

JSON-RPC API guide

Version 1.1.

The latest JSON-RPC API guide is available here: <https://docs.safedns.com/books/json-rpc-api/page/json-rpc-api-guide>

JSON-RPC API methods are designed for interaction with the service and making changes to the settings.

The API was designed using JSON-RPC protocol and is compatible with the following versions: 1.0 and 2.0. More info is available here: <http://www.jsonrpc.org/specification>

An example of a CURL request for JSON-RPC 1.0:

```
curl -X POST https://www.safedns.com/api/json/v1-L-d
'{"id": 0, "method": "methodName", "params": {"param1": "param1"}}'
```

An example of a CURL request for JSON-RPC 2.0:

```
curl -X POST https://www.safedns.com/api/json/v2-L-d
'{"id": 0, "jsonrpc": "2.0", "method": "methodName", "params": {"param1": "param1"}}'
```

Some API methods require basic authentication. In this case, authentication details must be sent in the request. Example:

```
curl -X POST https://www.safedns.com/api/json/v2 -L --user username:password -d
'{"id": 0, "jsonrpc": "2.0", "method": "methodName", "params": {"param1": "param1"}}'
```

Response format: JSON.

A response example of a successful API call:

```
{
  "jsonrpc": "2.0",
  "id": 0,
  "result": "result"
}
```

A method result is included in the “result” key.

In case of an incorrect request the response will be the following:

```
{
  "jsonrpc": "2.0",
  "id": 0,
  "error": {"code": "code", "name": "name", "message": "message", "data": "data"}
}
```

Table of available methods:

#	Value	Description
1	systemInfo	Gets information about the system configuration
2	myip	Gets the current client's IP
3	testAuth	User's authentication
4	register	Registers a new user
5	userInfo	Gets user information
6	profiles	Gets a list of available filtering policies
7	addProfile	Creates a new filtering policy
8	removeProfile	Deletes a filtering policy
9	renameProfile	Renames a filtering policy
10	categories	Gets a list of available filtering categories
11	userFilter	Gets a list of filtering categories applied for a user
12	setFilterCat	Changes filtering settings
13	domains	Gets a list of Allow/Denylisted domains for a certain filtering policy
14	addDomain	Adds a domain to the Allow/Denylist for a certain filtering policy

15	removeDomain	Deletes a domain from a certain filtering policy
16	clearDomains	Clears the list of domains for a certain filtering policy
17	updateNic	Updates a dynamic IP
18	setWhiteListOnly	Enables/disables the "Allow list only" mode
19	setSafeSearchEnabled	Enables/disables "Safe Search mode" for Google and Bing
20	setSafeYoutubeEnabled	Enables/disables "Restricted YouTube mode"
21	multiSchedule	Gets information on the filtering schedule for a certain policy
22	setSchedule	Sets filtering schedule for a certain policy
23	setScheduleEnabled	Enables/disables filtering schedule for a certain policy
24	profileScheduleActivity	Gets the time left before a change of the filtering policy according to the filtering schedule
25	getProfile	Creates AgentInfo, gets AgentInfo status or updates AgentInfo policy depending on the parameters
26	setProfile	Sets a filtering policy for AgentInfo
27	feedback	Sends user's feedback
28	getCategoriesDailyStats	Gets daily categories stats of a policy
29	getAdvertising	Gets the information on promo
30	getPlans	Gets the list of available billing plans
31	getPlan	Gets the user's billing plan
32	getAPCVersion	Gets mobile app's version: minimal supported and the current one

Available methods:

1. systemInfo

Gets information on the system settings.

Response:

```
{
  "public_dns": <public_dns>,
  "nxdomain": <nxdomain>,
  "blockpage": <blockpage>,
  "blockpage_token": <blockpage_token>,
  "blockapi": <blockapi>
}
```

2. myip

Gets current client's IP.

Response:

```
<ip address>
```

3. testAuth

User's authentication.

Mandatory parameters:

- username - client's username.
- password - client's password.

Response:

```
<true| false>
```

4. register

Registers a new user.

Mandatory parameters:

- username - client's username.
- password - client's password.

Response:

In case of a successful registration:

```
[ true]
```

If an error occurred:

```
[ false, <error>]
```

5. userInfo

Gets user information. Authorization is required.

Response:

```
{
  "plan": {
    "name": <plan.name>,
    "code": <plan.code>,
    "features": <features>
  },
  "tz": <timezone_offset / 3600>,
  "tz_minutes": <timezone_offset / 60>
}
```

6. profiles

Gets a list of available filtering policies. Authorization is required.

Response:

```
[
  {
    "name": <name>,
    "default": <true| false>,
    "white_list_only": <true| false>,
    "token": <token>,
    "safe_search_enabled": <true| false>,
    "is_schedule_enabled": <true| false>,
    "id": <id>
  }
]
```

7. addProfile

Creates a new filtering policy. Authorization is required.

Mandatory parameter:

- name - policy name

Response:

```
{
  "name": <name>,
  "default": <true| false>,
  "white_list_only": <true| false>,
  "token": <token>,
  "safe_search_enabled": <true| false>,
  "is_schedule_enabled": <true| false>,
  "id": <id>
}
```

8. removeProfile

Deletes a filtering policy. Authorization is required.

Mandatory parameter:

- profile_id - id of the policy that should be deleted.

Response:

```
null
```

9. renameProfile

Renames a filtering policy. Authorization is required.

Mandatory parameters:

- profile_id - policy id.
- name - policy name.

Response:

<name>

10. categories

Gets a list of available filtering categories. Authorization is required.

Response:

```
[
  {
    "title": <group1>,
    "items": [{"title": <item1>, "id": <id>}]
  }
]
```

11. userFilter

Gets a list of filtering categories applied for a user. Authorization is required.

Mandatory parameter:

- profile_id - policy id.

Response:

```
[ <cat_id1>, <cat_id2>, <cat_id3>]
```

12. setFilterCat

Changes filtering settings. Authorization is required.

Mandatory parameters:

- profile_id - policy id.
- cat - category id.
- state - boolean value (add/remove the category from the filtering settings).

Response:

```
null
```

13. domains

Gets a list of Allow/Denylisted domains for a certain filtering policy. Authorization is required.

Mandatory parameters:

- profile_id - policy id.
- choice - domain type (available options: “black”, “white”, “alias”).

Response:

If the domain type is “black” or “white”

```
[{"domain": <domain>, "id": <id>}]
```

If the domain type is “alias”:

```
[{"domain": <domain>, "id": <id>, "ip": <ip>}]
```

14. addDomain

Adds a domain to the Allow/Denylist for a certain filtering policy. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- choice - domain type (available options: “black”, “white”, “alias”).
- domain – domain.

Optional parameter:

- ip - IP address, the parameter should be used if the “alias” domain is being added.
- comment - adds a comment for a domain.

Response:

```
<id of added domain>
```

15. removeDomain

Deletes a domain from the Allow/Denylist for a certain filtering policy. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.

- choice - domain type (available options: "black", "white", "alias").
- domain - domain.

Response:

```
null
```

16. clearDomains

Clears the list of Allow/Denylisted domains for a certain filtering policy. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- choice - domain type (available options: "black", "white", "alias").

Response:

```
null
```

17. updateNic

Updates a dynamic IP. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- hostname - hostname.

Response:

```
<IP address>
```

18. setWhiteListOnly

Enables/disables the "Allow list only" mode. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- flag - enable/disable option (boolean value).

Response:

```
null
```

19. setSafeSearchEnabled

Enables/disables "Safe Search mode" for Google and Bing. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- flag - boolean value (enable/disable option).

Response:

```
null
```

20. setSafeYoutubeEnabled

Enables/disables "Safe Search mode" for Google and Bing. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- flag - boolean value (enable/disable option).

Response:

```
null
```

21. setBlockUnknownEnabled

Enables/disables "Restricted YouTube mode". Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- flag - boolean value (enable/disable option).

Response:

```
null
```

22. multiSchedule

Gets information on the filtering schedule for a certain policy. Authorization is required.

Mandatory parameter:

- profile_id - policy ID.

Response:

```
[[<offset1>, <true|false>], [<offset2>, <true|false>]]
```

23. setSchedule

Sets filtering schedule for a certain policy. Authorization is required.

Mandatory parameters:

- profile_id - policy ID.
- segments - a list of segments for scheduling: [[<offset1>, <true|false>], [<offset2>, <true|false>]].

Response:

```
[ true]
```

24. setScheduleEnabled

Enables/disables filtering schedule for a certain policy. Authorization is required

Mandatory parameters:

- profile_id - policy ID.
- flag - boolean value (enable/disable option).

Response:

```
<true|false>
```

25. profileScheduleActivity

Gets the time left before a change of the filtering policy according to the filtering schedule. Authorization is required

Mandatory parameter:

- profile_id - policy ID

Response:

```
[ <true| false>, <mins_until_change>]
```

Note:

If the scheduling is turned off or there are no other states: mins_until_change = -1.

26. getProfile

This method includes 3 functions, depending on the parameters. Authorization is required.:

1. creates AgentInfo;
2. gets AgentInfo status;
3. updates AgentInfo policy

Mandatory parameters for creating Agent info and getting a policy:

- uid - ""
- hostname - hostname.
- version - version.
- os_info - information on the OS.
- address - IP address.

Mandatory parameters for getting the current policy of AgentInfo:

- uid - uid AgentInfo.
- hostname - "".
- version - "".
- os_info - "".
- address - "".

Mandatory parameters for updating AgentInfo:

- uid - uid AgentInfo.
- hostname - new hostname.
- version - new version.
- os_info - new information on the OS.
- address - new IP address.

Response:

```
[ <uid>, <token>, <profile_id>]
```

27. setProfile

Sets a filtering policy for AgentInfo. Authorization is required

Mandatory parameters:

- uid - uid AgentInfo.
- profile_id - policy ID.

Response:

```
null
```

28. feedback

Sends user's feedback. Authorization is required.

Mandatory parameters:

- title - the subject line.
- message - user's message.

Response:

```
null
```

29. getCategoriesDailyStats

Gets daily categories stats of a policy. Authorization is required.

Mandatory parameter:

- profile_id - policy ID.

Optional parameters:

- lang - response language.
- categories_list - defines a list of categories. if it's not set - the result will show all the categories.

Response:

```
{<category_name>: <value>}
```

30. getAdvertising

Gets the information on promo. Authorization is required

Response:

```
{<strAdvUrl>, <strImageUrl>}
```

31. getPlans

Gets the list of available billing plans.

Optional parameter:

- mobile - gets the Plans list, and determines whether to respond with mobile plans only. Boolean value.

Response:

```
{  
  'name': <plan name>,  
  'code': <plan code>,  
  'isMobile': <is mobile plan>,  
  'period': <period>,  
  'trialPeriod': <is trial>,  
  'price': <price>,  
  'description': <path>,  
}
```

32. getPlan

Gets user's billing plan. Authorization is required.

Response:

```
{  
  'name': <plan name>,  
  'expired': <expired_days>,  
  'isMobile': <true| false>,  
}
```

33. getAPCVersion

Gets mobile app's version: minimal supported and the current one.

Optional parameter:

- tag - tag for getting the versions (the value "default" is set by default).

Response:

```
{  
  "minimalSupported": <minimum supported version (integer, non-negative number)>,  
  "current": <current version of the mobile agent (integer, non-negative number)>  
}
```

Revision #10

Created 2 September 2022 01:44:19 by Val Redman

Updated 4 March 2025 22:04:02