EMM REST API

Last Update 20 January 2023

/api/tokens/generatetoken (GET/POST)

Description: Generate session token for passed credentials. API requests must then present. GET displays form, POST to submit request.

HTTP Headers

None

Request Parameters

f	Response format. Default html, or json
username	
password	

Response

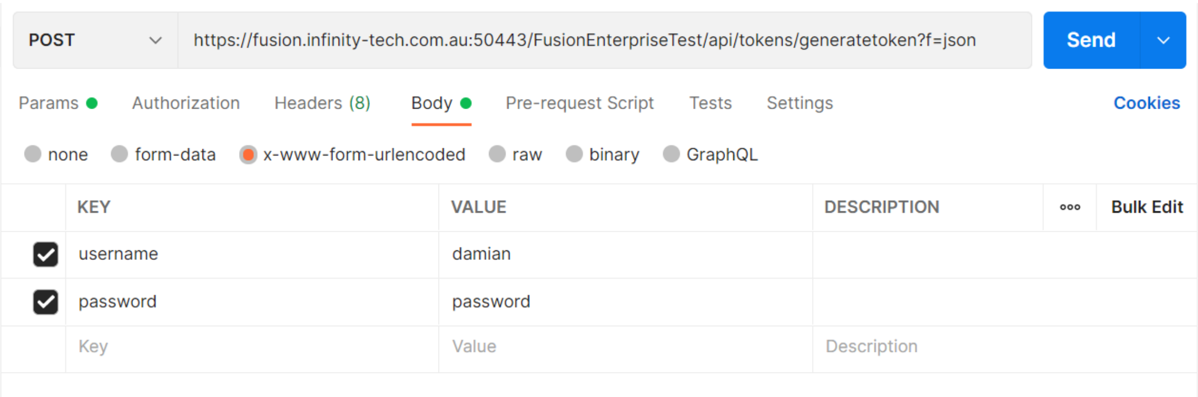
200	{
	“token”: “<token>”
	}

200	`{"error":{"api":"generatetoken","description":"<reason>"}}`

Implementation Notes

License option Max API Sessions.

Postman Example



POST https://fusion.infinity-tech.com.au:50443/FusionEnterpriseTest/api/tokens/generatetoken?f=json

	KEY	VALUE	DESCRIPTION
☑	username	damian	
☑	password	password	
	Key	Value	Description

/api/tokens/releaseToken (GET/POST)

Description: Release current token.

HTTP Headers

Authorization: Bearer <token>

/api/settings/webpush (GET)

Description: Retrieve WebPush VAPID public key.

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json

Response

200	{
	“vapid_publickey”: “<Base64 DER encoded public key>”
	}

200	

/api/recipients (GET)

Description: List recipients with optional pagination and filtering.

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
offset	Optional start offset
limit	Optional, required when start offset specified
filterField	Optional Filter by “type”, “name”, “local_id” search
filterValue	Optional “contact” “clint” “1”
orderby	Optional sort by “name”

Response

200	{
	"offset": <offset>,
	"count": <count this results>,
	"total": <total results>,
	"recipients":
	[
	{
	"id": <id>,
	"name": "<recipient name>",
	"type": <recipient type>
	},
	{
	…
	}
	]
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/recipients/{id} (GET)

Description: Retrieve recipients with specific id

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
id	Id of recipient to return

Response

200	{
	  "id": <id>,
	  "name": "<recipient name>",
	  "type": <recipient type>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users (GET)

Description: Return list of users

HTTP Headers

Authorization: Bearer <token>

Request Parameters

/api/users/{userid}/events/send (POST)

Description: Current user sends event (message) to gateway

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]

event	`"text": "message content",`
	`"priority": <0/1>`
	`"tone": <[A..Z]>`
	`"recipients": <csv id’s>`
	`"reply_message_id": <optional reply id>`

Response

200	`{`
	`  "message_id": <message id>`
	`}`

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events (GET)

Description: Current user list events (inbox)

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
offset	Optional start offset
limit	Optional, required when start offset specified
filterField	Optional Filter by “gateway”, “name” search
filterValue	Optional “clint*”

orderby	Optional order by
	`MQ.Priority`
	`MQ.QueueTM`
	`I.Status`
	`MQ.ThreadID`
	`MQ.FromUserID`
	`MQ.Msg`
	
	`Append DESC to reverse order`

Response

200	JSON list
	`{`
	`  "offset": <offset>,`
	`  "count": <count this results>,`
	`  "total": <total results>,`
	`  "events":`
	`  [`
	`    {`
	`      "id": <user inbox id>,`
	`      "message_id": <system message id>,`
	`      "queue_time": <64bit epoch timestamp>,`
	`      "message": "<message content>",`
	`      "status": <queued/in progress/complete>,`
	`      "status_text": “Status text”,`
	`      "status_time": <64bit epoch timestamp>`
	`      "current_cycle": <non zero is escalating message>,`
	`      "priority": <true/false>,`
	`      "thread_id": <unique id for replied messages>,`
	`      "from_user_id": <if received from user>,`
	`      "from_gateway_id": <if received via gateway>,`
	`      "from_address": “address”`
	`    },`
	`    {`
	`            …`
	`    }`
	`  ]`

	}
403	Forbidden – invalid bearer token
400	Bad request

```
enum
{
        MSGSTATUS_ERROR_SEND = 0,
        MSGSTATUS_QUEUED = 1,
        MSGSTATUS_DISPATCHED = 2,
        MSGSTATUS_SENT = 3,
        MSGSTATUS_RECEIVED = 4,
        MSGSTATUS_READ = 5,
        MSGSTATUS_FORCEESCALATE = 6,  // control status commands
        MSGSTATUS_ACKNOWLEDGE = 7,
        MSGSTATUS_CANCELALARM = 8,

        //inbox message status
        MSGSTATUS_REPLIED,
        MSGSTATUS_QUEUED_DELETE
};
```

/api/users/self/changes (GET)

Description: Fetch list of changes to state since last call for this user. A user can only retrieve their own state.

Recommend only calling this api minimum once every 30 seconds.

If user has been subscribed to WebPush, changes will not be queued and available via this api.

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json

Response

200	See WebPush JSON response examples
403	Forbidden – invalid bearer token
400	Bad request

/api/users/self/password (POST)

Description: Change current users password

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
Password	String

Response

200	{
	"status": 0 or 1
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{ eventid } (GET)

Description: Current user get event details (inbox)

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Id of event

Response

200	{
	    "id": <user inbox id>,
	    "message_id": <system message id>,
	    "queue_time": <64bit epoch timestamp>,
	    "message": "<message content>",
	    "status": <queued/in progress/complete>,
	    "status_text": “Status text”,
	    "status_time": <64bit epoch timestamp>
	    "current_cycle": <non zero is escalating message>,
	    "priority": <true/false>,
	    "thread_id": <unique id for replied messages>,
	    "from_user_id": <if received from user>,
	    "from_gateway_id": <if received via gateway>,
	    "from_address": “address”
	    "subevents":
	    [
	        {
	            "id": <id>,
	            "status": <queued/in progress/complete>
	            "queue_time": <timestamp>,
	        },
	        {
	            …
	        }
	    ]
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{ eventid }/read (POST)

Description: Current user has read or viewed the event

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Id of event

Response

200	{
	  "id": <id>,
	  "status": <MSGSTATUS_code>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{ eventid }/accept (POST)

Description: Current user accept event

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Id of event

Response

200	{
	  "id": <id>,
	  "status": <MSGSTATUS_code>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{ eventid }/cancel (POST)

Description: Current user cancel alarm associated with event

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Id of event

Response

200	{
	  "id": <id>,
	  "status": <MSGSTATUS_code>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{ eventid }/escalate (POST)

Description: Current user escalate alarm associated with event to next

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Id of event

Response

200	`{`
	`  "id": <id>,`
	`  "status": <MSGSTATUS_code>`
	`}`

403	Forbidden – invalid bearer token
400	Bad request

/api/users/{userid}/events/{eventid}/delete (POST)

Description: Current user delete event

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
userid	User Id [self/id]
eventid	Event Id of event

Response

200	{
	  "id": <id>,
	  "status": <MSGSTATUS_code>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/self/webpush/subscribe (GET,POST)

Description: Assign client push manager subscription details to current user. Or get if user has been subscribed and the time.

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
endpoint	Endpoint url of push server subscribed to by client
p256dh	Client push manager public key
auth	Client push manager secret token

Response

200	{
	  "status": <1= successfully subscribed, 0= error>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/users/self/webpush/unsubscribe (GET,POST)

Description: Remove client push manager subscription from current user.

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json

Response

200	{
	  "status": <1= successfully subscribed, 0= error>
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/alarms/active (GET)

Description: Current user list events (alarms)

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
offset	Optional start offset
limit	Optional, required when start offset specified
filterField	Optional Filter by “gateway”, “name” search
filterValue	Optional “clint*”
orderby	Optional sort by

Response

<table>
<tr><td>200</td><td>JSON list
<pre>
{
  "offset": <offset>,
  "count": <count this results>,
  "total": <total results>,
  "alarms":
  [
    {
      "id": <id>,
      "alarm-name ": "<name>",
      "activation-text": "<message content>",
      "reset-text": "<message content>",
      "status": <queued/in progress/complete>,
      "duration": <duration in seconds>,
      "recipient": <recipient id>,
      "priority": <0/1>
    },
    {
            …
    }
  ]
}
</pre></td></tr>
<tr><td>403</td><td>Forbidden – invalid bearer token</td></tr>
<tr><td>400</td><td>Bad request</td></tr>
</table>

/api/alarms/active/{id} (GET)

Description: Get alarm details

HTTP Headers

Authorization: Bearer <token>

Request Parameters

F	Response format. Default html, or json
id	Id of event

Response

200	{
	  "id": <id>,
	  "alarm-name ": "<name>",
	  "activation-text": "<message content>",
	  "reset-text": "<message content>",
	  "status": <queued/in progress/complete>,
	  "duration": <duration in seconds>,
	  "recipient": <recipient id>,
	  "priority": <0/1>
	},
	{
	  …
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/alarms/active/{id}/cancel (POST)

Description: Current user delete event

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
id	Id of event

Response

200	{
	  "id": <id>,
	  "status": "success/fail"
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/contacts (GET)

Description: Find contacts with optional pagination options.
Results sorted by name

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
offset	Optional start offset
limit	Optional, required when start offset specified
filterField	Optional Filter by “gateway”, “name” search
filterValue	Optional “clint*”
orderby	Optional sort by

Response

200	{
	  "offset": <offset>,
	  "count": <count this results>,
	  "total": <total results>,
	  "contacts":
	  [
	    {
	      "id": <id>,
	      "name": "<contact name>",
	      "gateway": <gateway id>,
	      "fieldx": "<valuex>",
	    },
	    {
	
	    }
	  ]
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/contacts/{id} (GET, POST)

Description: Retrieve contact with specific id

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
id	Id of contact to return (GET) or edit (POST)

contact	(POST only) JSON contact values to update
	{
	  "name": "<contact name>",
	  "gateway": <gateway id>,
	  "fieldx": "<valuex>",
	}

Response

200	{
	  "id": <id>,
	  "name": "<contact name>",
	  "gateway": <gateway id>,
	  "fieldx": "<valuex>",
	}

403	Forbidden – invalid bearer token
400	Bad request

/api/contacts/{id}/delete (POST)

Description: Delete contact with specific id

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json
id	(POST only) Id of recipient to delete (POST)

Response

200	{
	  "id": <id>,
	  "status": "success/in-use"
	  "in-use":
	  [
	    {
	      "name": "xx"
	    }
	  ]
	}

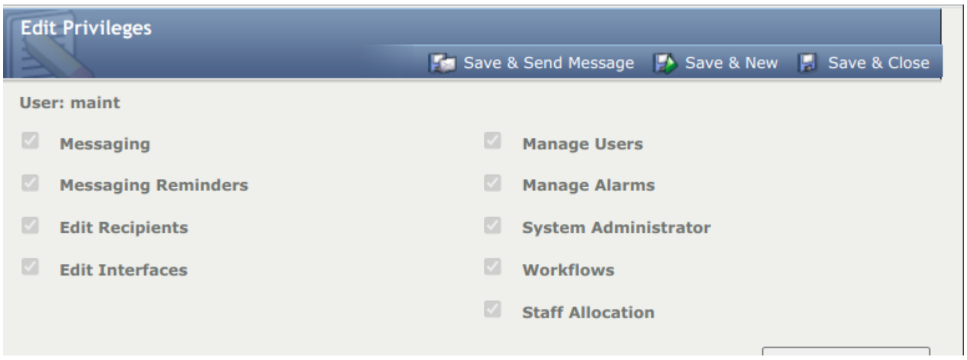
403	Forbidden – invalid bearer token
400	Bad request

/api/user/self (GET, POST)

/api/user/{id} (GET, POST)

Description: Get set user id / current user properties

If permission allows, the privileges can be set for user



User: maint

☑ Messaging

☑ Messaging Reminders

☑ Edit Recipients

☑ Edit Interfaces

☑ Manage Users

☑ Manage Alarms

☑ System Administrator

☑ Workflows

☑ Staff Allocation

HTTP Headers

Authorization: Bearer <token>

Request Parameters

f	Response format. Default html, or json

settings	(POST only) JSON contact values to update
	{
	"x": "<x value>",
	"y": "<y value>",
	}

Response

200	{
	"x": "<x value>",
	"y": "<y value>",
	}

403	Forbidden – invalid bearer token
400	Bad request

Web Push Notifications

User Event Notification

{"notifications":[{"type":"user_event","notification":{"id":1,"message_id":1,"queue_time":1664924064,"message":"11111111111111","escalation_cycle":0,"escalation_step":0}}]}

```
        writer.StartObject();
                writer.Key("notifications");
                writer.StartArray();
                        writer.StartObject();
                        writer.Key("type");
                        writer.String("user_event");
                        writer.Key("notification");
                                writer.StartObject();
                                // match APIUserEvent JSON key names
                                writer.Key("id");
                                writer.Int(m_UserInboxMessageID);
                                writer.Key("message_id");
                                writer.Int(m_MessageID);
                                writer.Key("queue_time");
                                writer.Int64((int64_t)m_ctTimeQueued);
                                writer.Key("message");
                                writer.String(m_strMsg);
                                writer.Key("escalation_cycle");
                                writer.Int(m_nCurrentCycle);
                                writer.Key("escalation_step");
                                writer.Int(m_nSourceEscalationStep);
                                writer.EndObject();
                        writer.EndObject();
                writer.EndArray();
        writer.EndObject();
```

```
{
    "notifications": [{
            "type": "user_event",
            "notification": {
                "id": 1,
                "message_id": 1,
                "queue_time": 1674146027,
                "message": "1111111111",
                "escalation_cycle": 0,
                "escalation_step": 0,
                "priority": false
            }
        } {
            "type": "user_event",
            "notification": {
                "id": 2,
                "message_id": 2,
                "queue_time": 1674146043,
                "message": "22222222222",
                "escalation_cycle": 0,
                "escalation_step": 0,
                "priority": false
            }
        } {
            "type": "user_event",
            "notification": {
                "id": 3,
                "message_id": 3,
                "queue_time": 1674146207,
                "message": "333333333",
                "escalation_cycle": 0,
                "escalation_step": 0,
                "priority": false
            }
        }
    ]
}
```

Active Alarm

```
writer.StartObject();
        writer.Key("notifications");
        writer.StartArray();
                writer.StartObject();
                        writer.Key("type");
                        writer.String("active_alarm");
                        writer.Key("notification");
                        writer.StartObject();
                                // match APIActiveAlarm JSON key names
                                writer.Key("id");
                                writer.Int(m_ID);
                                writer.Key("alarm_id");
                                writer.Int(m_AlarmID);
                                writer.Key("cleared");
                                writer.Bool(m_fCleared);        // alarm has been reset
                                writer.Key("active_time");
                                writer.Int64(m_AlarmActiveTime);
                                writer.Key("active_duration");
                                writer.Int64(m_AlarmActiveDuration);
                                writer.Key("msg_id");
                                writer.Int(m_MsgID);
                                writer.Key("reset_msg_id");
                                writer.Int(m_ResetMsgID);       // alarm has been reset
                                writer.Key("source_gateway_id");
                                writer.Int(m_GWID);
                                writer.Key("source_gateway_name");
                                writer.String(<gateway name>);
                                writer.Key("name");
                                writer.String(m_strName);
                                writer.Key("identifier");
                                writer.String(m_strIdentifier);
                                writer.Int(m_recipientID);
                                writer.Key("recipient_name");
                                if (m_recipientID == 0)
                                {
                                        writer.String("System");
                                }
                                else
                                {
                                        writer.String(<recipient name>);
                                }
                                writer.Key("active_text");
                                writer.String(m_strActiveMsg);
                                writer.Key("reset_text");
                                writer.String(m_strResetMsg);
                                writer.Key("priority");
                                writer.Bool(m_fPriority);
                                writer.Key("send_resets");
                                writer.Bool(m_fSendResets);
                                writer.Key("total_messages");
                                writer.Int(0);
                                writer.Key("total_messages_sent");
                                writer.Int(0);
                                writer.Key("total_messages_read");
                                writer.Int(0);
                        writer.EndObject();
                writer.EndObject();
        writer.EndArray();
writer.EndObject();
```