Linewize
by Family Zone

TECHNICAL GUIDE

Linewize School Manager API Documentation

Linewize

Contents

© Family Zone Pty Ltd 2020

Linewize

Key Concepts

Linewize School Manager employs a restful HTTP API following industry standards for cloud services. API requests are made using HTTP to our API endpoint secured using HTTPS TLS.

Currently we do not publish a SDK so integrators are free to use a language and toolset of their choosing as long as it supports HTTPS and standard HTTP 1.1. Responses are provided in JSON and most POST and PUT requests to the API expect JSON.

Device Identifier

Most API requests with Linewize School Manager rely on the notion of a “deviceid”. The “deviceid” is an organizational level arbitrary identifier for a network appliance. This identifier is configured on the Linewize Appliance and is used as a key identifier for configuration and statistics in the cloud.

Normally when making API requests to the Linewize School Manager cloud platform, you will specify the “deviceid” as part of the request URI.

Example: fw1.jacksonvilleprimary.school

It is safe to assume that this identifier will remain static for the lifetime of a customer regardless of hardware, so this identifier is static.

Account

External API access depends on the requester holding an account with Linewize School Manager. This account can be used to login to the Linewize School Manager cloud platform located at https://cloud.linewize.net and when presented with a API Access Keyset, the Cloud API as well.

Accounts hold permissions to manage and view statistics for particular devices identified by the “deviceid” detailed above. It is important not to confuse an account in Linewize School Manager with a network identity. Accounts in Linewize School Manager are used for management purposes, whereas network identities identify a particular user on the network.

Authentication

API usage requires a API access keyset and an authentication session token.

The API access keyset is provided by Linewize School Manager on request and the authentication session token is acquired by authenticating with a username and password using the keyset. Authentication session tokens have a limited lifetime and must be refreshed periodically to maintain a high level of security.

Example Request:

```
GET /devices?access_token=session_token HTTP 1.0
Hostname: api2.linewize.net
Authorization: Basic <encoded keyset>
```

The Authorization HTTP header is the base64 encoded product of the access key ID and access key secret. The access_token query parameter is the session token acquired by authenticating against the API using the process below.

Authentication Life-cycle Example

The first step in any session is to authenticate against the API with your account. This authentication step will provide a session token for future access.

```
POST /authenticate
{"username": "jack.sparrow@awesome.com", "password":"pirates4life"}
```

```
curl
-H "Content-Type: application/json"
-H "Authorization: Basic c3VxZpp"
-X POST
-d '{"username":"jack.sparrow@awesome.com","password":"pirates4life"}' https://api2.
linewize.net/authenticate
```

```
Response:
{
  "result": {
    "id": 1,
    "token": "8db63181-9d6c-44fa-8dad-1ff1ecc764b0",
    "two_factor": false
  }
}
```

If the authentication request was successful, you can make subsequent requests using that token.

```
GET /account/1?access_token=<access_token>
```

```
curl
-H "Content-Type: application/json"
-H "Authorization: Basic c3VxZpp"
-X GET
https://api2.linewize.net/account?token=x
```

```
Response:
{
  "result": {
    "admin": false,
    "customerid": 1,
    "demo_account": false,
    "email": "jack.sparrow@awesome.com",
    "enabled": true,
    "msp": "",
    "phone_number": "",
    "support_admin": false,
    "two_factor": false
  }
}
```

Linewize

Error Handling

Error handling in the Linewize School Manager API is handled two-fold. Firstly, if an error occurs you will not receive a response with a 200 http status code. This should be you first indication of an error. If you do not receive a status code matching 200 then there will be an error object returned in the response as JSON.

For example:

```
HTTP/1.1 401 UNAUTHORIZED
Content-Type: application/json
Request-Id: b0b33eb0-79df-48fd-819d-2b85909981be
Content-Length: 353
Connection: keep-alive

{
  "error": {
    "causing_application": "apidispatcher",
    "device_id": null,
    "exception": "You do not have a valid user session",
    "from_url": "https://api2.linewize.net/account/161?access_token=x ",
    "message": null,
    "stack_trace": null,
    "to_status_code": null,
    "to_url": null
  }
}
```

Versioning

Linewize School Manager currently maintains one version of our external API. Methods are periodically added to this API for new functionality and enhancements. If fundamental changes are made to our API we will provide 60 day's notice to all parties using the API.

Status and Uptime

The Linewize School Manager API is a 'first class citizen' - all our management front-ends depend on the API and have no direct access to internals or database systems.

We strive for 99.95% uptime, though periodically we have outages and scheduled maintenance.

Linewize

Conventions and Assumptions

From this point onwards, we assume that you are following the authentication cycle as detailed above and requests conform to normal HTTP standards.

Assumptions

- The Authorisation parameter with all API requests
- The “access_token” query parameter is included with all API requests
- Requests are made using HTTPS
- The current API endpoint is api2.linewize.net

General Management

When you authenticate with Linewize School Manager, the access session token will be granted for access to a group of devices. This group of devices is tied to the account that you authenticate with. Most end users will only have one device, whereas integrators and MSP’s will have access to multiple devices.

To retrieve a list of available devices, you can use the “/account/<account_id>/permissions” method.

```
GET /account/<account_id>/permissions
```

```
Account_id: This is a numeric identified provided in the authentication response
```

```
Response:
{
  “result”: [
    {
      “accountid”: “161”,
      “deviceid”: “tideturner.primary.school”
    },
    {
      “accountid”: “161”,
      “deviceid”: “landlubber.high.school”
    }]
}
```

Devices that are listed as part of this query will be accessible under your session.

Statistical Queries

As a crucial part of Linewize School Manager’s value proposition, we collect and aggregate detailed statistics on network usage. This information provides the basis for our cloud based products, but also can be queried and integrated into our products using our API.

Linewize

Standard Query Parameters

For most statistical queries, there are a static set of query parameters that can be supplied.

startDate	%Y-%m-%d	2019-01-01
endDate	%Y-%m-%d	2019-01-07
limit	Number	10000
offset	Number	100
include_noise	Boolean	true

Transfer totals

Get the total data transferred on a daily basis for a period.

```
GET /device/<device_id>/statistics/transfer
```

```
device_id: The requested device
All general parameters listed above can be used
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-13",
      "download": 2807928091,
      "upload": 400231837
    },
    {
      "day": "2017-08-14",
      "download": 1695149498,
      "upload": 233010370
  ],
  "use_decimal": true
}
```

Applications and Websites

Linewize School Manager abstracts out applications and websites into what is known as signatures. Signatures at the highest level are categories and at the lowest level are complex heuristics for individual applications.

This approach results in two different result entries in our types and websites API call: individual websites that are not part of a low level signature, and applications or websites.

In the example resultset provided, you have Youtube traffic which is represented as as a low level signature "sphirewall.application.youtube" and "scdn.co" which is a individual website but categorized as part of a CDN.

Get a list of top applications and websites:

```
GET /device/<device_id>/statistics/typesandwebsites
```

```
device_id: The requested device
All general parameters listed above can be used
```

```
Response:
{
  "result": [
    {
      "all_domains": [
        "youtube.com",
        "ytimg.com",
        "googlevideo.com"
      ],
      "application": "sphirewall.application.youtube",
      "categoryId": "sphirewall.application.artsandentertainment",
      "day": "2017-08-12",
      "domain": null,
      "download": 3797959804,
      "hits": 854,
      "subCategoryId": "sphirewall.application.media",
      "tag": "sphirewall.application.youtube",
      "transfer": 71294976,
      "upload": 154581374,
      "website": null
    },
    {
      "application": "sphirewall.application.cdnandcloud",
      "categoryId": "sphirewall.application.internetandtelecom",
      "day": "2017-08-12",
      "domain": "scdn.co",
      "download": 762393113,
      "hits": 5915,
      "subCategoryId": "sphirewall.application.cdnandcloud",
      "tag": "sphirewall.application.cdnandcloud",
      "transfer": 784317254,
      "upload": 21918226,
      "website": "scdn.co"
    }]
}
```

Get Users of a particular application:

```
GET /device/<device_id>/statistics/types/<type>/users
```

```
device_id: The requested device
type: The traffic type signature identifier. This can be an individual application,
categoryId or subCategoryId.
All general parameters listed above can be used
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 15072684,
      "transfer": 16948988,
      "upload": 1875211,
      "user": "charlotte.belt"
    },
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 7787366,
      "transfer": 8047122,
      "upload": 259668,
      "user": "helen.johnson"
    }]
}
```

Get Users of a particular website:

```
GET /device/<device_id>/statistics/website /<website >/users
```

```
device_id: The requested device
type: The website domain
All general parameters listed above can be used
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 13404187,
      "transfer": 14910530,
      "upload": 1506050,
      "user": "charlotte.belt"
    },
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 1900772,
      "transfer": 1987176,
      "upload": 86390,
      "user": "john.hogg@linewizedemo.com"
    }]
}
```

Get a drilldown to individual access of a website by a user

Individual access times can be provided using the Timeline routes. Timeline routes provide a individual connection by connection log of internet activity for users over a short period of time.

Users centric statistics

When users are authenticated on a network, stats can be queried centric to that user.

Get top users by data transfer

```
GET /device/<device_id>/statistics/users
```

```
device_id: The requested device
All general parameters listed above can be used
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 245989728,
      "transfer": 271216190,
      "upload": 25221454,
      "user": "maria.jakeman@linewizedemo.com"
    },
    {
      "day": "2017-08-12",
      "day_time": "2017-08-12T12:00:00.000Z",
      "download": 96102629,
      "transfer": 111777714,
      "upload": 15669695,
      "user": "charlotte.belt"
    }]
}
```

Get top websites and applications for an individual user

Returning top apps and websites for a user employs the same query you were introduced to in the section above. The difference being the use of the "filter_user" parameter.

```
GET /device/<device_id>/statistics/typesandwebsites
```

```
device_id: The requested device
filter_user: The user you wish to query for
```

```
Response:
As referenced in the section above.
```

Get dates a user accessed a website or application

The typesandwebsites query provides dates of access for individual websites and applications, but you can also query this in a user centric way for an individual website or application.

This query is faster than using the typesandwebsites query above and for an individual user and website or application we recommend using it.

```
GET /device/<device_id>/statistics/<website>/<username>/times
```

```
device_id: The requested device
username: The user you wish to query for
website: The website domain you wish to query for
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-01",
      "download": 522883,
      "upload": 162022
    },
    {
      "day": "2017-08-17",
      "download": 121004,
      "upload": 39232
    }
  ],
  "use_decimal": true
}
```

```
GET /device/<device_id>/statistics/type/<type>/<username>/times
```

```
device_id: The requested device
username: The user you wish to query for
type: The signature id or category you wish to query for
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-01",
      "download": 522883,
      "upload": 162022
    },
    {
      "day": "2017-08-17",
      "download": 121004,
      "upload": 39232
    }
  ],
  "use_decimal": true
}
```

Get a detailed access timeline

Our lowest level of data points are individual connections.

Individual connections offer more data about access, and provide an absolute time when the access was made. Due to the nature of network access on devices, the volume of this data is high and so we restrict query width over time. Timeline queries can only be run on an individual day and you must specify a start and end time.

```
GET /device/<device_id>/statistics/user/<username>/time
```

```
device_id: The requested device
user: The requested username
startDate: The date this query will be run on. For example 2012-01-02
startTime: The time this query will start at. For example 00:50
endTime: The time this query will end at: For example 06:00

All general params listed above can be used
```

```
Response:
{
  "result": [
       {
     "app_filtering_denied": false,
     "categoryId": "sphirewall.application.internetandtelecom",
     "contenttype": "image/gif",
     "destIp": "67.226.210.15",
     "destPort": 80,
     "download": 719,
     "final_connection_object": true,
     "fingerprint": "sphirewall.fingerprint.genericlinux",
     "geoip_destination": "US",
     "geoip_source": "",
     "httpHost": "dt.videohub.tv",
     "http_request_uris": [
       "GET /v1/usync/tt?userId=8335663417512805508 HTTP/1.1"
     ],
     "hwAddress": "00:50:56:85:4e:48",
     "inputDev": "br0",
     "noise": true,
     "outputDev": "br0",
     "protocol": 6,
     "referer": "http://www.huffingtonpost.com/",
     "sourceHostname": "unknown",
     "sourceIp": "192.168.179.112",
     "sourcePort": 56770,
     "string_time": "2017/08/17 11:37:41",
     "subCategoryId": "sphirewall.application.advertisingandtrackers",
     "tag": "sphirewall.application.advertisingandtrackers",
     "time": 1502926661,
     "upload": 653,
     "user": "helen.johnson",
     "useragent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/59.0.3071.115 Safari/537.36",
     "verdict_filter_rule": "13f18659-b7f3-4faa-a76d-7e1d72469d42"
   },
   {
     "app_filtering_denied": false,
     "categoryId": "sphirewall.application.artsandentertainment",
     "contenttype": "text/plain; charset=utf-8",
     "destIp": "52.7.227.229",
```

```
      "destPort": 80,
      "download": 1262,
      "final_connection_object": true,
      "fingerprint": "sphirewall.fingerprint.genericlinux",
      "geoip_destination": "US",
      "geoip_source": "",
      "httpHost": "stg-data-collector.playbuzz.com",
      "http_request_uris": [
        "POST /mcd/events HTTP/1.1",
        "POST /mcd/events HTTP/1.1",
        "POST /mcd/events HTTP/1.1"
      ],
      "hwAddress": "00:50:56:85:4e:48",
      "inputDev": "br0",
      "noise": false,
      "outputDev": "br0",
      "protocol": 6,
      "referer": "http://www.huffingtonpost.com/",
      "sourceHostname": "unknown",
      "sourceIp": "192.168.179.112",
      "sourcePort": 41668,
      "string_time": "2017/08/17 11:36:48",
      "subCategoryId": "",
      "tag": "sphirewall.application.artsandentertainment",
      "time": 1502926608,
      "upload": 3020,
      "user": "helen.johnson",
      "useragent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/59.0.3071.115 Safari/537.36",
      "verdict_filter_rule": "13f18659-b7f3-4faa-a76d-7e1d72469d42"
    }]
}
```


Hosts

Identical data can be queried for by network host rather than by user. This allows the same level of granularity on network access to be presented even when users are not authenticated on the network.

For details on these routes we can customise documentation for you, please contact our team.

Filtering violations

Regardless of whether a filtering rule was violated or triggered, Linewize School Manager keeps comprehensive network access records as you have seen above. Identifying filtering violations can be done using the API methods below.

With the exception of the detailed connection timeline query, filtered access is omitted from general statistics. This is to prevent confusion for the end user.

Get total violations count for policies

Filtering violations are centered around a policy. You can get violations by policy using the following query.

```
GET /device/<device_id>/webfiltering/violations/policies
```

```
device_id: The requested device
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "hits": 212,
      "policy": "200f0456-6c5f-4d9e-b965-3b9fe33c37e8"
    }
  ],
  "use_decimal": true
}
```

Get top offenders

To get a list of top offenders with disregard to policy, you can use the following query.

```
GET /device/<device_id>/webfiltering/violations/users
```

```
device_id: The requested device
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "hits": 152,
      "user": "lizzie.crawford@linewizedemo.com"
    },
    {
      "day": "2017-08-12",
      "hits": 60,
      "user": "charlotte.belt"
    }
  ],
  "use_decimal": true
}
```

Find which users violated a policy

Once you have identified a policy that has been offended, you can drill down into the individual users who have violated that policy.

```
GET /device/<device_id>/webfiltering/violations/policy/<policy_id>/users
```

```
device_id: The requested device
policy_id: The policy requested
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "hits": 152,
      "user": "lizzie.crawford@linewizedemo.com"
    },
    {
      "day": "2017-08-12",
      "hits": 60,
      "user": "charlotte.belt"
    }
  ],
  "use_decimal": true
}
```

Find which policies a user has violated

Conversely to the query above, you can also query for the policies that a particular user has violated.

```
GET /device/<device_id>/webfiltering/violations/user/<user>/policies
```

```
device_id: The requested device
user: The user requested
```

```
Response:
{
  "result": [
    {
      "day": "2017-08-12",
      "hits": 60,
      "policy": "200f0456-6c5f-4d9e-b965-3b9fe33c37e8"
    }
  ],
  "use_decimal": true
}
```

Get a timeline of violations for a user

Much like the connection timeline for Users detailed above, an absolute timeline of user violations can also be retrieved.

```
GET /device/<device_id>/webfiltering/violations/user/<user>/individual
```

```
device_id: The requested device
user: The user requested
```

```
Response:
{
  "result": [
    {
      "app_filtering_denied": true,
      "categoryId": "sphirewall.application.offensive",
      "contenttype": "",
      "destIp": "66.115.170.140",
      "destPort": 443,
      "download": 13361,
      "final_connection_object": true,
      "fingerprint": "sphirewall.fingerprint.genericlinux",
      "geoip_destination": "US",
      "geoip_source": "",
      "httpHost": "onlytease.com",
      "http_request_uris": [],
      "hwAddress": "00:50:56:85:b6:66",
      "inputDev": "br0",
      "noise": false,
      "outputDev": "br0",
      "protocol": 6,
      "referer": "",
      "sourceHostname": "unknown",
      "sourceIp": "192.168.179.111",
      "sourcePort": 47132,
      "string_time": "2017/08/17 11:21:32",
      "subCategoryId": "sphirewall.application.porn",
      "tag": "sphirewall.application.porn",
      "time": 1502925692,
      "upload": 305,
      "user": "charlotte.belt",
      "useragent": "",
      "verdict_application_rule": "200f0456-6c5f-4d9e-b965-3b9fe33c37e8",
      "verdict_application_rule_fire_event": false,
      "verdict_filter_rule": "13f18659-b7f3-4faa-a76d-7e1d72469d42"
    },
    {
      "app_filtering_denied": true,
      "categoryId": "sphirewall.application.offensive",
      "contenttype": "",
      "destIp": "66.115.170.140",
      "destPort": 443,
      "download": 13361,
      "final_connection_object": true,
      "fingerprint": "sphirewall.fingerprint.genericlinux",
      "geoip_destination": "US",
      "geoip_source": "",
      "httpHost": "onlytease.com",
      "http_request_uris": [],
      "hwAddress": "00:50:56:85:b6:66",
```

```
    "inputDev": "br0",
    "noise": false,
    "outputDev": "br0",
    "protocol": 6,
    "referer": "",
    "sourceHostname": "unknown",
    "sourceIp": "192.168.179.111",
    "sourcePort": 47136,
    "string_time": "2017/08/17 11:21:33",
    "subCategoryId": "sphirewall.application.porn",
    "tag": "sphirewall.application.porn",
    "time": 1502925693,
    "upload": 305,
    "user": "charlotte.belt",
    "useragent": "",
    "verdict_application_rule": "200f0456-6c5f-4d9e-b965-3b9fe33c37e8",
    "verdict_application_rule_fire_event": false,
    "verdict_filter_rule": "13f18659-b7f3-4faa-a76d-7e1d72469d42"
  }]
}
```

Linewize

Events

Linewize School Manager details different events and triggers for users, devices and for general network and appliance events. Events are dynamic in structure and can be used to debug network issues, identify user signings and much more.

Note: Events are only retained in the system for 2 weeks.

On large networks, we do not recommend wildcard querying of events as the data volumes are high and queries will take a long time to return if at all. For the purpose of this document, we will just focus on user-centric events.

Get a list of Events for a user

Querying for events is done using one route with different filtering parameters.

```
GET /device/<device_id>/events
```

```
device_id: The requested device
username: Username that we are filtering on.
```

```
Response:
{
  "result": [
    {
      "day": "Thu, 17 Aug 2017 02:00:00 GMT",
      "deviceid": "bridge.demo1.linewize",
      "ip": "192.168.179.111",
      "key": "event.userdb.session.timeout",
      "time": "2017-08-17 14:38:40",
      "user": "charlotte.belt"
    },
    {
      "absolute_timeout": 1502938593,
      "day": "Thu, 17 Aug 2017 01:00:00 GMT",
      "deviceid": "bridge.demo1.linewize",
      "idle_timeout": 1800,
      "ip": "192.168.179.111",
      "key": "event.userdb.auth.success",
      "mac": "00:50:56:85:b6:66",
      "provider": "api.linewizecp.chrome",
      "time": "2017-08-17 13:56:33",
      "type": "192.168.179.111",
      "user": "charlotte.belt"
    }]
}
```

Summary

This documentation offers some common uses for the Linewize School Manager API, which should lay a solid foundation for possibilities. Our team are available to support specific requests, and can provide customised documentation.

About Linewize

Linewize is committed to making student Internet management easy, and keeping students safe online on any device, anywhere, any time.

Learn more

Email sales@familyzone.com
Visit us at linewize.com