SLightliMon

Street Lighting lite Monitor

Street lighting free monitoring

Architecture example

Document version: 1.0/0
Date: October, 18 2015
Author: Adamo Ferro

Website:
http://af-projects.it/slightlimon

Introduction

The purpose of this document is to provide some guidelines for the deployment of an architecture for the monitoring of street lighting based on free open-source software. Implementation and configuration details won't be covered. An architecture based on these guidelines has been practically deployed for the monitoring of almost 100 power cabinets in a medium-sized municipality (around 100.000 inhabitants) and is still growing. However, the same principles can be applied to monitor smaller systems. The presented solution could be deployed at a low cost specially by medium/large-size entities with an existing network infrastructure on the territory.

Cabinet electrical parameters are collected in real time and analyzed offline using SLightliMon in order to automatically detect possible anomalies occurred during the previous night and to effectively plan maintenance activities. The presented architecture can potentially work with a large set of commercial power meters, as it does not require equipments of a particular brand or model. Although it cannot provide the amount of information usually available from so-called “Smart Lighting” commercial systems (which also often include direct control capabilities besides monitoring), the presented architecture showed to already provide an effective tool for street lighting monitoring.

Note: the presented example shows that it is possible to achieve the goal of street lighting monitoring with the help of existing and general open-source software tools. At the time of writing, there are open-source projects aimed to create a whole Internet of Things (IoT) generic monitoring platform, including the monitoring of street lighting. One example is the Open Smart Grid Platform (osgplatform.wordpress.com).

Network and devices

As shown in Figure 1, the presented architecture is based on a network of power meters which are connected to a server by a set of fiber optic- or copper-cabled switches, and/or WiFi bridges that can extend the cabled network. The way devices are connected to the server is not the main focus of this document. The main requirement is that the server could reach every power meter through a reliable IP connection.

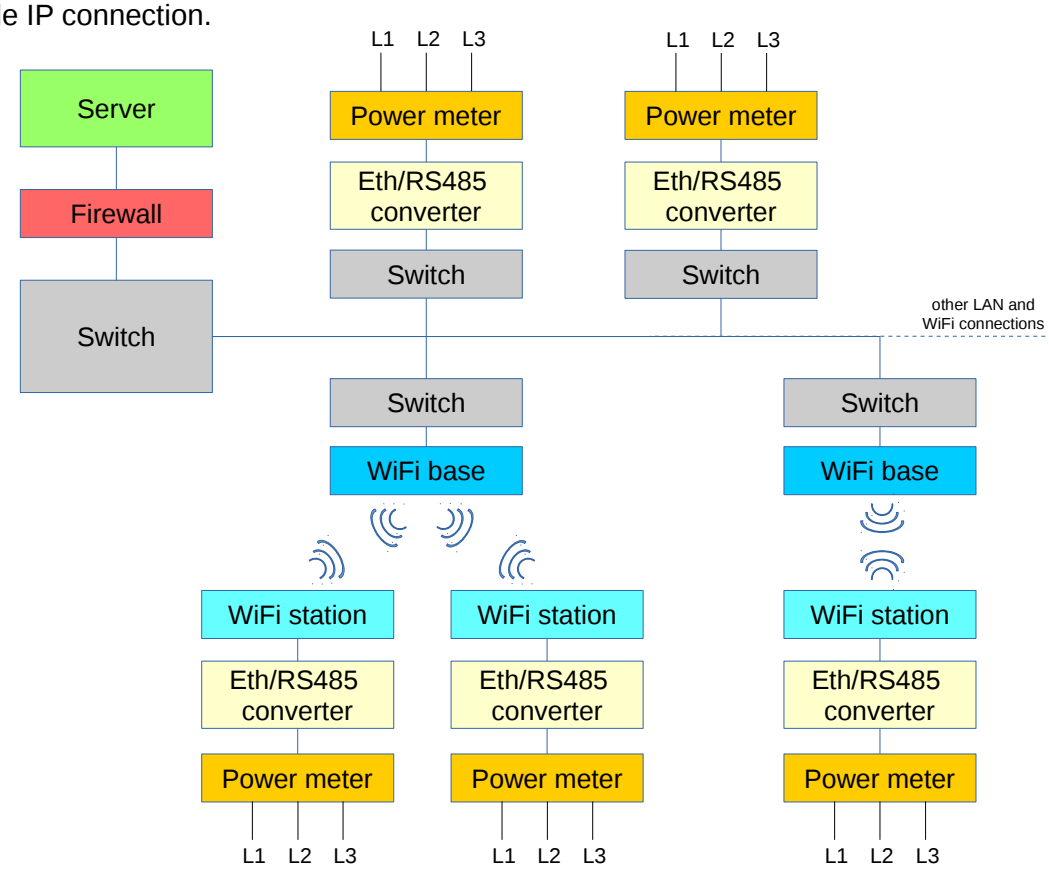

Figure 1

In this example it is assumed that power meters are commercial devices that can be queried via MODBUS/RTU over RS485. For this reason, Ethernet/RS485 converters are also shown. However, IP power meters that work with MODBUS/TCP or other protocols are also available on the market. For such devices no Ethernet/RS485 converters would be needed.

Software components

Overview

The server should be equipped with:

- Linux
- MODBUS libraries, if needed
- a free SCADA software able to store the collected data
- software plugins for interfacing the SCADA to the in-field power meters
- SLighliMon

Linux

Any free distribution of Linux should be suited for implementing the concepts described in this document. The resources requirements for running the presented monitoring system are very low. As an example, we are monitoring almost 100 power cabinets using a Debian 7 installation hosted on a one-core virtual machine with 1 GB of RAM (and the average CPU usage is about 30%).

SCADA software

Several free SCADA software that can be extended with ad-hoc plugins are available on the net. In our implementation I used Nagios (www.nagios.org) with other complementary tools. These are:

- pnp4nagios (docs.pnp4nagios.org)
- optional: gearman (gearman.org) + mod-gearman (mod-gearman.org)

In the following some details about such components are reported. However, every SCADA software with similar characteristics could be used. The described implementation dates back to the first months of 2014 and now I can say it's stable. In the meanwhile new stuff came out, have a look for example at Naemon (www.naemon.org).

Nagios

Nagios performs the MODBUS queries to the power meters through dedicated plugins (which I self-developed for our devices). The goal of the plugins is to retrieve from the power meters only the values of the MODBUS registers of interest and return them in a format that Nagios can interpret as so-called “performance data”. In order to retrieve reliable data, Nagios has been tuned to perform a query to a single power meter every 20 seconds, which are then averaged in 1 minute data by a Round Robin Database (RRD), as explained later. In our case each query retrieves all the electrical parameters at the same time.

pnp4nagios

This component allows Nagios to store the power meter measurements into RRD databases. One RRD database is created for each measurement of each power meter. In our setup we set pnp4nagios for operating with RRD databases with a time step of 1 minute. Data are also stored at 15, 30 and 60 minute resolution for convenience (lower resolution data are used to speed up data visualization when large time intervals are required by the user). pnp4nagios also provides a basic web-based data visualization interface that can be open directly from Nagios.

gearman + mod-gearman

These components are not mandatory. In very brief, they allow the detachment of the execution of power meter checks from Nagios to a set of “workers”. Such workers operate independently and inform Nagios about the result of the checks. Moreover, results can be also sent to a second Nagios instance. This system allows the distribution of checks on different cores and also on different machines. In addition, it enables the possibility to perform checks locally in a network segment and send the results both locally and to a remote server. This avoids data gaps due to possible network problems between periphery and center, as data are always stored also in the periphery.

SLightliMon

In this architecture SLightliMon can be used to analyze the data stored into the RRD databases created by pnp4nagios. If the number of power cabinets to be monitored is high, a database with the specific check parameters related to each power cabinet could be created (e.g., containing the consumed power during OFF state for each phase for each cabinet). Then, SLightliMon could be called with automated shell scripts or through a dedicated web interface that generate a report of the last night, week, or of the desired period.

Conclusion

This document briefly presented an example of architecture for street lighting monitoring based on free and open-source standard software. The main idea presented is that by means of a properly tuned standard free SCADA software and SLightliMon it is possible to perform street lighting monitoring using standard commercial power meters connected via IP. This solution may be suited to medium-size entities which already have a deployed network infrastructure on the territory, or that can extend the existing one in order to reach power cabinets. The main advantage of this solution is that the monitoring system is free and open, thus avoiding service fees and allowing the use of any commercial equipment instead of particular brands/models. An architecture based on these principles has been already deployed and is nowadays effectively used for the monitoring of almost 100 power cabinets.