Proliphix IP Devices:
HTTP API for NT Series Thermostats

Revision: 1.11

Date: Wednesday, June 20, 2007

Proliphix, Inc. ▪ *66 Tadmuck Road, Suite #1* ▪ *Westford, MA 01886* ▪ **ph** *866.IP.LIVING* ▪ **fx** *978.692.3378* ▪ **www.proliphix.com**

1. Table of Contents

2. Introduction

The Proliphix Device API leverages the HTTP protocol to get and set runtime/configuration variables on the Proliphix family of thermostats. Using the HTTP protocol, 3rd party software can control and query Proliphix devices. Control and status information is stored in device variables that are indexed by an Object Identification (OID) number. This document covers the access of the OIDs and some specific guidelines for manipulating the OIDs.

The general mechanism consists of:

1. Opening a TCP socket to the thermostats' HTTP server
2. Sending a header containing the requested information
3. Receiving a response from the thermostat
4. Closing the TCP socket

The simplicity of the protocol allows clients to be written easily in any programming language. Third-party HTTP libraries are available to further simplify writing clients.

One URL is used for performing a get and one URL is used for setting OIDs. Both use formatting of form submission, specified in the HTML RFC 1866. The form submission is made up for a list of keys and values, separated by ampersands ("&"). In the API, each key is the string "OID" and then the numerical representation of OID with which you are working. The API also specifies the use of HTTP POST to submit the request. The response is specific to a get vs. a set. Chapters 5 and 6 cover the GET and SET operations. Chapter 7 includes several coding examples in several programming languages.

3. Definitions

- Device – An embedded piece of electronics made by Proliphix, Inc., supporting Ethernet, IP, and HTTP
- OID – Specific variable stored on the device
- PIB – Proliphix Information Base, the embedded database storing all the OIDs on the thermostat

3.1 References

- HTTP 1.1 – http://www.ietf.org/rfc/rfc2616.txt
- HTTP Authentication: Basic and Digest Access Authentication -- http://www.ietf.org/rfc/rfc2617.txt
- "application/x-www-form-urlencoded" – http://www.ietf.org/rfc/rfc1866.txt

3.2 Disclaimer

THIS DOCUMENT IS PROVIDED BY PROLIPHIX, INC. ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL PROLIPHIX, INC. BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION; OR CUSTOMER EQUIPMENT) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENT, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
PROLIPHIX, INC. IS UNDER NO OBLIGATION TO UPDATE THIS DOCUMENT AND WILL NOT PROVIDE DIRECT SUPPORT OF THIS DOCUMENT. THE RELEASE OF THIS DOCUMENT IS FOR INFORMATIONAL PURPOSES ONLY.

3.3 Changes for Revision 1.11

Section	Description	Type
10.1.2	Change in HVAC State numbering to add Auxillary Heat indication	Functional
10.1.4	Correct documentation of HVAC Fan State	Documentation
10.1.11	Added definitions for Non-Pro series devices	Addition
10.5	Added Special days description	Addition

4. HTTP Authentication

Proliphix devices implement the Basic Access Authentication protocol specified in RFC 2617. All queries must provide proper credentials based upon the admin username/password pair. The device implements a timeout mechanism such that 20 minutes after the last successful access, the user is required to resend authentication information. This will come in the form of a 401 response even if proper credentials were sent in the current request.

5. API GET

The URL used is /get. An API GET request is a list of OIDs where their value is not specified. A properly formatted request should provide the Content-Length header. . The <credentials> entry is the encoded basic authentication word (See RFC 2617 -HTTP Authentication: Basic and Digest Access Authentication).

5.1 Request

```
POST /get HTTP/1.1
Authorization: Basic <credentials>
Content-Type: application/x-www-form-urlencoded
User-Agent: Jakarta Commons-HttpClient/2.0.2
Host: 192.168.111.114:8214
Content-Length: 92

OID1.10.9=&OID1.2=&OID1.1=&OID1.4=&OID1.8=&OID2.7.1=&
```

5.2 Response

```
HTTP/1.1 200 OK
Cache-control: no-cache
Server: Ubicom/1.1
Content-Length: 166

OID1.10.9=example@proliphix.com&OID1.2=SW Dev 114&OID1.1=therm_rev_2
0.1.40&OID1.4=192.168.111.114&OID1.8=00:11:49:00:00:58&OID2.7.1=NT100
```

6. API SET

The URL used is /pdp . An API SET is similar to the API GET for the request message, except that the desired value is provided at the equals sign. The response is formatted differently. The <credentials> entry is the encoded basic authentication word (See RFC 2617 -HTTP Authentication: Basic and Digest Access Authentication). The last item in the request must be “submit=Submit”. Do not include an ‘&’ after the "submit=Submit".

6.1 Request

```
POST /pdp HTTP/1.1
Authorization: Basic <credentials>
Content-Type: application/x-www-form-urlencoded
User-Agent: Jakarta Commons-HttpClient/2.0.2
Host: 192.168.111.114:8214
Content-Length: 193

OID1.2=SW+Dev+114&OID1.4=192.168.111.114&OID1.5=255.255.255.0&OID1.6=192.168.
111.1&OID1.7=3&OID1.9=1&OID1.10.1=2&OID1.10.3=24.41.5.92&OID1.10.5=20&OID1.10
.9=example%40proliphix.com&submit=Submit
```

6.2 Response

```
HTTP/1.1 200 OK
Cache-control: no-cache
Server: Ubicom/1.1
Content-Length: 308

OID1.2=SW Dev
114&OID1.4=192.168.111.114&OID1.5=255.255.255.0&OID1.6=192.168.111.1&OID1.7=3
&OID1.9=1&OID1.10.1=2&OID1.10.3=24.41.5.92&OID1.10.5=20&OID1.10.9=example@pro
liphix.com
```

7. Code Examples

The following are example written in various languages.

7.1 Curl

Curl is a command line tool provided with many Linux distributions. It provides a mechanism to grab OIDs with a single line on the UNIX prompt.

7.1.1 Get

```
curl –u hostname:password –-data OID1.1= http://192.168.1.100:8100/get
```

7.1.2 Set

```
curl –u hostname:password --data OID1.10.5=120 --data submit=Submit
http://192.168.1.100:8100/pdp
```

7.2 PHP

PHP is a web-server specific scripting language, akin to mod_perl. It integrates well into Apache and offer many web-specific libraries as part of the base system.

7.2.1 Get

```
$oids = array('OID1.4'=>'', // commonIpAddr
              'OID1.10.5'=>'',
              'submit'=>'Submit'); // commonCallhomeInterval
$url = "http://192.168.1.100:8100/get";
$ch = curl_init($url);
curl_setopt($ch, CURLOPT_HTTPGET, false);
curl_setopt($ch, CURLOPT_TIMEOUT, 5);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$myHeader = array("Content-Type: application/x-www-form-urlencoded" );
curl_setopt($ch, CURLOPT_HTTPHEADER, $myHeader);
curl_setopt($ch, CURLOPT_POSTFIELDS, http_build_query($oids));
$response = curl_exec($ch);
curl_close($ch);
$oids = array();
parse_str($response, $oids); // converts '.' to underscore
$localip = $oids['OID1_4'];
$interval = $oids['OID1_10_5']; // in minutes
```

7.3 Internet Explorer

A quick method of performing an API SET or API GET is via Internet Explorer or any web browser. This is possible since form parameters can be specified in the URL. This method is not recommended for a large list of OIDs. Just type in the following formats to access/set variables on the thermostat.

7.3.1 Get

Gets assume the following format: http://IPADDR:PORT/get?OIDoidid=.

The following example gets the current zone temperature of the thermostat at IP Address 192.168.1.100, Port 8100.

***http://192.168.1.100:8100/get?OID4.1.13*=**

Typing this into a browser's URL line returns:

```
OID4.1.13=795&
```

The value returned is in deci-degrees so the zone temperature is 79.5°F.

7.3.2 Set

Sets assume the following format: ***http://IPADDR:PORT/pdp?OIDoidid=value&submit=Submit***. Do not include an '&' after the 'submit=Submit'.

The following example sets the HVAC mode of the thermostat at IP Address 192.168.1.100, Port 8100 to Heat.

http://192.168.1.100:8100/pdp?OID4.1.1=2&submit=Submit

The following example sets the HVAC mode of the thermostat at IP Address 192.168.1.100, Port 8100 to Cool.

http://192.168.1.100:8100/pdp?OID4.1.1=3&submit=Submit

The following example sets the HVAC mode of the thermostat at IP Address 192.168.1.100, Port 8100 to Off.

http://192.168.1.100:8100/pdp?OID4.1.1=1&submit=Submit

The following example sets the device name of the thermostat at IP Address 192.168.1.100, Port 8100 to Lobby One.

http://192.168.1.100:8100/pdp?OID1.2=Lobby+One&submit=Submit

8. Limitations on the Frequency of Operations

While the device can handle numerous back to back requests, it is advisable to not sustain an operation frequency higher than 1 request per 60 second period for a substantial amount of time. In other words, sustained polling of the device should not exceed a few requests per minute. A prolonged burst of requests higher than 1 request per 60 second period may degrade the main function of the thermostat. Also, if one is making a PIB set and wishes to observe a reaction from the device, it is advisable to wait for at least 1 second between the set request and subsequent get request.

9. Warnings and Restrictions

9.1 General Disclaimer

This document describes the API for the NT series of thermostats ONLY. Refer to the TM Series API document for instruction on how to program the TM series of devices.

Only PIB variables described in Section 10 should be accessed. While other variables are exposed to the Web and API interfaces, the behavior of non-documented variables can change without warning in future releases. In addition, setting of undocumented variables may produce results that are not expected since the expected behavior has not been documented.

9.2 Warnings and Restrictions on "Sets" to the PIB

The ACCESS attribute of a PIB object defaults to READWRITE, if the ACCESS is not specified.

Only objects with an ACCESS of READWRITE may be ***set***. Attempts to ***set*** a READONLY object will result in and error message: OIDx.x..=FAILED6.

thermSetbackHeat	If the thermHvacMode is AUTO, this should not be set within 2 degrees of thermSetbackCool, or the current cool setting. This is taken care of via JavaScript validation on the web pages.
thermSetbackCool	See above.
thermActivePeriod	Do not set this object. The system sets this based on the scheduling configuration.

9.3 Error Codes

Sets and gets can return errors or the form OID#=FAILEDn.

9.3.1 FAILED5

This error message is return due to an invalid OID number during a GET operation.

9.3.2 FAILED6

This error message is return due to an invalid OID set parameter number during an SET operation. This can also be return when attempting to set a READONLY parameter.

9.3.3 FAILED8

This error message is return due to an invalid OID number during a SET or GET operation.

9.4 General Errors

If the HTTP server is very busy during a SET or GET operation or is servicing a web browser, a GET/SET may return a "Please try again" message.

10. Proliphix Thermostat API

In the following tables, an OID number is given by the addr="x.y.z..." statement. For example, **thermHvacState** has an OID number of 4.1.2. When an object has an ENUM table of values, the numeric value is returned by a query operation. A set operation expects a numeric parameter. Unless otherwise stated in the table, temperatures are represented in deci-degree Fahrenheit notation. For example a query of thermAverageTemp (4.1.13) will return 725 when the average temperature is 72.5 degrees Fahrenheit.

Some objects exist in a table format where the base OID number is indexed. Table entries will be marked. Table 1 shows some commonly used thermostat objects.

Object	OID Number	Description
thermAverageTemp	4.1.13	Average of all enabled sensors.
thermSensorTemp(**1**)	4.3.2.**1**	Zone Temperature (local sensor)
thermSensorTemp(**2**)	4.3.2.**2**	Remote Sensor #1
thermSensorTemp(**3**)	4.3.2.**3**	Remote Sensor #2
thermHvacState	4.1.2	Current State of the HVAC State Machine
thermFanState	4.1.4	Current State of the Fan

Table 1: Commonly Used Thermostat Objects

10.1 State Related Objects

10.1.1 thermHvacMode

This variable controls the major HVAC operation of the thermostat. This variable is controlled through the web interface on the **Status & Control** page in the **HVAC Settings** section. Make sure when setting to Auto mode that the temperature setback rules outlined in Section 10.4.2 and 10.4.3 are observed.

Object Description		
Field	Description	
Object	thermHvacMode	
OID #	4.1.1	
Type	ENUM	
Description	Current mode of HVAC operation	
Access	R/W	
Caveat	N/A	
ENUM Description		
ENUM	Value	Description
Off	1	HVAC is not active
Heat	2	Heat only
Cool	3	Cool only
Auto	4	Heat or cool selected by system

Table 2: thermHvacMode

10.1.2 thermHvacState

This object displays the current state of the HVAC system. This object can be polled to determine if an HVAC system is inactive or is currently heating or cooling. This object is displayed through the web interface on the **Status & Control** page in the **HVAC Settings** section

Field	Description
Object	thermHvacState
OID #	4.1.2
Type	ENUM
Description	Current state of HVAC operation
Access	R/O
Caveat	N/A

ENUM Description		
ENUM	Value	Description
Initializing	1	At device reset before operation
Off	2	Not operating
Heat	3	Heating, first stage
Heat2	4	Heating first+second stage (Professional series only)
Heat3	5	Auxillary heat (Heat Pump Only)
Cool	6	Cooling, first stage
Cool2	7	Cooling, first+second stage (Professional series only)
Delay	8	Waiting for compressor delay to timeout
ResetRelays	9	Intermediate state where relays are returned to inactive

Table 3: thermHvacState

10.1.3 thermFanMode

This object controls the current fan mode. This object is controlled through the web interface on the **Status & Control** page in the **HVAC Settings** section

Object Description	
Field	Description
Object	thermFanMode
OID #	4.1.3
Type	ENUM
Description	Current mode of fan operation
Access	R/W
Caveat	N/A

ENUM Description		
ENUM	Value	Description
Auto	1	Fan is automatically turned-on as needed by the thermostat.
On	2	The fan is always on.
Schedule	3	The mode is a combination of auto mode and the defined schedule. When the fan isn't turned on by the schedule, it operates in auto mode.

Table 4: thermFanMode

10.1.4 thermFanState

This object displays the current fan state.

Object Description		
Field	Description	
Object	thermFanState	
OID #	4.1.4	
Type	ENUM	
Description	Current state of fan operation	
Access	R/O	
Caveat	N/A	
ENUM Description		
ENUM	Value	Description
Init	0	State at reset
Off	1	The fan is off
On	2	The fan is on

Table 5: thermFanState

10.1.5 thermSetbackHeat

This object is the current setpoint in Fahrenheit deci-degrees for heat operation. Setting of this object will override the schedule heat setpoint. This object will be overwritten by the scheduling system on a change of period/class. This and all internal temperature settings are always in the Fahrenheit scale even when Celsius display mode is selected.

This object is controlled through the web interface on the **Status & Control** page in the **Temperature** section(Heat Setting).

Note that if the setback is a value not defined in the pull-down menus (or their equivalent), the web page will display the maximum value.

Object Description		
Field	Description	
Object	thermSetbackHeat	
OID #	4.1.5	
Type	Variable	
Description	Current heat setpoint in deci-degrees Fahrenheit	
Access	R/W	
Caveat	N/A	
Variable Description		
Variable	Value(s)	Description
INTEGER16	450…950	450 is 45.0 degrees in Fahrenheit.

Table 6: thermSetbackHeat

10.1.6 thermSetbackCool

This object is the current setpoint in Fahrenheit deci-degrees for cool operation. Setting of this object will override the schedule cool setpoint. This object will be overwritten by the scheduling system on a change of period/class. This and all internal temperature settings are always in the Fahrenheit scale even when Celsius display mode is selected.

This object is controlled through the web interface on the **Status & Control** page in the **Temperature** section(Cool Setting).

Note that if the setback is a value not defined in the pull-down menus (or their equivalent), the web page will display the maximum value.

Object Description		
Field	Description	
Object	thermSetbackCool	
OID #	4.1.6	
Type	Variable	
Description	Current cool setpoint in deci-degrees Fahrenheit	
Access	R/W	
Caveat	N/A	
Variable Description		
Variable	Value(s)	Description
INTEGER16	450…950	450 is 45.0 degrees in Fahrenheit.

Table 7: thermSetbackCool

10.1.7 thermConfigHumidityCool (NT150 only)

This object controls the current relative humidity setpoint to activate a cooling cycle. The object has a value in terms of percentage relative humidity from 5 to 95

This object is controlled through the web interface on the **Status & Control** page in the **Temperature** section(Cool Setting).

Object Description		
Field	Description	
Object	thermConfigHumidityCool	
OID #	4.2.22	
Type	Variable	
Description	Current relative humidity A/C trigger setting	
Access	R/W	
Caveat	N/A	
Variable Description		
Variable	Value(s)	Description
UNSIGNED8	0, 5-95	0 disables feature, 5 – 95 sets the relative humidity activation point.

Table 8: thermSetbackCool

10.1.8 thermSetbackStatus

This object controls the current setback status.

This object is displayed through the web interface on the **Status & Control** page in the **Temperature** section(Override and Hold Mode).

Object Description		
Field	Description	
Object	thermSetbackStatus	
OID #	4.1.9	
Type	ENUM	
Description	Current state of setback operation	
Access	R/W	
Caveat	N/A	
ENUM Description		
ENUM	Value	Description
Normal	1	The current heat/cool setback can change according to the current schedule.
Hold	2	Hold disables scheduling and the current setback values are maintained. Note: Hold mode can be selected to time out after a period of time.
Override	3	Override indicates that the current setback has been manually changed. The setpoint will return to Normal at the next scheduled setback event.

Table 9: thermSetbackStatus

10.1.9 thermCurrentPeriod

This object displays the current period. The period will be changed by the scheduling system at the next schedule boundary. This object is labeled as R/W, however, there is no provision for the API to set this value and any sets to this variable will be overridden by the thermostats internal logic.

Object Description		
Field	Description	
Object	thermCurrentPeriod	
OID #	4.1.10	
Type	ENUM	
Description	Current Period	
Access	R/W	
Caveat	Writes will be overridden by internal logic	
ENUM Description		
ENUM	Value	Description
Morn	1	Morning
Day	2	Day
Eve	3	Evening
Night	4	Night

Table 10: thermCurrentPeriod

10.1.10 thermActivePeriod

This object displays the current active period. This is a combination of the current period and Hold/Override settings.

Object Description		
Field	Description	
Object	thermActivePeriod	
OID #	4.1.12	
Type	ENUM	
Description	Current active period	
Access	R/O	
Caveat	N/A	
ENUM Description		
Name	Value	Description
Morn	1	Scheduled Morning
Day	2	Schedule Day
Eve	3	Scheduled Evening
Night	4	Scheduled Night
Hold	5	Hold
Override	6	Temporary Temperature Override

Table 11: thermActivePeriod

10.1.11 thermCurrentClass

This object displays the current day class. The day class will be changed by the scheduling system at the next schedule boundary.

Object Description		
Object	thermCurrentClass	
OID #	4.1.11	
Type	Variable	
Description	Current Class	
Access	R/O	
Caveat	N/A	
Variable Description		
Variable	Value	Description
	1	Occupied (Pro) / In
	2	Unoccupied (Pro) / Out
	3	Other (Pro) / Away

Table 12: thermCurrentClass

10.1.12 Alarms

The current alarm state should be queried using the **commonAlarmStatus** table OID. Once an alarm has been detected, the alarm can be cleared using the appropriate thermConfig(Low|High|FilterReminder|HighHumidity)TempPending variable. The alarm condition should be resolved before the clear, or the alarm will immediately go back to the active state.

10.1.12.1 commonAlarmStatus

This object displays the current alarm status for each alarm. This table object can be polled to determine the current state of alarms on the device.

Object Description		
Field	Description	
Object	commonAlarmStatus	
OID #	1.13.2.(1,2,3)	
Type	Table	
Description	Current alarm Status	
Access	R/O	
Caveat	N/A	
Table Description		
Index	Description	
1	Low Temp Alarm (Red)	
2	High Temp Alarm (Red)	
3	Filter Reminder (Yellow)	
4	High Humidity (NT150 only) (Red)	
ENUM Description		
Name	Value	Description
Green	1	No Alarm/Reminder
Yellow	2	Yellow Alarm Condition
Red	3	Alarm Pending

Table 13: commonAlarmStatus

10.1.12.2 thermConfigLowTempPending

This object displays the current alarm pending state for the **Zone Low Temperature Alarm**. This variable is R/W with the thermostat setting the variable to a value of 1 or 2 depending upon the alarm state evaluation. The API can be used to write the value to 3 to force a clear of the alarm state variable. If the alarm is still enabled and the alarm condition still active, the alarm will immediately return to the **Yes** state after the **Clear** command.

Object Description		
Object	thermConfigLowTempPending	
OID #	4.2.11	
Type	Variable	
Description	Zone Low Temperature Alarm Pending State	
Access	R/W	
Caveat	API write of the value 3 only!	
Variable Description		
Variable	Value	Description
	1	No – (Read) No alarm
	2	Yes – (Read) Alarm Present
	3	Clear – Write to clear alarm

Table 14: thermConfigLowTempPending

10.1.12.3 thermConfigHighTempPending

This object displays the current alarm pending state for the **Zone High Temperature Alarm**. This variable is R/W with the thermostat setting the variable to a value of 1 or 2 depending upon the alarm state evaluation. The API can be used to write the value to 3 to force a clear of the alarm state variable. If the alarm is still enabled and the alarm condition still active, the alarm will immediately return to the **Yes** state after the **Clear** command.

Object Description		
Object	thermConfigHighTempPending	
OID #	4.2.13	
Type	Variable	
Description	Zone High Temperature Alarm Pending State	
Access	R/W	
Caveat	API write of the value 3 only!	
Variable Description		
Variable	Value	Description
	1	No – (Read) No alarm
	2	Yes – (Read) Alarm Present
	3	Clear – Write to clear alarm

Table 15: thermConfigHighTempPending

10.1.12.4 thermConfigFilterReminderPending

This object displays the current alarm pending state for the **Filter Reminder Alert**. This variable is R/W with the thermostat setting the variable to a value of 1 or 2 depending upon the alarm state evaluation. The API can be used to write the value to 3 to force a clear of the alarm state variable. If the alarm is still enabled and the alarm condition still active, the alarm will immediately return to the **Yes** state after the **Clear** command.

Object Description		
Object	thermConfigFilterReminderPending	
OID #	4.2.9	
Type	Variable	
Description	Filter Reminder Alert Pending State	
Access	R/W	
Caveat	API write of the value 3 only!	
Variable Description		
Variable	Value	Description
	1	No – (Read) No alarm
	2	Yes – (Read) Alarm Present
	3	Clear – Write to clear alarm

Table 16: thermConfigFilterReminderPending

10.1.12.5 thermConfigHighHumidityrPending (NT150 only)

This object displays the current alarm pending state for the **High Humidity Alarm**. This variable is R/W with the thermostat setting the variable to a value of 1 or 2 depending upon the alarm state evaluation. The API can be used to write the value to 3 to force a clear of the alarm state variable. If the alarm is still enabled and the alarm condition still active, the alarm will immediately return to the **Yes** state after the **Clear** command.

Object Description		
Object	thermConfigHighHumidityPending	
OID #	4.2.17	
Type	Variable	
Description	High Humidity Alarm Pending State	
Access	R/W	
Caveat	API write of the value 3 only! / (NT150 only)	
Variable Description		
Variable	Value	Description
	1	No – (Read) No alarm
	2	Yes – (Read) Alarm Present
	3	Clear – Write to clear alarm

Table 17: thermConfigHighHumidityPending

10.1.13 thermSensorCorrection (Models with external sensors only)

This object stores the value that is added to an external sensor to compensate for system properties so that sampled sensor temperature matches actual temperature.

Object Description	
Field	Description
Object	thermSensorCorrection
OID #	4.3.4.(2,3)
Type	Table
Description	Remote sensor correction
Access	R/W
Caveat	Index 1 (Local) is not writable
Table Description	
Index	Description
2	RS #1
3	RS #2
Variable Description	
Value	Description
INTEGER16	+- value added to sampled value in deci-degrees.

Table 18: thermSensorCorrection

10.1.14 External Sensor Related Variables

10.1.14.1 thermSensorName (Models with external sensors only)

This object stores the current name of each of the external sensors.

Object Description	
Field	Description
Object	thermSensorName
OID #	4.3.5.(2,3)
Type	Table
Description	Remote Sensor Names
Access	R/W
Caveat	Index 1 (Local) is not writable
Table Description	
Index	Description
2	RS #1
3	RS #2
Variable Description	
Value	Description
String	Name

Table 19: thermSensorName

10.1.14.2 thermSensorState (Models with external sensors only)

This object controls the current state of sensors. Use this variable to enable and disable sensors. Note that if you disable a sensor, you must also disable the averaging of the sensor.

Object Description		
Field	Description	
Object	thermSensorState	
OID #	4.3.6.(1,2,3)	
Type	Table	
Description	Current sensor state	
Access	R/W	
Caveat	Only use the Disabled/Enabled ENUMs. Also check value of thermSensorAverage	
Table Description		
Index	Description	
1	Local	
2	RS #1	
3	RS #2	
ENUM Description		
Name	Value	Description
NotPresent	0	Sensor not present
Disabled	1	Sensor is disabled
Enabled	2	Sensor is enabled

Table 20: thermSensorState

10.1.14.3 thermSensorAverage (Models with external sensors only)

This object controls whether a sensor is included into the zone average. Note that if you disable a sensor, you must also disable the averaging of the sensor.

Object Description	
Field	Description
Object	thermSensorAverage
OID #	4.3.8.(1,2,3)
Type	Table
Description	Controls whether a sensor is used to determine the zone (averaged) temperature.
Access	R/W
Caveat	N/A

Table Description	
Index	Description
1	Local
2	RS #1
3	RS #2

ENUM Description		
Name	Value	Description
Disabled	1	Sensor is not averaged
Enabled	2	Sensor is averaged

Table 21: thermSensorAverage

10.1.14.4 thermSensorType (Models with external sensors only)

This object controls the type of remote sensor.

Object Description	
Field	Description
Object	thermSensorType
OID #	4.3.9.(2,3)
Type	Table
Description	Controls the type of remote sensor
Access	R/W
Caveat	Index 1 is not accessible

Table Description	
Index	Description
2	RS #1
3	RS #2

ENUM Description		
Name	Value	Description
Analog	1	Analog sensor
Thermistor	2	Thermistor sensor

Table 22: thermSensorType

10.2 Temperature Related Objects

10.2.1 thermAverageTemp (Zone Temperature)

This object stores the current averaged temperature (Zone Temperature) in deci-degrees of all sensors selected for averaging.

This object is displayed through the web interface on the **Status & Control** page in the **Temperature** section(Zone Temperature).

Object Description	
Field	Description
Object	thermAverageTemp
OID #	4.1.13
Type	Variable
Description	Current averaged temperature
Access	R/O
Caveat	N/A
Variable Description	
Value	Description
-300…+2000	Range from -30 Fahrenheit to +200 Fahrenheit

Table 23: thermAverageTemp

10.2.2 thermSensorTemp

This table object stores the current temperature in deci-degrees of each sensor.

This object is displayed through the web interface on the **Status & Control** page in the **Temperature** section(Local / RS #1 / RS #2).

Object Description	
Field	Description
Object	thermSensorTemp
OID #	4.3.2.(1,2,3)
Type	Table
Description	Current sensor temperature
Access	R/O
Caveat	N/A
Table Description	
Index	Description
1	Local Sensor
2	Remote Sensor 1
3	Remote Sensor 2
Variable Description	
Value	Description
-300…+2000	Range from -30 Fahrenheit to +200 Fahrenheit

Table 24: thermSensorTemp

10.2.3 thermRelativeHumidity (NT150 only)

This object displays the current relative humidity percentage.

This object is displayed through the web interface on the **Status & Control** page in the **Temperature** section.

Object Description	
Field	Description
Object	thermRelativeHumidity
OID #	4.1.14
Type	Variable
Description	Current relative humidity
Access	R/O
Caveat	Only updated once a minute
Variable Description	
Value	Description
5..95 %	Relative humidity

Table 25: thermRelativeHumidity

10.3 System Related Objects

10.3.1 systemUptime

This variable stores the uptime of the device. This value is the number of seconds elapsed since boot.

Object Description	
Field	Description
Object	systemUptime
OID #	2.1.1
Type	Variable
Description	Time in seconds since boot
Access	R/O
Caveat	N/A
Variable Description	
Value	Description
0 to 32 bit UNSIGNED Integer	Time in seconds since boot

Table 26: systemUptime

10.3.2 systemTimeSecs

This variable stores the current time of the device. This value is the number of seconds elapsed since Jan 1, 1970. Use this object to set the current time.

This object is controlled through the web interface on the **General Settings** page in the **Date and Time** section(Set Date and Time).

Object Description	
Field	Description
Object	systemTimeSecs
OID #	2.5.1
Type	Variable
Description	Time in seconds since Jan 1, 1970
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 to 32 bit UNSIGNED Integer	Time in seconds since Jan 1, 1970

Table 27: sytemTimeSecs

10.3.3 commonDevName

This object stores the current device name.

This object is controlled through the web interface on the **General Settings** page.

Object Description	
Field	Description
Object	commonDevName
OID #	1.2
Type	String Variable
Description	Device Name
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
String	Device Name

Table 28: commonDevName

10.3.4 systemMimModelNumber

This object stores the current device model number.

Object Description	
Field	Description
Object	systemMimModelNumber
OID #	2.7.1
Type	String Variable
Description	Device Name
Access	R/O
Caveat	N/A
Variable Description	
Value	Description
String	Currently one of the following: NT10/NT20/NT100/NT120/NT150

Table 29: systemMimModelNumber

10.4 Schedule

The following section describes the variables associated with setting up the device's schedule. Proliphix devices currently provide three day classes either In/Out/Away (Basic Model) or Occupied/Unoccupied/Other (Professional). Each day class is broken into four periods, Morn/Day/Eve/Night. A period is defined by its start time.

10.4.1 thermPeriodStart

This table object stores the start time for each day class / period pair. There are three classes (In/Out/Away or Occupied/Unoccupied/Other) and four periods. The start times are stored as minutes from midnight. For example, a value of 480 would mean a start time of 8:00am (480/60=8). Periods must be incrementing and non-overlapping. (i.e. period 2 must start after period 1 and period 3 after 2…)

Object Description	
Field	Description
Object	thermPeriodStart
OID #	4.4.1.3.(1/2/3).(1/2/3/4)
Type	Table / Variable
Description	Period start value
Access	R/W
Caveat	N/A
Table Description	
Class Index	Description
1	Occupied/In
2	Unoccupied/Out
3	Other/Away
Period Index	Description
1	Period 1
2	Period 2
3	Period 3
4	Period 4
Variable Description	
Value	Description
1 .. 1439	1 minute after midnight .. 1 minute before midnight

Table 30: thermPeriodStart

10.4.2 thermPeriodSetbackCool

This table object stores the cool setting for each class / period pair. The values are the cool setback at the start of the associated period.

Object Description	
Field	Description
Object	thermPeriodSetbackCool
OID #	4.4.1.5.(1/2/3).(1/2/3/4)
Type	Table /Variable
Description	Cool setback in deci-degrees
Access	R/W
Caveat	N/A
Table Description	
Class Index	Description
1	Occupied/In
2	Unoccupied/Out
3	Other/Away
Period Index	Description
1	Period 1
2	Period 2
3	Period 3
4	Period 4
Variable Description	
Value	Description
450 .. 950	Deci-degree Fahrenheit

Table 31: thermPeriodSetbackCool

10.4.3 thermPeriodSetbackHeat

This table object stores the heat setting for each class / period pair. The values are the heat setback at the start of the associated period.

Object Description	
Field	Description
Object	thermPeriodSetbackHeat
OID #	4.4.1.4.(1/2/3).(1/2/3/4)
Type	Table
Description	Heat setback in deci-degrees
Access	R/W
Caveat	N/A
Table Description	
Class Index	Description
1	Occupied/In
2	Unoccupied/Out
3	Other/Away
Period Index	Description
1	Period 1
2	Period 2
3	Period 3
4	Period 4
Variable Description	
Value	Description
450 .. 950	Deci-degree Fahrenheit

Table 32: thermPeriodSetbackHeat

10.4.4 thermPeriodSetbackFan (Professional models only, NT120/NT150)

This table object stores the fan on-time for each class / period pair.

Object Description	
Field	Description
Object	thermPeriodSetbackFan
OID #	4.4.1.6.(1/2/3).(1/2/3/4)
Type	Table / Variable
Description	Fan on-time for a particular period
Access	R/W
Caveat	Use only values defined in table below
Table Description	
Class Index	Description
1	Occupied/In
2	Unoccupied/Out
3	Other/Away
Period Index	Description
1	Period 1
2	Period 2
3	Period 3
4	Period 4
Variable Description	
Value	Description
0	Disable
15	15 Minutes
30	30 Minutes
45	45 Minutes
60	On

Table 33: thermPeriodSetbackCool

10.4.5 thermDefaultClassId

This table object stores the default class id for the default week. This table is set on the Weekly Schedule page of the Web Interface.

Object Description	
Field	Description
Object	thermDefaultClassId
OID #	4.4.3.2.(1/2/3/4/5/6/7)
Type	Table
Description	Default class for the default weekly schedule
Access	R/W
Caveat	Use only values defined in table below
Table Description	
Week Index	Description
1	Sunday
2	Monday
3	Tuesday
4	Wednesday
5	Thursday
6	Friday
7	Saturday
Variable Description	
Value	Description
1	Occupied/In
2	Unoccupied/Out
3	Other/Away

Table 34: thermDefaultClassId

10.5 Special Days

This table object stores the special-days information. There are a total of 20 special days available on the NT series of devices. Each special day is defined by:

- Special day index
- Start Day within the month
- Month
- Year
- Duration in days
- Class ID

To create a new special day, fill in the start day, month, year, duration, and class ID into an unused special day slot. To delete a special day, fill in 0 for every field in the slot to be deleted. The month and year fields can take the special value of 0 to indicate EVERY. When month/year is set to 0, every month/year on the indicated start date, the special day will become active.

Object Description	
Field	Description
Object	thermScheduleSpecial
OID #	4.4.4.(1-6).(1-20)
Type	Table / Variable
Description	Special days
Access	R/W
Caveat	Take care to not overlap special day periods
Table Description	
Variable Index	Description
1	Index (R/O)
2	Start day of the month (1-31)
3	Month (1-12) Set to 0 for EVERY Month
4	Year (2007,2008,2009) Rolling 3 year window from the current year. Set to 0 for EVERY year.
5	Duration (0-240)
6	Class Id (1,2,3) (See 10.1.11)
Special Day Index	Description
1-20	Index into table

Table 35: thermScheduleSpecial

10.6 Usage Statistics

Beginning in release 2.5.7 of the NT series of thermostats, there are OIDs related to usage. The following section describes the variables associated with getting information related to heat / cool / fan usage. There are several minute-accurate counters that track the on-time for the HEAT1, HEAT2, COOL1, COOL2, FAN, and EXTERNAL (NT120 only) relays. The relays can be reset to 0 by writing a 0 to the appropriate OID.

10.6.1 thermHeat1Usage

This OID counts the number of minutes that the HEAT1 state has been active.

Object Description	
Field	Description
Object	thermHeat1Usage
OID #	4.5.1
Type	Variable
Description	HEAT1 State Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 36: thermHeat1Usage

10.6.2 thermHeat2Usage

This OID counts the number of minutes that the HEAT2 state has been active. For release NT 2.5.7, this variable will also count the Aux Heat state when the thermostat is in Heat Pump mode.

Object Description	
Field	Description
Object	thermHeat2Usage
OID #	4.5.2
Type	Variable
Description	HEAT2 State Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 37: thermHeat2Usage

10.6.3 thermCool1Usage

This OID counts the number of minutes that the COOL1 state has been active.

Object Description	
Field	Description
Object	thermCool1Usage
OID #	4.5.3
Type	Variable
Description	COOL1 State Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 38: thermCool1Usage

10.6.4 thermCool2Usage

This OID counts the number of minutes that the COOL2 state has been active.

Object Description	
Field	Description
Object	thermCool2Usage
OID #	4.5.4
Type	Variable
Description	COOL2 State Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 39: thermCool2Usage

10.6.5 thermFanUsage

This OID counts the number of minutes that the fan relay has been active. (See thermUsageOptions for more details relating to what is counted by this OID)

Object Description	
Field	Description
Object	thermFanUsage
OID #	4.5.5
Type	Variable
Description	FAN on time in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 40: thermFanUsage

10.6.6 thermLastUsageReset

This OID stores the time at which the counters were last reset. This OID assumes that all usage counters are reset at the same time and this variable should be set to the current thermostat time (systemTimeSecs) when the usage counters are reset to 0.

Object Description	
Field	Description
Object	thermLastUsageReset
OID #	4.5.6
Type	Variable
Description	Last reset time in seconds since Jan 1, 1970
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Time in seconds since Jan 1, 1970

Table 41: thermLastUsageReset

10.6.7 thermExternalUsage (NT150 Only)

This OID counts the number of minutes that the EXTERNAL relay has been active. When any of the several conditions that can trigger an external relay to engage are present, this counter will increment. The polarity of the external relay activation does not matter.

Object Description	
Field	Description
Object	thermExternalUsage
OID #	4.5.7
Type	Variable
Description	EXTERNAL Relay Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 42: thermExternalUsage

10.6.8 thermUsageOptions

This OID contains configuration options for the usage counter mechanism.

Object Description		
Field	Description	
Object	thermUsageOptions	
OID #	4.5.8	
Type	Variable	
Description	BITFIELD	
Access	R/W	
Caveat	N/A	
BITFIELD Description		
OPTION	Value	Description
IncludeHeat	1	Include HEAT1/HEAT2 runtime in the fan usage calculation. If this is not set, the FAN will only reflect the runtime during COOL1, COOL2.

Table 43: therm2UsageOptions

10.6.9 thermHeat3Usage

This OID counts the number of minutes that the HEAT3 (Aux Heat) state has been active. This OID is only valid for release NT 3.0.0 and greater. When the thermostat is in single-stage or dual-stage heat-pump mode, this OID will count the Aux Heat usage. This counter is not used in Fuel-Burner mode.

Object Description	
Field	Description
Object	thermHeat3Usage
OID #	4.5.9
Type	Variable
Description	HEAT3 (Aux Heat) State Usage in Minutes
Access	R/W
Caveat	N/A
Variable Description	
Value	Description
0 .. 2^{32}-1	Minutes

Table 44: thermHeat3Usage