*D-Link Confidential*

# SO(Shared Object) File API for DNH

NMSDD, D-Link

Revision History

| Version | Date | Author | Reviewer | Description |
|---|---|---|---|---|
| 0.0.23 | 2020/11/11 | Jerome Cheng | | libLANSetting.so:<br>1. setLANSetting: config IPv4 by dhcpcd.conf.<br>2. getLANSetting: enhance the method to get IPv4 DNS from /etc/resolv.conf<br>3. setLANSettingIpv6: config IPv4 by dhcpcd.conf, /etc/resolv.conf.head, /etc/sysctl.conf<br>4. getLANSettingIpv6: get IPv6 LAN setting.<br>5. getIpv6Provision: get IPv6 provision type. |
| 0.0.22 | 2020/11/6 | Tony Ting | | libDB.so:<br>6. setFOTASetting: add 8: Every day in the 'nWeekday' parameter |
| 0.0.21 | 2020/11/3 | Jerome Cheng | | libFTPClient.so: enhance upload function with building destination path capability. |
| 0.0.20 | 2020/9/22 | Jerome Cheng | | libDatetimeSetting.so: add function for city datetime<br>1. setCityDatetime<br>2. getCityDatetime<br>3. getCityDst |
| 0.0.19 | 2020/9/22 | Tony Ting | | libDB.so: add function for FOTA<br>1. getFirmwareInfo<br>2. setFirmwareReleaseNote<br>3. setFirmwareDateTime |
| 0.0.18 | 2020/8/14 | Jerome Cheng | | libSystem.so: update function<br>1. getFotaStatus<br>2. setFotaStatus<br>libHdd.so: update function<br>1. getHddFlag<br>2. setHddFlag |
| 0.0.17 | 2020/8/13 | Tony Ting | | libDB.so: add function<br>1. getDeviceUUID<br>2. setFOTASetting<br>3. getFOTASetting<br>4. startFOTAScript<br>5. analyzeFirmwareInfo<br>6. setFOTAUpdateStatus<br>7. getFOTAUpdateStatus |
| 0.0.16 | 2020/7/21 | Jerome Cheng | | 1. Add getPerformance function to offer performance information.<br>2. Add getHddFlag/setHddFlag functions to handle HDD initialization messages.<br>3. Add getWebAccessPort/setWebAccessPort functions to access web access port data in system-data.db.<br>4. Add runMdns function to execute mDNS function.<br>5. Change updateDnssdRecord function name to runDnssd.<br>6. Add lanPortStatus, nvrVersion items in getSystemInfo return data. |
| 0.0.15 | 2020/6/29 | Jerome Cheng | | 1. Add hddEject function |

| | | | | |
|---|---|---|---|---|
| | | | | 2. Add getDaylightSaving function to offer daylight saving information.<br>3. Add web access port and daylight saving messages in getSystemInfo function<br>4. Add updateDnssdRecord function to maintain DNS-SD records.<br>5. Update set/getDatetime, set/getTimezone description. |
| 0.0.14 | 2020/6/8 | Jerome Cheng | | 1. Add setLANSettingIpv6, getLANSettingIpv6 to handle IPv6 function.<br>2. Add setDaylightSaving to handle daylight saving function.<br>3. Add FTP upload function. |
| 0.0.13 | 2020/5/26 | Jerome Cheng | | For DNH-200 SO API:<br>1. Enhance setTimezone with daylight saving.<br>2. Enhance setPowerLed to control power LED and setSsoLed to control blue LED.<br>3. Add setLANSettingIpv6, getLANSettingIpv6 to handle IPv6 function.<br>4. Enhance setLANSetting to change LED as IP provision.<br>5. Enhance getSystemInfo to offer system performance message.<br>6. Enhance SO API to sync DNS-SD messages.<br>7. Add libSystem.so shutdown.<br>Add libHdd.so hddInitialize, hddFormat |
| 0.0.12 | 2019/09/09 | Tony Ting | | Add libLog.so: addEventLog |
| 0.0.11 | 2019/07/08 | Tony Ting | | libDB.so: add function<br>1. getHWVersion<br>2. getNCVersion<br>3. getSSOStatus<br>4. getNCStatus<br>5. getDDPv5ClientVersion<br>6. setNCStatus<br>7. setSSOStatus<br>8. setDDPv5ClientVersion |
| 0.0.10 | 2019/06/13 | Jerome Cheng | | Remove 2 so files.<br>1. libReboot.so<br>2. libFactoryReset.so |
| 0.0.9 | 2019/05/31 | Tony Ting | | libDB.so: add function to get firmware version<br>1. getFirmwareVersion<br>2. getFirmwareVersionFull |
| 0.0.8 | 2019/05/03 | Jerome Cheng | | Add two fw upgrade APIs<br>1. getFirmwareUpgradeStatus<br>2. setFirmwareUpgradeStatus |
| 0.0.7 | 2019/04/26 | Tony Ting<br>Jerome Cheng | | 1. Add libFTPClient, libFWValidate, libUtilityTar<br>2. Add doChangDAA parameter for setLANSetting SO API |
| 0.0.6 | 2019/3/12 | Jerome Cheng | | Modify *setLANSetting* API implement on RK3328 |
| 0.0.5 | 2019/3/8 | Jerome Cheng | | Add the 4th Chapter: SO API in RK3328 SDK |
| 0.0.4 | 2019/03/05 | Austin Chen | | 1. Add fwUpgrade return value and process |
| 0.0.3 | 2019/3/4 | Redd Lin | | 2. Difference between Debian and embedded linux<br>3. Add partial so code<br>Add nodejs use .so |

| | | | | |
|---|---|---|---|---|
| 0.0.2 | 2019/2/27 | Redd Lin | | Difference between Debian and unknown OS |
| 0.0.1 | 2019/1/31 | Jerome Cheng | | Initial version |

**Table of Contents**

# 1. Requirement

Web、CLI、DDPv5 都會對 DNH 進行操作，為提升程式開發的效率與降低維護的成本，將這些操作以共享函式庫的方式來實作，就是實際執行的動作由 so 檔的執行，Web、CLI、DDPv5 呼叫 so 檔所提供的 API 即可。

# 2. SO API list

此章節列表中的函式是 DNH-100 中 Web、CLI、DDPv5 所有功能 SO API 的介面。

在呼叫 SO API 修改設定時有三種情境，一是暫時性的設定，在系統 Reboot 後就得恢復成原本的設定，二是永久性的設定，API 下了設定之後，系統 Reboot 後是修改後的設定，三是在做了暫時性的設定後，將暫時性的設定改變成永久性的設定。

SO API 透過參數<isPermanent>來決定要做暫時性或是永久性的設定。
SO API 修改設定時，會直接修改系統的設定檔，即為永久性的設定。為了因應暫時性的設定，在 API 做暫時性設定的動作，系統中會建立一個備用設定檔紀錄該設定值當前的設定，系統 Reboot 後透過程式讀取這個備用設定檔，將設定值設定到系統對應的設定檔裡頭。若程式檢查發現沒有這個備用設定檔，就不做動作。
所以，若當使用者想將暫時性的設定改變成永久性的設定時，我們只要刪除這個備用設定檔便可以了。這個刪除備用設定檔的動作會透過 libCLI.so 檔中的 save 函式進行。

| SO File Name | | | |
|---|---|---|---|
| Function | Description | Input Parameters | Return Value |
| libLANSetting.so | | | |
| int **setLANSetting** (<br>int isPermanent,<br>int isDHCP,<br>char *argIpAddress,<br>char *argNetmask,<br>char *argGateway,<br>int nDnsType,<br>char *argDNS1,<br>char *argDNS2,<br>int flagUpdateDaa<br>) | Set LAN setting<br>參數 isDHCP, strIPAddress, strNetmask, strGateway, strDNS1, strDNS2 的值若為 -1(“-1”)時，維持系統設定值<br><br>IP provision: Power LED blink orange | 1. isPermanent<br>0: temporary<br>1: permanent<br>2. isDHCP<br>0: static<br>1: DHCP<br>3. argIPAddress<br>4. argNetmask<br>5. argGateway<br>6. nDnsType<br>7. argDNS1 | Integer type:<br>0: Failed<br>1: Successful<br>2: Invalid permanet type<br>3: Invalid isDHCP value<br>4: Invalid IPv4 address<br>5: Invalid netmask<br>6: Invalid gateway |

| | | | |
|---|---|---|---|
| 3.1 setLANSetting | | 8. argDNS2<br>9. flagUpdateDaa | 7: Invalid DNS type<br>8: Invalid primary DNS<br>9: Invalid secondary DNS<br>10: Invalid flagUpdateDaa |
| char ***getLANSetting**()<br><br>3.40 getLANSetting | Get LAN setting information | void | JSON string type:<br>{<br>"ipmode": "<IP mode>",<br>"ip": "<IPv4 address>",<br>"netmask": "<netmask>",<br>"gateway": "<gateway>",<br>"dns1": "<DNS 1>",<br>"dns2": "<DNS 2>",<br>"tmpIp": "<IPv4 address>",<br>"tmpNetmask": "<netmask>",<br>"tmpGateway": "<gateway>",<br>"tmpDns1": "<DNS 1>",<br>"tmpDns2": "<DNS 2>",<br>"dnsType": <IPv4 DNS type><br>} |
| int **setLANSettingIpv6** (<br>int isPermanent,<br>char *argProvision,<br>char *argIpv6Address,<br>int nPrefixLength,<br>char *argGateway,<br>int nDnsType, | Set IPv6 LAN setting. | 1. isPermanent<br>0: temporary<br>1: permanent<br>2. argProvision<br>"Static",<br>"Auto",<br>"Local" | Integer type:<br>0: Failed<br>1: Successful<br>2: Invalid permanent type<br>3: Invalid IPv6 provision type |

| | | | |
|---|---|---|---|
| char *argDns1,<br>char *argDns2,<br>int flagUpdateDaa<br>)<br><br>3.41 setLANSettingIpv6 | | 3. argIpv6Address<br>4. nPrefixLength: 1~128<br>5. argGateway<br>6. nDnsType<br>7. argDns1<br>8. argDns2<br>9. flagUpdateDaa<br>0: No<br>1: Yes | 4: Invalid IPv6 address<br>5: Invalid prefix length<br>6: Invalid gateway<br>7: Invalid DNS type<br>8: Invalid primary DNS<br>9: Invalid secondary DNS<br>10: Invalid flagUpdateDaa |
| char ***getLANSettingIpv6**()<br><br>3.42 getLANSettingIpv6 | Get IPv6 LAN setting information | void | JSON string type:<br>{<br>"**ipv6Provision**": "<IPv6 provision>",<br>"**ipv6Address**": "<IPv6 address>",<br>"**ipv6PrefixLength**": <IPv6 prefix length>,<br>"**ipv6LinkLocal**": "<link-local address>"<br>"**ipv6Gateway**": "<IPv6 gateway>",<br>"**ipv6DnsType**": <IPv6 DNS type>,<br>"**ipv6Dns1**": "<IPv6 DNS 1>",<br>"**ipv6Dns2**": "<IPv6 DNS 2>",<br>"**ipv6TmpAddress**": "<IPv6 address>",<br>"**ipv6TmpPrefixLength**": <IPv6 prefix length>,<br>"**ipv6TmpGateway**": "<IPv6 gateway>", |

| | | | |
|---|---|---|---|
| | | | "**ipv6TmpDns1**": "<IPv6 DNS 1>",<br>"**ipv6TmpDns2**": "<IPv6 DNS 2>"<br>} |
| char ***getIpv6Provision**()<br>3.43 getIpv6Provision | Get IPv6 provision type. | void | String type:<br>1. "Static"<br>2. "Auto"<br>3. "Local" |
| libDatetimeSetting.so | | | |
| int **setDatetime** (<br>int isPermanent,<br>int enableNTPServer,<br>char* strServerAddress,<br>char* strDatetime<br>)<br>3.2 setDatetime | Set NTP server or datetime | 1. isPermanent<br>integer type<br>1: permanent<br>0: temporary<br>2. enableNTPServer<br>integer type<br>0: disable NTP<br>1: enable NTP<br>2: disable NTP and set datetime<br>3. strServerAddress<br>string type, domain name<br>4. strDatetime<br>string type and format is “YYYY-MM-DD hh:mm:ss”<br>1. | integer type<br>0: Failed<br>1: Success<br>2: Invalid permanent type<br>3: Invalid enable type<br>4: Invalid NTP Server<br>5: Invalid datetime<br>6: DB does not exist<br>7 Table does not exist |
| char ***getDatetime**(void) | | void | 1. Json string type:<br>{<br>“ntpEnable”: < ntpEnable>,<br>“ntpServer”: “< ntpServer>”,<br>“datetime”: “< datetime>”<br>}<br>2. “ErrDB”: Error: |

| | | | |
|---|---|---|---|
| | | | DB does not exist<br>3. “ErrTable”: Error: Table does not exist |
| int **setTimezone** (<br>int isPermanent,<br>char* strTimezone,<br>)<br><br>3.3 setTimezone | Set time zone | 1. isPermanent<br>0: temporary<br>1: permanent<br>2. strTimezone<br>1. time zone id in NC. E.x. “56”<br>2. area name. E.x. “Taipei” | integer type:<br>0: Failed<br>1: Success<br>2: Invalid permanent type<br>3: Invalid timezone id<br>4: DB does not exist<br>5: Table does not exist |
| char ***getTimezone**(void) | | void | 1. Json string type:<br>{<br>“tzID”: <tzID>,<br>“tzNameNC”: “< tzNameNC>”<br>}<br>2. “ErrDB”: Error: DB does not exist<br>3. “ErrTable”: Error: Table does not exist |
| int **setDaylightSaving** (<br>int isdst,<br>int offsetSeconds,<br>int startMonth,<br>int startWeek,<br>int startDay,<br>char* startTime,<br>int endMonth,<br>int endWeek,<br>int endDay,<br>char* endTime | Set daylight saving | 1. isdst<br>0: GMT time<br>1: daylight saving time<br>2. offsetSeconds<br>3. startMonth<br>4. startWeek<br>5. startDay<br>6. startTime<br>7. endMonth<br>8. endWeek | integer type:<br>0: Failed<br>1: Success<br>2: Invalid daylight saving type<br>3: Invalid offset seconds<br>4: Invalid start month value<br>5: Invalid start week value |

| | | | |
|---|---|---|---|
| ) | | 9. endDay<br>10. endTime | 6: Invalid start week day value<br>7: Invalid start time value<br>8: Invalid end month value<br>9: Invalid end week value<br>10: Invalid end week day value<br>11: Invalid end time value<br>12: Invalid start time and end time |
| char ***getDaylightSaving**() | | | Json string type:<br>{<br>“**isdst**”: <isdst>,<br>“**offsetSeconds**”: < offsetSeconds >,<br>“**startMonth**”: < startMonth >,<br>“**startWeek**”: < startWeek >,<br>“**startDay**”: < startDay >,<br>“**startTime**”: “< startTime >”,<br>“**endMonth**”: < endMonth >,<br>“**endWeek**”: < endWeek >,<br>“**endDay**”:<br>< endDay >,<br>“**endTime**”: “< endTime >”,<br>“**isInDstPeriod**”:<isInDstPeriod><br>} |

| | | | |
|---|---|---|---|
| int **setCityDatetime**(<br>char *strCityId,<br>int isManualTimezone,<br>int offsetSeconds,<br>int startMonth,<br>int startWeek,<br>int startDay,<br>char *startTime,<br>int endMonth,<br>int endWeek,<br>int endDay,<br>char *endTime) | 設定城市時間<br><br>當 strCityId 為 NULL，so api 取當前 City ID 進行城市時間設定。<br><br>自動配置:<br>isManualTimezone=0<br><br>手動配置:<br>isManualTimezone=1 以及第 3 個參數到第 11 個參數都要有值。 | 1. strCityId<br>2. isManualTimezone<br>3. offsetSeconds<br>4. startMonth<br>5. startWeek<br>6. startDay<br>7. startTime<br>8. endMonth<br>9. endWeek<br>10. endDay<br>11. endTime | integer type<br>0: Failed<br>1: Successful<br>2: Invalid city ID<br>3: Invalid isManualTimezone value<br>4: Invalid DST offset seconds<br>5: Invalid DST start month<br>6: Invalid DST start week<br>7: Invalid DST start weekday<br>8: Invalid DST start time<br>9: Invalid DST end month<br>10: Invalid DST end week<br>11: Invalid DST end weekday<br>12: Invalid DST end time<br>13: Invalid DST start time and end time |
| char ***getCityDatetime**() | 取得 DNH 當前設定的城市時間訊息。<br>若城市無夏令時間，offsetSeconds 的值為 0。 | N/A | json string type:<br>{<br>"tzID": "<City ID>",<br>"tzNameNC": "<City Name>",<br>"isManualTimezone" : <isManualTimezone>,<br>"offsetSeconds": <offsetSeconds>, |

| | | | |
|---|---|---|---|
| | | | "startMonth": <startMonth>,<br>"startWeek": <startWeek>,<br>"startDay": <startDay>,<br>"startTime": "<startTime>",<br>"endMonth": <endMonth>,<br>"endWeek": <endWeek>,<br>"endDay": <endDay>,<br>"endTime": "<endTime>",<br>"isInDstPeriod":<isInDstPeriod>,<br>"tzPosix": "<posix string>"<br>} |
| char ***getCityDst(**<br>char *strCityId<br>) | 取得城市 TZ file 的夏令時間，顯示在 Web UI 提供使用者設定。<br><br>若城市無夏令時間，offsetSeconds 的值為 0。 | 1. strCityId | {<br>"tzID": "<City ID>",<br>"offsetSeconds": <offsetSeconds>,<br>"startMonth": <startMonth>,<br>"startWeek": <startWeek>,<br>"startDay": <startDay>,<br>"startTime": "<startTime>",<br>"endMonth": <endMonth>,<br>"endWeek": <endWeek>,<br>"endDay": |

| | | | |
|---|---|---|---|
| | | | <endDay>,<br>"endTime":<br>"<endTime>"<br>} |
| libConsoleSetting.so | | | |
| int **setConsoleSetting**(<br>int enableConsole,<br>char *strProtocol,<br>int nTimeoutSec<br>) | Set console<br>-Telnet<br>-SSH<br>-disabled<br><br>Set console timeout<br>-300 seconds<br>-0 (Never) | enableConsole<br>1: enable<br>0: disable<br>-1: no action<br><br>strProtocol<br>“telnet”: Telnet<br>“ssh”: SSH<br>“-1”: no action<br><br>nTimeoutSec<br>300: 300 seconds<br>0: no timeout<br>-1: no action | integer type:<br>1: Success<br>2: enableConsole invalid<br>3: strProtocol invalid<br>4: nTimeoutSec invalid |
| char ***getConsoleSetting**( void) | | | Json string type:<br>{<br>“enable”: <enable>,<br>“protocol”: “< protocol>”,<br>“timeout”: <timeout><br>} |
| libCLI.so | | | |
| char ***getSystemInfo** ()<br><br>3.7 getSystemInfo | Get system information (device version, mac address, ip info, USB/Miscro SD usage, firmware version example, v1.00r010, NTP server, Datetime, Timezone)<br><br>For DNH-100/200 | | System information Json<br><br>DNH-100/200<br>{<br>“**webAccessPort**”: “<webAccessPort>”,<br>“**lanPortStatus**”: <lanPortStatus>,<br>“**dstStatus**”: <dstStatus>, |

| | | | |
|---|---|---|---|
| | - NC web access port<br>- LAN port status<br>- daylight saving information<br><br>For DNH-200<br>- NVR version | | “**dstOffset**”: <dstOffset>,<br>“**dstStart**”: <dstStart>,<br>“**dstEnd**”: <dstEnd><br>}<br><br>For DNH-200<br>{<br>“**nvrVersion**”: “<NVR version>”<br>} |
| int **setAdminPassword**(<br>char* password<br>)<br><br>3.9 setAdminPassword | Set admin password, and change “state” message in DNS-SD to “Operational”. | 1. password | integer type<br>0: Failed<br>1: Success |
| int **save**()<br><br>3.10 save | Save all settings to DNH-100 | | integer type<br>0: Failed<br>1: Success |
| int **setDeviceName**(char *strDeviceName) | Set device name, and change “name” message in DNS-SD. | strDeviceName | integer type<br>0: Failed<br>1: Success |
| char ***getPerformance**() | DNH-200 performance items:<br>1. CPU temperature (單位度 C)<br>2. CPU loading (%)<br>3. Memory usage (total/used)<br>4. System usage<br>5. USB usage<br>6. HDD model<br>7. HDD storage (total/used)<br>8. HDD NC partition (total/used)<br>9. HDD NVR log | | Json string type<br>{<br>"**cpuTemp**": "<CPU temperature>",<br>"**cpuLoading**": "<CPU loading>",<br>"**memUsage**": "<Memory usage>",<br>"**sysUsage**": "<System usage>",<br>"**usbUsage**": "<USB usage>",<br>"**hddModel**": "<HDD model>",<br>"**hddStorage**": "<HDD |

| | | | |
|---|---|---|---|
| | partition (total/used)<br>10. HDD NVR record partition (total/used)<br>11. Tx, Rx bit count per second | | storage>,<br>"**hddNc**": "<HDD NC>",<br>"**hddNvrLog**": "<HDD NVR log>",<br>"**hddNvrRecord**": "<HDD NVR record>",<br>"**rxtx**": "<Rx and Tx bit count per second>"<br>} |
| int **getWebAccessPort**(void) | Get web access port | | integer type<br><web access port> |
| int **setWebAccessPort**(int nPortNumber) | Set web access port | 1. nPortNumber | integer type<br>0: Failed<br>1: Success |
| **libFTPClient.so** | | | |
| int **download** (char* strIpAddr, int nPort, char* strlogin, char* strPassword, char* strSrcFilePath, char* strFileName, char* strDesFilePath)<br><br>3.11 download | Only support FTP portal (non-sFTP and non-tFTP) | 1. strIpAddr<br>2. nPort<br>3. strlogin<br>4. strPassword<br>5. strSrcFilePath<br>6. strFileName<br>7. strDesFilePath | integer type<br>1. Success<br>2. Error: FTP server does not exist<br>3. Error: FW file does not exist<br>4. Error: Destination path does not existent<br>5. Error: Timeout<br>6. Error: Login Fail |
| int **upload**(<br>char* strIpAddr,<br>int nPort,<br>char* strlogin,<br>char* strPassword,<br>char* strFileName,<br>char* strSrcFilePath,<br>int nBuildPath); | Support FTP upload | 1. strIpAddr<br>2. nPort<br>3. strlogin<br>4. strPassword<br>5. strFileName: destination path<br>6. strSrcFilePath<br>7. nBuildPath<br>0: Check path but not build path.<br>1: Check path and build path. | integer type<br>1. Success<br>2. Error: FTP server does not exist<br>3. Error: Destination file does not exist<br>4. Error: Source file does not exist<br>5. Error: Timeout<br>6. Error: Login Fail<br>7. Error: Invalid nBuildPath value |

| | | | |
|---|---|---|---|
| | | | 8. Error: Not permitted to build destination path |
| **libFWValidate.so** | | | |
| int **checkHeaderAndPayload** (char* strFilePath, char* strFileName)<br>3.12 checkHeaderrAndPayload | 1. Header model name is dnh100<br>2. Payload decryption via AES256-CTR | 1. strFilePath<br>2. strFileName | integer type<br>1: Valid<br>2: Invalid: FW file does not exist<br>3: Invalid: Header is incorrect<br>4: Invalid: Payload is incorrect |
| **libUtilityTar.so** | | | |
| int **unTar** (char* strFilePath, char* strFileName, char* strDesFilePath)<br>3.13 unTar | Support .tar and .tar.gz format | 1. strFilePath<br>2. strFileName<br>3. strDesFilePath | integer type<br>1: Success<br>2: Error: File does not exist<br>3: Error: File is not tar format<br>4: Error: Destination path does not exist. |
| **libDB.so** | | | |
| int **getFirmwareUpgradeStatus**()<br>3.14 getFirmwareUpgradeStatus | Get firmware upgrade status from system-data.db | N/A | integer type<br>0: Firmware upgrade is Idle<br>1: Firmware upgrade is busy<br>2: Error: DB of system-data.db does not exist.<br>3: Error: Table does not exist. |
| int **setFirmwareUpgradeStatus**(int nStatus)<br>3.15 setFirmwareUpgradeStat | Set firmware upgrade status into system-data.db | nStatus<br>0: Idle<br>1: Busy | integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of |

| | | | |
|---|---|---|---|
| us | | | firmware does not exist.<br>4: Firmware upgrade is busy<br>5: Error: input parameter is invalid. |
| char* **getFirmwareVersion**()<br>3.16 getFirmwareVersion | Get current firmware version without svn version | N/A | string type “<x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of firmware does not exist.<br>“4”: Error: Table of firmware has no data. |
| char* **getFirmwareVersionFull**()<br>3.17 getFirmwareVersionFull | Get current firmware version with svn version | N/A | string type “<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of firmware does not exist.<br>“4”: Error: Table of firmware has no data. |
| char* **getHWVersion**()<br>3.18 getHWVersion | Get hardware version | N/A | string type “<A-Z><1-999>” |
| char* **getNCVersion**()<br>3.19 getNCVersion | Get Nuclias Connect Version | N/A | string type “<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of Start_NC does not |

| | | | |
|---|---|---|---|
| | | | exist.<br>“4”: Error: Table of Start_NC has no data. |
| char* **getSSOStatus**()<br><br>3.20 getSSOStatus | Get Single Sign On Status | N/A | string type<br>“Enabled”<br>“Enabled (time out)<br>“Enable ( Nuclias Server is under maintenance )”<br>“Disabled”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of SSO does not exist.<br>“4”: Error: Table of SSO has no data. |
| int **getNCStatus**()<br><br>3.21 getNCStatus | Get NC status | N/A | integer type<br>0: stop<br>1: running |
| char* **getDDPv5ClientVersion**()<br><br>3.22 getDDPv5ClientVersion | Get current ddpv5 client version | N/A | string type<br>“<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of DDPv5 does not exist.<br>“4”: Error: Table of DDPv5 has no data. |
| int **setNCStatus**(int nStatus)<br><br>3.23 setNCStatus | Set NC Status into system-data.db | nStatus<br>0: stop<br>1: running | integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of Start_NC does not |

| | | | |
|---|---|---|---|
| | | | exist.<br>4: Error: Table of Start_NC has no data.<br>5: Error: input parameter is invalid. |
| int **setSSOStatus**(char* strStatus)<br>3.24 setSSOStatus | Set SSO Status into system-data.db | strStatus | integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of SSO does not exist.<br>4: Error: Table of SSO has no data. |
| int **setDDPv5ClientVersion**( char* strVersion)<br>3.25 setDDPv5ClientVersion | set ddpv5 client version | string type<br>strVersion<br>“<x.x.x.x>” | integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist.<br>3: Error: Table of DDPv5 does not exist.<br>4: Error: Table of DDPv5 has no data. |
| char* **getDeviceUUID**()<br>3.28 getDeviceUUID | Get device uuid of dnh100 | N/A | string type |
| int **setFOTASetting** (<br>int nEnable,<br>int nWeekday,<br>int nHour,<br>int nMinute,<br>int nUpdateBetaFw)<br>3.29 setFOTASetting | Record the FOTA setting in the database and start the FOTA script. | 1. nEnable<br>Integer<br>0: Disabled<br>1: Enabled<br>2. nWeekday<br>Integer, 1 ~ 8<br>1: Monday<br>2: Tuesday<br>3: Wednesday<br>4: Thursday | Integer<br>1: Success<br>2: Error: Invalid Input Parameter<br>3: Error: The system-data.db does not exist.<br>4: Error: The Firmware table does not exist. |

| | | | |
|---|---|---|---|
| | | 5: Friday<br>6: Saturday<br>7: Sunday<br>8: Every day<br>3. nHour<br>Integer, 0 ~ 23<br>4. nMinute<br>Integer, 0 ~ 59<br>5. nUpdateBetaFw<br>Integer<br>0: Disabled<br>1: Enabled<br><br>Note: If the “nEnable” is 0, the “nWeekday”, “nHour”, “nMinute”, and “nUpdateBetaFw” will not be updated to the database. | 5: Error: The Firmware table has no data. |
| char* **getFOTASetting** ()<br><br>3.30 getFOTASetting | Get FOTA auto update settings from the database. | N/A | String<br>JSON format |
| int **startFOTAScript**()<br><br>3.31 startFOTAScript | The FOTA script will be executed if the auto is enabled. | N/A | Integer<br>1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| char* **analyzeFirmwareInfo**(<br>int nMajor,<br>int nMinor, | Analysis firmware information and call the soapi **“setFotaStatus”** to | 1. nMajor<br>Integer, 1~99<br>2. nMinor<br>Integer, 0~999 | String<br>JSON format |

| | | | |
|---|---|---|---|
| int nRev,<br>char* strFirmwareInfoFromFOTA,<br>int nUpdateBetaFw)<br><br>3.32 analyzeFirmwareInfo | update the firmware status and DNS-SD | 3. nRev<br>Integer, 0~999<br>4. strFrimwareInfoFromFOTA<br>String<br>5. nUpdateBetwFw<br>Integer<br>0: Disabled<br>1: Enabled | |
| int **setFOTAUpdateStatus**(int nStatus)<br><br>3.33 setFOTAUpdateStatus | Record the FOTA firmware update status in the database | Integer<br>0: Idle<br>1: Running | Integer<br>1: Success<br>2: Error: Invalid Input Parameter<br>3: Error: The system-data.db does not exist.<br>4: Error: The Firmware table does not exist.<br>5: Error: The Firmware table has no data. |
| int **getFOTAUpdateStatus**()<br><br>3.34 getFOTAUpdateStatus | Get the FOTA firmware update status from the database | N/A | Integer<br>0: Idle<br>1: Running<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| char* **getFirmwareInfo**()<br><br>3.37 getFirmwareInfo | Get current DNH firmware information. | N/A | String<br>JSON format |
| int | Record the firmware | String | Integer |

| | | | |
|---|---|---|---|
| **setFirmwareReleaseNote(**<br>char* strReleaseNote)<br>3.38 setFirmwareReleaseNote | release note in the database | | 1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| int **setFirmwareDateTime**(char* strDateTime)<br>3.39 setFirmwareDateTime | Record the update date time | String | Integer<br>1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| **libLog.so** | | | |
| int addEventLog(char* strMessage, int nEventType)<br>3.26 addEventLog | add event log in ramdisk and external syslog server | strMessage: "….."<br>nEventType:<br>1: Device Management<br>3: Duplicate Task<br>4: Timeout Task<br>5: Invalid HTTP Messagge<br>6: Initialization<br>9: Firmware Upgrade | integer type<br>1: Success<br>2: Error: config-data.db doesn't exist<br>3: Error: config-data.db(CWM_Database_Setting) doesn't exist.<br>4: Error: log-data.db doesn't exist.<br>5: Error: log-data.db(CWM_Log_SystemEvent) doesn't exist.<br>6: Error: external syslog server doesn't |

<table>
<tr><td></td><td></td><td></td><td>exist.</td></tr>
<tr><td colspan="4"><b>libLed.so</b></td></tr>
<tr><td>int <b>setPowerLed</b>(int ledColor1, int ledColor2)</td><td>Set power LED color and control it with blink or not.</td><td>ledColor1<br>0: off<br>1: red<br>2: green<br>3: orange<br><br>ledColor2<br>-1: not blink<br>0: off<br>1: red<br>2: green<br>3: orange<br><br>Example<table>
<tr><th>c1</th><th>c2</th><th>LED</th></tr>
<tr><td>1</td><td>-1</td><td>Solid red</td></tr>
<tr><td>1</td><td>0</td><td>Blink red</td></tr>
<tr><td>1</td><td>2</td><td>Blink red /green</td></tr>
<tr><td>2</td><td>-1</td><td>Solid green</td></tr>
<tr><td>2</td><td>0</td><td>Blink green</td></tr>
<tr><td>3</td><td>-1</td><td>Solid orange</td></tr>
<tr><td>3</td><td>0</td><td>Blink orange</td></tr>
</table></td><td>integer type<br>0: Failed<br>1: Success</td></tr>
<tr><td>int <b>setSsoLed</b>(int nSsoMode)</td><td>Set SSO LED<br><br>For DNH-100<br>1. Disable: green<br>2. Enable: orange<br>3. Enable but failed: blink orange<br><br>For DNH-200</td><td>nSsoMode<br>0: Disabled<br>1: Enabled<br>2: Enabled, but timeout<br>3: Enabled, but server stop</td><td>integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of SSO does not exist.<br>4: Error: Table of SSO has no data.</td></tr>
</table>

| | | | |
|---|---|---|---|
| | 1. Disable: blue LED off<br>2. Enable: blue LED on<br>3. Enable but failed: blue LED off | | |
| **libSystem.so** | | | |
| int **runMdns**() | Update mDNS hostname | void | integer type<br>0: Failed<br>1: Success |
| int **runDnssd**() | Update DNS-SD records<br>1. manufacturer<br>2. type<br>3. line<br>4. series<br>5. model<br>6. hwVer<br>7. fwVariant<br>8. fwVer<br>9. hwId<br>10. uuid<br>11. name<br>12. state<br>13. fota | void | integer type<br>0: Failed<br>1: Success |
| int **getFotaStatus**(void) | Get FOTA status. | | integer type<br>1: UpToDate<br>2: Available<br>3: Mandatory |
| int **setFotaStatus**(int nFotaStatus) | Set FOTA status into DB and update DNS-SD records. | nFotaStatus<br>1: UpToDate<br>2: Available<br>3: Mandatory | integer type<br>0: Failed<br>1: Success<br>2: Invalid FOTA status |
| **libHdd.so** | | | |
| int **getHddFlag**(void)<br>3.35 getHddFlag | Recognize HDD status by the files in /tmp:<br>1. newhdds: HDD | void | integer type<br>**0**: There is no HDD mounted on DNH- |

|  |  |  |  |
|---|---|---|---|
|  | mounted<br>2. newhddu: HDD initialize error<br>3. newhdd1: HDD NC partition mounted<br>4. newhdd2: HDD NVR1 partition mounted<br>5. newhdd3: HDD NVR2 partition mounted |  | 200.<br>**1**: HDD is new or the formation is not valid for DNH-200.<br>**2**: HDD is failed to initialize on DNH-200.<br>**3**: HDD is ok to use for DNH-200 with the partition sizes for 1.0.0 version.<br>**-1**: error status |
| char ***setHddFlag**(int nResponse)<br><br>3.36 setHddFlag | Initialize HDD by /etc/hdd_init.sh | nResponse<br>**0**: No, do not initialize the HDD.<br>**1**: Yes, initialize the HDD. | integer type<br>**0**: Failed<br>**1**: Successful<br>**2**: Response value is invalid |
| **libBuzzer.so** |  |  |  |
| int runBuzzer() |  |  | integer type<br>0: Failed<br>1: Success |

# 3. SO API Definition

## 3.1 setLANSetting

| **Function** |
|---|
| int **setLANSetting**(int isPermanent, int isDHCP, char *argIpAddress, char *argNetmask, char *argGateway, int nDnsType, char *argDns1, char *argDns2, int flagUpdateDaa) |
| **Parameter** |
| 1. **isPermanent**: integer, 0: LAN setting is temporary, 1: LAN setting is permanent.<br>2. **isDHCP**: integer, 0: Static, 1: DHCP.<br>3. **argIpAddress**: string, IPv4 address format，若要使用當前系統設定值，可帶"-1".<br>4. **argNetmask**: string, IPv4 address format，若要使用當前系統設定值，可帶"-1".<br>5. **argGateway**: string, IPv4 address format，若要使用當前系統設定值，可帶"-1"，若不設定 |

gateway，帶"".

6. **nDnsType**: integer, 0: Auto, 1: Manual.
7. **argDns1**: string, IPv4 address format，若要使用當前系統設定值，可帶"-1"，若不設定 gateway，帶"".
8. **argDns2**: string, IPv4 address format，若要使用當前系統設定值，可帶"-1"，若不設定 gateway，帶"".
9. **flagUpdateDaa**: integer, for static setting, 0: do not sync IPv4 address to DAA, 1: sync IPv4 to DAA. 當設定 DHCP，參數值給 0.

**Return value**

Integer type:

0: Failed

1: Successful

2: Invalid permanet type

3: Invalid isDHCP value

4: Invalid IPv4 address

5: Invalid netmask

6: Invalid gateway

7: Invalid DNS type

8: Invalid primary DNS

9: Invalid secondary DNS

10: Invalid flagUpdateDaa

**Content**

■ 驗證參數

■ 若 isPermanent 值為 1，表示是永久性設定，將 IPv4 設定值存入 DB system-data.db 的 **LAN** 表。

| # | 欄位 | 變數 | 說明 |
|---|---|---|---|
| 1 | lanSettingType | isDHCP | 值為 0 存入"static"<br>值為 1 存入"dhcp" |
| 2 | ipAddress | argIpAddress | IPv4 format |
| 3 | netMask | argNetmask | IPv4 format |
| 4 | gateway | argGateway | IPv4 format |
| 5 | primaryDNS | argDns1 | IPv4 format |
| 6 | secondaryDNS | argDns2 | IPv4 format |
| 7 | dnsType | nDnsType | 0: Auto, 1: Manual |

以下動作不管 isPermanent 值為 0 或 1 都會進行。

- 當 isDHCP 值為 0 且 flagUpdateDaa 值為 1，將 argIpAddress 資料同步到 DAA，DAA 的欄位為[system-data.db].System.device_access_address。
  當 isDHCP 值為 1，忽略 flagUpdateDaa。

- 檢查要做 IPv4 的設定與當前的 IPv4 設定相同，若相同則不進行後續設定，回傳 1 (Success)。

- 設定 IPv4 LAN setting
  - **判斷 isDHCP 值為 0, Static**

| | |
|---|---|
| 設定 IP 與 Netmask /etc/dhcpcd.conf 設定 | static ip_address="<argIpAddress>/<argNetmask prefix length>" |
| 若設定 gateway 在 /etc/dhcpcd.conf 設定 | static routers="<argGateway>" |
| 若設定 IPv4 DNS 在 /etc/dhcpcd.conf 設定，兩組 DNS 中間用空格區分 | static domain_name_servers="<argDns1> <argDns2>" |
| 執行指令 | ifconfig eth0 down<br>ifconfig eth0 up |

  - **判斷 isDHCP 值為 1, DHCP**

| | |
|---|---|
| /etc/dhcpcd.conf | 清掉靜態的 LAN setting (IP, Netmask, gateway, DNS) |
| 若 nDnsType 值為 1 則設定 IPv4 DNS 在 /etc/dhcpcd.conf 設定，兩組 DNS 中間用空格區分 | static domain_name_servers="<argDns1> <argDns2>" |
| 執行指令 | ifconfig eth0 down<br>ifconfig eth0 up |

- 將數值存入 DB system-data.db 的 **LAN_Temp** 表中以下欄位。

| # | 欄位 | 變數 | 說明 |
|---|---|---|---|
| 1 | lanSettingType | isDHCP | 值為 0 存入"static"<br>值為 1 存入"dhcp" |
| 2 | ipAddress | argIpAddress | IPv4 format |
| 3 | Netmask | argNetmask | IPv4 format |

|  |  |  |  |
|---|---|---|---|
| 4 | Gateway | argGateway | IPv4 format |
| 5 | primaryDNS | argDns1 | IPv4 format |
| 6 | secondaryDNS | argDns2 | IPv4 format |
| 7 | dnsType | nDnsType | 0: Auto, 1: Manual |

## 3.2 setDatetime

函數

int **setDatetime** (int isPermanent, int enableNTPServer, char* strServerAddress, char* strDatetime)

參數

1. isPermanent
   0: temporary
   1: permanent
2. enableNTPServer
   0: disable
   1: enable
3. strServerAddress
4. strDatetime: the value should be “YYYY/MM/DD hh:mm:ss”.

回傳值

0: Fail
1: Success

內容

此 API 是 libDatetimeSetting.so 檔設定 NTP Server 的 API，系統環境是 Debian。

**<修改系統設定檔之前>**

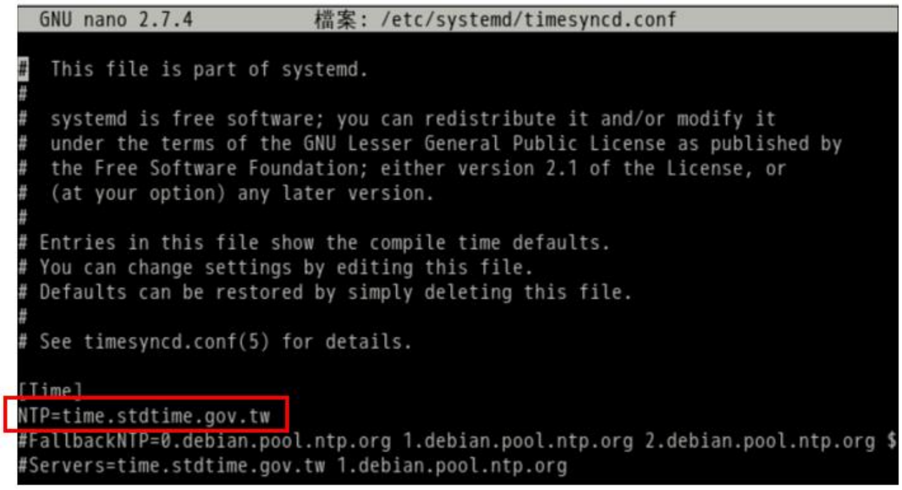
將數值更新到 DB CWM_LanSetting_Info table。

**<修改系統設定檔>**

根據<enable NTP server>參數進行 Date Time 的設定。

1. 若<enable NTP server>為 1，Enable NTP server
   A. 設定 NTP server, 修改檔案 /etc/systemd/timesyncd.conf, 檔案內容格式為

```
[Time]
NTP={NTP Server}
```



```
GNU nano 2.7.4              檔案: /etc/systemd/timesyncd.conf

#  This file is part of systemd.
#
#  systemd is free software; you can redistribute it and/or modify it
#  under the terms of the GNU Lesser General Public License as published by
#  the Free Software Foundation; either version 2.1 of the License, or
#  (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
#
# See timesyncd.conf(5) for details.

[Time]
NTP=time.stdtime.gov.tw
#FallbackNTP=0.debian.pool.ntp.org 1.debian.pool.ntp.org 2.debian.pool.ntp.org $
#Servers=time.stdtime.gov.tw 1.debian.pool.ntp.org
```

B. 設定完 NTP server 後，再利用以下指令將 NTP 功能開啟，開啟後系統即會去檔案所指定的位置進行 NTP 的時間同步

$ timedatectl set-ntp true

C. (補充說明) NTP 同步的間隔在 32 秒~2048 秒間(為 default 值)

https://www.freedesktop.org/software/systemd/man/timesyncd.conf.html

2. 若<enable NTP server>為 0，Disable NTP server

A. 利用以下指令將 NTP 功能關閉，設定完成系統就會將 NTP 給關閉，不再自動與 NTP 做時間同步。

$ timedatectl set-ntp false

B. 利用以下指令設定系統 date time.

$ date -s '{Datetime}'

C. (補充說明)若要手動調整時間，必須先將 NTP 關閉，不然時間又會同步回來

**<若無法修改系統設定檔>**

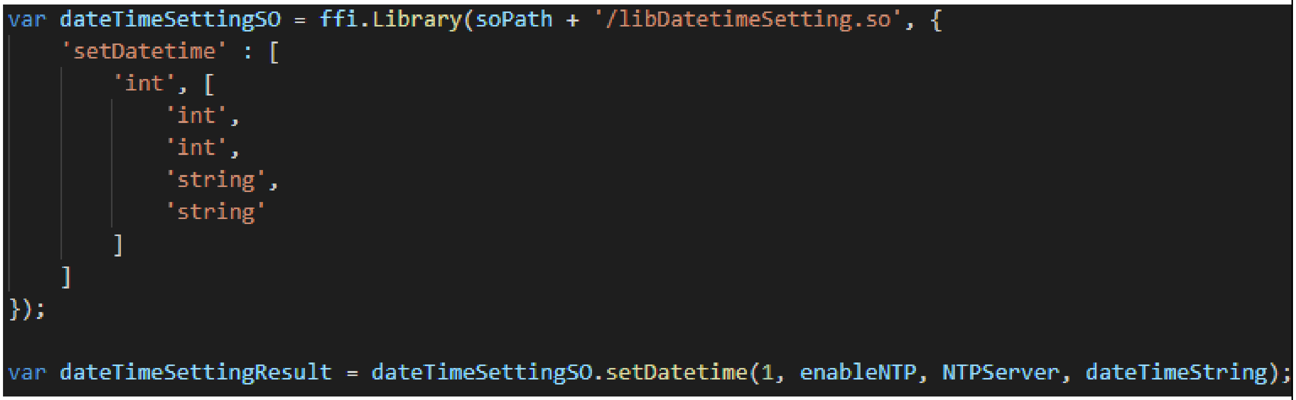
將 DB CWM_DateTimeSetting_Info table 的數值還原。

.so 檔 (.c 檔的 code) setDatetime、setTimezone、setNTPDDisable 會寫在同個.so 檔裡

```
#include <stdio.h>
#include <stdlib.h>

int setDatetime(int isPermanent, int enableNTPServer, char*
strServerAddress, char* strDatetime) {

  // 目前只寫了殼，實作邏輯待補
  return 1;

}

int setTimezone(int isPermanent, char* strTZName) {
  // 目前只寫了殼，實作邏輯待補

  return 1;
}

Int setNTPDDisable() {
  // 目前只寫了殼，實作邏輯待補

  return 1;
}
```

Nodejs use .so

需先安裝 ffi 套件



```
var dateTimeSettingSO = ffi.Library(soPath + '/libDatetimeSetting.so', {
    'setDatetime' : [
        'int', [
            'int',
            'int',
            'string',
            'string'
        ]
    ]
});

var dateTimeSettingResult = dateTimeSettingSO.setDatetime(1, enableNTP, NTPServer, dateTimeString);
```

| **指令 in Debian** | **指令 in embedded linux** |
|---|---|
| **設定時間**<br>date -s ‘YYYY/MM/DD hh:mm:ss’ | **[格式需修改]**<br>**設定時間**<br>~~date -s ‘YYYY/MM/DD hh:mm:ss’~~<br>date -s ‘YYYY-MM-DD hh:mm:ss’ |
| **設定 ntp**<br>利用 timedatectl 套件<br>1. nano /etc/systemd/timesyncd.conf<br>2. timedatectl set-ntp true | **[需要安裝套件 timedatectl]**<br>**設定 ntp**<br>無 timedatectl 可使用，但目前不知如何安裝套件，暫無替代方案。 |

## 3.3 setTimezone

函數

int **setTimezone**(int isPermanent, char* strTimezone, int isdst, int offsetSeconds, int startMonth, int startWeek, int startDay, char* startTime, int endMonth, int endWeek, int endDay, char* endTime)

參數

1. isPermanent
   0: temporary
   1: permanent
2. strTimezone: the value should be timezone id or timezone name.
3. isdst
   0: GMT time
   1: daylight saving time
4. offsetSeconds
5. startMonth
6. startWeek
7. startDay
8. startTime
9. endMonth
10. endWeek
11. endDay
12. endTime

回傳值

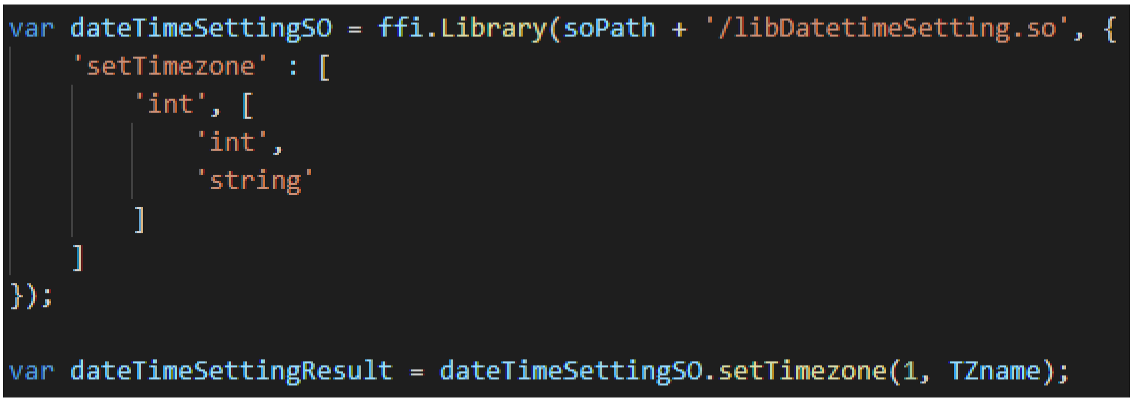
0: Fail
1: Success

內容

此 API 是 libDatetimeSetting.so 檔設定 Datetime 的 API。
利用以下指令設定 time zone, <timezone>為對應的時區
$ ln -sf /usr/share/zoneinfo/Etc/<timezone> /etc/localtime

Nodejs use .so

需先安裝 ffi 套件



```
var dateTimeSettingSO = ffi.Library(soPath + '/libDatetimeSetting.so', {
    'setTimezone' : [
        'int', [
            'int',
            'string'
        ]
    ]
});

var dateTimeSettingResult = dateTimeSettingSO.setTimezone(1, TZname);
```

## 3.7 getSystemInfo

| 函數 |
|---|
| char ***getSystemInfo** () |
| 參數 |
| 無 |
| 回傳值 |
| [“<device version>”, “<mac address>”, “<IP mode>”, “<IP address>”, “ <Netmask>”, “<Gateway>”, “<DNS server 1>”, “<DNS server 2>”, “<USB usage>”, “<Micro SD usage>”, “<Firmware version>”, “<NTP Server>”, “<Datetime>”, “<Timezone>”]<br>依陣列的索引來放置數值並回傳，若未取得數值，則給”Not found” |
| 內容 |
|  |

## 3.9 setAdminPassword

| 函數 |
|---|
| int **setAdminPassword**(char* password) |
| 參數 |
| 1. password |
| 回傳值 |
| 0: Fail<br>1: Success |
| 內容 |
|  |

## 3.10 save

| 函數 |
|---|
| int **save**() |
| 參數 |
| 無 |
| 回傳值 |
| 0: Fail<br>1: Success |
| 內容 |
|  |

## 3.11 download

| 函數 |
|---|
| int **download** (char* strIpAddr, int nPort, char* strlogin, char* strPassword, char* strSrcFilePath, char* strFileName, char* strDesFilePath) |
| 參數 |
| 1. strIpAddr<br>2. nPort<br>3. strlogin<br>4. strPassword<br>5. strSrcFilePath<br>6. strFileName<br>7. strDesFilePath |
| 回傳值 |
| integer type<br>1. Success<br>2. Error: FTP server does not exist<br>3. Error: FW file does not exist<br>4. Error: Destination path does not existent<br>5. Error: Timeout<br>6. Error: Login Fail |
| 內容 |
| Only support FTP portal (non-sFTP and non-tFTP) |

## 3.12 checkHeaderrAndPayload

| 函數 |
|---|
| int **checkHeaderAndPayload** (char* strFilePath, char* strFileName) |
| 參數 |
| 1. strFilePath<br>2. strFileName |
| 回傳值 |
| integer type<br>1. Valid<br>2. Invalid: FW file does not exist<br>3. Invalid: Header is incorrect<br>4. Invalid: Payload is incorrect |
| 內容 |
| 1. Header<br>model name is dnh100<br>2. Payload<br>decryption via AES256-CTR |

## 3.13 unTar

| 函數 |
|---|
| int **unTar** (char* strFilePath, char* strFileName, char* strDesFilePath) |
| 參數 |
| 1. strFilePath<br>2. strFileName<br>3. strDesFilePath |
| 回傳值 |
| integer type<br>1. Success<br>2. Error: File does not exist<br>3. Error: File is not tar format<br>4. Error: Destination path does not exist. |
| 內容 |
| Support .tar and .tar.gz format |

## 3.14 getFirmwareUpgradeStatus

| 函數 |
|---|
| int **getFirmwareUpgradeStatus**() |
| 參數 |
| N/A |
| 回傳值 |
| integer type<br>0. Firmware upgrade is Idle<br>1. Firmware upgrade is busy<br>2. Error: DB does not exist.<br>3. Error: Table does not exist. |
| 內容 |
| Get firmware upgrade status from system-data.db |

## 3.15 setFirmwareUpgradeStatus

| 函數 |
|---|
| int **setFirmwareUpgradeStatus**(int nStatus) |
| 參數 |
| nStatus<br>- data type: integer<br>- content:<br>0. Idle<br>1. Busy |
| 回傳值 |
| integer type<br>1. Success<br>2. Error: DB does not exist<br>3. Error: Table does not exist.<br>4. Firmware upgrade is busy.<br>5: Error: input parameter is invalid. |
| 內容 |
| Set firmware upgrade status into system-data.db |

## 3.16 getFirmwareVersion

| 函數 |
|---|

| |
|---|
| char* **getFirmwareVersion**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br>“<x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of firmware does not exist.<br>“4”: Error: Table of firmware has no data.<br><br>**e.g.**<br>**normal case:**<br>“1.0.0”<br>**abnormal case:**<br>“2”, “3” or “4” |
| 內容 |
| Get current firmware version of DNH-100 without svn version(SDK) |

## 3.17 getFirmwareVersionFull

| 函數 |
|---|
| char* **getFirmwareVersionFull**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br>“<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of firmware does not exist.<br>“4”: Error: Table of firmware has no data.<br><br>**e.g.**<br>**normal case:**<br>“1.0.0.246”<br>**abnormal case:**<br>“2”, “3” or “4” |
| 內容 |
| Get current firmware version of DNH-100 with svn version(SDK) |

## 3.18 getHWVersion

| 函數 |
|---|
| char* **getHWVersion**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br>“<A-Z><1-999>”<br><br>**e.g.**<br>**normal case:**<br>“A1”<br>**abnormal case: (The hardware version wasn’t assigned)**<br>“” |
| 內容 |
| Get current hardware version of DNH-100 |

## 3.19 getNCVersion

| 函數 |
|---|
| char* **getNCVersion**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br>“<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of Start_NC does not exist.<br>“4”: Error: Table of Start_NC has no data.<br><br>**e.g.**<br>**normal case:**<br>“1.0.0.1”<br>**abnormal case:**<br>“2”, “3” or “4” |
| 內容 |

| |
|---|
| Get current active version of nuclias connect |

## 3.20 getSSOStatus

| 函數 |
|---|
| char* **getSSOStatus**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br><br>“Enabled”<br>“Enabled (time out)<br>“Enable ( Nuclias Server is under maintenance )”<br>“Disabled”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of SSO does not exist.<br>“4”: Error: Table of SSO has no data. |
| 內容 |
| Get the configuration of single sign on |

## 3.21 getNCStatus

| 函數 |
|---|
| int **getNCtatus**() |
| 參數 |
| N/A |
| 回傳值 |
| integer type<br><br>0: stop<br>1: running |
| 內容 |
| Get current nuclias connect status |

## 3.22 getDDPv5ClientVersion

| 函數 |
|---|
| char* **getDDPv5ClientVersion**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br>“<x.x.x.x>”<br>“2”: Error: DB of system-data.db does not exist.<br>“3”: Error: Table of DDPv5 does not exist.<br>“4”: Error: Table of DDPv5 has no data.<br><br>**e.g.**<br>**normal case:**<br>“1.0.0.17”<br>**abnormal case:**<br>“2”, “3” or “4” |
| 內容 |
| Get current firmware version of DNH-100 |

## 3.23 setNCStatus

| 函數 |
|---|
| int **setNCStatus**(int nStatus) |
| 參數 |
| nStatus<br>- Data Type: integer<br>- Content:<br>0: stop<br>1: running |
| 回傳值 |
| integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of Start_NC does not exist.<br>4: Error: Table of Start_NC has no data.<br>5: Error: input parameter is invalid. |

| |
|---|
| 內容 |
| update nuclias connect status |

## 3.24 setSSOStatus

| |
|---|
| 函數 |
| int **setSSOStatus**(char* strStatus) |
| 參數 |
| strStatus<br>- Data Type: string<br>- Content:<br>“Enabled” or<br>“Enabled (time out)” or<br>“Enable ( Nuclias Server is under maintenance )” or<br>“Disabled” |
| 回傳值 |
| integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist<br>3: Error: Table of SSO does not exist.<br>4: Error: Table of SSO has no data. |
| 內容 |
| update single sign on status |

## 3.25 setDDPv5ClientVersion

| |
|---|
| 函數 |
| int **setDDPv5ClientVersion**(char* strVersion) |
| 參數 |
| strVersion<br>- Data Type: string<br>- Content: “<x.x.x.x>” |
| 回傳值 |
| integer type<br>1: Success<br>2: Error: DB of system-data.db does not exist.<br>3: Error: Table of DDPv5 does not exist. |

| |
|---|
| 4: Error: Table of DDPv5 has no data. |
| 內容 |
| Set DDPv5 client version of DNH-100 |

## 3.26 addEventLog

| |
|---|
| 函數 |
| int **addEventLog**(char* strMessage, int nEventType) |
| 參數 |
| strMessage<br>- Data Type: string<br>- Content: "….."<br><br>nEventType<br>- Data Type: integer<br>- Content:<br>1: Device Management<br>3: Duplicate Task<br>4: Timeout Task<br>5: Invalid HTTP Messagge<br>6: Initialization<br>9: Firmware Upgrade |
| 回傳值 |
| integer type<br>1: Success<br>2: Error: config-data.db doesn't exist<br>3: Error: config-data.db(CWM_Database_Setting) doesn't exist.<br>4: Error: log-data.db doesn't exist.<br>5: Error: log-data.db(CWM_Log_SystemEvent) doesn't exist.<br>6: Error: external syslog server doesn't exist. |
| 內容 |
| Based on auto backup setting to add event log in ramdisk and external syslog server |

## 3.27 runDnssd

| |
|---|
| 函數 |
| int **runDnssd**() |

參數

N/A

回傳值

Integer

0: failed

1: successful

內容

The runDnssd function maintains the file **/etc/avahi/services/http.service**.

The content of http.service, e.g.,

```
<?xml version="1.0" standalone='no'?><!--*-nxml-*-->
<!DOCTYPE service-group SYSTEM "avahi-service.dtd">

<!--
   This file is part of avahi.

   avahi is free software; you can redistribute it and/or modify it
   under the terms of the GNU Lesser General Public License as
   published by the Free Software Foundation; either version 2 of the
   License, or (at your option) any later version.

   avahi is distributed in the hope that it will be useful, but
   WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
   General Public License for more details.

   You should have received a copy of the GNU Lesser General Public
   License along with avahi; if not, write to the Free Software
   Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA
   02111-1307 USA.
-->

<!-- See avahi.service(5) for more information about this configuration file -->

<service-group>

   <name replace-wildcards="yes">%h</name>
   <service>
      <type>_http._tcp</type>
      <port>80</http>
      <txt-record>manufacturer=D-Link</txt-record>
      <txt-record>type=controller</txt-record>
      <txt-record>line=Nuclias</txt-record>
      <txt-record>series=DNH</txt-record>
      <txt-record>model=DNH-100</txt-record>
      <txt-record>hwVer=A1</txt-record>
      <txt-record>fwVariant=Default</txt-record>
      <txt-record>fwVer=1.0.1.3</txt-record>
      <txt-record>hwId=AA:BB:CC:11:EB:23</txt-record>
      <txt-record>uuid=a4aff40093a8</txt-record>
      <txt-record>name=DNH-100-EB23</txt-record>
      <txt-record>state=FactoryDefault</txt-record>
      <txt-record>fota=UpToDate</txt-record>
   </service>

</service-group>
```

p.s. %h in c code should be %%hhh

All content of http.service is generated by the function, and the data of the changeable records are retrieved from CLI command 'vendorstorage', so api or system-data.db as below list.

| Name | Value from |
|---|---|
| **hwVer** | The value of CLI command 'vendorstorage read HW_VER' |
| **fwVer** | The value of getFirmwareVersionFull function in libDB.so |
| **hwId** | The value of CLI command 'vendorstorage read LAN_MAC' |
| **uuid** | The value of CLI command 'vendorstorage read DEVICE_UUID' |
| **name** | The value of CLI command 'vendorstorage read LAN_MAC', and the last 4 MAC characters with uppercase appended to "DNH-100-" or "DNH-200-". E.g. "DNH-200-EB23" |
| **state** | The integer value of device_state field in System table in system-data.db.<br>0: "FactoryDefault"<br>1: "Operational" |
| **fota** | The integer value of fota_state field in Firmware table in system-data.db.<br>0: "UpToDate"<br>1: "Available"<br>2: "Mandatory" |

## 3.28 getDeviceUUID

| 函數 |
|---|
| char* **getDeviceUUID**() |
| 參數 |
| N/A |
| 回傳值 |
| string type<br><br>**e.g.**<br>**normal case:**<br>“123456789112”<br>**abnormal case: (The return value is empty)**<br>“” |
| 內容 |
| Get current device uuid of DNH-100 |

## 3.29 setFOTASetting

| Function |
|---|
| int **setFOTASetting**(int nEnable, int nWeekday, int nHour, int nMinute, int nUpdateBetaFw) |
| Input |
| 1. nEnable<br>Integer<br>0: Disabled<br>1: Enabled<br>2. nWeekday<br>Integer, 1 ~ 8<br>1: Monday<br>2: Tuesday<br>3: Wednesday<br>4: Thursday<br>5: Friday<br>6: Saturday<br>7: Sunday<br>8: Every day<br>3. nHour |

Integer, 0 ~ 23

4. nMinute

   Integer, 0 ~ 59

5. nUpdateBetaFw

   Integer

   0: Disabled

   1: Enabled

Note: If the “nEnable” is 0, the “nWeekday”, “nHour”, “nMinute”, and “nUpdateBetaFw” will not be updated to the database.

**Return**

Integer

1: Success

2: Error: Invalid Input Parameter

3: Error: The system-data.db does not exist.

4: Error: The Firmware table does not exist.

5: Error: The Firmware table has no data.

**Description**

Record the FOTA setting in the database and start the FOTA script. The DNH-200 uses the Linux’s ***“crond”*** module to arrange the schedule.

Step1. First of all, stop ***“crond”*** daemon

Step2. Modify schedule script.

- Auto
  - Enabled: modify the “**/opt/crontabs/root**” file as below.
    - ***<nMinute> <nHour> * * <nWeekday> <executable file>***
  - Disabled: remove the above content from “**/opt/crontabs/root**”

Step3. Final, executes “***crond -c /opt/crontabs***” commands if the Auto is enabled.

Note.

1. The definition of crontabs

```
---------------------- minute (0 - 59)
|    ----------------- hour (0 - 23)
|    |    ------------ day of month (1 - 31)
|    |    |    --------- month (1 - 12)
|    |    |    |    ---- day of week (0 - 6) (Sunday=0 or 7)
|    |    |    |    |
*    *    *    *    *    <command to be executed>
```

e.g.
1.1 every 5 minute : */5 * * * * /usr/sbin/dnh/fotaClient
1.2 every 5 hour : 0 */5 * * * /usr/sbin/dnh/fotaClient
1.3 every day 00:00 : 0 0 * * * /usr/sbin/dnh/fotaClient
1.4 every Monday 02:30: 30 2 * * 1 /usr/sbin/dnh/fotaClient

2. The example of crond
2.1 Background command: ***crond -c /opt/crontabs***
2.2 Foreground command: ***crond -f -c /opt/crontabs***

## 3.30 getFOTASetting

| Function |
|---|
| char* **getFOTAAutoUpdate**() |
| Input |
| N/A |
| Return |
| String<br>The JSON format is as below<br><br>{<br>"result" : <result>,<br>// 1: Success<br>// 2: Error: The system-data.db does not exist.<br>// 3: Error: The Firmware table does not exist.<br>// 4: Error: The Firmware table has no data.<br>// if the result is 1(Success)<br>"enable" : <enable>,<br>"weekday" : <weekday>,<br>"hour" : <hour>,<br>"minute" : <minute>,<br>"updateBetaFw" : <updateBetaFw>,<br>} |
| Description |
| Get FOTA auto update settings from the database. |

## 3.31 startFOTAScript

| Function |
|---|
| int **startFOTAScript**() |
| Input |
| N/A |
| Return |
| Integer<br>1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| Description |
| The FOTA script will be executed if the auto is enabled. |

## 3.32 analyzeFirmwareInfo

| Function |
|---|
| char* **analyzeFirmwareInfo**(int nMajor, int Minor, int nRev, char* strFirmwareInfoFromFOTA, int nUpdateBetaFw) |
| Input |
| 1. nMajor<br>Integer, 1~99<br>2. nMinor<br>Integer, 0~999<br>3. nRev<br>Integer, 0~99999<br>4. strFrimwareInfoFromFOTA<br>char*<br>e.g.<br>"{"stable":{"ver":{"major":"3","minor":"1","rev":"0"},"url":"https://fw.fota.dlink.com/COVR/COVR-3902/Ax/Default/stable/stable.bin","release_note":"Security improvement.","ts":1519962505},"beta":{"ver":{"major":"3","minor":"2","rev":"8"},"url":"https://fw.fota.dlink.com/COVR/COVR-3902/Ax/Default/beta/beta.bin","release_note":"Security improvement.","ts":1519962505},"intermediate":{"ver":{"major":"2","minor":"2","rev":"0"},"url":"https://fw.fota.dlink.com/COVR/COVR-3902/Ax/Default/intermediate/intermediate.bin","release_note":"Security |

improvement.","ts":1519962505},"enforced_ver":{"major":"2","minor":"0","rev":"0"}}"

5. nUpdateBetwFw

   Integer

   0: Disabled

   1: Enableed

Note:

The format of the DNH-200 firmware version is v<Major>.<Minor>.<Rev>

**Return**

String

JSON format is as below

```
{
    "result" : <result>,
                                            // 1: Success
                                            // 2: Error: Invalid Input Parameter
    "firmwareStatus" : <firmwareStatus>,
                                            // 1: UpToDate
                                            // 2: Available
                                            // 3: Mandatory

    // if the ‘firmwareStatus’ is 2(available) or 3(Mandatory), the following items are the
    // target firmware information.
    "major" : <major>,
    "minor" : <minor>,
    "rev" : <rev>,
    "url" : "<url>",
    "releaseNote" : "< releaseNote >"
}
```

And save the releaseNote in /userdata/fwTmp/firmwareReleaseNote.txt

**Description**

Analysis firmware information and call the soapi **“setFotaStatus”** to update the firmware status and DNS-SD

## 3.33 setFOTAUpdateStatus

**Function**

| |
|---|
| int **setFOTAUpdateStatus**(int nUpdateStatus) |
| Input |
| Integer<br>0: Idle<br>1: Running |
| Return |
| 1: Success<br>2: Error: Invalid Input Parameter<br>3: Error: The system-data.db does not exist.<br>4: Error: The Firmware table does not exist.<br>5: Error: The Firmware table has no data. |
| Description |
| Record the FOTA firmware update status in the database |

## 3.34 getFOTAUpdateStatus

| Function |
|---|
| int **getFOTAUpdateStatus**() |
| Input |
| N/A |
| Return |
| Integer<br>0: Idle<br>1: Running<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| Description |
| Get the FOTA firmware update status from the database |

## 3.35 getHddFlag

| Function |
|---|
| int **getHddFlag**(void) |
| Input |
| N/A |
| Return |
| Integer |

| |
|---|
| 0: There is no HDD mounted on DNH-200.<br>1: HDD is new or the formation is not valid for DNH-200.<br>2: HDD is failed to initialize on DNH-200.<br>3: HDD is ok to use for DNH-200 with the partition sizes for 1.0.0 version.<br>-1: error status |
| Description |
| Detect below files to know HDD status.<br>1. /tmp/newhdds: HDD mounted.<br>2. /tmp/newhddu: HDD initialized error.<br>3. /tmp/newhdd1: HDD partition 1 hdd_NC mounted.<br>4. /tmp/newhdd2: HDD partition 2 hdd_NVR1 mounted.<br>5. /tmp/newhdd3: HDD partition 3 hdd_NVR2 mounted.<br>Only 3 partitions are ALL mounted, HDD is ready.<br><br>There is no file existed, return 0.<br>There is only newhdds existed, return 1.<br>There are newhdds and newhddu existd, return 2.<br>There are newhdds, newhdd1, newhdd2 and newhdd3 all existed, also newhddu is not existed, return3.<br>Others, error status, NC/NVR 3 partitions are not all mounted, umount NC/NVR partition(s) for doing initialization. Return -1. |

## 3.36 setHddFlag

| Function |
|---|
| int **setHddFlag**(int nResponse) |
| Input |
| nResponse:<br>0: No, do not initialize the HDD.<br>1: Yes, initialize the HDD. |
| Return |
| Integer<br>0: Failed<br>1: Successful<br>2: Response value is invalid |
| Description |
| Validate HDD status should be 1 or -1, if it is not 1 or -1, return 0.<br>If the nResponse is 1, run **/etc/hdd_init.sh** to initialize the HDD. After the initialization finished, |

| |
|---|
| check the HDD status, 3 means successful, return 1. Otherwise, return 0.<br>If the nResponse is 0, do nothing and return 1. |

## 3.37 getFirmwareInfo

| Function |
|---|
| char* **getFirmwareInfo**() |
| Input |
| N/A |
| Return |
| String<br>The JSON format is as below<br><br>{<br>"result" : <result>,<br>// 1: Success<br>// 2: Error: The system-data.db does not exist.<br>// 3: Error: The Firmware table does not exist.<br>// 4: Error: The Firmware table has no data.<br>// if the result is 1(Success)<br>"version" : "<version>",<br>"releaseNote" : "<releaseNote>",<br>"updateDateTime" : "<updateDateTime>"<br>} |
| Description |
| Get current DNH firmware information. |

## 3.38 setFirmwareReleaseNote

| Function |
|---|
| int **setFirmwareReleaseNote**(char* strReleaseNote) |
| Input |
| N/A |
| Return |
| integer type |

|  |
|---|
| 1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| Description |
| Record the firmware release note in the database |

## 3.39 setFirmwareDateTime

| Function |
|---|
| int **setFirmwareDateTime**(char* strDateTime) |
| Input |
| N/A |
| Return |
| integer type<br>1: Success<br>2: Error: The system-data.db does not exist.<br>3: Error: The Firmware table does not exist.<br>4: Error: The Firmware table has no data. |
| Description |
| Record the update date time |

## 3.40 getLANSetting

|  |
|---|
|  |
| char ***getLANSetting**() |
| **Parameter** |
| N/A |
| **Return value** |
| JSON string type:<br>{<br>"ipmode": "<IPv4 mode>",<br>"ip": "<IPv4 address>",<br>"netmask": "<IPv4 netmask>",<br>"gateway": "<IPv4 gateway>",<br>"dns1": "<IPv4 DNS 1>", |

```
    "dns2": "<IPv4 DNS 2>",
    "tmpIp": "<IPv4 address in LAN_Temp>",
    "tmpNetmask": "<IPv4 netmask in LAN_Temp>",
    "tmpGateway": "<IPv4 gateway in LAN_Temp>",
    "tmpDns1": "<IPv4 DNS 1 in LAN_Temp>",
    "tmpDns2": "<IPv4 DNS 2 in LAN_Temp>",
    "dnsType": <IPv4 DNS type>
}
```

**Content**

JSON 資料項目對應資料如下。

| # | 項目 | 資料 |
|---|---|---|
| 1 | ipmode | [system-data.db].LAN_Temp.lanSettingType |
| 2 | ip | 使用以下指令取 IPv4 address, 取 inet addr:的值<br># ifconfig \| grep "inet addr" |
| 3 | netmask | 使用以下指令取 IPv4 netmask, 取 Mask:的值<br># ifconfig \| grep "inet addr" |
| 4 | gateway | 使用以下指令取 IPv4 default gateway<br># ip route \| grep 'default via' \| awk '{print $3}' |
| 5 | dns1 | 使用以下指令取 IPv4 primary DNS，第一組為 primary DNS，若未取得 IPv4 DNS 資料則 primary DNS 為空，dns1 項目數值為空字串。第二組為 secondary DNS，若未取得 IPv4 DNS 資料或者沒有第二組 IPv4 DNS 則 secondary DNS 為空，dns2 項目數值為空字串。<br># cat /etc/resolv.conf \| grep nameserver \| grep '\.' \| head -n 2 \| awk '{print $2}' |
| 6 | dns2 | 參考上方欄位。 |
| 7 | tmpIp | [system-data.db].LAN_Temp.ipAddress |
| 8 | tmpNetmask | [system-data.db].LAN_Temp.Netmask |
| 9 | tmpGateway | [system-data.db].LAN_Temp.Gateway |
| 10 | tmpDns1 | [system-data.db].LAN_Temp.primaryDNS |
| 11 | tmpDns2 | [system-data.db].LAN_Temp.secondaryDNS |
| 12 | dnsType | [system-data.db].LAN_Temp.dnsType |

## 3.41 setLANSettingIpv6

**Function**

int **setLANSettingIpv6**(int isPermanent, char *argProvision, char *argIpv6Address, int nPrefixLength, char *argGateway, int nDnsType, char *argDns1, char *argDns2, int flagUpdateDaa)

**Parameter**

1. **isPermanent**: integer, 0: LAN setting is temporary, 1: LAN setting is permanent.
2. **argProvision**: string, "Static", "Auto", "Local"
3. **argIpv6Address**: string, IPv6 address format, 若要使用當前系統設定值, 可帶"-1".
4. **nPrefixLength**: integer, 1~128, 若要使用當前系統設定值, 可帶-1.
5. **argGateway**: string, IPv6 address format, 若要使用當前系統設定值, 可帶"-1", 若不設定 gateway, 帶"".
6. **nDnsType**: integer, 0: Auto, 1: Manual.
7. **argDns1**: string, IPv6 address format, 若要使用當前系統設定值, 可帶"-1", 若不設定 gateway, 帶"".
8. **argDns2**: string, IPv6 address format, 若要使用當前系統設定值, 可帶"-1", 若不設定 gateway, 帶"".
9. **flagUpdateDaa**: integer, for static setting, 0: do not sync IPv6 address to DAA, 1: sync IPv6 to DAA. 當設定 Auto configuration 或是 Link-local, 參數值給 0.

**Return value**

Integer type:

0: Failed

1: Successful

2: Invalid permanent type

3: Invalid IPv6 provision type

4: Invalid IPv6 address

5: Invalid prefix length

6: Invalid gateway

7: Invalid DNS type

8: Invalid primary DNS

9: Invalid secondary DNS

10: Invalid flagUpdateDaa

**Content**

- 驗證參數

- 若 isPermanent 值為 1, 表示是永久性設定，將 IPv6 設定值存入 DB system-data.db 的 **LAN_IPv6** 表。

| # | 欄位 | 變數 | 說明 |
|---|---|---|---|
| 1 | provision | argProvision | "Static", "Auto", "Local" |
| 2 | ipv6Address | argIpv6Address | IPv6 format |
| 3 | prefixLength | nPrefixLength | 1~128 |
| 4 | gateway | argGateway | IPv6 format |
| 5 | primaryDNS | argDns1 | IPv6 format |
| 6 | secondaryDNS | argDns2 | IPv6 format |
| 7 | dnsType | nDnsType | 0: Auto, 1: Manual |

以下動作不管 isPermanent 值為 0 或 1 都會進行。

■ 當 argProvision 值為"Static"且 flagUpdateDaa 值為 1，將 argIpv6Address 資料同步到 DAA，DAA 的欄位為[system-data.db].System.device_access_address。
當 argProvision 值不是"Static"，忽略 flagUpdateDaa。

■ 檢查要做 IPv6 的設定與當前的 IPv6 設定相同，若相同則不進行後續設定，回傳 1 (Success)。

■ 設定 IPv6 LAN setting
在/etc/dhcpcd.conf 的初始內容中刪去 slaac private 並加入
slaac hwaddr
interface eth0

◆ **判斷 argProvision 值為"Static"**

| | |
|---|---|
| 設定 IP 與 prefix length 在 /etc/dhcpcd.conf 設定 | ipv6ra_noautoconf<br>noipv6rs<br>static ip6_address="<argIpv6Address>/<nPrefixLength>" |
| 若設定 primary DNS 或 secondary DNS 在 /etc/resolv.conf.head 設定，若無設定則清空 | nameserver "<argDns1>"<br>nameserver "<argDns2>" |
| /etc/sysctl.conf 設定 | net.ipv6.conf.eth0.autoconf = 0<br>net.ipv6.conf.all.autoconf = 0<br>net.ipv6.conf.default.autoconf = 0<br><br>net.ipv6.conf.all.accept_ra = 0 |

| | |
|---|---|
| | net.ipv6.conf.default.accept_ra = 0<br>net.ipv6.conf.eth0.accept_ra = 0 |
| /etc/init.d/S99dnh 加入設定 | 增加以下這行，位置在 is_recovery() 這行之前<br>sysctl -pq |
| 執行指令 | sysctl -pq<br>ifconfig eth0 down<br>ifconfig eth0 up |
| 若設定 gateway 執行指令新增 gateway | ip -6 route add "<argGateway>" dev eth0<br>ip -6 route add default via "<argGateway>" |

- **判斷 argProvision 值為"Auto"**

| | |
|---|---|
| /etc/dhcpcd.conf 設定 | 清除 Static 及 Link-local 的設定 |
| 若 nDnsType 為 1 則設定 primary DNS 或 secondary DNS 在 /etc/resolv.conf.head 設定，若無設定則清空 | nameserver "<argDns1>"<br>nameserver "<argDns2>" |
| /etc/sysctl.conf 設定 | N/A |
| /etc/init.d/S99dnh 加入設定 | 增加以下這行，位置在 is_recovery() 這行之前<br>sysctl -pq |
| 執行指令 | sysctl -pq<br>ifconfig eth0 down<br>ifconfig eth0 up |

- **判斷 argProvision 值為"Local"**

| | |
|---|---|
| /etc/dhcpcd.conf 設定 | ipv6ra_noautoconf<br>noipv6rs |
| /etc/resolv.conf.head 設定 | N/A |
| /etc/sysctl.conf 設定 | net.ipv6.conf.eth0.autoconf = 0<br>net.ipv6.conf.all.autoconf = 0<br>net.ipv6.conf.default.autoconf = 0 |
| /etc/init.d/S99dnh 加入設定 | 增加以下這行，位置在 is_recovery() 這行之前<br>sysctl -pq |

| | |
|---|---|
| 執行指令 | sysctl -pq<br>ifconfig eth0 down<br>ifconfig eth0 up |

■ 將數值存入 DB system-data.db 的 **LAN_IPv6_Temp** 表中以下欄位。

| # | 欄位 | 變數 | 說明 |
|---|---|---|---|
| 1 | provision | argProvision | "Static", "Auto", "Local" |
| 2 | ipv6Address | argIpv6Address | IPv6 format |
| 3 | prefixLength | nPrefixLength | 1~128 |
| 4 | gateway | argGateway | IPv6 format |
| 5 | primaryDNS | argDns1 | IPv6 format |
| 6 | secondaryDNS | argDns2 | IPv6 format |
| 7 | dnsType | nDnsType | 0: Auto, 1: Manual |

## 3.42 getLANSettingIpv6

**Function**

char ***getLANSettingIpv6**()

**Parameter**

N/A

**Return value**

JSON string type:

```
{
    "ipv6Provision": "<IPv6 provision>",
    "ipv6Address": "<IPv6 address>",
    "ipv6PrefixLength": <IPv6 prefix length>,
    "ipv6LinkLocal": "<link-local address>"
    "ipv6Gateway": "<IPv6 gateway>",
    "ipv6DnsType": <IPv6 DNS type>,
    "ipv6Dns1": "<IPv6 DNS 1>",
    "ipv6Dns2": "<IPv6 DNS 2>",
    "ipv6TmpAddress": "<IPv6 address in LAN_IPv6_Temp>",
    "ipv6TmpPrefixLength": <IPv6 prefix length in LAN_IPv6_Temp>,
    "ipv6TmpGateway": "<IPv6 gateway in LAN_IPv6_Temp>",
    "ipv6TmpDns1": "<IPv6 DNS 1 in LAN_IPv6_Temp>",
    "ipv6TmpDns2": "<IPv6 DNS 2 in LAN_IPv6_Temp>"
}
```

**Content**

JSON 資料項目對應資料如下。

| # | 項目 | 資料 |
|---|---|---|
| 1 | ipv6Provision | [system-data.db].LAN_IPv6_Temp.provision |
| 2 | ipv6Address | 使用以下指令取 IPv6 global address，不取"/<prefix length>"的部分<br># ifconfig \| grep inet6 \| grep Global \| awk '{print $3}' |
| 3 | ipv6PrefixLength | 使用以下指令取 IPv6 prefix length，只取"/"之後 prefix length 的數值<br># ifconfig \| grep inet6 \| grep Global \| awk '{print $3}' |
| 4 | ipv6LinkLocal | 使用以下指令取 IPv6 link-local address<br># ifconfig \| grep inet6 \| grep Link \| awk '{print $3}' |
| 5 | ipv6Gateway | 使用以下指令取 IPv6 default gateway<br># ip -6 route \| grep 'default via' \| awk '{print $3}' |
| 6 | ipv6DnsType | [system-data.db].LAN_IPv6_Temp.dnsType |
| 7 | ipv6Dns1 | 使用以下指令取 IPv6 primary DNS，第一組為 primary DNS，若未取得 IPv6 DNS 資料則 primary DNS 為空，ipv6Dns1 項目數值為空字串。第二組為 secondary DNS，若未取得 IPv6 DNS 資料或者沒有第二組 IPv6 DNS 則 secondary DNS 為空，ipv6Dns2 項目數值為空字串。<br># cat /etc/resolv.conf \| grep nameserver \| grep : \| head -n 2 \| awk '{print $2}' |
| 8 | ipv6Dns2 | 參考上方欄位。 |
| 9 | ipv6TmpAddress | [system-data.db].LAN_IPv6_Temp.ipv6Address |
| 10 | ipv6TmpPrefixLength | [system-data.db].LAN_IPv6_Temp.prefixLength |
| 11 | ipv6TmpGateway | [system-data.db].LAN_IPv6_Temp.gateway |
| 12 | ipv6TmpDns1 | [system-data.db].LAN_IPv6_Temp.primaryDNS |
| 13 | ipv6TmpDns2 | [system-data.db].LAN_IPv6_Temp.secondaryDNS |

## 3.43 getIpv6Provision

**Function**

char ***getIpv6Provision**()

| **Parameter** |
|---|
| N/A |
| **Return value** |
| String type:<br>4. "Static"<br>5. "Auto"<br>6. "Local" |
| **Content** |
| 取[system-data.db].LAN_IPv6_Temp.provision 的資料回傳 |

# 4. SO API in RK3328 SDK

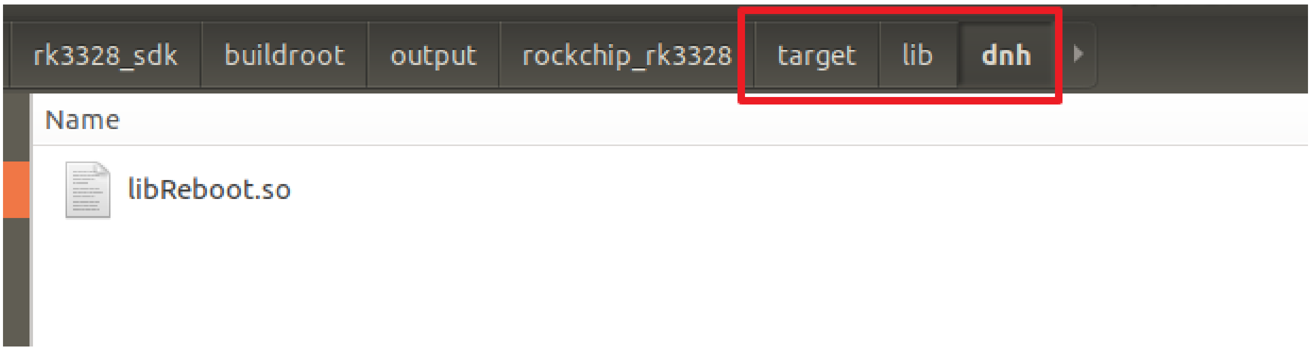
最終，SO API 要提供給 Web、CLI、DDPv5 操作，因此我們 DNH 所開發的這些 API 放置在共同的資料夾底下，以便各個介面調用 SO API。並且，SO API 要在 RK3328 核心上運行，所以在開發環境必須生成 RK3328 核心可以執行的 SO API。

我們在這個章節說明這兩個部分。

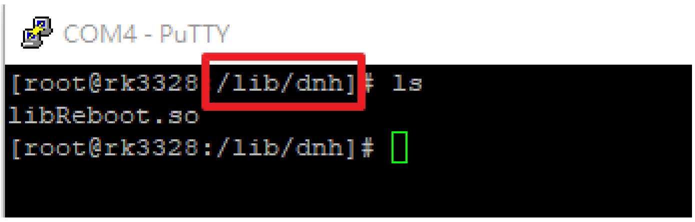
## 4.1 SO API path in SDK and Image

我們的目標是把 DNH 所用到的 SO API 放到系統存放函式庫的資料夾“lib”，並且在這個資料夾裡新增一個 DNH SO API 專屬的資料夾，取名“dnh”。

這個操作並不複雜，在 SDK 中對應到 Image 上“lib”資料夾的路徑新增“dnh”資料夾即可。
~/rk3328_sdk/buildroot/output/rockchip_rk3328/target/lib/dnh



這裡要做一個提醒，當我們新增資料夾之後，必須重製根目錄系統(./build.sh rootfs)。
在 RK3328 上運行修改後的 image，便可在根目錄“lib”資料夾底下看到我們所新增“dnh”資料夾。



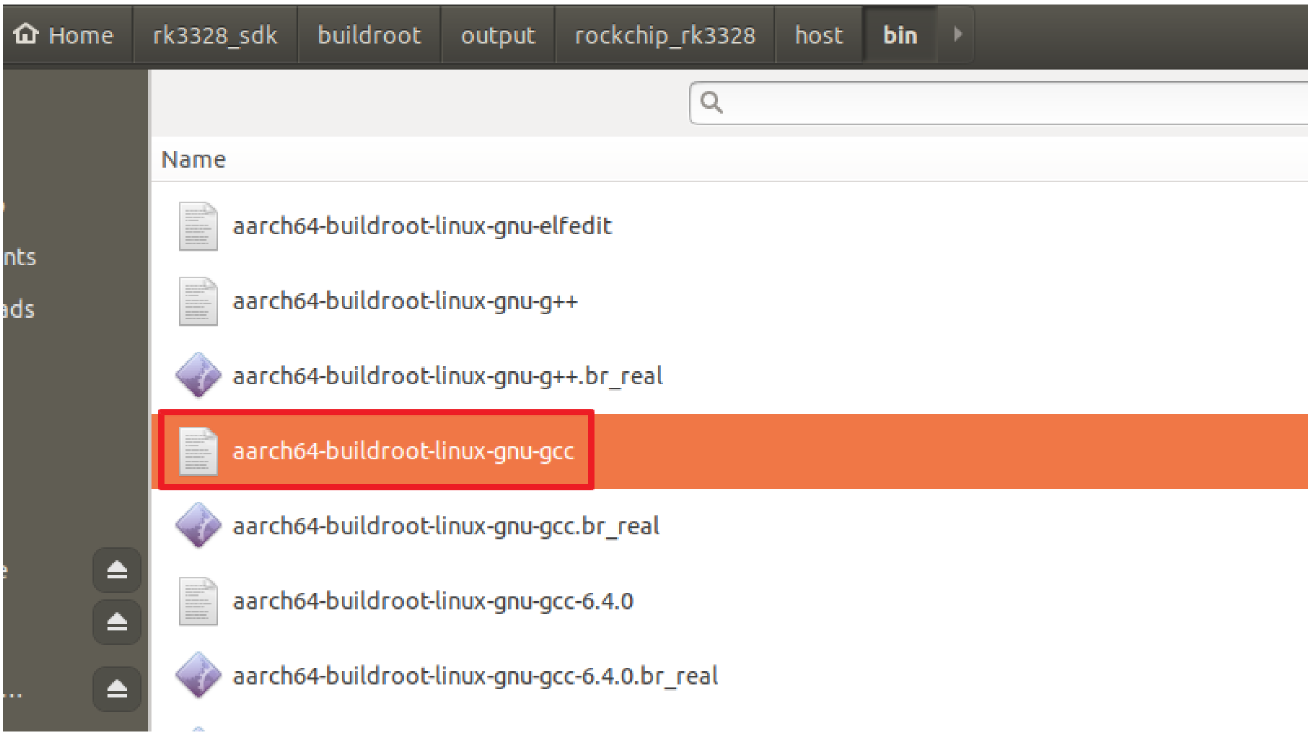
```
[root@rk3328:/lib/dnh]# ls
libReboot.so
[root@rk3328:/lib/dnh]#
```

## 4.2 Cross compile C file

要在 RK3328 上運行 SO API 時，有一個問題必須克服，我們目標環境和開發環境的核心不同，RK3328 是 ARMv8 核心，開發環境是 Intel 的核心，因此 RK3328 無法運行開發環境的 gcc 編譯的 SO API。

這時候我們必須對 SO API 的 C 檔做交叉編譯，將執行檔編譯成 RK3328 能夠解讀、執行的 SO API。在 SDK 中提供了 ARMv8 的 gcc 編譯器，透過它來編譯 SO API 的 C 檔，如此 SO API 就可以在 RK3328 上運行了。

SDK 提供的 ARMv8 gcc 編譯器所在路徑如下：
~/rk3328_sdk/buildroot/output/rockchip_rk3328/host/bin



aarch64-buildroot-linux-gnu-elfedit
aarch64-buildroot-linux-gnu-g++
aarch64-buildroot-linux-gnu-g++.br_real
aarch64-buildroot-linux-gnu-gcc
aarch64-buildroot-linux-gnu-gcc.br_real
aarch64-buildroot-linux-gnu-gcc-6.4.0
aarch64-buildroot-linux-gnu-gcc-6.4.0.br_real