Connecting to Parallels RAS using the Connection Broker API

March 02, 2026

Parallels International GmbH
Vordergasse 59
8200 Schaffhausen
Switzerland
Tel: + 41 52 672 20 30
www.parallels.com

Table of Contents

Summary **4**

API Structure **5**

Request Headers 5

Request Body 5

Request Parameters 8

Priming the session 11

Response Body **12**

Status Codes 12

Successful API Body 12

Successful Response Parameters (200) 13

Unauthorized (401) 13

Bad Requests (400) 13

Error Codes **17**

Policy **20**

Summary

Starting with Parallels RAS version 21.1 Parallels has made it possible to manage connections to Parallels RAS without using its own Connection Broker. This is made possible via a specific API which runs on the Parallels Connection Broker.

Parallels RAS still delivers the sessions to the users, but the API allows session management to be separated from the Parallels Connection Broker.

Enabling the Connection Broker API

Use of the API needs to be enabled within the Parallels RAS Management console. For more information on the configuration steps needed, please refer to the Admin Guide under the following sections:

- Assigning a certificate
- Enabling the Connection Broker API

Using the API

The API Allows two “phases” of operation.

Priming

The first phase is called “Priming” this is an optional step an organisation may take. Here, a session can be created and an application initiated prior to the user connecting. The session is ready for a user to connect to later which can accelerate connection time for the user. This is a similar approach to “Session Pre-launch." Priming is done centrally and requires no end-user intervention at all.

As part of the Priming operation, a **Named Event** is used. This event is created by the RAS Shell prior to an application starting in the Primed Session. The RAS shell will monitor the state of this event and will ask the client to disconnect when the event is set, which will need to be done by the user of the API.

The Named event is “TunnelSession-{A893A070-C05E-415F-82CF-6DB853BC8F3F}”.

Tunnelling

The second step is required and utilises the Parallels Secure Gateway for a secure connection through the Parallels Clients. Organisations can determine how the session is managed via the API, to connect to an existing ‘Primed” session or launch a new session, programmatically.

The structure of the API is shown below.

API Structure

To fulfil the requirements the following API is provided:

Endpoint: Prime or tunnel session
URL: `PUT /api/directsessiontunnel`
Description: Optionally prime a session using the specifications provided and return links to use to open a tunnel to this session.

Request Headers

Content-Type: application/json
Authorization: <apitoken>

Request Body

```
{

    "connectionname": "myconnection",

    "ras_host": "rashost",

    "ras_port": 443,

    "endpoint_host": "endpoint host",

    "endpoint_port": 3389,

    "ras_portal_url": "ras portal url",

    "endsession_url": "https://mycompany.com",

    "parallelsclient_use_endsession_url": false,

    "refreshpage_url": "https://mycompany.com",

    "username": "username",

    "password": "password",

    "app_mode": true,

    "ras_urlschemaredir": true,
```

```
    "prime_session": 1,
    "prime_secret": "base64 string", //this string needs to be added to the
     client registry
    "timeout": 50,
    "app_name": "Application Name",
    "app_path": "AppPath.exe",
    "app_dir": "c:\\",
    "app_params": "arguments arguments",
    "app_wintype": 0,
    "ttl": 60,
    "reconnectionttl": 60,
    "detectproxy": true,
    "detectproxyurl": ""
    "policy": {
        "ConnProps": {
            "AllowClpbrdFmtClientServer": 0,
            "AllowClpbrdFmtServerClient": 0,
            "ClipboardDirection": 0,
            "RedirectDrives": 0,
        }
    }
}
```

Request Parameters

Field	Type	Required	Validation	Description
connectionname	string	Yes	Min Len: 1 Max Len: 260	This name is used by the client to persist any configuration to be used in future connections
ras_host	string	Yes	Min Len: 1 Max Len: 260	Server which Parallels client connects to
ras_port	int	No	Min: 1 Max: 65535	Port to reach RAS server. Default is 443
endpoint_host	string	Yes	Min Len: 1 Max Len: 260	Host where the RDP session is to run
endpoint_port	int	No	Min: 1 Max: 65535	Port for RDP session. Default is 3389
ras_portal_url	string	Yes	Min Len: 1 Max Len: 2000	URL to reach User portal
endsession_url	string	No	Min Len: 0 Max Len: 4096	When the session closes the browser will redirect to this page
parallelsclient_use_endsession_url	bool	No	true/false	(false) Parallels client will not make use of the “endsession_url” argument. (true) Parallels client will make use of the “endsession_url” argument. Default is false.
refreshpage_url	string	No	Min Len: 0 Max Len: 4096	If the user confirms refreshing the page the browser will redirect to this page
username	string	Yes	Min Len: 1 Max Len: 260	Username used to establish the RDP session
password	string	Yes	Min Len: 1 Max Len: 2000	Password used to establish the RDP session

app_mode	bool	No	true/false	(false) Desktop mode (true) Published app mode Default is Yes.
app_name	string	No	Min Len: 0 Max Len: 260	If set this will be displayed by the client while connecting.
app_path	string	Yes, if appmode is 1	Min Len: 1 Max Len: 260	The target application to be started after initial session handshake
app_dir	string	No	Min Len: 0 Max Len: 260	Folder to start target application
app_params	string	No	Min Len: 0 Max Len: 1024	Optional arguments to the target application
app_wintype	int	No	0-2	Application is shown as (0) Normal (1) Maximized (2) Minimized when launched. Note application needs to process such argument. Default is (0) Normal
ras_urlschemaredir	bool	No	true/false	(false) Returns tuxclient protocol schema (true) Returns HTTPS URL to redirect for the tuxclient protocol schema. Default is false.
prime_session	int	No	0-4	(0) No priming is done + return tunnel tokens (1) RAS backend will prime a session + return tunnel tokens (2) Returns command line to prime session. + return tunnel tokens (3) Returns command line to prime session + tunnel token is not returned.

				(4) RAS backend will prime a session + tunnel token is not returned. When returning the command to prime the session, it does not include the installation path of the Parallels client and need to be prepended. Please refer to the section **Priming the session** for more information. Default (1) RAS backend
prime_secret	string	No	Min Len: 0 Max Len: 260	Base64 of a secret key to be used for priming. The key needs to be 16, 24 or 32 bytes in size. The same value is to be written where the priming is going to happen. Please see section “Changing the priming secret”
timeout	int	No	Min: 0 Max: 1800	Number of seconds to wait for the session to be primed. If set to 0 or 2, the API will not wait for the session to be primed to return. Default is 120 (2 minutes)
ttl	int	No	Min: 5 Max: 86400	Time that the token can be used in seconds, Default is 60, Minimum is 5 and maximum is **86400**, which is one day.
reconnectionttl	int	No	Min: 5 Max: 86400	Time the client requires to refresh the reconnection token in seconds, Default is **3600**, Minimum is **30** and maximum is **86400**, which is one day.

detectproxy	bool	No	true/false	When set the client will try to auto detect an http 1.1 proxy.
detectproxyurl	string	No	Min: 0 Max: 4096	Used to detect proxy. If not set "http://www.parallels.com" will be used
policy	string	No	n/a	Policy settings to apply to the session when the client establishes the RDP session. Please check the policy section in this document.

Priming the session

Priming a session is an optional step where an organisation may wish to “pre-launch” sessions which are then disconnected and will allow a user to connect to them later thought the Parallels Clients.

The API setting to prime a session prime_session offers 5 possible values

Mode	Returns Tunnelling URLs	Description
0	Yes	No priming.
1	Yes	RAS Backend will prime the session. If timeout is set to non zero, the API will return when the priming is ready or an error has occurred. If timeout is set to zero the priming is done asynchronously.
2	Yes	API will provide command line argument to prime the session. The priming can be triggered by the caller. When starting the session, you need to wait for the process to exit and read its exit code. Exit codes are documented as part of the **Error Codes** section. It is recommended to have a timeout for this to happen. i.e. if the process does not close within a stipulated time, the session is to be declared not valid.
3	No	API will provide command line arguments to prime the session. The priming can be triggered by the caller. When starting the session, you need to wait for the process to exit and read its exit

		code. Exit codes are documented as part of the **Error Codes** section. It is recommended to have a timeout for this to happen. i.e. if the process does not close within a stipulated time, the session is to be declared not valid.
4	No	RAS Backend will prime the session. If timeout is set to non zero, the API will return when the priming is ready or an error has occurred. If timeout is set to zero the priming is done asynchronously.

Response Body

The response will be in json format with Headers to include Content-Type: application/json

Status Codes

Code	Meaning
200	OK
400	Bad request
405	Method not allowed (Only PUT is supported)
401	Unauthorized

Successful API Body

```
{
    "primecmd": \"TSClient.exe urlSchema "tuxclient:///?Command=TunnelSessionPrelaunch&Token=DATA\"",
    "parallelsclient": "https://localhost/URLSchemaRedirector?Command=TunnelSession&Token=DATA",
    “userportal”: “https://userportal.com/tunnelsession&token=DATA”,
    "sessionprimetime": 12890
}
```

Successful Response Parameters (200)

Field	Type	Description
primecmd	string	command line argument to prime the session (only returned if prime_session is set to 2)
parallelsclient	string	link to start the tunnelled RDP session using Parallels client
userportal	string	link to start the tunnelled RDP session using Parallels User portal (web client using HTML5)
sessionprimetime	int	number of milliseconds taken to prepare the session. (only returned if prime_session is set to 1)

Unauthorized (401)

An unauthorized request, i.e. one with an invalid api token, will receive the following json reply

```
{

    "error": "unauthorized",

    "message": "Invalid or missing API token."

}
```

Bad Requests (400)

Possible bad requests while validating parameters

Error	Message	Parameter	Code	Reason
BadRequest	Unable to parse request payload. Please check the format and try again.	n/a	3	Failed to parse request data.
InvalidRequest	This endpoint requires a valid JSON payload in the request body.	n/a	4	Request does not contain json content or

				json content was invalid.
MissingArgument	Can contain multiple errors	yes	100	A required parameter was not specified
ValueOutOfRange	The value for '%s' must be between %min and %max.	yes	101	A value of a parameter is out of range
StringTooShort	The '%s' field must be at least %min characters long.	yes	102	A string value is too short
StringTooLong	The '%s' field must not exceed %max characters.	yes	103	A string value is too long
InvalidArgument	The 'apikey' field is not valid.	n/a	104	Value of the apikey is not valid.

Examples of possible errors are as follows:

InvalidRequest


```
{
    "error": "InvalidRequest",
    "message": "This endpoint requires a valid JSON payload in the request body."
    "code": 4
}
```


InvalidArgument


```
{
    "error": "InvalidArgument",
    "message": "The 'apikey' field is not valid."
    "code": 104
```

```
}
```

MissingArgument

```
{
    "error": "MissingArgument",
    "message": "The 'ras_host' field is required but was not provided.",
    "parameter": "ras_host",
    "code": 100
}
```

ValueOutOfRange

```
{
    "error": "ValueOutOfRange",
    "message": "The value for 'prime_session' must be between 0 and 2.",
    "parameter": "prime_session",
    "code": 101
}
```

Possible bad requests while priming the session

Error	Message	Code	Reason
InvalidRequest	Parallels client failed to start	200	Process tsclient.exe failed to start
InvalidRequest	Session timed out	201	API timed out
InvalidRequest	Session terminated abnormally	202	Process closed with unexpected error code
InvalidRequest	Endpoint server not available	203	Generic error if something unexpected fails.

InvalidRequest	Session failed to create event	3001	Failed to create named event
InvalidRequest	Session failed to start application	3002	Application specified could not be started
InvalidRequest	Parallels Client failed to start remote application	3003	Parallels client did not send request to start remote application (not implemented yet).
InvalidRequest	Parallels Client received invalid data to start remote application	3004	Parallels client sent Invalid data to start the remote application.
InvalidRequest	Session priming failed: Token signature invalid	3005	The priming token is not valid.
InvalidRequest	Session priming failed: Token expired	3006	The priming token has expired.
InvalidRequest	Session priming failed: Invalid Priming Secret	3007	The priming secret is invalid. Check registry settings.
InvalidRequest	Parallels Client failed to establish a session with the end point, error code: %u	>3100	Session was not established. The error codes are shown in the **Error Code** section

Examples of possible errors are as follows:

Session timed out before priming was complete

```
{
    "error": "InvalidRequest",
    "message": "Session priming timed out",
    "code": 201
}
```

Application failed to start (not available or user does not have permission to start)

```
{
    "error": "InvalidRequest",
```

```
    "message": "Session priming failed to start application",
    "code": 3002
}
```


Endpoint server cannot be reached


```
{
    "error": "InvalidRequest",
    "message": "Parallels Client failed to establish a session with the end point, error code: 3360",
    "code": 3360
}
```


Bad credentials


```
{
    "error": "InvalidRequest",
    "message": "Parallels Client failed to establish a session with the end point, error code: 5155",
    "code": 5155
}
```


A list of all possible error codes will be provided.

Error Codes

This section includes all documented error codes.

API code	process exit code	RDP error code	RDP Category	Description
203	< 3000	n/a	n/a	Process closed with unexpected error code

n/a	3000	n/a	n/a	Priming was completed successfully
3001	3001	n/a	Priming	Failed to create named event
3002	3002	n/a	Priming	Application specified could not be started
3003	3003	n/a	Priming	Parallels client did not send request to start remote application (not implemented yet).
3004	3004	n/a	Priming	
3005	3005	n/a	Priming	Fatal error while validating token signature
3006	3006	n/a	Priming	Token expired
>3100	> 3100	n/a	n/a	Session failed to establish.
3360	3360	260	Connection	Invalid host name
3362	3362	262	Internal	Out of memory
3364	3364	264	Connection	Connection timeout
3616	3616	516	Connection	Host too busy
3620	3620	520	Connection	Invalid host name
3618	3618	518	Internal	Out of memory
3872	3872	772	NetworkError	Network error
3874	3874	774	Internal	Out of memory
3876	3876	776	Connection	Invalid host name
4128	4128	1028	NetworkError	Network error
4130	4130	1030	NetworkError	Network error (invalid data)
4131	4131	1031	NetworkError	Network error (invalid data)
4132	4132	1032	Internal	Internal error

4386	4386	1286	NetworkError	Network error (invalid encryption method)
4388	4388	1288	Connection	Invalid host name
4640	4640	1540	Connection	Invalid host name
4642	4642	1542	Connection	Invalid host name (neg flags mismatch)
4644	4644	1544	Internal	Internal timer error
4645	4645	1545	NetworkError	Network error ()
4896	4896	1796	Connection	Connection timeout
4898	4898	1798	Connection	Invalid host name (Certificate unpack error)
5152	5152	2052	Connection	Invalid host name
5155	5155	2055	Credentials	Bad Credentials
5156	5156	2056	RDPLicense	No RD license servers available
5408	5408	2308	NetworkError	Network error
5410	5410	2310	RDPLicense	No RD license servers available
5412	5412	2312	RDPLicense	RD licensing timeout
5666	5666	2566	RDPLicense	No RD license servers available
5667	5667	2567	Credentials	Invalid credentials
5922	5922	2822	NetworkError	Network error (invalid data)
5923	5923	2823	Account	Account is disabled
6178	6178	3078	NetworkError	Network error (invalid data)
6179	6179	3079	Account	The account is restricted.
6180	6180	3080	NetworkError	Network error (invalid data)
64435	6435	3335	Account	Account is locked out

6691	6691	3591	Account	The account is expired.
6947	6947	3847	Account	Password expired
7715	7715	4615	Account	Password expired
8739	8739	5639	Account	The policy does not support delegation of credentials to the target server.
8995	8995	5895	Account	Delegation of credentials to the target server is not allowed unless mutual authentication has been achieved.
9251	9251	6151	Account	No authority could be contacted for authentication.
10019	10019	6919	Connection	The remote PC certificate is expired or invalid
10275	10275	7175	Connection	An incorrect PIN was presented to the smart card.
11555	11555	8455	Connection	The server authentication policy does not allow connection requests using saved credentials.
11811	11811	8711	Connection	The smart card is blocked.

Policy

To force any RAS policy settings when using the api, you need to add the required settings in the JSON body of the request. Please note that some policy settings are not to be used since they will not have any effect on the session (e.g. Client options → Appearance, Redirection category). The policy part is as follows:

```
"policy": {
    "ConnProps": {
        "AllowClpbrdFmtClientServer": 0,
        "AllowClpbrdFmtServerClient": 0,
        "ClipboardDirection": 0,
```

```
            "ComputerName": "mycomp",
            "DriveCacheMode": 2,
            "Drives": "(*:)",
            "DynDrives": 1,
            "LimitClipboardToTextOnly": 0,
            "RedirectClipboard": 0,
            "RedirectDrives": 1,
            "RedirectReadonlyDrives": 0
        }
    }
```

In order to simplify the creation of such data, please use the following flow:

1. Use RAS Management Console to create your policies
2. Select the policy you want to use
3. Right click to open the context menu or click on the Tasks button on the top right of the list.
4. Select “Copy policy (json)”
5. The policy data will be added to your clipboard
   e.g.

```
{ <== do not copy
"policy": {
        "ConnProps": {
            "AllowClpbrdFmtClientServer": 0,
            "AllowClpbrdFmtServerClient": 0,
            "ClipboardDirection": 0,
            "ComputerName": "mycomp",
            "DriveCacheMode": 2,
            "Drives": "(*:)",
            "DynDrives": 1,
            "LimitClipboardToTextOnly": 0,
            "RedirectClipboard": 0,
            "RedirectDrives": 0,
            "RedirectReadonlyDrives": 0
        }
} <== do not copy
```

6. To use this data, remove the enclosing curly brackets and add to the API request json body

Note: For development and testing purposes, API exploration, playgrounds (e.g., postman…) can be used to validate the requests and responses provided by this REST API.