table rows that reference this session.
"value":[string]	The value of the attribute.

support_session_skill

Stores skills assigned to active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"code_name":[string]	The code name assigned to the skill.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"support_session_id":[integer]	The foreign key that links this table row to other table rows that reference this session.

presentation_session

Stores data about the presentation sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"lsid":[guid]	The logging session ID that uniquely identifies this session. This LSID can be used on /login > Reports and in the Reporting API, among other places.
"presentation_name":[string]	The name of the presentation.

attende

Stores all information about attendees who have joined a presentation.

"connected":[boolean]	1: The presentation client connected to the B Series Appliance and joined the presentation. 0: The presentation client did not connect to the B Series Appliance.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"hostname":"[string]"	The attendee's hostname.
"name":"[string]"	The name of the attendee.
"operating_system":"[string]"	The attendee's operating system.
"presentation_session_id":[integer]	The logging session ID that uniquely identifies this presentation.
"private_ip":"[string]"	The attendee computer's private IP address.
"public_ip":"[string]"	The attendee computer's public IP address.
(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.

presenter

Stores the information about the presenter who has started presenting their screen.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"presentation_session_id":[integer]	The logging session ID that uniquely identifies this presentation.
"representative_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.

JSON Tables and Fields for session_event

These data elements are returned for session events. All but the **data** element occur for each session event.

"data":[object]	Contains details about the session event. These details are described in the tables below.
"lsid":"[string]"	The logging session ID that uniquely identifies this session. This LSID can be used in /login > Reports , with the reporting API, in the integration client, etc.
"name":"[string]"	The name of the session event. These session events are described in the tables below.
"timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this event occurred.
"type":"[string]"	The type of table. The value is fixed to session_event for all session events.

callback_button_deployed

"callback_button_id":[integer]	The ID of the Support Button that was deployed.
"description":"[string]"	The description of the Support Button as entered by the representative.
"expiration":[timestamp]	A UNIX timestamp (UTC) representing the time at which the Support Button will expire.
"profile_id":[integer]	The ID of the Support Button profile used.
"profile_name":"[string]"	The name of the Support Button profile used.
"representative_id":[integer]	The unique ID of the representative who deployed the Support Button.

callback_button_removed

"callback_button_id":[integer]	The ID of the Support Button that was removed.
"representative_id":[integer]	The unique ID of the representative who removed the Support Button.

canned_script_executed

"script_name":"[string]"	The name of the canned script that was executed.
--------------------------	--

chat_message

"body": "[string]"	The message to be rendered in the recipient's chat window. The body can contain the %name% macro. Integrations should substitute it with the name of the performed_by member.
"destination": [object]	The list of recipients receiving the chat message.
"performed_by": [object]	The list of the name-value pairs showing who performed the chat operation.

command_shell_session_started

"download_url": "[string]"	The URL to download the command shell recording.
"instance": [integer]	The index of the command shell instance.
"representative_id": [integer]	The unique ID of the representative who started the command shell operation.
"view_url": "[string]"	The URL to view the command shell recording.

customer_exit_survey

"responses": [object]	The list of question-answer pairs in the customer exit survey.
-----------------------	--

directory_created

"destination": [object]	The list of name-value pairs showing on which member's system the directory was created.
"failure_reason": "[string]"	Optional: The reason the directory creation failed, shown only if an error occurred.
"path": "[string]"	The absolute path of the directory.
"performed_by": [object]	The list of name-value pairs showing who created the directory.

file_deleted

"destination": [object]	The list of name-value pairs showing from which member's system the file was deleted.
"failure_reason": "[string]"	Optional: The reason the file deletion failed, shown only if an error occurred.
"name": "[string]"	The name of the deleted file.
"performed_by": [object]	The list of name-value pairs showing who deleted the file.
"size": [integer]	The size of the deleted file, given in bytes.

file_download

"destination":[object]	The list of name-value pairs showing to which member's system the file was downloaded.
"name":"[string]"	The name of the downloaded file.
"performed_by":[object]	The list of name-value pairs showing from which member's system the file was downloaded.
"size":[integer]	The size of the downloaded file, given in bytes.

file_download_incomplete

"bytes_transferred":[integer]	The number of bytes transferred successfully before the file download failed.
"destination":[object]	The list of name-value pairs showing to which member's system the file was being downloaded.
"failure_reason":"[string]"	The reason the file download failed.
"name":"[string]"	The name of the file being downloaded.
"performed_by":[object]	The list of name-value pairs showing from which member's system the file was being downloaded.

file_moved

"destination":[object]	The list of name-value pairs showing on which member's system the file was moved.
"failure_reason":"[string]"	Optional: The reason the move failed, shown only if an error occurred.
"new_path":"[string]"	The absolute path of the file after it was moved.
"old_path":"[string]"	The absolute path of the file before it was moved.
"performed_by":[object]	The list of name-value pairs showing who moved the file.

file_upload

"destination":[object]	The list of name-value pairs showing to which member's system the file was uploaded.
"name":"[string]"	The name of the uploaded file.
"performed_by":[object]	The list of name-value pairs showing from which member's system the file was uploaded.
"size":[integer]	The size of the uploaded file, given in bytes.

file_upload_incomplete

"bytes_transferred":[integer]	The number of bytes transferred successfully before the file upload failed.
"destination":[object]	The list of name-value pairs showing to which member's system the file was being uploaded.
"failure_reason":"[string]"	The reason the file upload failed.
"name":"[string]"	The name of the file being uploaded.
"performed_by":[object]	The list of name-value pairs showing from which member's system the file was being uploaded.

files_shared

"files":[object]	<p>The list of shared files and their attributes. This object contains the following elements:</p> <p><name> The name of the shared file.</p> <p><size> The size of the shared file.</p> <p><type> The file type. Possible values of this field are file and dir.</p>
"performed_by":[object]	The list of name-value pairs showing which member shared the files.

legal_agreement_response

"response":"[string]"	The action taken by the end-user. Possible values are accept , decline , and timeout .
"text":"[string]"	The text of the legal agreement shown in the message box to the customer.
"title":"[string]"	The title of the message box where the legal agreement was shown to the customer.

registry_exported

"representative_id":[integer]	The unique ID of the representative who exported the registry.
"root_path":"[string]"	The root of the exported tree.

registry_imported

"representative_id":[integer]	The unique ID of the representative who imported the registry.
-------------------------------	--

registry_key_added

"key": "[string]"	The name of the added registry key.
"path": "[string]"	The registry key's parent path.
"representative_id": [integer]	The unique ID of the representative who added the registry key.

registry_key_deleted

"key": "[string]"	The name of the deleted registry key.
"path": "[string]"	The registry key's parent path.
"representative_id": [integer]	The unique ID of the representative who deleted the registry key.

registry_key_renamed

"new_key": "[string]"	The registry key's name after modification.
"old_key": "[string]"	The registry key's name before modification.
"path": "[string]"	The registry key's parent path.
"representative_id": [integer]	The unique ID of the representative who renamed the registry key.

registry_value_added

"data": "[string]"	The data of the added registry key value.
"name": "[string]"	The name of the added registry key value.
"path": "[string]"	The key path to the added registry key value.
"representative_id": [integer]	The unique ID of the representative who added the registry key value.
"type": "[string]"	The type of the added registry key value data.

registry_value_deleted

"name": "[string]"	The data of the deleted registry key value.
"path": "[string]"	The key path to the deleted registry key value.
"representative_id": [integer]	The unique ID of the representative who deleted the registry key value.

registry_value_modified

"name": "[string]"	The name of the modified registry key value.
"new_data": "[string]"	The data of the registry key value after modification.
"old_data": "[string]"	The data of the registry key value before modification.
"path": "[string]"	The key path to the modified registry key value.
"representative_id": [integer]	The unique ID of the representative who modified the registry key value.
"type": "[string]"	The type of the modified registry key value data.

registry_value_renamed

"new_name": "[string]"	The name of the registry key value after modification.
"old_name": "[string]"	The name of the registry key value before modification.
"path": "[string]"	The key path to the registry key value.
"representative_id": [integer]	The unique ID of the representative who renamed the registry key value.

representative_survey

"responses": [object]	The list of question-answer pairs in the representative survey.
-----------------------	---

screen_recording

"download_url": "[string]"	The URL to download the screen recording.
"representative_id": [integer]	The unique ID of the representative in session during this recording.
"view_url": "[string]"	The URL to view the screen recording.

screenshot_captured

"representative_id": [integer]	The unique ID of the representative who captured this screenshot.
--------------------------------	---

session_assigned

"representative_id": [integer]	The unique ID of the representative who received the session assignment alert.
"timeout": [integer]	Optional: The number of seconds before the alert timed out if it was not accepted.

session_assignment_response

"response": "[string]"	Text representing the response to the session assignment request.
------------------------	---

session_foreground_window_changed

"exe_name": "[string]"	The executable name of the foreground window.
------------------------	---

"window_name": "[string]"	The title of the foreground window.
---------------------------	-------------------------------------

session_note_added

"body": "[string]"	The message to be rendered as a session note.
--------------------	---

"number": [integer]	The index of the session note that was created for this session.
---------------------	--

"representative_id": [integer]	The unique ID of the representative who added the session note.
--------------------------------	---

show_my_screen_recording

"download_url": "[string]"	The URL to download the show my screen recording.
----------------------------	---

"instance": [integer]	The index of the show my screen recording instance.
-----------------------	---

"representative_id": [integer]	The unique ID of the representative showing their screen.
--------------------------------	---

"view_url": "[string]"	The URL to view the show my screen recording.
------------------------	---

special_action_executed

"action_name": "[string]"	The name of the special action that was executed.
---------------------------	---

system_information_retrieved

"system_information": [object]	This object contains multiple data sets, each with a category name , a columns element with a list of field names for the category, and a rows element with field values for each field name.
--------------------------------	--

JSON Tables and Fields for Presentation Session Events

The Presentation Session Event shows the events that are logged into the event archive file.



Note: The following session events are excluded from the archive file: **Session Start, Session End, Conference Member Added, Conference Member Removed, and Conference Member State Changed**. These are excluded because the data needed to log these events can be created from existing state model tables like **presentation_session, presenter, and attendee**. Please see "[System State Model of the Real-Time API](#)" on page 158 for more information.

Required Data Elements

Chat Message

"performed_by":[object]	The list of the name-value pairs showing who performed the chat operation.
"destination":[object]	The list of the recipients who will be receiving the chat message.
"body":"[string]"	Message to be rendered in the recipient's chat window. The body can contain the %name macro. The integrations should substitute it with the name of the performed_by member.

Files Shared

"performed_by":[object]	The list of the name-value pairs showing who shared the files.
"files":"[object]"	The list of the shared files and their attributes. This object contains following elements: <name> is the name of the file. <size> is the size of the file. <type> is the file type. valid values of this field are file and dir .

Screen Recording

"presenter_id":[integer]	The unique ID of the presenter.
"download_url":"[string]"	The URL to download the screen recording.
"view_url":[integer]	The URL to view the screen recording.
"appliance_guid":"[string]"	The GUID of the B Series Appliance where the recording was created.

Use Cases

Integrations can use the available data to generate metrics as needed. Below are some examples of how certain metrics could be generated from the data.

Average Work Time

The average amount of time a representative spends actively working on a single support session. Work time for a single session includes all time where the following are true:

- The representative console has focus (`representative.console_in_focus = true`)
- The support session is selected (`representative.selected_support_session_id = support_session.id`)
- The customer client is connected (`customer_client.connected = true`)
- The representative is not idle (`representative.routing_idle = false`)

The average work time can be computed for a set of sessions over a given time range. For example, compute the average work time for all sessions between 8am and 5pm on a given date.

Total Active Time

The cumulative amount of time the representative was responsible for an active support session. Active time for a single session includes all time where the following are true:

- The support session is in the representative's personal queue (`representative.id = queue.representative_id AND queue.id = support_session.queue_id`)
- The representative is in the support session (`representative_support_session.support_session_id = support_session.id AND representative_support_session.representative_id = representative_id`)
- The customer client is connected (`customer_client.connected = true`)

The total active time is the sum of active time for all sessions over a given time range.

Rep Login/out Time

When a representative logs in, a new row is inserted into the `representative` table.

When a representative logs out, a row is removed from the `representative` table.

The `representative.connected` field can be used to know when a representative is logged in but not connected.

Number of Users Logged In during a Given Period of Time

Count the number of rows in the `representative` table. The period of time can be replayed using the event archive to compute the minimum, maximum, and average number of representatives logged in.

Representative Time without a Session

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative is not in any support sessions (no rows where `representative_support_session.representative_id = representative.id`)
- The representative is connected (`representative.connected = true`)

Amount of Time in Concurrent Sessions

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative is in X support sessions (X is the number of rows where `representative_support_session.representative_id = representative.id`)
- The representative is connected (`representative.connected = true`)

Time in Auto Assign Mode

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative has auto-assign enabled (`representative.routing_enabled = true`)

Time Available

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative is available for routing (`representative.routing_available = true`)

Representative Time Idle

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative is idle (`representative.routing_idle = true`)

Time Waiting for a Customer to Reconnect

Sum the number of seconds where the following are true:

- The representative is logged in (has a row in the `representative` table)
- The representative is in the session (`representative_support_session.representative_id = representative_id` AND `representative_support_session.support_session_id = support_session.id`)
- The customer is in the session (`customer_client.support_session_id = support_session.id`)
- The customer is not connected (`customer_client.connected = false`)

Query Example for Archive

Download state archive 50 for October 1
2016

```
https://support.example.com/api/reporting?  
generate_report=Archive&type=state&date=2016-10-01&  
index=50
```

Download event archive 50 for October 1
2016

```
https://support.example.com/api/reporting?  
generate_report=Archive&type=event&date=2016-10-01&  
index=50
```

Download Reports with LicenseUsage

The **LicenseUsage** query returns an overview of peak license usage times, grouped by hour, day, or month. Data is added to this report when at least 90% of your BeyondTrust licenses are in use. You may use any of the following sets of parameters to generate reports:

- **start_date**, **duration**, and **group_by**
- **start_time**, **duration**, and **group_by**

The API account must have the permission **Allow Access to License Usage Reports**.

Parameters for LicenseUsage

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report should return peak license usage data beginning on or after this date.
<code>start_time=[timestamp]</code>	Specifies that the report should return peak license usage data beginning at or after this time. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date is specified, duration represents days; if start_time is specified, duration represents seconds.
<code>group_by</code>	Specifies whether the data should be grouped by hour , day , or month .

XML Response for LicenseUsage Query

<code><license_usage></code>	<p>Contains a <license_time_intervals> element. If no license usage data is returned, this element will contain no license usage elements. If an error occurs during the search, it will contain an <error> element describing the problem.</p> <p>Also contains <start_time> and <end_time> elements displaying the time parameters in the system time and with a timestamp attribute in UTC. A <group_by> element shows whether the data is grouped by hour, day, or month.</p>
------------------------------------	---

Element Names and Attributes

	<i>//license_usage/license_time_intervals</i>
<code><license_time_interval></code>	Contains a <license_time_interval> element for each time at which peak license usage was logged.
	<i>//license_usage/license_time_intervals/license_time_interval</i>
timestamp (attribute)	The timestamp at which peak license usage was logged.
datetime (attribute)	The date and time at which peak license usage was logged.
<code><license_count></code>	Contains a <license_count> element for each potential type of license usage. This displays the number of licenses in use at that time.

//license_usage/license_time_intervals/license_time_interval/license_count

license_type (attribute)	The type of license used by the representative.
reason (attribute)	Can be either login or extended_contact , or this attribute may not be present. The login reason indicates that a license was in use due to a user being logged into the representative console. The extended_contact reason indicates that a license was being consumed due to a user being in extended availability mode. If no reason is listed, the license count is the total number of licenses in use.

Query Examples for LicenseUsage

License usage starting October 1 2016 to present, grouped by hour	<code>https://support.example.com/api/reporting?generate_report=LicenseUsage&start_date=2016-10-01&duration=0&group_by=hour</code>
License usage during the month of October 2016, grouped by day	<code>https://support.example.com/api/reporting?generate_report=LicenseUsage&start_date=2016-10-01&duration=31&group_by=day</code>
License usage starting 8:00 AM October 1 2016 to present, grouped by month	<code>https://support.example.com/api/reporting?generate_report=LicenseUsage&start_time=1475308800&duration=0&group_by=month</code>
License usage starting 8:00 AM October 1 2016 to 6:00 PM October 1 2016, grouped by hour	<code>https://support.example.com/api/reporting?generate_report=LicenseUsage&start_time=1475308800&duration=36000&group_by=hour</code>

Download Reports with VaultAccountActivity

The **VaultAccountActivity** query returns full information for all Vault account activity events that match given search parameters. You can use any of the following sets of parameters to generate reports:

- **start_date** and **duration**
- **start_time** and **duration**
- **end_date** and **duration**
- **end_time** and **duration**

The API account must have the permission **Allow Access to Vault Account Activity Reports**.

Parameters for VaultAccountActivity

<code>start_date=[YYYY-MM-DD]</code>	Specifies that the report returns all events that happened on or after this date and that are within the duration specified below.
<code>start_time=[timestamp]</code>	Specifies that the report returns all sessions, those still in progress, that began at or after this time as well as that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>end_date=[YYYY-MM-DD]</code>	Specifies that the report returns only closed sessions that ended on or after this date

	and that are within the duration specified below.
<code>end_time=[timestamp]</code>	Specifies that the report returns only closed sessions that ended at or after this time and that are within the duration specified below. The time must be a UNIX timestamp (UTC).
<code>duration=[integer]</code>	Length of time from the specified date or time for which you wish to pull reports, or 0 to pull from the specified date to present. If start_date or end_date is specified, duration represents days; if start_time or end_time is specified, duration represents seconds.

Optional Parameter for VaultAccountActivity

<code>limit=[string]</code>	<p>The category by which to filter results. Can be one of the following:</p> <p>all Returns all results.</p> <p>rep:[id] Returns sessions owned by a representative, specified by user ID. To get a representative's ID, see "API Command: get_logged_in_reps" on page 23.</p> <p>account: [id] Returns all events involving a specific account.</p>
-----------------------------	---

XML Response for VaultAccountActivity Query

<code><vault_account_activity_list></code>	Contains a <code><vault_account_activity></code> element for each event that matches the given criteria. If no events are returned, this element contains no <code><vault_account_activity></code> elements. If an error occurs during the search, it contains an <code><error></code> element describing the problem.
--	--

Element Names and Attributes

timestamp (attribute)	The system time at which the event occurred.
Account	The ID of the Vault account.

<code>event_type</code> (attribute)	<p>The type of event which occurred. Event types include the following:</p> <ul style="list-style-type: none"> Account Created Account Deleted Credentials Checked Out Credentials Checked In Password Changed Password Rotation Failed Credentials Used Credentials Force Checked In
<code><performed_by></code>	<p>The entity that performed the action. Indicates the entity's ID and also its type, indicating whether this action was performed by the system, a representative, or an API account.</p>
<code><data></code>	<p>The value of this attribute depends on the event_type. For a Password Changed event, it contains values like Manually Edited, Manually Rotated, or Rotate after check in. For a Password Rotation Failed event, it contains an error string explaining the reason for its failure. For a Credential Checked Out event, if the credentials were used in a session, then it contains the LSID of the session.</p>

Real-Time State API

The real-time state API provides information about the current state of the BeyondTrust Appliance B Series. Data include logged-in representatives, support sessions in progress, and queue statuses. The real-time state API provides a summary of the current system state rather than a history of events. It can answer such questions as:

- What sessions are waiting in a queue?
- What sessions have a representative in them?
- What sessions have been in the system for longer than X minutes?
- What is the estimated wait time for a session?
- How many chat support sessions are in progress?
- How many RDP sessions are in progress?
- How many sessions started from a session key or the issue submission form are in progress?
- Which representatives are logged in?
- Which representatives are available to accept sessions from a specific queue?
- In which sessions is a representative participating?
- How long has a representative been logged in?
- Which representatives are idle, are busy, or have automatic session assignment disabled?

The real-time state API is an authenticated API. For instructions on using authenticated APIs using OAuth, see ["Authenticate to the Remote Support API" on page 7](#).

The real-time state API is structured as tables (see ["System State Model of the Real-Time API" on page 158](#)). These data must be parsed by the integration, following specific procedures (see ["Protocol of the Real-Time State API" on page 154](#)). A JavaScript library is provided to facilitate use of the socket-based real-time state API in a browser (see ["JavaScript Library for the Real-Time State API" on page 164](#)).



Note: The real-time state API is not able to answer questions about historical data. For example, some questions the real-time state API cannot answer include:

- How many sessions has a representative participated in during the last X hours?
- What is the total amount of time a representative has been idle during the last X hours?
- What is the total amount of time a representative was participating in at least Y sessions during the last X hours?
- How long has a representative been idle, busy, or available?

For this type of historical question, see ["Download Reports with Archive" on page 131](#).

Protocol of the Real-Time State API

In order to retrieve and maintain data using the real-time state API, a certain protocol must be followed. The protocol is divided into four main parts, shown in order of execution:

- ["Connection" on page 154](#)
 - The integration connects to the B Series Appliance via a secure connection.
- ["Authentication" on page 155](#)
 - The integration sends credentials for authentication.
 - The server authenticates the credentials. The account used must have permission to use the real-time state API.
 - The server sends an authentication response to the integration.
- ["Model Subscriptions" on page 155](#)
 - The integration subscribes to one or more tables in the model.
 - The server sends a full copy of the subscribed tables to the integration.
 - The integration is expected to maintain a copy of the tables.
- ["Model Updates" on page 156](#)
 - The server pushes future updates of the subscribed tables to the integration.
 - The integration is expected to update its copy of the tables.

The connection, authentication, and model subscription phases are serial. All messages in those phases must be sent in the correct order, and all messages are required. Messages in the model updates phase can arrive in any order. Messages are encoded using JSON (JavaScript Object Notation), except for the connection phase.

The real-time state API has certain limits:

- A maximum of thirty connected integrations may receive model updates simultaneously.
- Connected integrations receive updates with a maximum latency of twenty seconds.
- Integrations must have sufficient bandwidth to receive updates. Integrations that get too far behind the real-time stream are automatically disconnected.



IMPORTANT!

Information will be sent from the B Series Appliance within binary websocket messages, and data sent to the B Series Appliance can be sent within text websocket messages. Each message sent to or received from the B Series Appliance is terminated with a newline character (`\n`). It may be necessary to trim this character before parsing the JSON received. Similarly, a newline character must be appended to the end of all messages sent.

Connection

During the connection phase of the protocol, the integration makes a secure web socket connection to the B Series Appliance.

Integration connects to the B Series Appliance using a secure web socket

```
sock = new WebSocket ("wss://<hostname>/nw")
sock.SubProtocol = "ingredi state api"
```



Note: The subprotocol must be set appropriately. It must also be URL encoded when sent to the server (e.g. `"ingredi%20state%20api"`)

Authentication

During the authentication phase of the protocol, the integration authenticates with the B Series Appliance.

Integration sends credentials to the B Series Appliance

Integration → B Series Appliance

```
{
  "type" : "authenticate",
  "credentials" :
  {
    "bearer_token" : "<OAuth 2.0 bearer token>"
  }
}\n
```

B Series Appliance verifies the credentials and sends an authentication response to the integration

B Series Appliance → Integration

```
{
  "type" : "authenticate_response",
  "success" : true,
  // or
  // "success" : false,
  "reason" : "reason if success == false"
}\n
```

After authenticating, the integration must subscribe to one or more tables in the model.

Model Subscriptions

During the model subscriptions phase of the protocol, the integration tells the B Series Appliance the parts of the system state model for which it wants to receive updates.

Integration subscribes to the model

Integration → B Series Appliance

```
{
  "type" : "subscribe",
  "tables" : "all"
  // or
  // "tables" : ["customer_client", "..."]
}\n
```

The `tables` name/value pair can be either "all" to subscribe to all tables or an array of table names in the model. For a list and description of tables, see ["System State Model of the Real-Time API" on page 158](#).

B Series Appliance confirms the subscription

B Series Appliance → Integration

```
{
  "type" : "subscribe_response",
  "timestamp" : <UNIX timestamp>,
  "success" : true,
  // or
  // "success" : false,
  "reason" : "reason if success == false"
}\n
```

In the subscribe response, the timestamp is the B Series Appliance's current time. This is useful for doing time calculations when the integration's clock is skewed from the B Series Appliance's clock.

After receiving the subscribe response, the integration starts to receive model updates from the B Series Appliance.

Model Updates

During the model updates phase of the protocol, the integration receives system state model updates from the B Series Appliance. The integration does not request model updates. Instead, model updates are sent automatically by the B Series Appliance as resources permit.



Note: If the B Series Appliance sends a large update, the data may be broken up multiple parts. Check to see if the message terminates with a newline character (`\n`). If it does not, append further messages until you receive a message ending with a newline, indicating you have reached the end of the data.

Update model message

B Series Appliance → Integration

```
{
  "type" : "model_update"
  // "insert" is specified only if rows need to be inserted.
}
```

```

"insert" :
{
  // One or more table names are specified as keys.
  "<table>" :
  {
    // One or more row IDs are specified as keys.
    "<id>" :
    {
      // One or more field names are specified as keys.
      // Some fields in the table may not be listed.
      "<field>" : "<value>",
      // ...
    },
    // ...
  },
  // "update" is specified only if rows need to be updated
  "update" :
  {
    // This object has the same syntax as "insert"
  },
  // "delete" is specified only if rows need to be deleted
  "delete" :
  {
    // One or more table names are specified as keys.
    // Each value is an array of row IDs to delete.
    "<table>" : ["<id>", "<id>", ...],
    // ...
  },
},
}\n

```

At least one of `insert`, `update`, or `delete` will be specified, and a combination thereof could also be specified. When the message is received, the integration should update its copy of the state model:

- For `insert`, the integration should insert into the specified `<table>`s a row for each given `<id>` and having the given `<value>`s for the specified `<field>`s.
- For `update`, the integration should update existing rows for the given `<table>`s and the given `<id>`s, modifying them to have the given `<value>`s for the specified `<field>`s.
- For `delete`, the integration should delete rows in the given `<table>`s with the given `<id>`s.

Truncate model message

B Series Appliance → Integration

```

{
  "type" : "truncate_model"
}\n

```

After receiving this message, the integration should delete all rows from all tables.

System State Model of the Real-Time API

The system state model of the real-time state API functions similarly to a database.

- It is composed of tables.
- Each table has one or more fields.
- Each table has zero or more rows.
- Tables can relate to each other via rows that serve as functional keys.

In the initial state of the model, no rows exist in any tables. The system state model is updated in real time to reflect system state transitions, such as when clients connect and support sessions begin. The system state model persists until the B Series Appliance's processes are started. After a process restart, the model is reset to its initial state. The time between the model's initial state and the next reset is referred to as the model's lifetime.

Tables in the Real-Time System State Model

The section below details the tables that exist in the system state model, along with the fields that exist in each table.

- ["customer_client" on page 159](#)
- ["queue" on page 159](#)
- ["representative" on page 160](#)
- ["representative_queue" on page 160](#)
- ["representative_support_session" on page 161](#)
- ["support_session" on page 161](#)
- ["support_session_attribute" on page 162](#)
- ["support_session_skill" on page 162](#)
- ["presentation_session" on page 162](#)
- ["attendee" on page 163](#)
- ["presenter" on page 163](#)

customer_client

Stores all customer clients connected to the B Series Appliance. Note that a **support_session** can exist without a **customer_client**. This typically occurs when the customer closes the customer client and the representative does not immediately terminate the session in the representative console.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"client_type": "[string]"	The type of customer client running on the endpoint or communicating with the endpoint. May be one of desktop , mobile , rdp , vnc , vpro , shell , or web_chat .
"created_timestamp": [timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"elevated": [boolean]	1 : The customer client is running in an elevated context. 0 : The customer client is running in a user context.
"hostname": "[string]"	The endpoint's hostname.
"operating_system": "[string]"	The endpoint's operating system.
"support_session_id": [integer]	The foreign key that links this table row to other table rows that reference this session.

Stores all customer clients connected to the B Series Appliance. Note that a **support_session** can exist without a **customer_client**. This typically occurs when the customer closes the customer client and the representative does not immediately terminate the session in the representative console.

queue

Stores all active support session queues. A queue is active if one or more of its members are logged in or if it is a persistent queue.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"code_name": "[string]"	The code name assigned to the support team. Personal queues do not have code names.
"created_timestamp": [timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"name": "[string]"	The support team's display name.
"support_team_id": "[string]"	The support team's unique identifier. The ID is not shown for personal queues.
"type": "[string]"	The type of queue. May be one of prewait , team , or personal .

representative

Stores all representatives logged into a representative console

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"private_display_name":[string]	The representative's private display name.
"public_display_name":[string]	The representative's public display name.
"queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue.
"routing_available":[boolean]	1: The representative is available to handle new support sessions. 0: The representative is unavailable to handle new support sessions.
"routing_busy":[boolean]	1: The representative has reached their maximum number of support sessions for session assignment. 0: The representative is not marked as busy.
"routing_enabled":[boolean]	1: The representative has automatic session assignment enabled. 0: The representative has automatic session assignment disabled.
"routing_idle":[boolean]	1: The representative is idle. 0: The representative is active.
"skill_code_names":[string]	A comma-separated list of skill code names assigned to this representative.
"type":[string]	The type of representative connection. May be one of normal or invited .
"user_id":[integer]	The representative's unique ID which identifies them in BeyondTrust.
"username":[string]	The username this representative uses to authenticate to BeyondTrust.

representative_queue

Stores logged-in representatives in relation to their support queues. While team leads and managers have access to their team members' personal queues, that access is not represented in this table.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue.
"representative_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.
"role":[string]	The user's role in the team. May be one of member , lead , or manager .

representative_support_session

Stores representatives participating in support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"representative_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.
"support_session_id":[integer]	The foreign key that links this table row to other table rows that reference this session.

support_session

Stores all active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"customer_company":[string]	The end user's company name.
"customer_company_code":[string]	The end user's company code.
"customer_description":[string]	The issue description as provided by the customer.
"customer_name":[string]	The end user's name.
"estimated_pickup_timestamp":[timestamp]	The time at which the system expects this session to be accepted or transferred. This could be a time in the past or future.
"issue_code_name":[string]	The code name of the issue selected by the customer on the front-end survey.
"issue_description":[string]	The description of the issue selected by the customer on the front-end survey.
"lsid":[string]	The logging session ID that uniquely identifies this session. This LSID can be used in /login > Reports , with the reporting API, in the integration client, etc.
"primary_team_queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue. This field stores the last team queue that owned the session before the session ended.
"priority":[integer]	The session's priority. May be one of 1 (high), 2 (medium), or 3 (low).
"public_site_id":[integer]	The ID of the public site associated with the session.
"public_site_name":[string]	The name of the public site associated with the session.
"queue_entry_timestamp":[timestamp]	The time at which this session entered its current queue.
"queue_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this queue. This field stores the queue in which the session currently resides.

"start_method":["string"]

The method with which the session was started. May be one of **session_key** (the seven-digit code or the URL), **rep_list**, **issue_submission**, **local_jump**, **remote_jump**, **shell_jump**, **local_rdp**, **remote_rdp**, **local_vnc**, **remote_vnc**, or **vpro**.

support_session_attribute

Stores custom session attributes assigned to active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"code_name":["string"]	The code name assigned to the support session attribute.
"created_timestamp":["timestamp"]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"support_session_id":["integer"]	The foreign key that links this table row to other table rows that reference this session.
"value":["string"]	The value of the attribute.

support_session_skill

Stores skills assigned to active support sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"code_name":["string"]	The code name assigned to the skill.
"created_timestamp":["timestamp"]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"support_session_id":["integer"]	The foreign key that links this table row to other table rows that reference this session.

presentation_session

Stores data about the presentation sessions.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":["timestamp"]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"lsid":["guid"]	The logging session ID that uniquely identifies this session. This LSID can be used on /login > Reports and in the Reporting API, among other places.
"presentation_name":["string"]	The name of the presentation.

attende

Stores all information about attendees who have joined a presentation.

"connected":[boolean]	1: The presentation client connected to the B Series Appliance and joined the presentation. 0: The presentation client did not connect to the B Series Appliance.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"hostname":[string]	The attendee's hostname.
"name":[string]	The name of the attendee.
"operating_system":[string]	The attendee's operating system.
"presentation_session_id":[integer]	The logging session ID that uniquely identifies this presentation.
"private_ip":[string]	The attendee computer's private IP address.
"public_ip":[string]	The attendee computer's public IP address.
(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.

presenter

Stores the information about the presenter who has started presenting their screen.

(id)	An integer that identifies this table row. This ID is unique for the lifetime of the model.
"created_timestamp":[timestamp]	A UNIX timestamp (UTC) representing the time at which this table row was created.
"presentation_session_id":[integer]	The logging session ID that uniquely identifies this presentation.
"representative_id":[integer]	The unique ID serving as the foreign key that links this table row with other table rows that reference this representative.



Note: For BeyondTrust versions 16.2 and later, 64-bit unsigned integers are used to generate the id value. The maximum allowed value is 18446744073709551615.

JavaScript Library for the Real-Time State API

This JavaScript library is provided to facilitate use of the socket-based real-time state API in a browser (see "[Protocol of the Real-Time State API](#)" on page 154). It simplifies interactions with the API in the following ways:

- Direct interaction with the WebSocket API is not necessary.
- The connection, authentication, and table registration steps are handled for you using the credentials, B Series Appliance information, and table names you provide.
- The server's JSON messages are buffered and decoded automatically.
- Model changes are provided as JavaScript objects instead of JSON strings.
- Several methods of subscribing to model changes are provided to help filter, transfer, and store the data more easily.

Browser Requirements

This API will only work in [browsers that support binary WebSockets](https://caniuse.com/#feat=websockets) (see <https://caniuse.com/#feat=websockets>).

Usage

To use this JavaScript library, your HTML page must reference the `state.js` file located on your BeyondTrust Appliance B Series.

A script tag would look something like the example below, where "support.example.com" is your B Series Appliance's hostname:

```
<script src="https://support.example.com/api/state.js"></script>
```


This script tag must be included in your HTML page before any of your own code that uses the API.



Note: Examples in this document use a simple script tag and the API's global `BomgarState()` function. However, `state.js` also contains a universal module definition (UMD) wrapper that allows it to be used with various JavaScript module systems such as Browserify or RequireJS.

Basic Use of the JavaScript Library

For basic functionality of the JavaScript library for the real-time state API, only the following two functions are necessary.


 **Note:** A complete list of table names and their data structures can be found in "[System State Model of the Real-Time API](#)" on page 158.

BomgarState(options)

Make one call to the `BomgarState()` function to provide the API with connection and authentication information and with the tables you wish to receive updates for.


Arguments

- `options (Object)`
 - `bearerToken (String)` - Required
 - `host (String)` - Required.
 - `port (Integer)` - Optional. Defaults to 443.
 - `company (String)` - Required.
 - `tables (String | String[])` - Optional. Can be an array of table names or the string **all**. Defaults to **all**.

 For more information on how to generate the `bearerToken`, please see [Authenticate to the Remote Support API](https://www.beyondtrust.com/docs/remote-support/how-to/integrations/api/authentication.htm) at <https://www.beyondtrust.com/docs/remote-support/how-to/integrations/api/authentication.htm>.

Returns

- `(Observable<Object>)`: An [observable](#) sequence of all model changes as objects, parsed from the original JSON (see <https://reactivex.io/documentation/observable.html>).

 **Note:** You must call `.subscribe()` on this returned observable object to actually initiate the connection to the server. Calling `BomgarState()` merely defines the necessary connection and authentication parameters.

Example

```
//Provide the API with the connection and authentication parameters
var allChangesObservable = BomgarState({
  host: 'companyname.support.example.com',
  port: 443, // Optional
  company: 'companyname',
  bearerToken: 'bearerToken',
  tables: ['customer_client', 'representative'] // Or 'all'
});

// This triggers the actual connection and authentication:
var subscription = allChangesObservable.subscribe();
```

.subscribe([onChange], [onError], [onCompleted])

Calling the `subscribe()` method on the observable returned by `BomgarState()` initiates the connection and authentication with your B Series Appliance over a WebSocket. If successful, the API then sends your table list to the server. `subscribe()` also allows you to optionally register callback functions for model changes, errors, and connection closures.

Arguments

- `onChange (Function)`: Function that the API invokes for every model change received from the server. The function should accept one argument: an object describing the model changes. The structure of this object is described in the **Model Updates** section of "[Protocol of the Real-Time State API](#)" on page 154.
- `onError (Function)`: Function that the API invokes 0 or 1 times if a fatal error occurs during connection, authentication, or table registration.
- `onCompleted (Function)`: Function that the API invokes once when the connection closes normally.

Returns

- `(Object)`: Returns an object representing the subscription. If you need to programmatically close the connection, call `.dispose()` on this object.

Example

```
// This causes the connection and authentication to occur.
var subscription = allChangesObservable.subscribe(
  // This onChange function is invoked every time any model change - insertion, update
  // deletion, or truncation - occurs in any table you have registered for
  function onChange(changeObject) {
    switch (changeObject.type) {
      case 'model_update':
        // Test for 'insert', 'update', or 'delete' properties on changeObject, then find
        // the table name and local data accordingly
        break;
      case 'truncate_model':
        // Delete all local data in all tables
        break;
    }
  },
  function onError(error) {
    console.error('An error occurred: %s', error);
  },
  function onCompleted() {
    console.log('Connection closed. No more messages will be received.');
```

Detailed Use of the JavaScript Library

The following methods are available on the object returned by `BomgarState()` (the object called `allChangesObservable` in the example in "[Basic Use of the JavaScript Library](#)" on page 165). These methods allow you to subscribe to more focused changes based on table names and change types. Some developers may find these more convenient to use than registering a single callback function for all tables and change types using `subscribe()` as seen with the basic usage.

`.changesForTable(tableName)`

Calling this function subscribes only to model changes for the given table name.

Arguments

- `tableName (String)`: The name of a table you requested from `BomgarState()` using the `tables` option.

Returns

`(Observable<ChangeObject>)`: An observable of changes to the given table. Change objects have `"name"` and `"data"` properties:

```
{
  "name": "<change type>", // "insert", "update", or "delete"
  "data": {
    "<id>": {
      // Varies depending on table and change type
    },
    // Other rows
  }
}
```

Example

```
// Get all the change messages for the 'representative' table in one subscription
allChangesObservable.changesForTable('representative')
  .subscribe(function onRepTableChange(repChangeObject) {
    // 'repChangeObject.name' contains the change name 'insert', 'update', or 'delete'
    // 'repChangeObject.data' contains the data related to the change from
    // changeObject[change][tableName] in the '.subscribe()' example
    switch (repChangeObject.name) {
      case 'insert':
        // Use 'repChangeObject.data' to perform an insert
        break;
      case 'update':
        // Use 'repChangeObject.data' to perform an update
        break;
      case 'delete':
        // Use 'repChangeObject.data' to perform a delete
        break;
    }
  });
```

.tableInserts(tableName)

Calling this function subscribes only to inserts for the given table name.

Arguments

- `tableName (String)`: The name of a table you requested from `BomgarState()` using the `tables` option.

Returns

`(Observable<InsertObject>)`: An observable of inserts into the given table. Insert objects contain one or more property names that are IDs for the table row and also contain property values that are objects containing the data to insert.

```
{
  "<id>": {
    // Varies depending on table
  },
  // Other inserts
}
```

Example

```
// Create a subscription for inserts into the 'customer_client' table
allChangesObservable.tableInserts('customer_client')
  .subscribe(function onCustomerClientInsert(insertObject) {
    // Insert rows
  });
```

.tableUpdates(tableName)

Calling this function subscribes only to updates for the given table name.

Arguments

- `tableName (String)`: The name of a table you requested from `BomgarState()` using the `tables` option.

Returns

`(Observable<UpdateObject>)`: An observable of updates for the given table. Update objects contain one or more property names that are IDs for the table row and also contain property values that are objects containing the data to update.

```
{
  "<id>": {
    // Varies depending on table
  },
  // Other updates
}
```

Example

```
// Create a subscription for updates to the 'customer_client' table
allChangesObservable.tableUpdates('customer_client')
```



```
.subscribe(function onCustomerClientUpdate(updateObject) {
    // Update rows
});
```

.tableDeletions(tableName)

Calling this function subscribes only to deletions for the given table name.

Arguments

- `tableName (String)`: The name of a table you requested from `BomgarState()` using the `tables` option.

Returns

`(Observable<Deletion[]>)`: An observable of deletions for the given table. Deletion objects are arrays of row IDs:

```
[
    "<id>",
    // Other IDs
]
```

Example

```
// Create a subscription for deletions from the 'customer_client' table
allChangesObservable.tableDeletions('customer_client')
    .subscribe(function onCustomerClientDeletion(deletionObject) {
        // Delete rows
    });
```

.truncations()

Calling this function subscribes to notifications that all tables should be truncated.

Arguments

- None

Returns

`(Observable<TruncationObject>)`: An observable of truncations. Truncation objects contain a property named `type` whose value is `truncate_model`.

Example

```
// Create a subscription for truncations
allChangesObservable.truncations()
    .subscribe(function() {
        // Delete all data from all tables
    });
```

Advanced Use of the JavaScript Library

In addition to the methods documented here, every observable returned by this API exposes many other operator functions documented by the Reactive Extensions / RxJS (v4) project. If you wish to explore these other operators, you may find the [resources on ReactiveX.io](https://reactivex.io/resources-on-ReactiveX.io) helpful (see <https://reactivex.io/documentation/operators.html>). If you encounter sections that are language-specific, only the RxJS content is applicable.

Working Demonstration of the JavaScript Library

The code examples below are the source of a working demo that shows the JSON data emitted by the three types of subscriptions.

index.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <title>BeyondTrust Real-Time State API Demo</title>
  <meta name="description" content="">
  <meta name="viewport" content="width=device-width">
  <link rel="stylesheet" href="styles.css">
</head>
<body>

<div id="container">
  <div class="row">
    <section id="cust">
      <h2>Customer Messages
        <span class="subtitle">.tableInserts('customer_client'), .tableUpdates('customer_
client'), .tableDeletions('customer_client')
      </h2>
      <div class="messages"></div>
    </section>
    <section id="rep">
      <h2>Representative Messages
        <span class="subtitle">.changesForTable('representative')</span>
      </h2>
      <div class="messages"></div>
    </section>
  </div>
  <div class="row">
    <section id="all">
      <h2>Raw Messages
        <span class="subtitle">.subscribe(onNext, onError, onCompleted)</span>
      </h2>
      <div class="messages"></div>
    </section>
  </div>
</div>

<script src="https://[YOUR B SERIES APPLIANCE HOSTNAME HERE]/api/state.js"></script>
<script type="text/javascript">
  var repSection = document.getElementById('rep');
  var custSection = document.getElementById('cust');
  var allSection = document.getElementById('all');

  var messages$ = bomgarState({
    host: '[YOUR B SERIES APPLIANCE HOSTNAME HERE]',
    port: 443,
```

```
company: '[YOUR COMPANY NAME HERE]',
username: '[YOUR USERNAME HERE]',
password: '[YOUR PASSWORD HERE]',
tables: 'all' // or an array like ['customer_client', 'representative']
});

var subscription = messages$.subscribe(
  function onNext(message) {
    appendMessage('Message:', message, allSection);
  },
  function onError(error) {
    appendMessage('An error occurred:', error, allSection, 'red');
    console.error('An error occurred: %s', error);
  },
  function onCompleted() {
    appendMessage('The End', 'Connection closed.', allSection, '#f50');
    console.warn('Connection closed. No more messages will be received.');
```

```
});

messages$.changesForTable('representative')
  .subscribe(function onRepChange(message) {
    appendMessage('Change: ', message, repSection);
  });
```

```
messages$.tableInserts('customer_client')
  .subscribe(function onCustInsert(message) {
    appendMessage('Insert: ', message, custSection, 'green');
```

```
});

messages$.tableUpdates('customer_client')
  .subscribe(function onCustUpdate(message) {
    appendMessage('Update: ', message, custSection, 'blue');
```

```
});

messages$.tableDeletions('customer_client')
  .subscribe(function onCustDeletion(message) {
    appendMessage('Deletion: ', message, custSection, 'red');
```

```
});

function appendMessage(label, message, section, color) {
  var messages = section.querySelector('.messages');
  var div = document.createElement('div');
  var hr = document.createElement('hr');
  var h4 = document.createElement('h4');
  h4.textContent = label;

  var p = document.createElement('p');
  p.style.color = color || 'black';
  p.textContent = JSON.stringify(message, null, '\t');

  div.appendChild(h4);
  div.appendChild(p);
  messages.appendChild(hr);
  messages.appendChild(div);
}
```

```
        setTimeout(function() {
            messages.scrollTop = messages.scrollHeight;
        }, 250);
    }
</script>
</body>
</html>
```

styles.css

```
* {
    box-sizing: border-box;
}

html, body {
    height: 100%;
    position: relative;
    min-height: 100%;
}

body {
    font-family: 'Open Sans', 'Helvetica Neue', sans-serif;
    display: flex;
    flex-direction: column;
    margin: 0;
    padding: 0;
    overflow: hidden;
}

#container {
    display: flex;
    flex: 1 1 100%;
    max-height: 100%;
    max-width: 100%;
    flex-direction: column;
}

.row {
    display: flex;
    flex-direction: row;
    flex: 1 1 1px;
    position: relative;
    min-height: 0;
    max-height: 100%;
    max-width: 100%;
    width: 100%;
    overflow: hidden;
}

section {
    box-shadow: inset 0 1px 1px rgba(0, 0, 0, .05);
    border: 1px solid #a0a0a0;
    background-color: #f5f5f5;
}
```

```
padding: 19px;
display: flex;
flex-direction: column;
flex: 1 1 0%;
min-width: 0;
max-width: 100%;
position: relative;
}

section h2 {
margin: 0 0 19px 0;
text-align: right;
}

section h2 .subtitle {
display: block;
font-size: 12px;
color: #828282;
}

section hr {
border-color: #E0E0E0;
border-style: solid;
}

section .messages {
overflow: auto;
}

section .messages hr:first-of-type {
display: none;
}

p {
tab-size: 2;
font-family: monospace;
white-space: pre-wrap;
}
```

Backup API

The backup API enables you to automatically back up your BeyondTrust software configuration on a recurring basis. The backup file includes all your configuration settings and logged data except for recordings and some large files from the file store. The backup includes only files from the file store less than 200 KB in size and no more than 50 files total. In the event of a hardware failure, having a backup file helps to speed the disaster recovery process.

The backup API is an authenticated API. For instructions on using authenticated APIs using OAuth, see "[Authenticate to the Remote Support API](#)" on page 7. The API account used to issue this command must have access to the backup API.

Commands are executed by sending a simple HTTP request to the B Series Appliance. The request can be sent using any HTTPS-capable socket library, scripting language module, or a URL fetcher such as **cURL** or **wget**. Either **GET** or **POST** may be used as the request method.

The backup API URL is <https://support.example.com/api/backup>.

Query Example

```
backup
```

```
https://support.example.com/api/backup
```

Test Scenario

To get started with this basic API integration, follow the steps below.

1. Log in to your BeyondTrust administrative interface and go to **Management > API Configuration**. Check the box to **Enable XML API**.
2. Create an API account and copy the client secret. This secret can be viewed only once and must be regenerated if lost.

```
OAuth Client ID: e52a9aa6fc0508ddf3a40601a736b230a1bebcd1
OAuth Client Secret: BU5u0fVEb1qEWuHdBK9AR6q9+O1CB26squ1susfJ0LsK
```

3. It is necessary to base64 encode these values ("Client ID:Client Secret") for use in the authorization header.

```
Base64 Encoded:
ZTUyYTlhYTZmYzAlMDhkZGYzYTQwNjAxYTczNmIyMzBhMWJlYmNkMTpCVTV1MGZWRWIxcUVXdUhkQks5QVI2cTkrTzFD
QjI2c3F1MXN1c2ZKMExzSw==
```

4. We will use cURL to illustrate generating a token using a BeyondTrust API account and using that token to make requests to the BeyondTrust web API.
 - a. First, we request a Bearer Token using the OAuth client ID and client secret.

```
curl -H "authorization: Basic
ZTUyYTlhYTZmYzAlMDhkZGYzYTQwNjAxYTczNmIyMzBhMWJlYmNkMTpCVTV1MGZWRWIxcUVXdUhkQks5QVI2cT
krTzFDQjI2c3F1MXN1c2ZKMExzSw==" --data "grant_type=client_credentials"
https://support.example.com/oauth2/token
```

- b. This results in a JSON response containing the bearer token.

```
{
  "access_token": "23MS6S2L42WCriESVzGbuwsiQwdbxuAJ3Zj4DxO",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

- c. We can now use that token to make a request to the API.

```
curl -H "authorization: Bearer 23MS6S2L42WCriESVzGbuwsiQwdbxuAJ3Zj4DxO"
https://support.example.com/api/command?action=get_api_info
```

- d. This results in an XML response for the requested API.



Note: If you receive any errors such as **Document Not Found**, check that the API account has the necessary permissions. Also, make sure that a user is logged into the site while you are testing.

API Change Log

API Version 1.24.1 for RS 24.1.x

- Configuration API:
 - Force the logging out of users - great if building API connectors leveraging anomaly detection.
 - List user group policy membership, making it easier to control access programmatically.

API Version 1.23.1 for RS 23.3.x

- Configuration API:
 - Added "Windows Local" and "Domain" accounts and attributes to the Vault Account configuration API (GET).
 - Added an *endpoint* filter to the Vault Account configuration API (GET).

API Version 1.22.2 for RS 22.1.x

- Reporting API:
 - Added **Syslog** query. This downloads a ZIP file containing all Syslog files available on the appliance. Syslog files include all changes made on the /login administrative interface within the last 30 days.

API Version 1.22.2 for RS 22.3.x

- Configuration API:
 - Added GET, PATCH, and DELETE APIs for the Protocol Tunnel Jump Item type.
 - Added GET and PATCH APIs to allow administrators to update the available groups for existing SAML Security Provider resources.

API Version 1.22.2 for RS 22.2.x

- Version update
- Configuration API:
 - Enhanced Group Policy Configuration APIs (GET, POST, and PATCH) to allow administrators to read and set access permission settings.

API Version 1.22.1 for RS 22.1.x

- Version update
- Expanded **EndpointLicenseUsage** to make it possible to download a ZIP file containing detailed information (English only) on BeyondTrust license usage.

API Version 1.21.1 for RS 21.3.x

- Version update
- Expanded the current `send_chat_message` Command API operation to send a chat message to Team Chats.
- Added an API that copies the existing Jump Client resource with the given `<id>`.
- Added command API: `set_rep_status` to set the status for representatives logged into the representative console.

API Version 1.19.2 for RS 19.2.x and 20.1.x

- Added ["Configuration API" on page 19](#)

API Version 1.19.0 for RS 19.1.x

- Version update

API Version 1.18.0 for RS 18.2.x

- Added `web_console` as a client type.

API Version 1.16.0 for RS 17.1.x

- Use OAuth 2.0 authentication for the real-time state API and endpoint credential manager connections.
- When importing a Jump Item several changes have been made:
 - Specify a name for Jump Items.
 - Import VNC Jump Items.
 - Specify a local address for Protocol Tunnel Jump Items.
 - For Web Jump Items, set if the certificate should be verified.
 - ["API Command: import_jump_shortcut" on page 53](#)

API Version 1.15.1 for RS 16.2.x

- Granularly define the accounts used for API access to the specific roles they serve. Additionally, OAuth 2.0 authentication is now used for authenticating API accounts.
 - ["Reporting API" on page 92](#)
 - ["Remote Support Command API" on page 21](#)
 - ["Backup API" on page 175](#)
- Specify the timezone offset of a generated session so that the customer client can be downloaded from the nearest traffic node of an Atlas cluster, resulting in faster session start times.
 - ["Session Generation API" on page 78](#)

API Version 1.15.0 for RS 16.1.x

- Import multiple shortcuts using the `import_jump_shortcuts` command.
 - ["API Command: import_jump_shortcut" on page 53](#)
- Use the Presentation Archive API to view presentation event data.
 - ["Download Reports with PresentationSession" on page 110](#)
 - ["Download Reports with PresentationSessionListing" on page 116](#)
- **PresentationRecording** has been deprecated in favor of **PresentationSessionRecording**. **PresentationRecording** is still available for backward compatibility.
 - ["Download Reports with PresentationSessionRecording" on page 118](#)
- A JavaScript library with the ability to interact with the BeyondTrust real-time state API has been implemented.
 - ["JavaScript Library for the Real-Time State API" on page 164](#)

API Version 1.13.1 for RS 15.2.x

- Use the Archive API to view session event data.
 - ["Download Reports with Archive" on page 131](#)
- View the Company API Name in the XML output of the `get_api_info` Command API operation.
 - ["API Command: get_api_info" on page 61](#)

API Version 1.13.0 for RS 15.1.x

- Use BeyondTrust Representative Console Scripts to start sessions through Jumpoints, start RDP sessions, or start Shell Jump sessions.
 - ["API Script Command: push_and_start_remote" on page 69](#)
 - ["API Script Command: start_rdp_session" on page 72](#)
 - ["API Script Command: start_shell_jump_session" on page 75](#)
- Add custom session attributes to sessions started with a BeyondTrust Representative Console Script.
 - ["API Script Command: generate_session_key" on page 67](#)
 - ["API Script Command: push_and_start_local" on page 68](#)
 - ["API Script Command: start_vpro_session" on page 77](#)
- View real-time data for support center activity in your organization.
 - ["Real-Time State API" on page 153](#)
- View archives of the system state of your BeyondTrust Appliance B Series to analyze support center activity in your organization.
 - ["Download Reports with ArchiveListing" on page 129](#)
 - ["Download Reports with Archive" on page 131](#)

API Version 1.12.0 for RS 14.2.x and 14.3.x

- The file extension `.ns` has been deprecated from the API calls. It is still available for backward compatibility.
 - <https://support.example.com/api/command.ns> is now <https://support.example.com/api/command>
 - https://support.example.com/api/client_script.ns is now https://support.example.com/api/client_script
 - https://support.example.com/api/start_session.ns is now https://support.example.com/api/start_session
 - <https://support.example.com/api/reporting.ns> is now <https://support.example.com/api/reporting>
 - <https://support.example.com/api/backup.ns> is now <https://support.example.com/api/backup>
- Use two new API commands to help automate failover.
 - ["API Command: check_health" on page 49](#)
 - ["API Command: set_failover_role" on page 51](#)
- Set custom session attributes when starting or during a session. Customer details, skills, and custom fields can now be added to sessions regardless of session start type (session key, rep selection, issue submission). Additionally, certain fields have been deprecated in favor of a more consistent format. All fields are still available for backward compatibility. Deprecated fields are noted in their respective sections.
 - ["API Command: generate_session_key" on page 28](#)
 - ["API Command: set_session_attributes" on page 38](#)
 - ["Session Generation API" on page 78](#)
 - ["Start Sessions with Session Key Acceptance" on page 82](#)
 - ["Use JavaScript to Start Click-to-Chat or Full Client Sessions" on page 83](#)
 - ["Start Sessions with External Keys \(TicketID\)" on page 90](#)
- Retrieve custom session attributes during a session.
 - ["API Command: get_session_attributes" on page 39](#)

API Version 1.11.0 for RS 14.1.x

Use three new commands to view information about your BeyondTrust Appliance B Series and connected software clients.

- ["API Command: get_appliances" on page 41](#)
- ["API Command: get_connected_client_list" on page 42](#)
- ["API Command: get_connected_clients" on page 44](#)

Download reports of license usage data.

- ["Download Reports with LicenseUsage" on page 149](#)

Command and reporting APIs return XML that declare a namespace.

- Reporting API: <https://www.beyondtrust.com/namespaces/API/reporting>
- Command API: <https://www.beyondtrust.com/namespaces/API/command>



Note: The above namespaces are returned XML data and are not functional URLs.

Specify the language to use for the customer client.

- ["Session Generation API" on page 78](#)
- ["Use JavaScript to Start Click-to-Chat or Full Client Sessions" on page 83](#)

API Version Reference

The following table shows the relationship between the API and BeyondTrust versions.

API Version	BeyondTrust Version
1.24.1	24.x
1.23.1	23.x
1.22.2	22.2.x, 22.3.x
1.22.1	22.1.x
1.21.1	21.3.x
1.21.0	21.1.x, 21.2.x
1.19.2	19.2.x, 20.1.x
1.19.0	19.1.x
1.18.0	18.2.x
1.16.0	17.1.x
1.15.1	16.2.x
1.15.0	16.1.x
1.13.1	15.2.x
1.13.0	15.1.x
1.12.0	14.2.x, 14.3.x
1.11.0	14.1.x
1.10.0	13.1.x
1.9.0	12.3.x
1.8.0	12.2.x
1.7.1	12.1.4+
1.7.0	12.1.x
1.6.0	11.1.x
1.5.0	10.6.x
1.4.2	10.5.2+
1.4.1	10.5.1
1.4.0	10.5.0
1.3.2	10.4.1+
1.3.0	10.4.0
1.2.2	10.3.4+
1.2.1	10.3.2, 10.3.3
1.2.0	10.3.0, 10.3.1
1.1.1	10.2.5+

API Version	BeyondTrust Version
1.1.0	10.2.(0-4)
1.0.0	10.1.x

Disclaimers, Licensing Restrictions, and Tech Support

Disclaimers

This document is provided for information purposes only. BeyondTrust Corporation may change the contents hereof without notice. This document is not warranted to be error-free, nor subject to any other warranties or conditions, whether expressed orally or implied in law, including implied warranties and conditions of merchantability or fitness for a particular purpose. BeyondTrust Corporation specifically disclaims any liability with respect to this document and no contractual obligations are formed either directly or indirectly by this document. The technologies, functionality, services, and processes described herein are subject to change without notice.

All Rights Reserved. Other trademarks identified on this page are owned by their respective owners. BeyondTrust is not a chartered bank or trust company, or depository institution. It is not authorized to accept deposits or trust accounts and is not licensed or regulated by any state or federal banking authority.

Licensing Restrictions

One BeyondTrust Remote Support license enables one support representative at a time to troubleshoot an unlimited number of remote computers, whether attended or unattended. Although multiple accounts may exist on the same license, two or more licenses (one per concurrent support representative) are required to enable multiple support representatives to troubleshoot simultaneously.

Tech Support

At BeyondTrust, we are committed to offering the highest quality service by ensuring that our customers have everything they need to operate with maximum productivity. Should you need any assistance, please log into the [Customer Portal](#) at <https://beyondtrustcorp.service-now.com/csm> to chat with Support.

Technical support is provided with annual purchase of our maintenance plan.