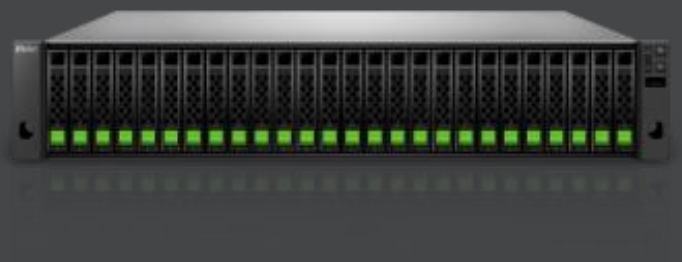
XCubeFAS XEVO RESTful API Software Manual

Applicable Models:
XF2026D



QSAN Technology, Inc.
www.QSAN.com

QSAN

August 2019

This edition applies to QSAN XCubeFAS XEVO (an all-flash array operating system). QSAN believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

Notices

This XCubeFAS XEVO user's manual is applicable to the following XCubeFAS models:

XCubeFAS Storage System 2U 19" Rack Mount Model

Model Name	Controller Type	Form Factor, Bay Count, and Rack Unit
XF2026D	Dual Controller	SFF 26-disk 2U Chassis

Information contained in this manual has been reviewed for accuracy. But it could include typographical errors or technical inaccuracies. Changes are made to the document periodically. These changes will be incorporated in new editions of the publication. QSAN may make improvements or changes in the products. All features, functionality, and product specifications are subject to change without prior notice or obligation. All statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Table of Contents

Figures

Tables

Preface

About This Manual

This manual provides technical guidance for designing and implementing QSAN XCubeFAS series AFA (All-Flash Array) system, and it is intended for use by system administrators, SAN designers, storage consultants, or anyone who has purchased these products and is familiar with servers and computer networks, network administration, storage system installation and configuration, storage area network management, and relevant protocols.

Related Documents

There are related documents which can be downloaded from the website.

- All XCubeFAS Documents
- XCubeFAS QIG (Quick Installation Guide)
- XCubeFAS Hardware Manual
- XCubeFAS XEVO Software Manual
- Compatibility Matrix
- White Papers
- Application Notes

Technical Support

Do you have any questions or need help trouble-shooting a problem? Please contact QSAN Support, we will reply to you as soon as possible.

- Via the Web: https://www.qsan.com/technical_support
- Via Telephone: +886-2-77206355
  (Service hours: 09:30 - 18:00, Monday - Friday, UTC+8)
- Via Skype Chat, Skype ID: qsan.support
  (Service hours: 09:30 - 02:00, Monday - Friday, UTC+8, Summer time: 09:30 - 01:00)
- Via Email: support@qsan.com

Information, Tip and Caution

This manual uses the following symbols to draw attention to important safety and operational information.

INFORMATION:
INFORMATION provides useful knowledge, definition, or terminology for reference.

TIP:
TIP provides helpful suggestions for performing tasks more effectively.

CAUTION:
CAUTION indicates that failure to take a specified action could result in damage to the system.

Conventions

The following table describes the typographic conventions used in this manual.

Conventions	Description
Bold	Indicates text on a window, other than the window title, including menus, menu options, buttons, fields, and labels. Example: Click the **OK** button.
<*Italic*>	Indicates a variable, which is a placeholder for actual text provided by the user or system. Example: copy <*source-file*> <*target-file*>.
[ ] square brackets	Indicates optional values. Example: [a | b] indicates that you can choose a, b, or nothing.
{ } braces	Indicates required or expected values. Example: { a | b } indicates that you must choose either a or b.
| vertical bar	Indicates that you have a choice between two or more options or

	arguments.
/ Slash	Indicates all options or arguments.
underline	Indicates the default value. Example: [a | b]

1. RESTful API Overview

QSAN meets the trend of the times and supports the RESTful API which is becoming the most popular API design specification.

1.1. What is RESTful API

A RESTful API is based on REST (REpresentational State Transfer) technology, an architectural style and approach to communications often used in web services development. A RESTful API is an API (Application Program Interface) that uses HTTP requests to GET, POST, PUT, and DELETE data.

RESTful API design is designed to take advantage of existing protocols. It usually takes advantage of HTTP when used for Web APIs. This means that developers do not need to install libraries or additional software in order to take advantage of a REST API design. REST API is notable for its incredible layer of flexibility and leverages less bandwidth, making it more suitable for internet usage. Since data is not tied to methods and resources, REST has the ability to handle multiple types of calls, return different data formats and even change structurally with the correct implementation of hypermedia.

1.2. How RESTful API Work

A RESTful API breaks down a transaction to create a series of small modules. Each module addresses a particular underlying part of the transaction. This modularity provides developers with a lot of flexibility, but it can be challenging for developers to design from scratch.

A RESTful API explicitly takes advantage of HTTP methodologies. They use GET to retrieve a resource; PUT to change the state of or update a resource, which can be an object, file or block; POST to create that resource; and DELETE to remove it.

2. Getting Started with RESTful API

There are some useful API tools to test before developing an application using the RESTful API. API testing is a software testing type which focuses on the determination if the developed APIs meet expectations regarding the functionality, reliability, performance, and security of the application. The following provides some API testing tools that testing teams can select to suit your needs.

Postman

Website: https://www.getpostman.com/

Postman is an easy-to-use REST client and rich interface which makes it easy to use.

SoapUI

Website: https://www.soapui.org/

SoapUI is a headless functional testing tool dedicated to API testing, allowing users to test REST and SOAP APIs and Web Services easily.

Katalon Studio

Website: https://www.katalon.com

Katalon Studio is a free test automation tool for API, Web and Mobile applications.

3. RESTful API Sets

This chapter is to help you find an API by name. Each API topic includes one or more of the following sections:

API The Application Program Interface

Description The API's purpose and note about usage

Parameters Descriptions of API's parameters

Example One or more examples of API's usage

The usage of each API is described in the following.

3.1. Login

API

POST <baseurl>/rest/login/account/<$username>/password/<$password>

Description

Login the system.

Parameters

Table 3-1 Login Parameters

Name	Type	Description
$username	string	Login account
$password	string	Login password

Example

For security reasons, you have to login the system for authentication before you can execute other RESTful APIs.

```
POST http://<IP or FQDN>/rest/login/account/admin/password/1234

 ------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

3.2. Help

API

GET <baseurl>/rest/

Description

Show all APIs.

Parameters

None.

Example

Display all RESTfull APIs.

```
GET http://<IP or FQDN>/rest

 ------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "Invalid param",
    "data": [
        "====================GET====================",
        "/rest/hwmonitor/enc",
        "/rest/hwmonitor/enc/$enclosure id",
        "/rest/hwmonitor/enc/$enclosure id/ssd",
        "/rest/hwmonitor/enc/$enclosure id/ctrl",
        "/rest/hwmonitor/enc/$enclosure id/ctrl/$controller id",
        "/rest/hwmonitor/enc/$enclosure id/psu",
        "/rest/hwmonitor/enc/$enclosure id/fan",
        "/rest/hwmonitor/ctrl/$controller id/bbm",
        "/rest/hwmonitor/flash",
        "/rest/hwmonitor/sysInfo",
        "/rest/hwmonitor/temp",
        "/rest/hwmonitor/ssd",
        "/rest/dashboard/latency",
        "/rest/dashboard/iops",
```



```
        "/rest/dashboard/throughput",
        "/rest/dashboard/arrayCapacity",
        "/rest/storage/pool",
        "/rest/storage/pool/$pool id",
        "/rest/storage/pool/$pool id/volume",
        "/rest/storage/pool/$pool id/volume/$volume id",
        "/rest/storage/volume",
        "/rest/host/hostlist",
        "====================GET====================",
        "",
        "",
        "====================POST====================",
        "/rest/login/account/$username/password/$password",
        "/rest/storage/pool/create",
        "/rest/storage/pool/delete",
        "/rest/storage/pool/addDisk",
        "/rest/storage/volume/create",
        "/rest/storage/volume/delete",
        "/rest/storage/volume/rename",
        "/rest/storage/volume/delete_by_pool",
        "/rest/host/create",
        "/rest/host/delete",
        "/rest/host/connectVolumes",
        "/rest/host/disconnectVolumes",
        "====================POST====================="
    ]
}
```

3.3. Hardware Monitor

This section includes the following API sets.

GET

- **enc:** Show enclosure information.
- **sysInfo:** Show system information.
- **ctrl:** Show controller information.
- **ssd:** Show SSD (Solid State Drive) information.
- **psu:** Show PSU (Power Supply Unit) information.
- **fan:** show fan information.
- **bbm:** Show BBM (Battery Backup Module) information.
- **flash:** Show flash module information.
- **temp:** Show temperature information.

3.3.1. Enclosure Information

API

GET <baseurl>/rest/hwmonitor/enc

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>

Description

Show all or specific enclosure information.

Parameters

Table 3-2 Enclosure Parameters

Name	Type	Description
$enclosure id	integer	Enclosure ID

Example

Display all enclosure information.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "enc_name": "0 (Head Unit: XF2026)",
            "enc_id": 0
        }
    ]
}
```

Display the enclosure information for the specific enclosure ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
```



```
    "data": [
        {
            "enc_name": "0 (Head Unit: XF2026)",
            "enc_id": 0
        }
    ]
}
```


3.3.2. System Information

API

GET <baseurl>/rest/hwmonitor/sysInfo

Description

Show system information.

Parameters

None.

Example

Display the basic information of the system.


```
GET http://<IP or FQDN>/rest/hwmonitor/sysInfo

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": {
        "Info": {
            "sys_name": "XCubeFAS",
            "model_name": "XF2026",
            "SN": "QW3160160I0xxxxxx",
            "ctrl_status": "Normal",
            "master_ctrl": "vg_ctr1"
        },
        "Ctrl_1": {
            "sn": "5001378008xxxxxx",
            "cpu": "Intel(R) Processor D-1500 4 Cores",
            "mem": "32 GB",
            "host_card_1": "4 x 10GbE iSCSI (SFP+)",
            "host_card_2": "empty",
            "fw_ver": "1.1.0  (build 201907xxxxxx) ",
            "ioc_ver": "07.00.01.00",
            "exp_ver": 1000
        },
```

```
        "Ctrl_2": {
            "sn": "5001378008xxxxxx",
            "cpu": "Intel(R) Processor D-1500 4 Cores",
            "mem": "32 GB",
            "host_card_1": "4 x 10GbE iSCSI (SFP+)",
            "host_card_2": "empty",
            "fw_ver": "1.1.0  (build 201907xxxxxx) ",
            "ioc_ver": "07.00.01.00",
            "exp_ver": 1000
        },
        "Bpl": {
            "BSN": "001378xxxxxx",
            "BID": "QW226",
            "mcu_ver": "1.2.1"
        }
    }
}
```


3.3.3. Controller Information

API

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>/ctrl

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>/ctrl/<$controller id>

Description

Show both or specific controller information in the enclosure.

Parameters

Table 3-3 Controller Parameters

Name	Type	Description
$enclosure id	integer	Enclosure ID
$controller id	integer	Controller ID

Example

Display the information of both controllers.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0/ctrl

------------------------------------ RESULT ------------------------------------

{
    "success": true,
```



```
    "msg": "",
    "data": [
        {
            "status": "Normal",
            "ctr_id": 0,
            "sn": "5001378008xxxxxx",
            "cpu": "Intel(R) Processor D-1500 4 Cores",
            "memory": "32 GB",
            "host_card_1": null,
            "host_card_2": null,
            "fw_ver": "1.1.0  (build 201907xxxxxx) ",
            "ioc_ver": "07.00.01.00",
            "exp_ver": 1000,
            "real_status": "OPERATIONAL"
        },
        {
            "status": "Normal",
            "ctr_id": 1,
            "sn": "5001378008xxxxxx",
            "cpu": "Intel(R) Processor D-1500 4 Cores",
            "memory": "32 GB",
            "host_card_1": null,
            "host_card_2": null,
            "fw_ver": "1.1.0  (build 201907xxxxxx) ",
            "ioc_ver": "07.00.01.00",
            "exp_ver": 1000
            "real_status": "OPERATIONAL",
        }
    ]
}
```


Display the controller information for the specific controller ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0/ctrl/0

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": {
        "status": "Normal",
        "ctr_id": 0,
        "sn": "5001378008xxxxxx",
        "cpu": "Intel(R) Processor D-1500 4 Cores",
        "memory": "32 GB",
        "host_card_1": null,
        "host_card_2": null,
        "fw_ver": "1.1.0  (build 201907xxxxxx) ",
        "ioc_ver": "07.00.01.00",
        "exp_ver": 1000,
        "real_status": "OPERATIONAL"
    }
}
```

3.3.4. SSD Information

API

GET <baseurl>/rest/hwmonitor/ssd

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>/ssd

Description

Show all SSD information in all enclosures or in a specific enclosure.

Parameters

Table 3-4 Enclosure Parameters

Name	Type	Description
$enclosure id	integer	Enclosure ID

Example

Display all SSD information in all enclosures.

```
GET http://<IP or FQDN>/rest/hwmonitor/ssd

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3451001459,
            "key": 3451001459,
            "slot": 1,
            "sizeMB": 381298,
            "sizeGB": 372,
            "raid_set": "N/A",
            "status": "Online",
            "health": "Good",
            "usage": "FR",
            "caching_owner": "N/A",
            "vendor": "SEAGATE",
            "sed": "None",
            "sed_status": "",
            "ssd": "Yes",
            "serial": "Z3F010RY0000Z3F010RY",
            "spinrate": 1,
            "model": "ST400FM0053",
            "fw_ver": 6,
            "rate": "SAS SSD    12.0Gb/s",
            "wce": "Enabled",
```



```
            "idletimer": "Disabled",
            "readahead": "Enabled",
            "cmdque": "Enabled",
            "enc_real_number": 0
        },
        {
            "id": 3121749056,
            "key": 3121749056,
            "slot": 2,
            "sizeMB": 381298,
            "sizeGB": 372,

            . . . (continue)

        },
        {
            "id": 2258039913,
            "key": 2258039913,
            "slot": 3,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },
        {
            "id": 2925818167,
            "key": 2925818167,
            "slot": 4,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },
        {
            "id": 480894376,
            "key": 480894376,
            "slot": 5,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },
        {
            "id": 2217841664,
            "key": 2217841664,
            "slot": 6,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        }
    ]
}
```

Display all SSD information for the specific enclosure ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0/ssd

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3451001459,
            "key": 3451001459,
            "slot": 1,
            "sizeMB": 381298,
            "sizeGB": 372,
            "raid_set": 1,
            "status": "Online",
            "health": "Good",
            "usage": "RD",
            "caching_owner": "N/A",
            "vendor": "SEAGATE",
            "sed": "None",
            "sed_status": "",
            "ssd": "Yes",
            "serial": "Z3F010RY0000Z3F010RY",
            "spinrate": 1,
            "model": "ST400FM0053",
            "fw_ver": 6,
            "rate": "SAS SSD    12.0Gb/s",
            "wce": "Enabled",
            "idletimer": "Disabled",
            "readahead": "Enabled",
            "cmdque": "Enabled",
            "enc_real_number": 0
        },

        . . . (continue)

    ]
}
```

3.3.5. PSU Information

API

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>/psu

Description

Show all PSU information in the enclosure.

Parameters

Table 3-5 PSU Parameters

Name	Type	Description
$enclosure id	integer	Enclosure ID

Example

Display all PSU information for the specific enclosure ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0/psu

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "name": "PSU1",
            "status": "Normal",
            "item": {
                "Type": "Power Supply",
                "Item": "PSU1",
                "Loc.": "BPL",
                "Status": "OK",
                "Error": "OK",
                "Curr_Value": "N/A",
                "Low_Warn": "",
                "High_Warn": "",
                "Low_Cri": "",
                "High_Cri": ""
            }
        },
        {
            "name": "PSU2",
            "status": "Normal",
            "item": {
                "Type": "Power Supply",
                "Item": "PSU2",
                "Loc.": "BPL",
                "Status": "OK",
                "Error": "OK",
                "Curr_Value": "N/A",
                "Low_Warn": "",
                "High_Warn": "",
                "Low_Cri": "",
                "High_Cri": ""
            }
        }
    ]
}
```

3.3.6. Fan Information

API

GET <baseurl>/rest/hwmonitor/enc/<$enclosure id>/fan

Description

Show all fan information in the enclosure.

Parameters

Table 3-6 Fan Parameters

Name	Type	Description
$enclosure id	integer	Enclosure ID

Example

Display all fan information for the specific enclosure ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/enc/0/fan

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "name": "FAN1",
            "status": "Normal",
            "speed": "5421 RPM",
            "item": {
                "Type": "Cooling",
                "Item": "FAN1",
                "Loc.": "BPL",
                "Status": "OK",
                "Error": "OK",
                "Curr_Value": "5421 RPM",
                "Low_Warn": "",
                "High_Warn": "",
                "Low_Cri": "",
                "High_Cri": ""
            }
        },
        {
            "name": "FAN2",
            "status": "Normal",
            "speed": "5273 RPM",
            "item": {
                "Type": "Cooling",
```



```
                "Item": "FAN2",
                "Loc.": "BPL",
                "Status": "OK",
                "Error": "OK",
                "Curr_Value": "5273 RPM",
                "Low_Warn": "",
                "High_Warn": "",
                "Low_Cri": "",
                "High_Cri": ""
            }
        },

        . . . (continue)

    ]
}
```

3.3.7. BBM Information

API

GET <baseurl>/rest/hwmonitor/ctrl/<$enclosure id>/bbm

Description

Show BBM information.

Parameters

Table 3-7 BBM Parameters

Name	Type	Description
$controller id	integer	Controller ID

Example

Display BBM information for the specific controller ID 0.

```
GET http://<IP or FQDN>/rest/hwmonitor/ctrl/0/bbm

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": {
        "status": "Normal"
    }
```

```
}
```

3.3.8. Flash Information

API

GET <baseurl>/rest/hwmonitor/flash

Description

Show flash module information.

Parameters

None.

Example

Display flash module information.

```
GET http://<IP or FQDN>/rest/hwmonitor/flash

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": "Normal"
}
```

3.3.9. Temperature Information

API

GET <baseurl>/rest/hwmonitor/temp

Description

Show all temperature information for the system.

Parameters

None.

Example

Display the temperature of the system components.

```
GET http://<IP or FQDN>/rest/hwmonitor/temp

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": {
        "CPU": {
            "tempature": "+56.0 (C)",
            "status": "normal"
        },
        "SAS Expander": {
            "tempature": "+61.7 (C)",
            "status": "normal"
        },
        "Location Left": {
            "tempature": "+27.2 (C)",
            "status": "normal"
        },
        "Location Middle": {
            "tempature": "+30.3 (C)",
            "status": "normal"
        },
        "Location Right": {
            "tempature": "+24.1 (C)",
            "status": "normal"
        }
    }
}
```

3.4. Dashboard

This section includes the following API sets.

GET

- **arrayCapacity:** Show total capacity of the array.
- **latency:** Show array latency for performance.
- **iops:** Show array IOPS (Input / Output Per Second) for performance.
- **throughput:** Show array throughput for performance.

3.4.1. Array Capacity

API

GET <baseurl>/rest/dashboard/arrayCapacity

Description

Show total capacity of the array.

Parameters

None.

Example

Display that the total capacity of the array is 3,813,960 MB of the array.

```
GET http://<IP or FQDN>/rest/dashboard/arrayCapacity

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": 3813960
}
```

3.4.2. Array Latency

API

GET <baseurl>/rest/dashboard/latency

Description

Show array latency for performance.

Parameters

None.

Example

Display the array latency. The unit is ms (millisecond).

```
GET http://<IP or FQDN>/rest/dashboard/latency

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        51.05,
        50.69,
        51.08,

        . . . (continue)

        44.02
    ]
}
```

3.4.3. IOPS Information

API

GET <baseurl>/rest/dashboard/iops

Description

Show array IOPS for performance.

Parameters

None.

Example

Display the array IOPS.

```
GET http://<IP or FQDN>/rest/dashboard/iops

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        2838,
```

```
        2839,
        2842,

        . . . (continue)

        2659
    ]
}
```

3.4.4. Throughput Information

API

GET <baseurl>/rest/dashboard/throughput

Description

Show array throughput for performance.

Parameters

None.

Example

Display the array throughput. The unit is MB/s.

```
GET http://<IP or FQDN>/rest/dashboard/throughput

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        1419.99,
        1421.94,
        1419.69,

        . . . (continue)

        1319.75
    ]
}
```


3.5. Storage

This section includes the following API sets.

GET

- **pool:** Show pool information.
- **volume:** Show volume information.

POST

- **pool_create:** Create a pool.
- **pool_delete:** Delete the pool.
- **pool_addDisk:** Add a disk group into the pool.
- **volume_create:** Create a volume in a pool.
- **volume_delete:** Delete the volume from the pool.
- **volume_rename:** Rename the volume.
- **volume_delete_by_pool:** Delete all volumes in the pool.

3.5.1. Create Pool

API

POST <baseurl>/rest/storage/pool/create

Description

Create a pool.

Parameters

Table 3-8 Create Pool Parameters

Name	Type	Description
$pool_name	string	Pool name
$pd_list	string	SSD list, which is needed to create a pool. The format is [SSD ID, SSD ID, ..., SSD ID].

Example

Create a pool named “Pool_02” with 2 SSDs in slot 3 and 4. First, get the SSD ID information.

```
GET http://<IP or FQDN>/rest/hwmonitor/ssd

------------------------------------ RESULT ------------------------------------


{
    "success": true,
    "msg": "",
    "data": [
        {

        . . . (continue)

        {
            "id": 2258039913,
            "key": 2258039913,
            "slot": 3,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },
        {
            "id": 2925818167,
            "key": 2925818167,
            "slot": 4,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },

        . . . (continue)

        }
    ]
}
```


Enter the parameters of pool_name and pd_list. The following figure is an example of Postman to enter the parameters.

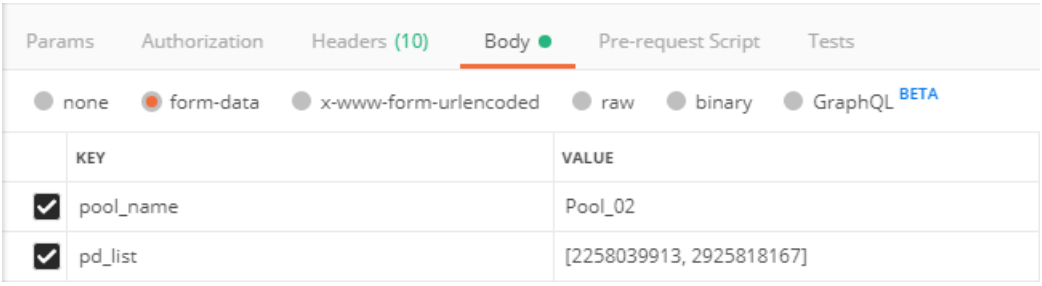


Figure 3-1 Create Pool Parameters

```
POST http://<IP or FQDN>/rest/storage/pool/create

---------------------------------- PARAMETERS -----------------------------------

pool_name = Pool_02
pd_list = [2258039913, 2925818167]

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "LVMERR_SUCCESS",
    "data": []
}
```

3.5.2. Pool Information

API

GET <baseurl>/rest/storage/pool

GET <baseurl>/rest/storage/pool/<$pool id>

Description

Show all or specific pool information.

Parameters

Table 3-9 Pool Parameters

Name	Type	Description
$pool id	integer	Pool ID

Example

Display all pools of the system. Currently, there two pools in the system.

```
GET http://<IP or FQDN>/rest/storage/pool

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3704982456,
            "enc_idx": 0,
            "enc_name": "Local",
            "owner": "CTRL 1",
            "powner": "CTRL 2",
            "hac": "A-A",
            "no": 1,
            "name": "Pool_01",
            "total_size_mb": 381298,
            "free_size_mb": 278897,
            "avail_size_mb": 278897,
            "used_size_mb": 102401,
            "virtual_total_size_mb": 0,
            "virtual_used_size_mb": 0,
            "thin": "Disabled",
            "pd_cnt": 2,
            "udv_cnt": 1,
            "status": "Online",
            "health": "Good",
            "raid": "RAID 1",
            "attributes": 512,
            "encryption": "Disabled",
            "wce": 1,
            "spindown": 0,
            "readahead": 1,
            "cmdque": 1,
            "maid": 0,
            "thin_policy": []
        },
        {
            "id": 3692465081,
            "enc_idx": 0,
            "enc_name": "Local",
            "owner": "CTRL 1",
            "powner": "CTRL 2",
            "hac": "A-A",
            "no": 2,
```



```
            "name": "Pool_02",

            . . . (continue)

        }
    ]
}
```

Display the pool information for the pool name “Pool_02” with the pool ID 3692465081.

```
GET http://<IP or FQDN>/rest/storage/pool/3692465081

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3692465081,
            "enc_idx": 0,
            "enc_name": "Local",
            "owner": "CTRL 1",
            "powner": "CTRL 2",
            "hac": "A-A",
            "no": 1,
            "name": "Pool_02",
            "total_size_mb": 1525682,
            "free_size_mb": 1525682,
            "avail_size_mb": 1525682,
            "used_size_mb": 0,
            "virtual_total_size_mb": 0,
            "virtual_used_size_mb": 0,
            "thin": "Disabled",
            "pd_cnt": 2,
            "udv_cnt": 0,
            "status": "Online",
            "health": "Good",
            "raid": "RAID 0",
            "attributes": 512,
            "encryption": "Disabled",
            "wce": 1,
            "spindown": 0,
            "readahead": 1,
            "cmdque": 1,
            "maid": 0,
            "thin_policy": []
        }
    ]
}
```

3.5.3. Add Disk Group

API

POST <baseurl>/rest/storage/pool/addDisk

Description

Add a disk group into the pool.

Parameters

Table 3-10 Add Disk Group Parameters

Name	Type	Description
pool_id	String	Pool ID
pd_list	String	SSD list, which is needed to add a disk group. The format is [SSD ID, SSD ID, …, SSD ID].

Example

Add a disk group with 2 SSDs in slot 5 and 6 into the pool name “Pool_02” with the pool ID 3692465081. First, get the SSD ID information.

```
GET http://<IP or FQDN>/rest/hwmonitor/ssd

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [

        . . . (continue)

        {
            "id": 480894376,
            "key": 480894376,
            "slot": 5,
            "sizeMB": 762841,
            "sizeGB": 744,

            . . . (continue)

        },
        {
            "id": 2217841664,
            "key": 2217841664,
            "slot": 6,
            "sizeMB": 762841,
```



```
            "sizeGB": 744,

            . . . (continue)

        }
    ]
}
```

Enter the parameters of pool_id and pd_list.

```
POST http://<IP or FQDN>/rest/storage/pool/addDisk

---------------------------------- PARAMETERS -----------------------------------

pool_id = 3692465081
pd_list = [480894376, 2217841664]

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

3.5.4. Delete Pool

API

POST <baseurl>/rest/storage/pool/delete

Description

Delete the pool.

Parameters

Table 3-11 Delete Pool Parameters

Name	Type	Description
pool_id	integer	Pool ID

Example

Delete the pool name “Pool_02” with the pool ID 3692465081.

```
POST http://<IP or FQDN>/rest/storage/pool/delete

----------------------------------- PARAMETERS -------------------------------------

pool_id = 3692465081

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "LVMERR_SUCCESS",
    "data": []
}
```

3.5.5. Create Volume

API

POST <baseurl>/rest/storage/volume/create

Description

Create a volume in a pool.

Parameters

Table 3-12 Create Volume Parameters

Name	Type	Description
volume_name	string	Volume name
pool_id	interger	Pool ID
capacity	integer	Volume capacity in GB

Example

Create a volume named “Volume_02” in the pool name “Pool_01” with the pool ID 3704982456. Its capacity is set to 150GB.

```
POST http://<IP or FQDN>/rest/storage/volume/create

---------------------------------- PARAMETERS -----------------------------------

volume_name = Volume_02
pool_id = 3704982456
capacity = 150

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

3.5.6. Volume Information

API

GET <baseurl>/rest/storage/volume

GET <baseurl>/rest/storage/pool/<$pool id>/volume

GET <baseurl>/rest/storage/pool/<$pool id>/volume/<$volume id>

Description

Show all or specific volume information.

Parameters

Table 3-13 Volume Parameters

Name	Type	Description
$pool id	integer	Pool ID
$volume id	integer	Volume ID

Example

Display all volumes of the system.

```
GET http://<IP or FQDN>/rest/storage/volume

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "rg_id": 3704982456,
            "no": 1,
            "name": "Volume_01",
            "id": 3697832144,
            "has_qreplica": 0,
            "has_clone": 0,
            "clone": "-",
            "target_id": 4294967295,
            "schedule": "N/A",
            "type": "RAID",
            "total_size_mb": 102400,
            "thin_used_size_mb": 102401,
            "thin_avail_size_mb": 0,
            "permission": "WB",
            "priority": "MD",
            "bg_rate": 4,
            "readahead": "Enabled",
            "status": "Online",
            "progress": " ",
            "health": "Optimal",
            "raid": "RAID 1",
            "av_media": "Disabled",
            "reclamation": "Disabled",
            "lun": 0,
            "snap_space_mb": "0 MB/0 MB",
            "snap_total_size_mb": 0,
            "snap_used_size_mb": 0,
            "free_rg_size_mb": 125297,
            "block_size": 4096,
            "usage": "100%"
        },
        {
            "rg_id": 3704982456,
            "no": 2,
            "name": "Volume_02",
            "id": 3699470545,

            . . . (continue)

        }
    ]
}
```


Display all volumes in the pool name “Pool_01” with the pool ID 3704982456.

```
GET http://<IP or FQDN>/rest/storage/pool/3704982456/volume

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3704982456,
            "enc_idx": 0,
            "enc_name": "Local",
            "owner": "CTRL 1",
            "powner": "CTRL 2",
            "hac": "A-A",
            "no": 1,
            "name": "Pool_01",
            "total_size_mb": 381298,
            "free_size_mb": 125297,
            "avail_size_mb": 125297,
            "used_size_mb": 256001,
            "virtual_total_size_mb": 0,
            "virtual_used_size_mb": 0,
            "thin": "Disabled",
            "pd_cnt": 2,
            "udv_cnt": 2,
            "status": "Online",
            "health": "Good",
            "raid": "RAID 1",
            "attributes": 512,
            "encryption": "Disabled",
            "wce": 1,
            "spindown": 0,
            "readahead": 1,
            "cmdque": 1,
            "maid": 0,
            "thin_policy": []
        },
        {
            "rg_id": 3704982456,
            "no": 1,
            "name": "Volume_01",
            "id": 3697832144,

            . . . (continue)

        },
        {
            "rg_id": 3704982456,
            "no": 2,
            "name": "Volume_02",
            "id": 3699470545,

            . . . (continue)

        }
    ]
}
```

Display the volume information for the volume name “Volume_02” with the volume ID 3699470545 in the pool name “Pool_01” with the pool ID 3704982456.

```
GET http://<IP or FQDN>/rest/storage/pool/3704982456/volume/3699470545

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "id": 3704982456,
            "enc_idx": 0,
            "enc_name": "Local",
            "owner": "CTRL 1",
            "powner": "CTRL 2",
            "hac": "A-A",
            "no": 1,
            "name": "Pool_01",

            . . . (continue)

        },
        {
            "rg_id": 3704982456,
            "no": 1,
            "name": "Volume_02",
            "id": 3699470545,

            . . . (continue)

        }
    ]
}
```

3.5.7. Rename Volume

API

POST <baseurl>/rest/storage/volume/rename

Description

Rename the volume.

Parameters

Table 3-14 Rename Pool Parameters

Name	Type	Description
volume_id	interger	Volume ID
new_name	String	New volume name

Example

Rename the volume name “Volume_02” with the volume ID 3699470545 to “New_Volume_02”.

```
POST http://<IP or FQDN>/rest/storage/volume/rename

---------------------------------- PARAMETERS -----------------------------------

volume_id = 3699470545
new_name = New_Volume_02

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

Display all volumes of the system. It renames to “New_Volume_02”.

```
GET http://<IP or FQDN>/rest/storage/volume

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "rg_id": 3704982456,
            "no": 1,
            "name": "New_Volume_02",
            "id": 3699470545,

            . . . (continue)

        },
        {
            "rg_id": 3704982456,
            "no": 2,
            "name": "Volume_01",
```

```
            "id": 3697832144,

            . . . (continue)

        }
    ]
}
```

3.5.8. Delete Volume

API

POST <baseurl>/rest/storage/volume/delete

POST <baseurl>/rest/storage/volume/delete_by_pool

Description

Delete the volume or delete all volumes in the pool.

Parameters

Table 3-15 Delete Volume Parameters

Name	Type	Description
volume_id	integer	Volume ID
pool_id	integer	Pool ID

Example

Delete the volume name “New_Volume_02” with the volume ID 3699470545.

```
POST http://<IP or FQDN>/rest/storage/volume/delete

---------------------------------- PARAMETERS -----------------------------------

volume_id = 3699470545

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```


Delete all volumes in the pool name “Pool_01” with the pool ID 3704982456.

```
POST http://<IP or FQDN>/rest/storage/volume/delete_by_pool

---------------------------------- PARAMETERS -----------------------------------

pool_id = 3700526002

{
    "success": true,
    "msg": "",
    "data": []
}
```

3.6. Hosts

This section includes the following API sets.

GET

- **hostlist:** List all host groups.

POST

- **create:** Create a host group.
- **delete:** Delete the host group.
- **connectVolumes:** Connect volumes into the host group.
- **disconnectVolumes:** Disconnect volumes from the host group.

3.6.1. Create Host Group

API

POST <baseurl>/rest/host/create

Description

Create a host group.

Parameters

Table 3-16 Create Host Group Parameters

Name	Type	Description
host_name	string	Host name
Type	string	Protocol type: [iscsi | FC]

Example

Create a host group named “HostGroup_001” with protocol type iSCSI.

```
POST http://<IP or FQDN>/rest/host/create

---------------------------------- PARAMETERS -----------------------------------

host_name = HostGroup_001
type = iscsi

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

3.6.2. Host List

API

GET <baseurl>/rest/host/hostlist

Description

List all host groups.

Parameters

None.

Example

List all host groups of the system.

```
GET http://<IP or FQDN>/rest/host/hostlist

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "host_name": "HostGroup_001",
            "type": "WITH_ISCSI",
            "chap_flag": "disable",
            "mutual_flag": "disable",
            "allow_hosts": [
                {
                    "dev_name": "Dev(*)",
                    "allow_hosts": "*",
                    "status": "Disconnected"
                }
            ],
            "chap_account": [],
            "mutual_name": "",
            "mutual_pwd": "",
            "target": [
                {
                    "alias": "-",
                    "ctrl": "Ctrl_1",
                    "target_id": 1,
                    "target_name": "iqn.2004-08.com.qsan:xf2026:dev1.ctr1",
                    "port": [
                        0,
                        1
                    ]
                },
                {
                    "alias": "-",
                    "ctrl": "Ctrl_2",
                    "target_id": 65537,
                    "target_name": "iqn.2004-08.com.qsan:xf2026:dev1.ctr2",
                    "port": [
                        0,
                        1
                    ]
                }
            ],
            "mapping": []
        }
    ]
}
```

3.6.3. Connect Volumes

API

POST <baseurl>/rest/host/connectVolumes

Description

Connect volumes into the host group.

Parameters

Table 3-17 Connect Volumes Parameters

Name	Type	Description
host_name	string	Host name
volume_id	string	Volume ID list, which is needed to connect volumes into the host group. The format is [Volume ID, Volume ID, …, Volume ID].

Example

Display all volumes of the system.

```
GET http://<IP or FQDN>/rest/storage/volume

------------------------------------ RESULT ------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "rg_id": 3704982456,
            "no": 1,
            "name": "Volume_01",
            "id": 3695079634,

            . . . (continue)

        },
        {
            "rg_id": 3704982456,
            "no": 2,
            "name": "Volume_02",
            "id": 3704713427,

            . . . (continue)

        }
```



```
    ]
}
```

Connect two volumes with the volume ID 3695079634 and 3704713427 into the host group name “HostGroup_001”.

```
POST http://<IP or FQDN>/rest/host/connectVolumes

---------------------------------- PARAMETERS -----------------------------------

host_name = HostGroup_001
volume_id = [3695079634, 3704713427]

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
}
```

List the host group.

```
GET http://<IP or FQDN>/rest/host/hostlist

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": [
        {
            "host_name": "HostGroup_001",
            "type": "WITH_ISCSI",

            . . . (continue)

            ],
            "mapping": [
                {
                    "vd_name": "Volume_01",
                    "type": "RAID",
                    "lun_id": 0,
                    "capacity": "100.00 GB",
                    "health": "Optimal",
                    "vd_id": 3695079634
                },
                {
                    "vd_name": "Volume_02",
                    "type": "RAID",
                    "lun_id": 1,
                    "capacity": "110.00 GB",
```

```
                    "health": "Optimal",
                    "vd_id": 3704713427
                }
            ]
        }
    ]
}
```

3.6.4. Disconnect Volumes

API

POST <baseurl>/rest/host/disconnectVolumes

Description

Disconnect volumes from the host group.

Parameters

Table 3-18 Disconnect Volumes Parameters

Name	Type	Description
host_name	string	Host name
volume_id	string	Volume ID list, which is needed to disconnect volumes from the host group. The format is [Volume ID, Volume ID, …, Volume ID].

Example

Disconnect two volumes with the volume ID 3695079634 and 3704713427 from the host group name “HostGroup_001”.

```
POST http://<IP or FQDN>/rest/host/disconnectVolumes

---------------------------------- PARAMETERS -----------------------------------

host_name = HostGroup_001
volume_id = [3695079634, 3704713427]

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": "",
    "data": []
```



```
}
```

3.6.5. Delete Host Group

API

POST <baseurl>/rest/host/delete

Description

Delete the host group.

Parameters

Table 3-19 Delete Host Group Parameters

Name	Type	Description
host_name	string	Host name

Example

Delete the host group name “HostGroup_001”.

```
POST http://<IP or FQDN>/rest/host/delete

---------------------------------- PARAMETERS -----------------------------------

host_name = HostGroup_001

------------------------------------ RESULT -------------------------------------

{
    "success": true,
    "msg": 40009,
    "data": []
}
```

4. Support and Other Resources

4.1. Getting Technical Support

After installing your device, locate the serial number on the sticker located on the side of the chassis or from the XEVO -> **System** -> **Maintenance** > **System Information** and use it to register your product at https://www.qsan.com/business_partnership. We recommend registering your product in QSAN partner website for firmware updates, document download, and latest news in eDM. To contact QSAN Support, please use the following information.

- Via the Web: https://www.qsan.com/technical_support
- Via Telephone: +886-2-77206355
  (Service hours: 09:30 - 18:00, Monday - Friday, UTC+8)
- Via Skype Chat, Skype ID: qsan.support
  (Service hours: 09:30 - 02:00, Monday - Friday, UTC+8, Summer time: 09:30 - 01:00)
- Via Email: support@qsan.com

Information to Collect

- Product name, model or version, and serial number
- Operating system name and version
- Firmware version
- Error messages or capture screenshots
- Product-specific reports and logs
- Add-on products or components installed
- Third-party products or components installed

Information for Technical Support

The following system information is necessary for technical support. Please refer to following for what and where to get the information of your XCubeFAS Series model.

If the technical support requests you to download the Service Package, please navigate in the XEVO -> **System** -> **Maintenance** > **System Information**, and then click the **Download Service Package** button to download. Then the system will automatically generate a zip file the default download location of your web browser.

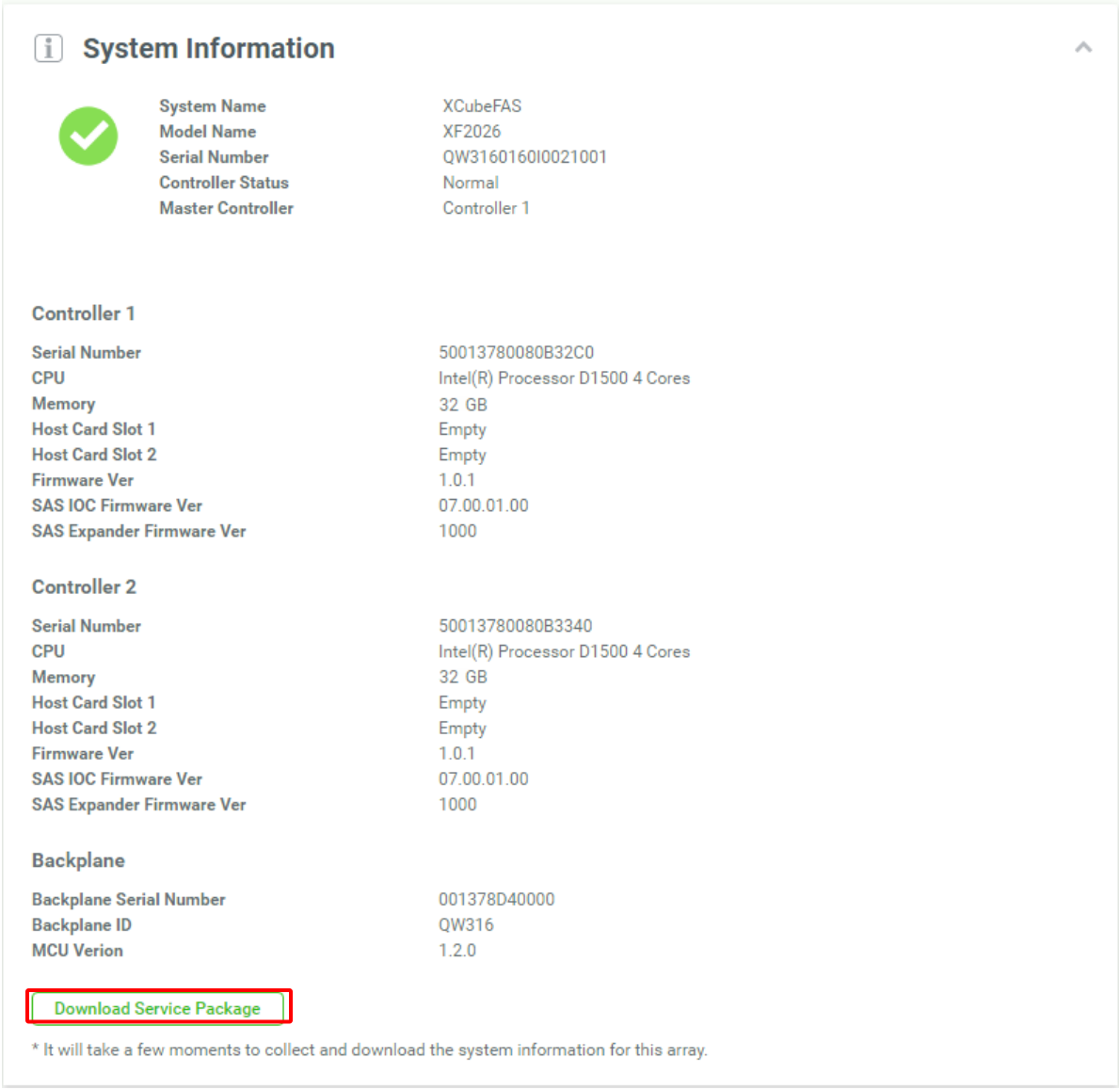


Figure 4-1 Download Service Package

4.2. Online Customer Support

For better customer support, every XCubeFAS series models include the console cable (two for dual controller models), one for single controller models) for online support. Please follow the procedures below to setup the online help environment for QSAN support team.

The following procedure will help you to setup the serial console via the console cable that is enclosed in the shipping carton. The following image is the appearance of the console cable.

Figure 4-2 Appearance of a Console Cable

Procedures to Setup the Serial Console

1. Setup the serial cable between the controller and one server/host like in the below image.

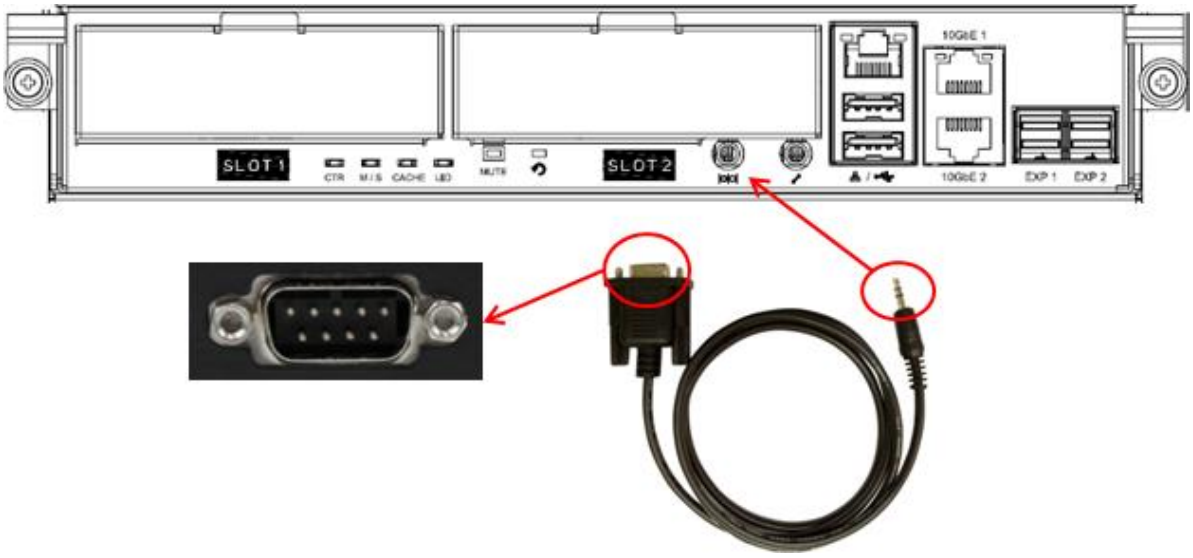


Figure 4-3 Connect the Console Cable

2. You must use terminal software such as HyperTerminal or Putty to open the console after the connection is made.

INFORMATION:
For more information about terminal software, please refer to

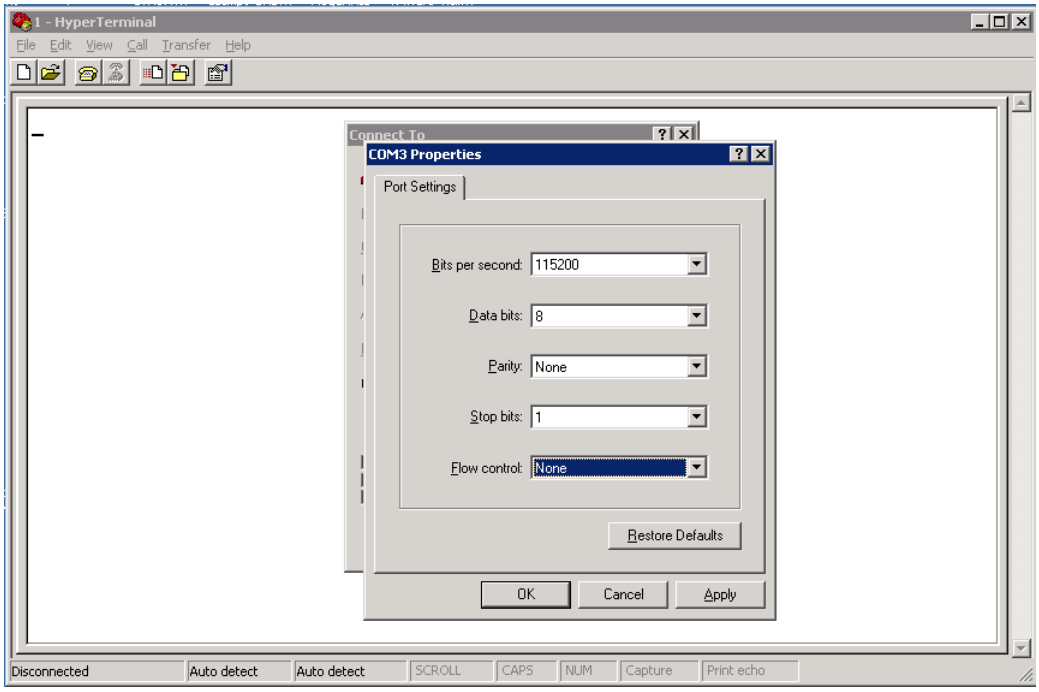
HyperTerminal: http://www.hilgraeve.com/hyperterminal/

PuTTY: http://www.putty.org/

3. Here we first demonstrate HyperTerminal. The console settings are on the following.

   Baud rate: 115200, 8 data bit, no parity, 1 stop bit, and no flow control

   Terminal type: vt100



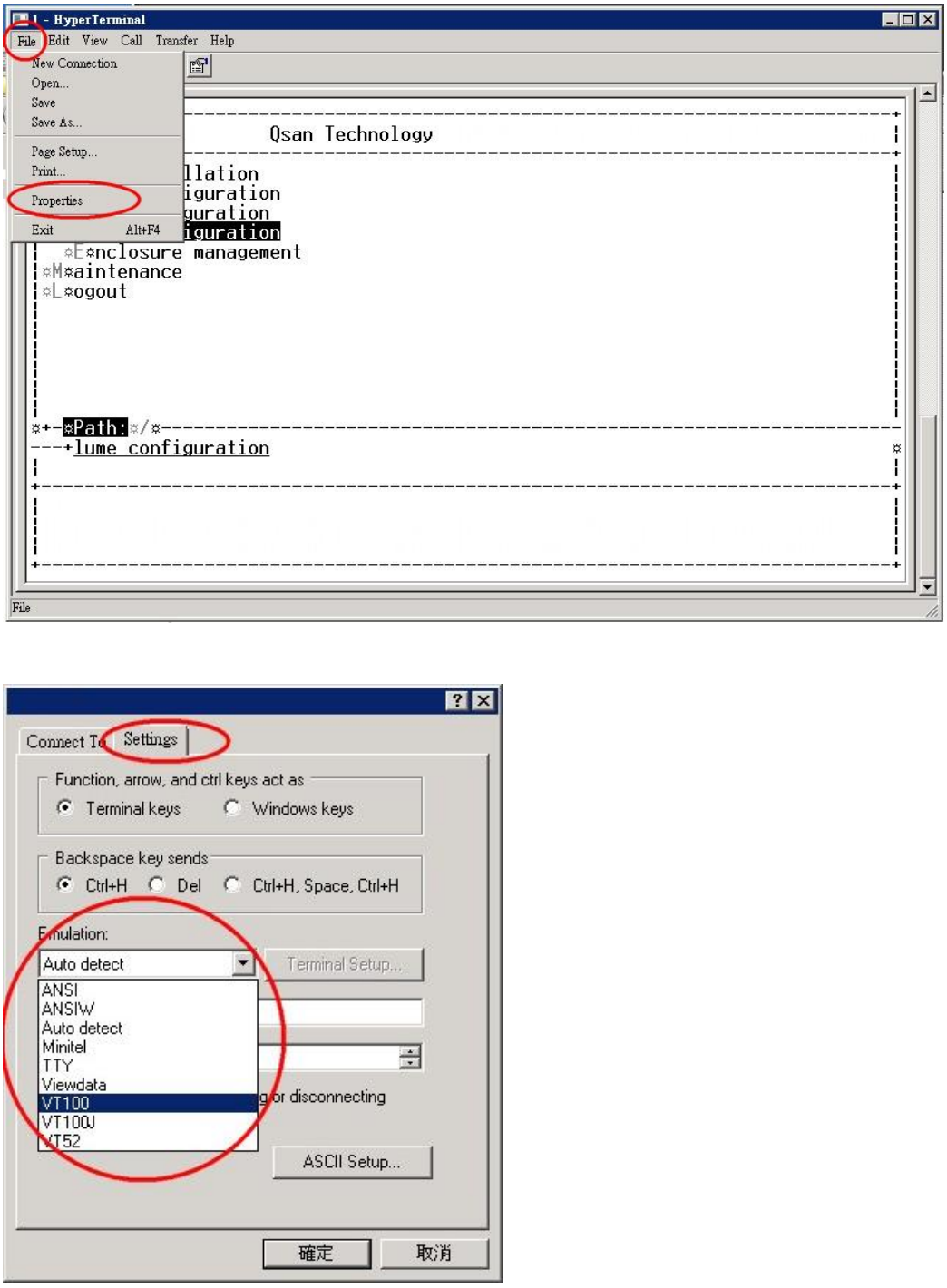
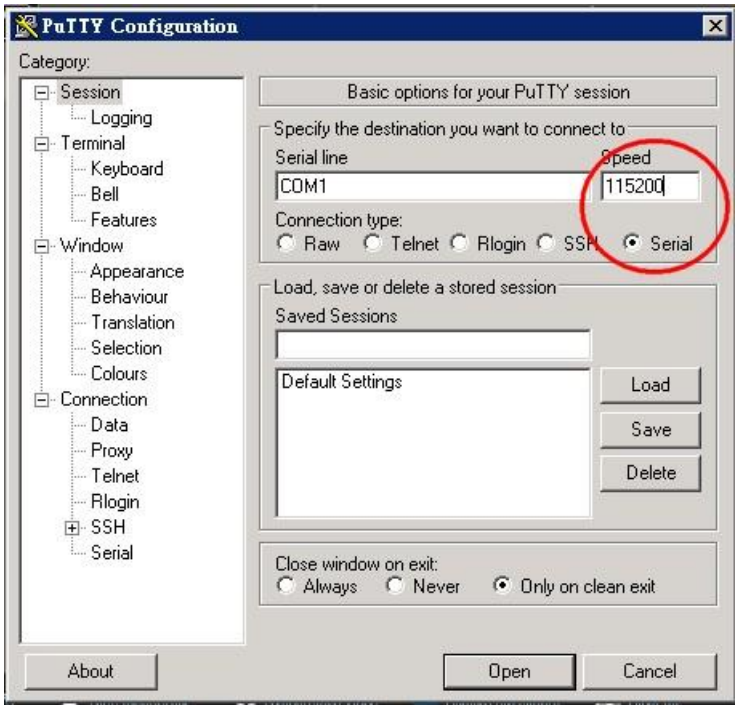


Figure 4-4 The Procedures of Setup Serial Console by HyperTerminal

4. If you are using PuTTY instead, please refer to below


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Basic options for your PuTTY session
Specify the destination you want to connect to
Serial line
Speed
COM1
115200
Connection type:
Raw
Telnet
Rlogin
SSH
Serial
Load, save or delete a stored session
Saved Sessions
Default Settings
Load
Save
Delete
Close window on exit:
Always
Never
Only on clean exit
About
Open
Cancel


PuTTY Configuration
Category:
Session
Logging
Terminal
Keyboard
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Serial
Options controlling local serial lines
Select a serial line
Serial line to connect to
COM1
Configure the serial line
Speed (baud)
115200
Data bits
8
Stop bits
1
Parity
None
Flow control
None
None
XON/XOFF
RTS/CTS
DSR/DTR
About
Open
Cancel

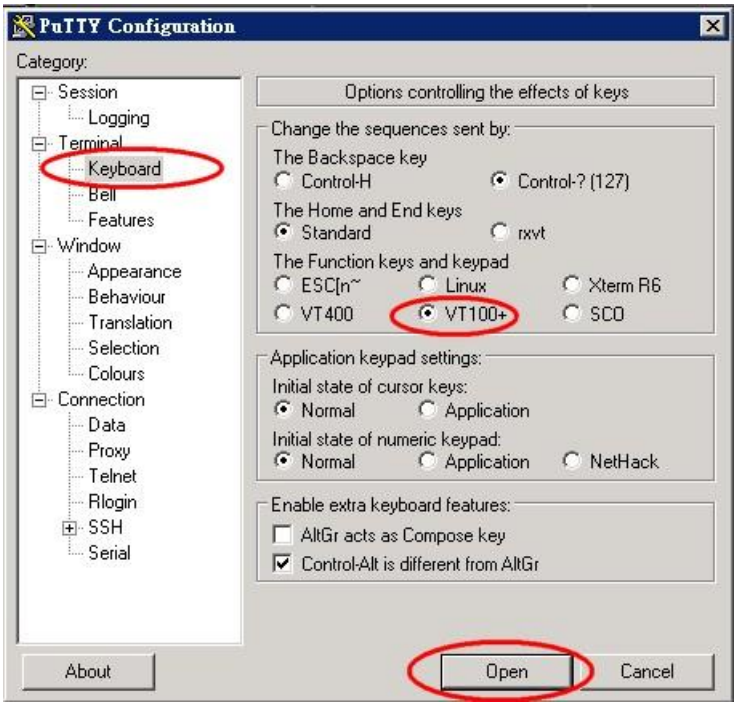


Figure 4-5 The Procedures of Setup Serial Console by PuTTY

5. Users should be able to login the controller system via console cable by following the procedures above.

Setup the Connection for Online Support

Following is the procedure to setup the connection for online support via TeamViewer:

1. Please download the TeamViewer from following hyper link: https://www.teamviewer.com/en/download/
2. Install TeamViewer.
3. Please provide the ID/password showed on the application to QSAN support team member to join the online support session.

4.3. Accessing Product Updates

To download product updates, please visit QSAN website:

https://www.qsan.com/download_center

4.4. Documentation Feedback

QSAN is committed to providing documentation that meets and exceeds your expectations. To help us improve the documentation, email any errors, suggestions, or comments to docsfeedback@qsan.com.

When submitting your feedback, include the document title, part number, revision, and publication date located on the front cover of the document.

Appendix

Glossary and Acronym List

Common Terminology

Item	Description
RAID	Redundant Array of Independent Disks. There are different RAID levels with different degree of data protection, data availability, and performance to host environment.
Disk	The Physical Disk belongs to the member disk of one specific RAID group.
Pool	A collection of removable media. One pool consists of a set of volumes and owns one RAID level attribute.
Volume	Each pool could be divided into several volumes. The volumes from one pool have the same RAID level, but may have different volume capacity.
LUN	Logical Unit Number. A logical unit number (LUN) is a unique identifier which enables it to differentiate among separate devices (each one is a logical unit).
WebUI	Web User Interface.
WT	Write-Through cache-write policy. A cache technique in which the completion of a write request is not signaled until data is safely stored in non-volatile media. Each data is synchronized in both data cache and accessed physical disks.
WB	Write-Back cache-write policy. A cache technique in which the completion of a write request is signaled as soon as the data is in cache and actual writing to non-volatile media occurs at a later time. It speeds up system write performance but needs to bear the risk where data may be inconsistent between data cache and the physical disks in one short time interval.
RO	Set the volume to be Read-Only.
DS	Dedicated Spare disks. The spare disks are only used by one specific

	RAID group. Others could not use these dedicated spare disks for any rebuilding purpose.
LS	Local Spare disks. The spare disks are only used by the RAID groups of the local enclosure. Other enclosure could not use these local spare disks for any rebuilding purpose.
GS	Global Spare disks. It is shared for rebuilding purpose. If some RAID groups need to use the global spare disks for rebuilding, they could get the spare disks out from the common spare disks pool for such requirement.
DG	DeGraded mode. Not all of the array's member disks are functioning, but the array is able to respond to application read and write requests to its virtual disks.
SCSI	Small Computer System Interface
SAS	Serial Attached SCSI
S.M.A.R.T.	Self-Monitoring Analysis and Reporting Technology
WWN	World Wide Name
HBA	Host Bus Adapter
SES	SCSI Enclosure Services
NIC	Network Interface Card
BBM	Battery Backup Module
SCM	Super Capacitor Module

FC / iSCSI / SAS Terminology

Item	Description
FC	Fibre Channel
FC-P2P	Point-to-Point
FC-AL	Arbitrated Loop
FC-SW	Switched Fabric
iSCSI	Internet Small Computer Systems Interface
LACP	Link Aggregation Control Protocol
MPIO	Multipath Input/Output
MC/S	Multiple Connections per Session
MTU	Maximum Transmission Unit
CHAP	Challenge Handshake Authentication Protocol. An optional security

	mechanism to control access to an iSCSI storage system over the iSCSI data ports.
iSNS	Internet Storage Name Service
SAS	Serial Attached SCSI

Dual Controller Terminology

Item	Description
SBB	Storage Bridge Bay. The objective of the Storage Bridge Bay Working Group (SBB) is to create a specification that defines mechanical, electrical and low-level enclosure management requirements for an enclosure controller slot that will support a variety of storage controllers from a variety of independent hardware vendors (“IHVs”) and system vendors.
6G MUX	Bridge board is for SATA II disk to support dual controller mode.

End-User License Agreement (EULA)

Please read this document carefully before you use our product or open the package containing our product.

YOU AGREE TO ACCEPT TERMS OF THIS EULA BY USING OUR PRODUCT, OPENING THE PACKAGE CONTAINING OUR PRODUCT OR INSTALLING THE SOFTWARE INTO OUR PRODUCT. IF YOU DO NOT AGREE TO TERMS OF THIS EULA, YOU MAY RETURN THE PRODUCT TO THE RESELLER WHERE YOU PURCHASED IT FOR A REFUND IN ACCORDANCE WITH THE RESELLER'S APPLICABLE RETURN POLICY.

General

QSAN Technology, Inc. ("QSAN") is willing to grant you (“User”) a license of software, firmware and/or other product sold, manufactured or offered by QSAN (“the Product”) pursuant to this EULA.

License Grant

QSAN grants to User a personal, non-exclusive, non-transferable, non-distributable, non-assignable, non-sub-licensable license to install and use the Product pursuant to the terms of this EULA. Any right beyond this EULA will not be granted.

Intellectual Property Right

Intellectual property rights relative to the Product are the property of QSAN or its licensor(s). User will not acquire any intellectual property by this EULA.

License Limitations

User may not, and may not authorize or permit any third party to: (a) use the Product for any purpose other than in connection with the Product or in a manner inconsistent with the design or documentations of the Product; (b) license, distribute, lease, rent, lend, transfer, assign or otherwise dispose of the Product or use the Product in any commercial hosted or service bureau environment; (c) reverse engineer, decompile, disassemble or attempt to discover the source code for or any trade secrets related to the Product, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation; (d) adapt, modify, alter, translate or create any derivative works of the Licensed Software; (e) remove, alter or obscure any copyright notice or other proprietary rights notice on the Product; or (f) circumvent or attempt to circumvent any methods employed by QSAN to control access to the components, features or functions of the Product.

Disclaimer

QSAN DISCLAIMS ALL WARRANTIES OF PRODUCT, INCLUDING BUT NOT LIMITED TO ANY MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, WORKMANLIKE EFFORT, TITLE, AND NON-INFRINGEMENT. ALL PRODUCTS ARE PROVIDE “AS IS” WITHOUT WARRANTY OF ANY KIND. QSAN MAKES NO WARRANTY THAT THE PRODUCT WILL BE FREE OF BUGS, ERRORS, VIRUSES OR OTHER DEFECTS.

IN NO EVENT WILL QSAN BE LIABLE FOR THE COST OF COVER OR FOR ANY DIRECT, INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, CONSEQUENTIAL OR SIMILAR DAMAGES OR LIABILITIES WHATSOEVER (INCLUDING, BUT NOT LIMITED TO LOSS OF DATA, INFORMATION, REVENUE, PROFIT OR BUSINESS) ARISING OUT OF OR RELATING TO THE USE OR INABILITY TO USE THE PRODUCT OR OTHERWISE UNDER OR IN CONNECTION WITH THIS EULA OR THE PRODUCT, WHETHER BASED ON CONTRACT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHER THEORY EVEN IF QSAN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Limitation of Liability

IN ANY CASE, QSAN’S LIABILITY ARISING OUT OF OR IN CONNECTION WITH THIS EULA OR THE PRODUCT WILL BE LIMITED TO THE TOTAL AMOUNT ACTUALLY AND ORIGINALLY PAID BY CUSTOMER FOR THE PRODUCT. The foregoing Disclaimer and Limitation of Liability will apply to the maximum extent permitted by applicable law. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the exclusions and limitations set forth above may not apply.

Termination

If User breaches any of its obligations under this EULA, QSAN may terminate this EULA and take remedies available to QSAN immediately.

Miscellaneous

- QSAN reserves the right to modify this EULA.
- QSAN reserves the right to renew the software or firmware anytime.
- QSAN may assign its rights and obligations under this EULA to any third party without condition.
- This EULA will be binding upon and will inure to User’s successors and permitted assigns.

- This EULA shall be governed by and constructed according to the laws of R.O.C. Any disputes arising from or in connection with this EULA, User agree to submit to the jurisdiction of Taiwan Shilin district court as first instance trial.