

csstel



**COME
TOGE
THER**

AN-860910

ComView & Web Server Integration

Document No.: AN-860910, Rev 1.00
Applicable Products: ComView NX(S/M/L/X)
Contact: support@csstel.com
Web: www.csstel.com

Introduction

ComView device provides a web API to let users perform http requests to get real-time data stream received from its physical interfaces, various types of data, and alarm records for further processing. Proactively, ComView can notify users of alarm conditions in real-time via http POST for immediate corrective action. It can also be scheduled to automatically push application data to one or more web servers for storage and/or user-specific data processing. These capabilities help simplify the integration of ComView device into user existing web server, more readily.

This application note is intended to describe how users can configure ComView and their web server to act as an alarm listener by using ComView alarm notification via http POST, as an alarm poller by issuing http POST requests to ComView for alarm log, and as a data collector for receiving application data from ComView.

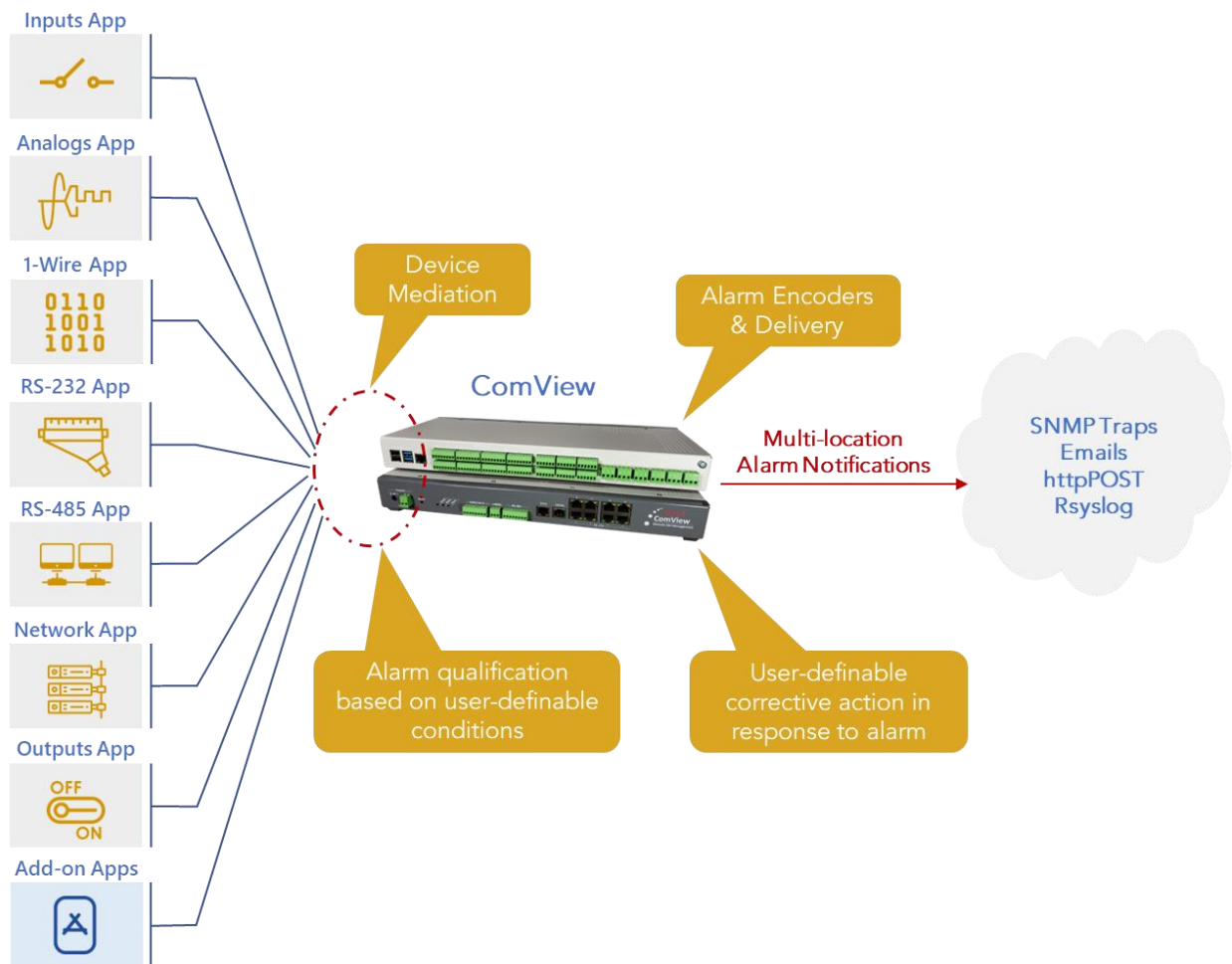
This application note does not provide detailed description of how to use ComView, its connectivity and configuration, and other supporting information, as these are beyond the scope of this document. Refer to other resources for more details.

References:

- [1]. ComView - User Guide
- [2]. PHP - <https://www.php.net/manual/en/intro-what-is.php>
- [3]. Apache web server - <https://httpd.apache.org/>
- [4]. PHP tutorials - <https://www.w3schools.com/php/>

Alarm Monitoring

ComView implements an advanced alarm processor that helps users simplify their remote site monitoring tasks and improve their operational efficiency.



The alarm processor can mediate with various types of onsite devices connected to its physical interfaces and monitor for user-definable alarm conditions. These conditions can range from simple detection of contact input state change, pattern matching of text-based signatures to a more complex protocol-based data acquisition with mathematical calculations and data processing.

On alarm condition, the alarm processor can automatically take corrective action that include activating output relay(s) to control external devices, execute user scripts to perform more complex tasks, and notify users of active alarm to multiple locations using various notification methods. In the meanwhile, it continues to monitor the condition and notify users again when the alarm has been cleared, helping alleviate potential operational overhead.

For flexibility, the alarm processor allows users to define each alarm condition with free text description for ease in identification and troubleshooting. Users can also select what alarm variables to include in an alarm message and what methods to deliver it.

Alarm Record

The alarm processor normalizes various alarm types into a common alarm record which is used by protocol-specific alarm encoders to encode an alarm message for delivery via various methods. The common alarm record has the following CSV format:

```
YYYYMMDD, HHMMSS, Interface, Source, Value, Name, Description
```

YYYYMMDD:

Date stamp

HHMMSS:

Time stamp

Interface:

Name of physical interface where alarm comes from (e.g., Inputs)

Source:

Name of alarm source (e.g., Input port 0)

Value:

Value of alarm trigger (e.g., Low)

Name:

Alternate use (e.g., unit of measurement) or secondary name of alarm source (e.g., Back door)

Description:

User-defined description of alarm condition

The alarm processor logs alarm records in interface-specific alarm files as well as in the consolidated alarm file for ease in troubleshooting, correlating, and accounting.

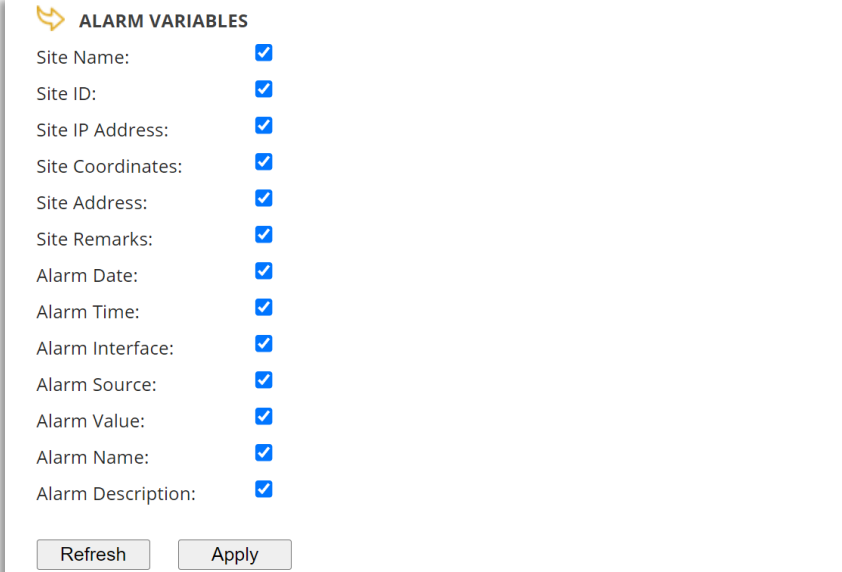
With alarms in CSV-formatted records, alarm files can be readily processed further according to user-specific requirements for analysis and reporting using common software such Microsoft Excel, Google Sheets, Python, and others.

Alarm Variables

Alarm variables are for site identification, timestamp, alarm identification, and alarm description that users can select for the alarm processor to encode an alarm message prior to its delivery. These variables include:

SiteName
SiteID
SiteIP
SiteCoordinates
SiteAddress
SiteRemarks
AlarmDate
AlarmTime
AlarmInterface
AlarmSource
AlarmValue
AlarmName
AlarmDescription

Users can select these variables via web interface (CONFIGURATION -> ALARM NOTIFICATION), as shown below:



ALARM VARIABLES

Site Name:	<input checked="" type="checkbox"/>
Site ID:	<input checked="" type="checkbox"/>
Site IP Address:	<input checked="" type="checkbox"/>
Site Coordinates:	<input checked="" type="checkbox"/>
Site Address:	<input checked="" type="checkbox"/>
Site Remarks:	<input checked="" type="checkbox"/>
Alarm Date:	<input checked="" type="checkbox"/>
Alarm Time:	<input checked="" type="checkbox"/>
Alarm Interface:	<input checked="" type="checkbox"/>
Alarm Source:	<input checked="" type="checkbox"/>
Alarm Value:	<input checked="" type="checkbox"/>
Alarm Name:	<input checked="" type="checkbox"/>
Alarm Description:	<input checked="" type="checkbox"/>

Variable names with the 'Site' prefix are for site identification, predefined during the device configuration in 'Site Info' page of the device web interface.

Variable names with the 'Alarm' prefix are for alarm details. These are predefined (except AlarmDate and AlarmTime) in various pages of the device web interface due to various sources of alarm.

Web Server as Alarm Listener

Users can configure their web server to act as an alarm listener to capture alarms delivered by ComView device using http POST alarm notification method.


Alarm Notification by HTTP POST

The alarm processor encodes an alarm record in JSON (JavaScript Object Notation) string and uses cURL (client URL) command line to send an http POST request to web servers (up to 3) to notify users of an alarm.

The following is a sample alarm in JSON string with "object": "value" pairs in CSV format:

```
{
  "SiteName": "CVSite",
  "SiteID": "00000001",
  "SiteIP": "192.168.0.100",
  "SiteCoordinates" = "43.64406,-79.38671",
  "SiteAddress"= "*** Undefined address ***",
  "SiteRemarks"= "*** Unconfigured device ***",
  "AlarmDate"= "20221021",
  "AlarmTime"= "141502",
  "AlarmInterface"= "Device",
  "AlarmSource"= "Internal",
  "AlarmValue"= "Every 15 minutes",
  "AlarmName"= "System heartbeat",
  "AlarmDescription"= "CVSite"
}
```

Users can configure ComView to deliver alarms by http POST via web interface (CONFIGURATION -> ALARM NOTIFICATION), as shown below:

 **NOTIFICATION BY HTTP POST**

HTTP POST Notification Enable: No Yes

HTTP POST URL0:

HTTP POST URL1:

HTTP POST URL2:

Alarm Reception PHP Script

For the web server to receive http POST requests from ComView, users must develop a backend script to run on the web server.

The following is a sample PHP script running on the Apache web server to receive alarm via http POST in JSON string format above:

```
<?php
$file = "/var/www/html/alarms.log";
$SName = $_POST["SiteName"];
$SID = $_POST["SiteID"];
$SIP = $_POST["SiteIP"];
$SCoordinates = $_POST["SiteCoordinates"];
$SAddress = $_POST["SiteAddress"];
$SRemarks = $_POST["SiteRemarks"];
$ADate = $_POST["AlarmDate"];
$ATime = $_POST["AlarmTime"];
$AInterface = $_POST["AlarmInterface"];
$ASource = $_POST["AlarmSource"];
$AValue = $_POST["AlarmValue"];
$AName = $_POST["AlarmName"];
$ADescription = $_POST["AlarmDescription"];
$alarm = <<<EOD

--- Begin ---
Site_name= $SName
Site_ID= $SID
Site_IP= $SIP
Site_coordinates= $SCoordinates
Site_address= $SAddress
Site_remarks= $SRemarks
Alarm_date= $ADate
Alarm_time= $ATime
Alarm_interface= $AInterface
Alarm_source= $ASource
Alarm_value= $AValue
Alarm_name= $AName
Alarm_description= $ADescription
--- End ---

EOD;
if (file_exists($file)){
    $handle = fopen($file, "a");
    fwrite($handle, $alarm ."\n");
    fclose($handle);
}
?>
```

This PHP script can be embedded in an html page or run on its own. In the latter, the script must be in a file; e.g., 'alarm-receiver.php' and stored in '/var/www/html/' directory in the Apache web server.

Alarm Log

The sample PHP script above logs alarms in the “/var/www/html/alarms.log” file, which must be pre-created with “write” permission. Below is a sample alarm collected and formatted by the script:

```
--- Begin ---
Site_name= CVSite
Site_ID= 00000001
Site_IP= 192.168.0.100
Site_coordinates= 43.64406,-79.38671
Site_address= *** Undefined address ***
Site_remarks= *** Unconfigured device ***
Alarm_date= 20221021
Alarm_time= 141502
Alarm_interface= Device
Alarm_source= Internal
Alarm_value= Every 15 minutes
Alarm_name= System heartbeat
Alarm_description= CVSite
--- End ---
```

Notes:

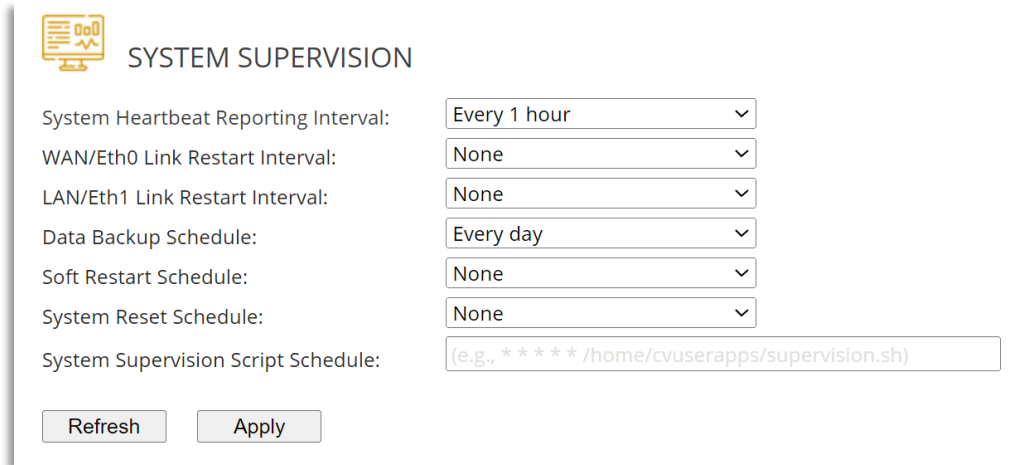
- [1]. The script assumes all alarm variables are used in the alarm record
- [2]. Each alarm received is encapsulated by ‘--- Begin ---’ / ‘--- End ---’ header/trailer pair for ease in parsing and identification
- [3]. Site_ID is an 8-digit user-defined value that can be used as an index in site database
- [4]. Site_IP can be used as a hyperlink for users to access ComView dashboard
- [5]. Site_coordinates is user-defined value that can be used as a hyperlink to Google Map


Once collected on the web server, alarms can be parsed and processed according to user-specific requirements; for example, displaying on alarm monitor, integrating with other network management platforms.

Alarm Delivery Verification

ComView device supports “heartbeat” that users can configure for periodic reporting to verify its operational status and its network connectivity. ComView sends its heartbeat in the same way as that of the alarm notification, and therefore it can be used to verify ComView alarm notification via http POST.

Users can configure ComView via web interface (CONFIGURATION -> SYSTEM SUPERVISION) to deliver its heartbeats as shown below:



 SYSTEM SUPERVISION

System Heartbeat Reporting Interval:

WAN/Eth0 Link Restart Interval:

LAN/Eth1 Link Restart Interval:

Data Backup Schedule:

Soft Restart Schedule:

System Reset Schedule:

System Supervision Script Schedule:

System heartbeat reporting interval can be set to every 15 minutes, 30 minutes, 1 hour, 2 hours, 4 hours, 6 hours, 12 hours, and 24 hours.

Web Server as Alarm Poller

Alternative to being an alarm listener, users can configure their web server to act as an alarm poller that regularly polls alarm records from ComView devices.

ComView logs all alarms including its heartbeats in its consolidated alarm file. To retrieve the last 50 alarm records from this file, users can issue an http GET request to URI `"/dashboard/recent_alarms"` as shown in the example below :

```
https://192.168.0.100/dashboard/recent_alarms
```

which returns the following sample alarm records:

```
{
  "recent": [
    "20230525,130001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,120001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,110002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,100001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,090002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,080001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,070001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,060002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,050002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,040001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,030002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,020001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,010001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230525,000003,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,230001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,220002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,210001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,200002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,190002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,180001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,170002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,160001,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,150002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,140002,Device,Internal,Every 1 hour,System heartbeat,DemoSystem ",
    "20230524,132552,1-Wire,Sensor0,23.437,C,Sensor0 < 24C alarm cleared",
    "20230524,130247,1-Wire,Sensor1,24.937,C,Sensor1 < 25C"
  ]
}
```

Notes:

- [1]. The records above follow the alarm record syntax as described previously
- [2]. When an alarm condition has been cleared, an alarm is sent with "alarm cleared" appended to the alarm description of the previously reported alarm
- [3]. All http requests require API key for ComView to authenticate the application access to its web API. Contact us for this key (at no cost) and how to use it.

Users can periodically issue http GET requests to get an updated list of alarm records from which new alarms can be derived based on timestamps.

For alarm and site correlation, users can issue an http GET request to URI “/configuration/site_info” for site information as shown in the example below:

```
https://192.168.0.100/configuration/site_info
```

which returns the following sample values:

```
{  
  "SiteName": "DemoSystem",  
  "SiteID": "00000001",  
  "SiteCoordinates": "43.64406,-79.38671000",  
  "SiteContactEmail": "siteadmin@xyz.com",  
  "SiteAddress": "Demo system at 192.168.0.100",  
  "SiteRemarks": "Demo system"  
}
```

These parameters can be used in a similar manner as described in the Alarm Log section.

Web Server as Data Collector

To collect application data (such as analog readings, fuel measurements, energy consumption data, and others) for central storage and for further application-specific data processing, users can configure their web server to act as a data collector to regularly receive data from ComView devices.

ComView device uses the popular secure copy (SCP) utility to transfer its data to a remote server, therefore, the web server must have support for SCP (which is commonly available in any Linux server).

Users can configure ComView via web interface to automatically push data files (CONFIGURATION -> FILE PUSH), as shown below:

FILE PUSH

Select Option: All

Push Time Interval: 1 Days

Inputs Application Files: Both

Outputs Application Files: Raw

Analogs Application Files: Both

1-Wire Application Files: Both

RS-485 Application Files: Both

RS-232 Application Files: Both

Network Application Files: Both

Alarm File: Yes

Destination Server IP Address: 192.168.0.230

Destination Server Path: /backup

Destination Server User ID: username

Destination Server User PW: password

Refresh Apply

Users can define the file push time interval in Minutes, Hours, and Days. When due, ComView will automatically zip all files, log in the server, and securely transfer the file to the server at the directory as specified.

ComView names the zipped file with the site name as its prefix followed by timestamp, i.e., <SiteName>_Files_YYYYMMSS_HH.MM.SS.tgz for ease in identification.

Summary

This application note illustrates how ComView normalizes alarm conditions from various physical interfaces into a common CSV-formatted alarm record. For alarm monitoring applications, users can configure their web server to receive these alarm records either by listening to http POST requests from ComView or periodically issuing http GET request to the recent alarm URI to poll the last 50 alarm records. The application note also shows how users can configure ComView to automatically push application data to the web server as required by centralized data acquisition applications.

About CSSTEL

CSSTEL is a privately held developer and manufacturer of ComView hardware and software solutions for secure, remote infrastructure site management since 1997 with installations in over 30 countries around the world.

We offer ComView solutions that are scalable and customizable to monitor and manage virtually the entire spectrum of remote site infrastructure and site conditions.

We help telecom service providers, carriers, financial institutions, healthcare providers, government agencies, utilities, and other public and private sector organizations maintain constant visibility and control over their remote site infrastructure.



IMPORTANT:

- CSSTEL Inc. assumes no responsibility for errors or omissions, or for damages resulting from the use of information contained in this document or from the use of programs and source code that accompany it. In no event shall CSSTEL Inc. be liable for any loss of profit, or any other commercial damage caused or alleged to have been caused directly or indirectly.
- No parts of this work may be reproduced in any form or by any means - graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems - without the written permission of CSSTEL Inc.
- Products that are referred to in this document may be either trademarks and/or registered trademarks of the respective owners. CSSTEL Inc. makes no claim to these trademarks.
- All rights reserved.

Revision History

Revision	Date	Description
1.00	2023-05-26	Initial release

*** End of document ***