Millistream Authentication and Authorization Service

Technical Description and Reference Guide

8 September 2021

Millistream Market Data AB
Box 5243, 402 24 Göteborg, Sweden

www.millistream.com
Org nr: 556763-4117

Introduction

The Millistream Authentication and Authorization Service (MAAS) is based on the OAuth2.0 standard with extensions where needed without sacrificing compatibility with OAuth2.0 applications.

To access the service a client have to provide authentication in the form of standard HTTP Basic authentication using the *client_id* and *client_secret* as the username and password. If keep-alive is used then only the initial request have to provide the client authentication since the connection is deemed authenticated until it's closed.

Connections are allowed to use keep-alive and can remain idle for up to 5 minutes before the connection is closed, there exists no limit on the number of requests per connection.

When posting data the caller can use either the default *application/x-www-form-urlencoded* (the only format supported by OAuth2.0), *application/json* or *application/xml* format by supplying a correct **Content-Type** header.

Likewise the caller can choose whether to receive replies in either *application/xml* (the default) or *application/json* by supplying a correct **Accept** header.

Note that when posting data in XML then the document/data must be enclosed by a dummy element as par with the XML standard, the name of this element does not matter and is ignored by MAAS. For XML replies MAAS will enclose the document/data with a element named **msg** and arrays will have each item enclosed by an element named **row**. **NULL** values will be indicated by a **xsi:nil="true"** attribute.

For *application/x-www-form-urlencoded* requests a value of "**null**" or a tag without value like "**tag1=**" or "**tag1=**&tag2=ff" can be used to indicate a **NULL** value.

For user authentication, two-factor authentication in the form of either Yubico OTP or TOTP (e.g the Google Authenticator of FreeOTP) is supported.

The access token can be added to each request with the "**X-On-Behalf-Of: Bearer {token string}**" header instead of e.g supplying it in POST requests with "**token={token string}**" for all requests where an access token have to be presented. This have been done in order to offer a consistent API even though it's not compatible with OAuth2.0 (since it's required by the **/files** Service which does not exist in OAuth2.0 anyway).

The service endpoint is https://maas.millistream.com:10200/

Concepts

- `client` - Clients are the users of the MAAS service itself. They can be services wanting to be able to authenticate end users for access to said service and they can be administrative users of MAAS for managing accounts etc. Clients are segregated with namespaces and clients can only access users, tokens and other objects issued by clients in the same namespace (except clients in the "Millistream" namespace; they have access to all other namespaces).
- `user` - This is the end user of the various services that the clients are providing. Like clients, users are segregated by namespaces enabling different services to have users with the same username without the risk of collision.
- `access token` - A 36-character long randomly generated string in the form of a UUID which users can use to gain access to services provided by clients, the clients then in turn provide the token to MAAS in order to check authorization and authentication. Each access token is connected to a user and client pair.
- `refresh token` - Shares the same characteristics as the access token, can not be used for authorization purposes but are instead used by clients to renew access tokens for users without having to authenticate the users with username+password.
- `scope` - The scope(s) define the sub services that the access token are authorized to access at the client service as the user.

All timestamps except the `last_modified` field will always be in UTC and have "YYYY-MM-DD HH:MM:SS" format, the `last_modified` timestamps have "YYYY-MM-DD HH:MM:SS.uuuuuu" as format to include microsecond resolution.

All strings are Unicode UTF-8 characters.

POST /tokens

Generates a new access token for a user. If `scope` contains "`trader`" and there already exists a valid access token for the same user+client then this existing access token will be automatically revoked.

Supported parameters:

Parameter	*Description*
`grant_type`	If "`password`" then username+password authentication is used, if "`refresh_token`" then a refresh token have to be supplied.
`username`	The user username for the "`password`" grant.
`password`	The user password for the "`password`" grant.
`scope`	Space separated list of the scopes the new token should be valid for. Valid values: "`mdf, mws, data-entry, symbol-lookup, mds, widgets, push, trader`".
`2fa`	Two-factor data for the "`password`" grant if two-factor authentication is required for the user.
`refresh_token`	The refresh token to use for the "`refresh_token`" grant.
`client_id`	The client to create the token under, if not present then the client_id from the requesting client is used.

Reply elements:

Element	*Description*
`access_token`	The new access token.
`token_type`	Currently the fixed string "bearer".
`expires_in`	The number of seconds before the access token expires.
`refresh_token`	The refresh token. Always returned but will be set to **`null`** if the refresh token expires before the access token as a hint to the caller that it will soon need to perform a password grant.

GET /tokens

List user tokens. Only tokens generated under the same client namespace as the requesting client are returned.

Supported query parameters:

Parameter	*Description*
`expired`	If included and set to any other value other than "0" then expired tokens will be included in the reply.
`userid`	Only include tokens created for this user **[numerical user id]**.
`client`	Only include tokens created by this client **[numerical client id]**.
`after`	Only include tokens that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`userid`	The user for who the token was created **[numerical user id]**.
`client`	The client that created the token **[numerical client id]**.
`token`	The token value.
`refresh_token`	The refresh token associated with this token. Set to **`null`** for refresh tokens, and for access tokens that never expire.
`valid_to`	Timestamp of when the token expires.
`type`	`access_token` or `refresh_token`.
`scope`	Space separated list of the scopes for the token.
`client_modified`	The client that last modified the token **[numerical client id]**.
`last_modified`	Timestamp of when the token was last modified.

POST /tokens/{token_value}

Modify expiry time and/or the scope of a token.

Supported parameters:

Parameter	*Description*
`valid_to`	A timestamp when the token should expire. If **`NULL`** the token never expires.
`scope`	Space separated list of the new set of scopes the token should be valid for. Valid values: "`mdf, mws, data-entry, symbol-lookup, mds, widgets, push, trader`".

MILLISTREAM

POST /introspect

Returns meta-data about a token. Supports rfc7662. Extended to support authorization.

Supported parameters:

Parameter	*Description*
`token`	The token to return meta-data for. The **`X-On-Behalf-Of`** header can be used instead.
`username`	The user username if no token is supplied and the client is authorized to perform user authorization.
`password`	The user password if no token is supplied and the client is authorized to perform user authorization.
`2fa`	Two-factor data if no token is supplied and the client is authorized to perform user authorization.
`auth_type`	Used if the caller also wants to receive the authorization data for the token. Valid values: "`trader, widgets, push, mws`".
`referer`	Referrer for the user, needed for `auth_type` requests if the user have one or more referrers set on its account.
`widget`	Name of the widget for `auth_type=widgets`.
`marketplace`	Comma separated list of marketplaces. Searching is done using OR.
`submarket`	Comma separated list of submarkets. Searching is done using OR.
`list`	Comma separated list of lists. Searching is done using OR.
`company`	Comma separated list of companies. Searching is done using OR.
`fund_company`	Comma separated list of companies. Searching is done using OR.
`insref`	Comma separated list of insrefs. Searching is done using OR.
`instrument_type`	Comma separated list of instrument types. Searching is done using AND.
`instrument_subtype`	Comma separated list of instrument subtypes. Searching is done using AND.

Reply elements:

Element	*Description*
`active`	Boolean indicator of whether or not the presented token is currently active. Will be either "`true`" or "`false`". Also if an `auth_type` fails in full then "`false`" will be returned.
`token_type`	Type of the token, either "`bearer`" or "refresh_`token`".
`exp`	Integer timestamp, measured in the number of seconds since January 1 1970 UTC, indicating when this token will expire. Absent for non expiring tokens.
`client_id`	Client identifier for the client that created this token.
`namespace`	The client namespace that this token is created under.
`userid`	The user for who the token was created **[numerical user id]**.
`username`	Username of the user for who this token was created.
`scope`	A string containing a space-separated list of scopes associated with this token.

Additional reply elements from "`auth_type=trader`":

Element	*Description*

`authorizations`	Object/element containing the below elements.
`widgets`	Array of strings describing the widgets that this token is authorized for.
`marketplaces`	Array of integers describing the marketplaces that this token is authorized for.
`newsagencies`	Array of integers describing the news agencies that this token is authorized for.

Additional reply elements from "`auth_type=widget`":

Element	*Description*
`authorizations`	Array of authorization elements, one for each of the requested items that the user is authorized for.
`marketplace`	Identifies a marketplace for this authorization element.
`submarket`	Identifies a submarket for this authorization element.
`list`	Identifies a list for this authorization element.
`company`	Identifies a company for this authorization element.
`fund_company`	Identifies a fund company for this authorization element.
`insref`	Identifies a insref for this authorization element.
`delay`	The authorized delay for this element. Valid values: "`realtime, 15-minutes, end-of-day`".
`historylen`	The maximum number of history days authorized for this element.
`package`	The authorized news package for this element.

Additional reply elements from "`auth_type=push`":

Element	*Description*
`authorizations`	Array of authorization elements, one array for each combination of mclass and delay.
`mclass`	Bitfield of the message classes that the array are for.
`delay`	The delay that the array are for. Valid values: "`realtime, 15-minutes`".
`instruments`	Array of the instruments that the user is authorized for in combination with the mclass and delay.
`insref`	Identifies a insref for this instruments element.
`package`	The authorized news package for this instruments element. Only present if not NULL.

Additional reply elements from "`auth_type=mws`":

Element	*Description*
`authorizations`	Array of authorization elements, one element for each combination of insref and delay.
`insref`	Specifies the insref for this element.
`mclass`	The authorized Message Classes for this element.
`ca_types`	The authorized CorporateAction types for this element. Only present if not NULL.
`ci_types`	The authorized CI/CIHistory types for this element. Only present if not NULL.
`news_package`	The authorized NewsHeadline/NewsContent package for this element. Only present if not NULL.
`estimates_source`	The authorized Estimates/EstimatesHistory source for this element. Only present if not NULL.
`delay`	The delay fort his element. Valid values: "`realtime, 15-minutes`".

For application caching purposes the following conditional HTTP headers are supported, note that the conditional will be for the requested token, so i.e the Last-Modified response header from a request on one token cannot be used as a conditional for another token.

HTTP Header	*Description*
`If-Modified-Since`	The server will send back a full reply only if it has been last modified after the given date+time. If the request has not been modified since, the response will be a 304 without any body; the **`Last-Modified`** response header of a previous request will contain the date of last modification.
`If-None-Match`	The server will send back the requested resource, only if it doesn't have an **`ETag`** matching the given one. If the **`Etag`** matches the response will be a 304 without any body; the last **`Etag`** response header of a previous request will contain the **`Etag`** value.

POST /revoke

Revokes a token. Supports rfc7009. If called on a refresh token then all access tokens associated with the refresh token will also be revoked.

Supported parameters:

Parameter	*Description*
`token`	The token to revoke. The **`X-On-Behalf-Of`** header can be used instead.

The server will make an empty 200 reply to all revocation requests.

GET /meta

Fetch enumeration meta data to populate various elements of the MAAS API.

Supported query parameters:

Parameter	*Description*
`element`	Only include meta data about the specified element. If not set then meta data for all elements will be returned. Valid values: "**`scope, mclass, instrument_types, ca_types, ci_types, news_packages`** and **`estimates_source`**".

The reply for all elements except for "news_packages" will be objects in the form of:

JSON: **`"element" : [ { "identifier" : value } … ]`**

XML: **`<element name="element"><enum id="identifier" value="value"/> … </element>`**

For "news_packages" the reply will be objects in the form of:

JSON: **`"news_packages" : [ { "insref" : insref, "packages" : [ { "identifier" : value} … ]}]`**

XML: **`<element name="news_packages"><insref>insref<enum id="identifier" value="value"/> … </insref></element>`**

Where "value" can be "**`null`**" for news agencies that have no packages.

GET /files

List files. If a access token is provided with the X-On-Behalf-Of header the the files existing under the user+client name space as indicated by the token are listed and the query parameters are not used.

If no access token is supplied then files belonging to all users and clients with the same client namespace as the requesting client is listed, further filtering can be applied with query parameters.

Supported query parameters:

Parameter	*Description*
`userid`	Only include files created by this user **[numerical user id]**.
`client`	Only include files created by this client **[numerical client id]**.
`filename`	Only include files whose name contains the supplied value as a substring.
`deleted`	If included and set to any other value than "0" then deleted files will be included in the reply.
`after`	Only include files that have a `last_modified` after the supplied value.

Reply elements when a access token is provided:

Element	*Description*
`filename`	The name of the file.
`filesize`	The size of the file in number of 8-bit bytes.
`ETag`	The ETag value of the file, HEX encoded SHA256 digest. If e.g the "sha256sum" command is run on a file the exact same value will be returned.
`last_modified`	Timestamp of when the file was last modified.

Reply elements when a access token is not provided:

Element	*Description*
`userid`	The user that created the file **[numerical user id]**.
`client`	The client that created the file **[numerical client id]**.
`filename`	The name of the file.
`filesize`	The size of the file in number of 8-bit bytes.
`ETag`	The ETag value of the file, HEX encoded SHA256 digest. If e.g the "sha256sum" command is run on a file the exact same value will be returned.
`deleted`	Boolean indicator of whether or not the file is deleted. Will be either "`true`" or "`false`".
`client_modified`	The client that last modified the file **[numerical client id]**.
`last_modified`	Timestamp of when the file was last modified.

GET /files/{filename}

Returns the contents of the file named {filename} owned by the user+client as indicated by the token from the ***X-On-Behalf-Of*** *header.*

For caching purposes the following conditional HTTP headers are supported:

HTTP Header	*Description*
`If-Match`	The server will send back the requested file, only if it matches the given **`ETag`**. If **`*`** is given as the value then it will always match. If the **`ETag`** does not match the response will be a 412 without any body; the last **`ETag`** response header of a previous request will contain the **`ETag`** value.
`If-None-Match`	The server will send back the requested file, only if it doesn't have an **`ETag`** matching the given one. If **`*`** is given as the value then it will never match. If the **`Etag`** matches the response will be a 304 without any body; the last **`Etag`** response header of a previous request will contain the **`Etag`** value.
`If-Unmodified-Since`	The server will send back the requested file only if it has not been last modified after the given date+time. If the file has been modified since, the response will be a 412 without any body; the **`Last-Modified`** response header of a previous request will contain the date of last modification. Ignored if the request contains a **`If-Match`** header.
`If-Modified-Since`	The server will send back the requested file only if it has been last modified after the given date+time. If the file has not been modified since, the response will be a 304 without any body; the **`Last-Modified`** response header of a previous request will contain the date of last modification. Ignored if the request contains a **`If-None-Match`** header.

PUT /files/{filename}

Creates or replaces the file named {filename} owned by the user+client as indicated by the token from the ***X-On-Behalf-Of*** *header with the contents of the request.*

Each user+client combination creates a separate filename namespace so the same user can have different files with the same filename at the same time created by different clients.

Note that all meta-data about the file except the filename will be ignored, the filename is limited to 36 characters in length, must contain characters in the 7-bit ascii range 32-125 with the exception of the ' ***/****' character and is case sensitive.*

The creation of a new file will be indicated by a 201 reply while the replacement of an existing file will be indicated by a 204 reply. The ***ETag*** *header will be a HEX encoded SHA256 digest of the contents of the file, and will be included in the reply together with a* ***Last-Modified*** *date+time header.*

For caching and validation purposes the following conditional HTTP headers are supported:

HTTP Header	*Description*
`If-Match`	The server will upload the file, only if it matches the given **ETag**. If ***** is given as the value then it will always match. If the **ETag** does not match the response will be a 412 without any body; the last **ETag** response header of a previous request will contain the **ETag** value.
`If-None-Match`	The server will upload the file, only if it doesn't have an **ETag** matching the given one. If ***** is given as the value then it will never match. If the **Etag** matches the response will be a 412 without any body; the last **Etag** response header of a previous request will contain the **Etag** value.
`If-Unmodified-Since`	The server will upload the file only if it has not been last modified after the given date+time. If the file has been modified since, the response will be a 412 without any body; the **Last-Modified** response header of a previous request will contain the date of last modification. Ignored if the request contains a **If-Match** header.
`If-Modified-Since`	The server will upload the file only if it has been last modified after the given date+time. If the file has not been modified since, the response will be a 412 without any body; the **Last-Modified** response header of a previous request will contain the date of last modification. Ignored if the request contains a **If-None-Match** header.

Further the "**Expect: 100-continue**" header is fully supported for clients wanting to check all the conditionals and other validity checks before sending a large file.

DELETE /files/{filename}

Deletes the file named {filename} owned by the user+client as indicated by the token from the ***X-On-Behalf-Of*** *header.*

For caching purposes the following conditional HTTP headers are supported:

HTTP Header	*Description*
`If-Match`	The server will delete the file, only if it matches the given **`ETag`**. If **`*`** is given as the value then it will always match. If the **`ETag`** does not match the response will be a 412 without any body; the last **`ETag`** response header of a previous request will contain the **`ETag`** value.
`If-None-Match`	The server will delete the file, only if it doesn't have an **`ETag`** matching the given one. If **`*`** is given as the value then it will never match. If the **`Etag`** matches the response will be a 412 without any body; the last **`Etag`** response header of a previous request will contain the **`Etag`** value.
`If-Unmodified-Since`	The server will delete the file only if it has not been last modified after the given date+time. If the file has been modified since, the response will be a 412 without any body; the **`Last-Modified`** response header of a previous request will contain the date of last modification. Ignored if the request contains a **`If-Match`** header.
`If-Modified-Since`	The server will delete the file only if it has been last modified after the given date+time. If the file has not been modified since, the response will be a 412 without any body; the **`Last-Modified`** response header of a previous request will contain the date of last modification. Ignored if the request contains a **`If-None-Match`** header.

POST /password

Generate a new random password or create an encrypted salt+password combination for a supplied clear text password. Can also generate a TOTP secret for two-factor authentication.

Supported parameters:

Parameter	*Description*
`password`	A password in clear text that the service will encrypt to a salt+password combination
`password_len`	A random password with the supplied length in number of characters will be created and both the clear text and encrypted salt+password combination will be returned. The given length must be between 1 and 128 characters.
`secret_bits`	A random TOTP secret with the supplied length in number of bits will be created, supported values are 80/128/160/256.

Reply elements:

Element	*Description*
`cleartext`	The generated password in cleartext so that the caller can inform the user about his/her new password.
`salt`	HEX encoded 256-bit salt used to encrypt the password.
`password`	HEX encoded 256-bit digest of the encrypted password.
`secret`	Base32 encoded TOTP secret.

GET /users

List users.

Supported query parameters:

Parameter	*Description*
`expired`	If included and set to any other value other than "0" then expired users will be included in the reply.
`client`	Only include users that have been last modified by this client **[numerical client id]**.
`username`	Perform a wildcard search for users whose username contains the supplied value.
`company`	Perform a wildcard search for users whose company contains the supplied value.
`namespace`	Perform a wildcard search for users whose client namespace contains the supplied value.

Reply elements:

Element	*Description*
`userid`	The id of the user **[numerical user id]**.
`username`	The username of the user.
`company`	The company of the user.
`namespace`	The client namespace of the user.
`user_type`	Can be '**`private`**', '**`professional`**' or **`NULL`**.
`valid_from`	Timestamp when the user will be enabled
`valid_to`	Timestamp when the user will be expired/disabled. Can be **`NULL`** if no valid_to time have been set for the user.
`client_modified`	The client that last modified the user **[numerical client id]**.
`last_modified`	Timestamp of when the user was last modified.

GET /users/log

List the log of changes done to all users.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the user was created in this event, or '**`modify`**' if the user was modified.
`userid`	The id of the user **[numerical user id]**.
`username`	Username.
`password`	Boolean that is **`true`** if the password changed with this log item, **`false`** if not.
`namespace`	Client namespace.
`company`	Company Name.
`contact`	Contact Person.
`email`	E-mail.
`phone`	Phone number.
`comment`	Free text comment.
`user_type`	Can be '**`private`**', '**`professional`**' or **`NULL`**.
`twofactor_type`	**`Yubikey`**, **`TOTP`** or **`NULL`** for no Two-Factor authentication.
`twofactor_data`	For **`Yubikey`**, a string in *`application/x-www-form-urlencoded`* format can be used to carry the following keys: `client_id` — Numerical id for authenticating the request to the verification server. If **`NULL`** then the default MAAS id will be used. `client_key` — Base64 encoded hash key for authentication the request to the verification server. If **`NULL`** then the default MAAS key will be used. `public_id` — The first 12 characters from the Yubikey OTP, used to connect the user to a specific Yubikey. For **`TOTP`**, this is the Base32 encoded secret.
`twofactor_server`	For **`Yubikey`**, this can be a space separated list of verification servers to use. **`NULL`** then the YubiCloud is used for the OTP verification.
`logins_max`	Maximum number of simultaneous logins on mdf.
`mdf_send`	**`Y`** if user is allowed to send data using mdf. NULL if not.
`mdf_create`	**`Y`** if user is allowed to create instruments using mdf. NULL if not.
`mdf_disable_naggle`	**`Y`** if user is allowed to disable the naggle algorithm for tcp/ip using mdf. NULL if not.
`mdf_disable_encryption`	**`Y`** if user is allowed to disable server to client encryption using mdf. NULL if not.

`referer`	Space separated list of allowed referrers for this user.
`scope`	Space separated list of the scopes that the user is authorized for.
`created`	Timestamp when the user was created.
`valid_from`	Timestamp when the user will be enabled.
`valid_to`	Timestamp when the user will be expired/disabled. **`NULL`** if no valid_to time have been set for the user.
`client_modified`	The client that modified the log item **[numerical client id]**.
`last_modified`	Timestamp of when the user was last modified.

GET /users/{username}|{userid}

Return data about the user identified by either a username or a userid.

Reply elements:

Element	*Description*
`userid`	The id of the user **[numerical user id]**.
`username`	Username.
`namespace`	Client namespace.
`company`	Company Name.
`contact`	Contact Person.
`email`	E-mail.
`phone`	Phone number.
`comment`	Free text comment.
`user_type`	Can be '**`private`**', '**`professional`**' or **`NULL`**.
`twofactor_type`	**`Yubikey`**, **`TOTP`** or **`NULL`** for no Two-Factor authentication.
`twofactor_data`	For **`Yubikey`**, a string in *`application/x-www-form-urlencoded`* format can be used to carry the following keys: *Key* / *Description* `client_id` — Numerical id for authenticating the request to the verification server. If **`NULL`** then the default MAAS id will be used. `client_key` — Base64 encoded hash key for authentication the request to the verification server. If **`NULL`** then the default MAAS key will be used. `public_id` — The first 12 characters from the Yubikey OTP, used to connect the user to a specific Yubikey. For **`TOTP`**, this is the Base32 encoded secret.
`twofactor_server`	For **`Yubikey`**, this can be a space separated list of verification servers to use. If **`NULL`** then the YubiCloud is used for the OTP verification.
`logins_max`	Maximum number of simultaneous logins on mdf.
`mdf_send`	**`Y`** if user is allowed to send data using mdf. NULL if not.
`mdf_create`	**`Y`** if user is allowed to create instruments using mdf. NULL if not.
`mdf_disable_naggle`	**`Y`** if user is allowed to disable the naggle algorithm for tcp/ip using mdf. NULL if not.
`mdf_disable_encryption`	**`Y`** if user is allowed to disable server to client encryption using mdf. NULL if not.
`referer`	Space separated list of allowed referrers for this user.
`scope`	Space separated list of the scopes that the user is authorized for.
`created`	Timestamp when the user was created in "YYYY-MM-DD HH:MM:SS" format, expressed in UTC.
`valid_from`	Timestamp when the user will be enabled.
`valid_to`	Timestamp when the user will be expired/disabled. **`NULL`** if no valid_to time have been set for the user.
`client_modified`	The client that last modified the user **[numerical client id]**.

`last_modified`	Timestamp of when the user was last modified.

GET /users/{username}|{userid}/log

List the log of changes done to a user identified by either a username or userid.

Supported query parameters:

Parameter	*Description*
`after`	Only include log items that have a `last_modified` after the supplied value.
`client`	Only include log items modified by this client **[numerical client id]**.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the user was created in this event, or '**`modify`**' if the user was modified.
`userid`	The id of the user **[numerical user id]**.
`username`	Username.
`password`	Boolean that is **`true`** if the password changed with this log item, **`false`** if not.
`namespace`	Client namespace.
`company`	Company Name.
`contact`	Contact Person.
`email`	E-mail.
`phone`	Phone number.
`comment`	Free text comment.
`user_type`	Can be '**`private`**', '**`professional`**' or **`NULL`**.
`twofactor_type`	**`Yubikey`**, **`TOTP`** or **`NULL`** for no Two-Factor authentication.
`twofactor_data`	For **`Yubikey`**, a string in *`application/x-www-form-urlencoded`* format can be used to carry the following keys: `client_id` — Numerical id for authenticating the request to the verification server. If **`NULL`** then the default MAAS id will be used. `client_key` — Base64 encoded hash key for authentication the request to the verification server. If **`NULL`** then the default MAAS key will be used. `public_id` — The first 12 characters from the Yubikey OTP, used to connect the user to a specific Yubikey. For **`TOTP`**, this is the Base32 encoded secret.
`twofactor_server`	For **`Yubikey`**, this can be a space separated list of verification servers to use. If **`NULL`** then the YubiCloud is used for the OTP verification.
`logins_max`	Maximum number of simultaneous logins on mdf.
`mdf_send`	**`Y`** if user is allowed to send data using mdf. NULL if not.
`mdf_create`	**`Y`** if user is allowed to create instruments using mdf. NULL if not.
`mdf_disable_naggle`	**`Y`** if user is allowed to disable the naggle algorithm for tcp/ip using mdf. NULL if not.
`mdf_disable_encryption`	**`Y`** if user is allowed to disable server to client encryption using mdf. NULL if not.

`referer`	Space separated list of allowed referrers for this user.
`scope`	Space separated list of the scopes that the user is authorized for.
`created`	Timestamp when the user was created.
`valid_from`	Timestamp when the user will be enabled.
`valid_to`	Timestamp when the user will be expired/disabled. **`NULL`** if no valid_to time have been set for the user.
`client_modified`	The client that modified the log item **[numerical client id]**.
`last_modified`	Timestamp of when the user was last modified.

POST /users

Create or modify a user.

The creation of a new user will be indicated by a 201 reply while modification of an existing user will be indicated by a 200 reply. Trying to add an already existing user will be indicated by a 409 reply.

*When creating a new user, the `username`, `salt` and `password` parameters are mandatory. When modifying, unchanged parameters can be omitted from the request (if `salt` and `password` are set to **`NULL`** they are treated as if they where omitted).*

Supported parameters:

Parameter	*Description*
`userid`	The id of the user **[numerical user id]**. Must be **`NULL`** or absent when creating a user. Must be included when editing a user, a request without the correct userid on a username that already exists will be replied with "409 Conflict".
`username`	Username. Must be unique within the client namespace. Maximum 64 characters.
`salt`	HEX encoded 256-bit salt used to encrypt the password.
`password`	HEX encoded 256-bit digest of the encrypted password.
`namespace`	Client namespace. If set to **`NULL`**, the namespace of the requesting client is used.
`company`	Company Name.
`contact`	Contact Person.
`email`	E-mail.
`phone`	Phone number.
`comment`	Free text comment. Maximum 2048 characters.
`user_type`	Can be '**`private`**', '**`professional`**' or **`NULL`**.
`twofactor_type`	**`Yubikey`**, **`TOTP`** or **`NULL`** for no Two-Factor authentication.
`twofactor_data`	For **`Yubikey`**, a string in *`application/x-www-form-urlencoded`* format can be used to carry the following keys: *Key* / *Description* `client_id` — Numerical id for authenticating the request to the verification server. If **`NULL`** then the default MAAS id will be used. `client_key` — Base64 encoded hash key for authentication the request to the verification server. If **`NULL`** then the default MAAS key will be used. `public_id` — The first 12 characters from the Yubikey OTP, used to connect the user to a specific Yubikey. For **`TOTP`**, this is the Base32 encoded secret.
`twofactor_server`	For **`Yubikey`**, this can be a space separated list of verification servers to use. If **`NULL`** then the YubiCloud is used for the OTP verification.
`logins_max`	Maximum number of simultaneous logins on mdf. **`NULL`** if unrestricted.
`mdf_send`	**`Y`** if user is allowed to send data using mdf. **`NULL`** if not.
`mdf_create`	**`Y`** if user is allowed to create instruments using mdf. **`NULL`** if not.
`mdf_disable_naggle`	**`Y`** if user is allowed to disable the naggle algorithm for tcp/ip using mdf. **`NULL`** if not.
`mdf_disable_encryption`	**`Y`** if user is allowed to disable server to client encryption using mdf. **`NULL`** if not.

`referer`	Space separated list of allowed referrers for this user. If **`NULL`** then no referrer check will be performed. The referrer from the default user for the namespace will be used if a **`NULL`** or absent value is posted.
`scope`	Space separated list of the scopes that the user is authorized for. The scope from the default user for the namespace will be used if a **`NULL`** or absent value is posted.
`valid_from`	Timestamp when the user will be enabled. If set to **`NULL`** then "now" is used on add.
`valid_to`	Timestamp when the user will be expired/disabled. **`NULL`** if no invalidation time exists.

Reply elements:

Element	*Description*
`userid`	The id of the newly created user **[numerical user id]**. Only sent in a 201 reply.

GET /clients

List clients.

Supported query parameters:

Parameter	*Description*
`expired`	If included and set to any other value other than "0" then expired clients will be included in the reply.
`client`	Only include clients that have been last modified by this client **[numerical client id]**.
`client_id`	Perform a wildcard search for clients whose client_id contains the supplied value.
`namespace`	Perform a wildcard search for clients whose client namespace contains the supplied value.

Reply elements:

Element	*Description*
`client`	The id of the client **[numerical user id]**.
`client_id`	The username of the client.
`namespace`	The client namespace of the client.
`valid_from`	Timestamp when the client will be enabled.
`valid_to`	Timestamp when the client will be expired/disabled. **`NULL`** if no valid_to time have been set for the client.
`client_modified`	The client that last modified the client **[numerical client id]**.
`last_modified`	Timestamp of when the client was last modified.

GET /clients/log

List the log of changes done to all clients.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the client was created in this event, or '**`modify`**' if the client was modified.
`client`	The id of the client **[numerical user id]**.
`client_id`	Username.
`comment`	Free text comment.
`namespace`	Client namespace.
`users_delete`	**`Y`** if client is allowed to delete users by set/decrease `valid_to`. **`NULL`** if not.
`users_add`	**`Y`** if client is allowed to add users. **`NULL`** if not.
`users_view`	**`Y`** if client is allowed to view users. **`NULL`** if not.
`users_edit`	**`Y`** if client is allowed to edit users. **`NULL`** if not.
`users_authorize`	**`Y`** if client is allowed to use `username,password and 2fa` on `/introspect`.
`clients_delete`	**`Y`** if client is allowed to delete clients by set/decrease `valid_to`. **`NULL`** if not.
`clients_add`	**`Y`** if client is allowed to add clients. **`NULL`** if not.
`clients_view`	**`Y`** if client is allowed to view clients. **`NULL`** if not.
`clients_edit`	**`Y`** if client is allowed to edit clients. **`NULL`** if not.
`feed_nodes_delete`	**`Y`** if client is allowed to delete feed-nodes. **`NULL`** if not.
`feed_nodes_add`	**`Y`** if client is allowed to add feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_view`	**`Y`** if client is allowed to view feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_edit`	**`Y`** if client is allowed to edit feed-nodes. **`NULL`** if not.
`stats_delete`	**`Y`** if client is allowed to delete statistics. **`NULL`** if not.
`stats_add`	**`Y`** if client is allowed to add statistics. **`NULL`** if not.
`stats_view`	**`Y`** if client is allowed to view statistics. **`NULL`** if not.
`stats_edit`	**`Y`** if client is allowed to edit statistics. **`NULL`** if not.
`tokens_delete`	**`Y`** if client is allowed to delete tokens. **`NULL`** if not.
`tokens_add`	**`Y`** if client is allowed to create tokens. **`NULL`** if not.
`tokens_view`	**`Y`** if client is allowed to view tokens. **`NULL`** if not.
`tokens_edit`	**`Y`** if client is allowed to edit tokens. **`NULL`** if not.
`tokens_authorize`	**`Y`** if client is allowed to use `auth_type` on `POST /introspect`. **`NULL`** if not.
`logins_delete`	

logins_add	
logins_view	
logins_edit	
packages_delete	**Y** if client is allowed to delete packages. **NULL** if not.
packages_add	**Y** if client is allowed to add packages. **NULL** if not.
packages_view	**Y** if client is allowed to view packages. **NULL** if not.
packages_edit	**Y** if client is allowed to edit packages. **NULL** if not.
authorizations_delete	**Y** if client is allowed to delete authorizations. **NULL** if not.
authorizations_add	**Y** if client is allowed to add authorizations. **NULL** if not.
authorizations_view	**Y** if client is allowed to view authorizations. **NULL** if not.
authorizations_edit	**Y** if client is allowed to edit authorizations. **NULL** if not.
files_delete	**Y** if client is allowed to delete files. **NULL** if not.
files_add	**Y** if client is allowed to add files. **NULL** if not.
files_view	**Y** if client is allowed to view files. **NULL** if not.
files_edit	**Y** if client is allowed to edit files. **NULL** if not.
default_user_delete	**Y** if client is allowed to delete the default user. **NULL** if not.
default_user_add	**Y** if client is allowed to add a default user. **NULL** if not.
default_user_view	**Y** if client is allowed to view the default user. **NULL** if not.
default_user_edit	**Y** if client is allowed to edit the default user. **NULL** if not.
is_maas	**Y** if client is another MAAS server. **NULL** if not.
scope	Space separated list of the scopes that the client is authorized for.
created	Timestamp when the client was created.
valid_from	Timestamp when the client will be enabled.
valid_to	Timestamp when the client will be expired/disabled. **NULL** if no valid_to time have been set for the client.
client_modified	The client that modified the log item **[numerical client id]**.
last_modified	Timestamp of when the client was last modified.

GET /clients/{client_id}|{client}

Return data about the client identified by either a client_id or a client. A client is always allowed to perform this on itself even if the client is not allowed to view clients.

Reply elements:

Element	*Description*
client	The id of the client **[numerical user id]**.
client_id	Username.
comment	Free text comment.
namespace	Client namespace.
users_delete	**Y** if client is allowed to delete users (and decrease valid_to). **NULL** if not.
users_add	**Y** if client is allowed to add users. **NULL** if not.
users_view	**Y** if client is allowed to view users. **NULL** if not.
users_edit	**Y** if client is allowed to edit users. **NULL** if not.
users_authorize	**Y** if client is allowed to use username,password and 2fa on /introspect.
clients_delete	**Y** if client is allowed to delete clients (and decrease valid_to). **NULL** if not.
clients_add	**Y** if client is allowed to add clients. **NULL** if not.
clients_view	**Y** if client is allowed to view clients. **NULL** if not.
clients_edit	**Y** if client is allowed to edit clients. **NULL** if not.
feed_nodes_delete	**Y** if client is allowed to delete feed-nodes. **NULL** if not.
feed_nodes_add	**Y** if client is allowed to add feed-nodes and insrefs. **NULL** if not.
feed_nodes_view	**Y** if client is allowed to view feed-nodes and insrefs. **NULL** if not.
feed_nodes_edit	**Y** if client is allowed to edit feed-nodes. **NULL** if not.
stats_delete	**Y** if client is allowed to delete statistics. **NULL** if not.
stats_add	**Y** if client is allowed to add statistics. **NULL** if not.
stats_view	**Y** if client is allowed to view statistics. **NULL** if not.
stats_edit	**Y** if client is allowed to edit statistics. **NULL** if not.
tokens_delete	**Y** if client is allowed to delete tokens. **NULL** if not.
tokens_add	**Y** if client is allowed to create tokens. **NULL** if not.
tokens_view	**Y** if client is allowed to view tokens. **NULL** if not.
tokens_edit	**Y** if client is allowed to edit tokens. **NULL** if not.
tokens_authorize	**Y** if client is allowed to use auth_type on POST /introspect. **NULL** if not.
logins_delete	
logins_add	
logins_view	
logins_edit	
packages_delete	**Y** if client is allowed to delete packages. **NULL** if not.
packages_add	**Y** if client is allowed to add packages. **NULL** if not.
packages_view	**Y** if client is allowed to view packages. **NULL** if not.

packages_edit	**Y** if client is allowed to edit packages. **NULL** if not.
authorizations_delete	**Y** if client is allowed to delete authorizations. **NULL** if not.
authorizations_add	**Y** if client is allowed to add authorizations. **NULL** if not.
authorizations_view	**Y** if client is allowed to view authorizations. **NULL** if not.
authorizations_edit	**Y** if client is allowed to edit authorizations. **NULL** if not.
files_delete	**Y** if client is allowed to delete files. **NULL** if not.
files_add	**Y** if client is allowed to add files. **NULL** if not.
files_view	**Y** if client is allowed to view files. **NULL** if not.
files_edit	**Y** if client is allowed to edit files. **NULL** if not.
default_user_delete	**Y** if client is allowed to delete the default user. **NULL** if not.
default_user_add	**Y** if client is allowed to add a default user. **NULL** if not.
default_user_view	**Y** if client is allowed to view the default user. **NULL** if not.
default_user_edit	**Y** if client is allowed to edit the default user. **NULL** if not.
is_maas	**Y** if client is another MAAS server. **NULL** if not.
scope	Space separated list of the scopes that the client is authorized for.
created	Timestamp when the client was created.
valid_from	Timestamp when the client will be enabled.
valid_to	Timestamp when the client will be expired/disabled. **NULL** if no valid_to time have been set for the client.
client_modified	The client that last modified the client **[numerical client id]**.
last_modified	Timestamp of when the client was last modified.

GET /clients/{client_id}|{client}/log

List the log of changes done to a client identified by either a client_id or client.

Supported query parameters:

Parameter	*Description*
`after`	Only include log items that have a `last_modified` after the supplied value.
`client`	Only include log items modified by this client **[numerical client id]**.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the client was created in this event, or '**`modify`**' if the client was modified.
`client`	The id of the client **[numerical user id]**.
`client_id`	Username.
`comment`	Free text comment.
`namespace`	Client namespace.
`users_delete`	**`Y`** if client is allowed to delete users (and decrease `valid_to`). **`NULL`** if not.
`users_add`	**`Y`** if client is allowed to add users. **`NULL`** if not.
`users_view`	**`Y`** if client is allowed to view users. **`NULL`** if not.
`users_edit`	**`Y`** if client is allowed to edit users. **`NULL`** if not.
`users_authorize`	**`Y`** if client is allowed to use `username,password and 2fa` on `/introspect`.
`clients_delete`	**`Y`** if client is allowed to delete clients (and decrease `valid_to`). **`NULL`** if not.
`clients_add`	**`Y`** if client is allowed to add clients. **`NULL`** if not.
`clients_view`	**`Y`** if client is allowed to view clients. **`NULL`** if not.
`clients_edit`	**`Y`** if client is allowed to edit clients. **`NULL`** if not.
`feed_nodes_delete`	**`Y`** if client is allowed to delete feed-nodes. **`NULL`** if not.
`feed_nodes_add`	**`Y`** if client is allowed to add feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_view`	**`Y`** if client is allowed to view feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_edit`	**`Y`** if client is allowed to edit feed-nodes. **`NULL`** if not.
`stats_delete`	**`Y`** if client is allowed to delete statistics. **`NULL`** if not.
`stats_add`	**`Y`** if client is allowed to add statistics. **`NULL`** if not.
`stats_view`	**`Y`** if client is allowed to view statistics. **`NULL`** if not.
`stats_edit`	**`Y`** if client is allowed to edit statistics. **`NULL`** if not.
`tokens_delete`	**`Y`** if client is allowed to delete tokens. **`NULL`** if not.
`tokens_add`	**`Y`** if client is allowed to create tokens. **`NULL`** if not.
`tokens_view`	**`Y`** if client is allowed to view tokens. **`NULL`** if not.
`tokens_edit`	**`Y`** if client is allowed to edit tokens. **`NULL`** if not.
`tokens_authorize`	**`Y`** if client is allowed to use `auth_type` on `POST /introspect`. **`NULL`** if not.
`logins_delete`	

logins_add	
logins_view	
logins_edit	
packages_delete	**Y** if client is allowed to delete packages. **NULL** if not.
packages_add	**Y** if client is allowed to add packages. **NULL** if not.
packages_view	**Y** if client is allowed to view packages. **NULL** if not.
packages_edit	**Y** if client is allowed to edit packages. **NULL** if not.
authorizations_delete	**Y** if client is allowed to delete authorizations. **NULL** if not.
authorizations_add	**Y** if client is allowed to add authorizations. **NULL** if not.
authorizations_view	**Y** if client is allowed to view authorizations. **NULL** if not.
authorizations_edit	**Y** if client is allowed to edit authorizations. **NULL** if not.
files_delete	**Y** if client is allowed to delete files. **NULL** if not.
files_add	**Y** if client is allowed to add files. **NULL** if not.
files_view	**Y** if client is allowed to view files. **NULL** if not.
files_edit	**Y** if client is allowed to edit files. **NULL** if not.
default_user_delete	**Y** if client is allowed to delete the default user. **NULL** if not.
default_user_add	**Y** if client is allowed to add a default user. **NULL** if not.
default_user_view	**Y** if client is allowed to view the default user. **NULL** if not.
default_user_edit	**Y** if client is allowed to edit the default user. **NULL** if not.
is_maas	**Y** if client is another MAAS server. **NULL** if not.
scope	Space separated list of the scopes that the client is authorized for.
created	Timestamp when the client was created.
valid_from	Timestamp when the client will be enabled.
valid_to	Timestamp when the client will be expired/disabled. **NULL** if no valid_to time have been set for the client.
client_modified	The client that modified the log item **[numerical client id]**.
last_modified	Timestamp of when the client was last modified.

POST /clients

Create or modify a client.

The creation of a new client will be indicated by a 201 reply while the modification of an existing client will be indicated by a 200 reply.

*When creating a new client, the `client_id`, `salt` and `client_secret` parameters are mandatory. When modifying, unchanged parameters can be omitted from the request (if `salt` and `client_secret` are set to **`NULL`** they are treated as if they where omitted).*

Supported parameters:

Element	*Description*
`client`	The id of the client **[numerical user id]**. If present then the request is seen as an edit, if not (or set to **`NULL`**) then the request is seen as and add.
`client_id`	Username. Must be unique regardless of namespace. Maximum 64 characters.
`comment`	Free text comment. Maximum 2048 characters.
`salt`	HEX encoded 256-bit salt used to encrypt the password.
`client_secret`	HEX encoded 256-bit digest of the encrypted password.
`namespace`	Client namespace. If set to **`NULL`** on add, the namespace of the requesting client is used.
`users_delete`	**`Y`** if client is allowed to delete users (and decrease `valid_to`). **`NULL`** if not.
`users_add`	**`Y`** if client is allowed to add users. **`NULL`** if not.
`users_view`	**`Y`** if client is allowed to view users. **`NULL`** if not.
`users_edit`	**`Y`** if client is allowed to edit users. **`NULL`** if not.
`users_authorize`	**`Y`** if client is allowed to use `username,password and 2fa` on `/introspect`.
`clients_delete`	**`Y`** if client is allowed to delete clients (and decrease `valid_to`). **`NULL`** if not.
`clients_add`	**`Y`** if client is allowed to add clients. **`NULL`** if not.
`clients_view`	**`Y`** if client is allowed to view clients. **`NULL`** if not.
`clients_edit`	**`Y`** if client is allowed to edit clients. **`NULL`** if not.
`feed_nodes_delete`	**`Y`** if client is allowed to delete feed-nodes. **`NULL`** if not.
`feed_nodes_add`	**`Y`** if client is allowed to add feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_view`	**`Y`** if client is allowed to view feed-nodes and insrefs. **`NULL`** if not.
`feed_nodes_edit`	**`Y`** if client is allowed to edit feed-nodes. **`NULL`** if not.
`stats_delete`	**`Y`** if client is allowed to delete statistics. **`NULL`** if not.
`stats_add`	**`Y`** if client is allowed to add statistics. **`NULL`** if not.
`stats_view`	**`Y`** if client is allowed to view statistics. **`NULL`** if not.
`stats_edit`	**`Y`** if client is allowed to edit statistics. **`NULL`** if not.
`tokens_delete`	**`Y`** if client is allowed to delete tokens. **`NULL`** if not.
`tokens_add`	**`Y`** if client is allowed to create tokens. **`NULL`** if not.
`tokens_view`	**`Y`** if client is allowed to view tokens. **`NULL`** if not.
`tokens_edit`	**`Y`** if client is allowed to edit tokens. **`NULL`** if not.
`tokens_authorize`	**`Y`** if client is allowed to use `auth_type` on `POST /introspect`. **`NULL`** if not.
`logins_delete`	

logins_add	
logins_view	
logins_edit	
packages_delete	**Y** if client is allowed to delete packages. **NULL** if not.
packages_add	**Y** if client is allowed to add packages. **NULL** if not.
packages_view	**Y** if client is allowed to view packages. **NULL** if not.
packages_edit	**Y** if client is allowed to edit packages. **NULL** if not.
authorizations_delete	**Y** if client is allowed to delete authorizations. **NULL** if not.
authorizations_add	**Y** if client is allowed to add authorizations. **NULL** if not.
authorizations_view	**Y** if client is allowed to view authorizations. **NULL** if not.
authorizations_edit	**Y** if client is allowed to edit authorizations. **NULL** if not.
files_delete	**Y** if client is allowed to delete files. **NULL** if not.
files_add	**Y** if client is allowed to add files. **NULL** if not.
files_view	**Y** if client is allowed to view files. **NULL** if not.
files_edit	**Y** if client is allowed to edit files. **NULL** if not.
default_user_delete	**Y** if client is allowed to delete the default user. **NULL** if not.
default_user_add	**Y** if client is allowed to add a default user. **NULL** if not.
default_user_view	**Y** if client is allowed to view the default user. **NULL** if not.
default_user_edit	**Y** if client is allowed to edit the default user. **NULL** if not.
is_maas	**Y** if client is another MAAS server. **NULL** if not.
scope	Space separated list of the scopes that the client is authorized for.
valid_from	Timestamp when the client will be enabled. If **NULL** then "now" is used on add.
valid_to	Timestamp when the client will be expired/disabled. **NULL** if no valid_to time have been set for the client.

GET /packages

List the available authorization packages.

Reply elements:

Element	*Description*
`name`	Name of the package
`namespace`	Namespace of the package

GET /packages/log

List the log of changes done to all packages.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.
`after_id`	Only include log items that have a `id` larger than the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the package row was created in this event, '**`modify`**' if the package row was modified in this event and '**`delete`**' if the package row was deleted in this event.
`replaces`	The id of the original package row that this package row replaces if event is '**`modify`**' or '**`delete`**'.
`id`	The id of the package row.
`namespace`	The namespace of the package.
`name`	The name of the package.
`comment`	Free-text comment of the package row.
`package`	Name of a package that will be brought in by this package row and whose authorization items will be overridden by any non-**`NULL`** values from this package row.
`widget`	Name of a widget that will be authorized by this package row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the package row will take effect.

`valid_to`	Timestamp when the package row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the package row **[numerical client id]**.
`last_modified`	Timestamp when the package row was modified

GET /packages/{name}

List the authorizations of the package identified by name.

Reply elements:

Element	*Description*
`id`	The id of the package row.
`namespace`	The namespace of the package.
`name`	The name of the package.
`comment`	Free-text comment of the package row.
`package`	Name of a package that will be brought in by this package row and whose authorization items will be overridden by any non-**`NULL`** values from this package row.
`widget`	Name of a widget that will be authorized by this package row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`esitmates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization will take effect.
`valid_to`	Timestamp when the package row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the package row **[numerical client id]**.
`last_modified`	Timestamp when the package row was modified

GET /packages/{name}/log

List the log of changes done to the package identified by name.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.
`after_id`	Only include log items that have a `id` larger than the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the package row was created in this event, '**`modify`**' if the package row was modified in this event and '**`delete`**' if the package row was deleted in this event.
`replaces`	The id of the original package row that this package row replaces if event is '**`modify`**' or '**`delete`**'.
`id`	The id of the package row.
`namespace`	The namespace of the package.
`name`	The name of the package.
`comment`	Free-text comment of the package row.
`package`	Name of a package that will be brought in by this package row and whose authorization items will be overridden by any non-**`NULL`** values from this package row.
`widget`	Name of a widget that will be authorized by this package row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the package row will take effect.

`valid_to`	Timestamp when the package row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the package row **[numerical client id]**.
`last_modified`	Timestamp when the package row was modified

POST /packages/{name}

Modify the authorizations of the package identified by name.

The post-data must contain the complete authorization of the package (absent rows will be seen as deletes) and can only be in either JSON or XML since urlencoded does not support arrays.

Supported parameters:

Element	*Description*
`id`	The id of the authorization row. Must be **`NULL`** or absent for inserts.
`namespace`	Package namespace. If **`NULL`**, the namespace of the requesting client is used.
`comment`	Free-text comment of the authorization row.
`package`	Name of a package that will be brought in by this authorization row and whose authorization items will be overridden by any non-**`NULL`** values from this authorization row.
`widget`	Name of a widget that will be authorized by this authorization row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization row will take effect. If set to **`NULL`** then "now" is used when inserting a new row.
`valid_to`	Timestamp when the authorization row will be expired/disabled. **`NULL`** if no valid_to time have been set.

GET /authorizations/log

List the log of changes done to all authorizations.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.
`after_id`	Only include log items that have a `id` larger than the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the authorization row was created in this event, '**`modify`**' if the authorization row was modified in this event and '**`delete`**' if the authorization row was deleted in this event.
`replaces`	The id of the original authorization row that this authorization row replaces if event is '**`modify`**' or '**`delete`**'.
`id`	The id of the authorization row.
`userid`	The id of the user **[numerical user id]**.
`comment`	Free-text comment of the authorization row.
`package`	Name of a package that will be brought in by this authorization row and whose authorization items will be overridden by any non-**`NULL`** values from this authorization row.
`widget`	Name of a widget that will be authorized by this authorization row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`trial`	**`Y`** if this authorization row is part of a trial period, **`N`** if not.

`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization row will take effect.
`valid_to`	Timestamp when the authorization row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the authorization row **[numerical client id]**.
`last_modified`	Timestamp when the authorization row was modified

GET /authorizations/{userid}

List the authorizations of the user identified by userid.

Reply elements:

Element	*Description*
`id`	The id of the authorization row.
`userid`	The id of the user **[numerical user id]**.
`comment`	Free-text comment of the authorization row.
`package`	Name of a package that will be brought in by this authorization row and whose authorization items will be overridden by any non-**`NULL`** values from this authorization row.
`widget`	Name of a widget that will be authorized by this authorization row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`trial`	**`Y`** if this authorization row is part of a trial period, **`N`** if not.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization row will take effect.
`valid_to`	Timestamp when the authorization row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the authorization row **[numerical client id]**.
`last_modified`	Timestamp when the authorization row was modified

GET /authorizations/{userid}/log

List the log of changes done to the authorizations for the user identified by userid.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.
`after_id`	Only include log items that have a `id` larger than the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the authorization row was created in this event, '**`modify`**' if the authorization row was modified in this event and '**`delete`**' if the authorization row was deleted in this event.
`replaces`	The id of the original authorization row that this authorization row replaces if event is '**`modify`**' or '**`delete`**'.
`id`	The id of the authorization row.
`userid`	The id of the user **[numerical user id]**.
`comment`	Free-text comment of the authorization row.
`package`	Name of a package that will be brought in by this authorization row and whose authorization items will be overridden by any non-**`NULL`** values from this authorization row.
`widget`	Name of a widget that will be authorized by this authorization row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`trial`	**`Y`** if this authorization row is part of a trial period, **`N`** if not.

`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization row will take effect.
`valid_to`	Timestamp when the authorization row will be expired/disabled. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that modified the authorization row **[numerical client id]**.
`last_modified`	Timestamp when the authorization row was modified

POST /authorizations/{userid}

Modify the authorizations of the user identified by userid.

The post-data must contain the complete authorization of the user (absent rows will be seen as deletes) and can only be in either JSON or XML since urlencoded does not support arrays.

Supported parameters:

Element	*Description*
`id`	The id of the authorization row. Must be **`NULL`** or absent for inserts.
`comment`	Free-text comment of the authorization row.
`package`	Name of a package that will be brought in by this authorization row and whose authorization items will be overridden by any non-**`NULL`** values from this authorization row.
`widget`	Name of a widget that will be authorized by this authorization row.
`marketplace`	Filter the authorization on this Marketplace (insref).
`submarket`	Filter the authorization on this Submarket (insref).
`list`	Filter the authorization on this List (insref).
`company`	Filter the authorization on this Company (insref).
`fund_company`	Filter the authorization on this Fund Company (insref).
`insref`	Filter the authorization on this Insref.
`mclass`	Filter the authorization on these mclasses, array of integers.
`instrument_types`	Filter the authorization on these Instrument Types, array of integers.
`ca_types`	Filter the authorization on these Corporate Action types, array of integers.
`ci_types`	Filter the authorization on these Company Information types, array of integers.
`news_packages`	Filter the authorization on these News Packages, array of integers.
`hist_years`	Number of allowed years for historical requests.
`estimates_source`	Filter the authorization on this source for Estimates (string with max 15 chars).
`qps`	Allowed Queries Per Second.
`qpm`	Allowed Queries Per Month.
`delay`	Allow requests for this delay: '**`realtime`**', '**`delay`**', '**`end-of-day`**', '**`next-day`**' or '**`t+1`**'.
`trial`	**`Y`** if this authorization row is part of a trial period, **`N`** if not.
`push`	**`Y`** if push requests are allowed, **`N`** if not.
`valid_from`	Timestamp when the authorization row will take effect. If set to **`NULL`** then "now" is used when inserting a new row.
`valid_to`	Timestamp when the authorization row will be expired/disabled. **`NULL`** if no valid_to time have been set.

GET /feed_nodes

List the feed-nodes that MAAS listens to in order to maintain the instruments database for the authorizations.

Supported query parameters:

Parameter	*Description*
`expired`	If included and set to any other value other than "0" then expired feed-nodes will be included in the reply.

Reply elements:

Element	*Description*
`uri`	The URI of the feed-node
`zone`	
`node`	
`created`	Timestamp when the feed-node was created.
`valid_to`	Timestamp when the feed-node will be expired. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that last modified the feed-node **[numerical client id]**.
`last_modified`	Timestamp when the feed-node was modified.

GET /feed_nodes/log

List the log of changes done to the list of feed-nodes.

Supported query parameters:

Parameter	*Description*
`client`	Only include log items that have been modified by this client **[numerical client id]**.
`after`	Only include log items that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`event`	Contains '**`add`**' if the feed-node was created in this event, or '**`modify`**' if the feed-node was modified.
`uri`	The URI of the feed-node
`zone`	
`node`	
`created`	Timestamp when the feed-node was created.
`valid_to`	Timestamp when the feed-node will be expired.. **`NULL`** if no valid_to time have been set.
`client_modified`	The client that last modified the feed-node **[numerical client id]**.
`last_modified`	Timestamp when the feed-node was modified.

POST /feed_nodes

Add or edit a feed-node.

The creation of a new feed-node (add) will be indicated by a 201 reply while the edit of an existing feed-node will be indicated by a 204 reply.

When creating a new feed-node, the `uri`, `zone` and `node` parameters are mandatory. On edit, unchanged parameters can be omitted from the request.

Supported parameters:

Element	*Description*
`uri`	The URI of the feed-node to add or edit. Maximum 191 characters.
`zone`	
`node`	
`valid_to`	Timestamp when the feed-node will be expired. **`NULL`** if no valid_to time have been set.

GET /insrefs

List the available allocation ranges of insrefs that MAAS can assign to feed-handlers.

Supported query parameters:

Parameter	*Description*
`after`	Only include rows that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`from`	Start value of the allocation range.
`to`	End value of the allocation range.
`deleted`	Boolean indicator of whether or not the allocation range is exhausted. Will be either "`true`" or "`false`".
`last_modified`	Timestamp when the allocation range was modified.

POST /insrefs

Add a new allocation range of insrefs that MAAS can assign to feed-handlers.

Supported parameters:

Element	*Description*
`from`	Start value of the allocation range.
`to`	End value of the allocation range.

GET /stats

Return the monthly statistics variables.

Supported query parameters:

Parameter	*Description*
`userid`	Only include the stats for this user **[numerical user id]**.
`key`	Only include stats for the supplied key/name.
`period`	Only include stats for the supplied period, must be in format "YYYY-MM".
`client`	Only include stats that have been modified by this client **[numerical client id]**.
`after`	Only include stats that have a `last_modified` after the supplied value.

Reply elements:

Element	*Description*
`userid`	The id of the user **[numerical user id]**
`key`	The key/name of the monthly statistic.
`period`	The monthly period in “YYYY-MM” format.
`value`	The value of the monthly statistic.
`client_modified`	The client that last modified the monthly statistic **[numerical client id]**.
`last_modified`	Timestamp when the monthly statistic was modified.

POST /stats

Update a monthly statistic variable.

The creation of a new statistic will be indicated by a 201 reply while the update of an existing statistic will be indicated by a 204 reply.

The unique key of a statistic is userid+key+period and the value can be either set directly to a new value or be increased/decreased by sending a delta value.

Supported parameters:

Element	*Description*
`userid`	The id of the user **[numerical user id]** for whom the monthly statistic should be updated.
`key`	The key/name of the monthly statistic to update. Keys starting with “maas.” are internally managed keys by MAAS and cannot be edited by clients. Maximum 64 characters (that can use the full UTF-8 universe).
`period`	The monthly period to update, format must by “YYYY-MM”.
`value`	The new value to set the monthly statistic to.
`delta`	The monthly statistic will be updated by this value, if positive the value will increase and if negative the value will decrease. If the statistic is created with a delta then the value will be set to that of the delta.