



Implementing Power Visibility for Off-Grid Solar Hybrid Sites

Implementing Power Visibility for Off-Grid Solar Hybrid Sites using the SiteBoss Site Controller

“A range of physical interfaces are being made to different site devices like fuel cell controllers, cameras, relays, and other sensors at the site to create the single dashboard in the SiteBoss UI.”

Mountaintops are common places to locate telecom sites due to the vast distances communication services can cover from a single site. Mountaintops and other particularly remote sites are often a first place that a telecom operator will look to deploy a telecom site automation solution, as servicing one of these locations is expensive, time-consuming, and may be impossible during certain times of year. Often AC grid power at these sites is impractical and another power source needs to be provisioned.

Customer Challenge

Asentria began working with a North American mobile network operator to address issues for remote telecom sites with no AC grid power in the Southwestern United States. The site pictured is powered by 7 photovoltaic solar (PV) arrays during the day which also charges the batteries onsite. The site has a fuel cell that provides power when the solar array can't charge batteries enough by themselves.

The network operation center (NOC) personnel had no easy visibility of how the site was being powered at any one time. If problems would occur, alarm messages didn't adequately indicate what the actual problem was on site to know how to prepare for a costly service visit. Cell technicians were many hours away from the sites by car, and some sites were only easily accessible via an expensive visit by helicopter.



Objective

Improve site resilience by getting more and better data related to site power. Allow for remote or automatic ability

to clear a troublesome alarm at sites. Enable remote IP access for NOC to remote security cameras and other remote equipment for site troubleshooting. Reduce number and improve efficiency of any needed truck rolls.



Power System Monitoring

The following variables were desired to be measured to allow the operator's network operation center (NOC) to be able to remotely gain visibility to how a site was being powered at any time, as well as indicate any alarm conditions.

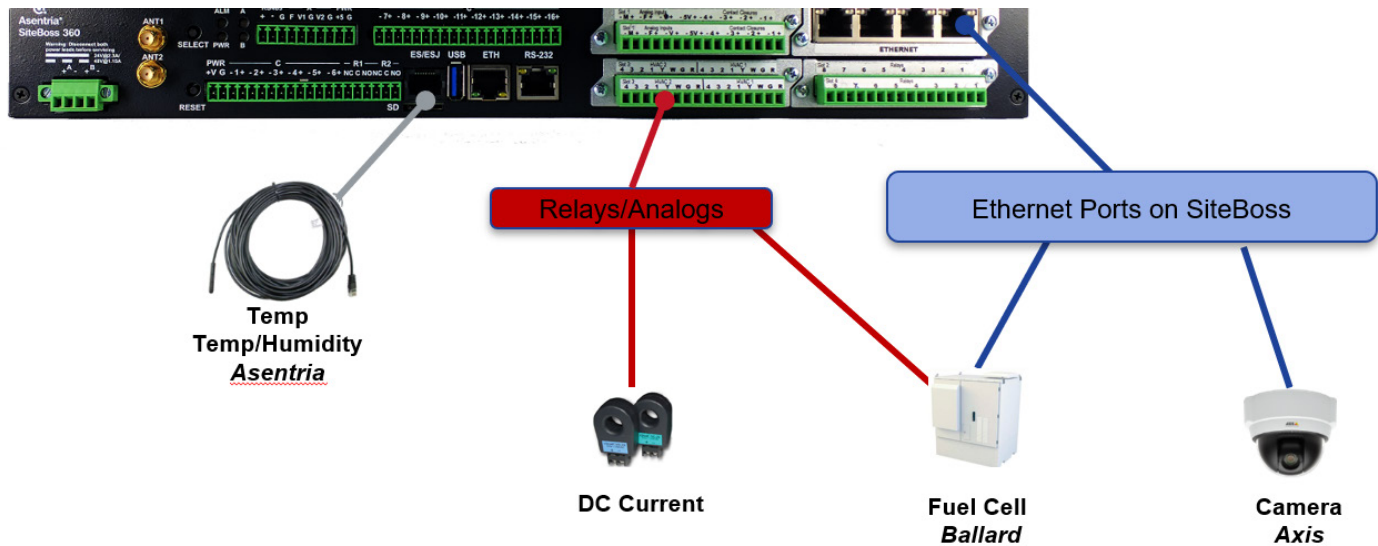
1. Fuel Cell
 - a. Output Current
 - b. Output Voltage
 - c. Status
 - d. Fault status

3. Batteries
 - a. Battery Voltage
 - b. Battery Current

Other Monitoring and Remote Access

1. Shelter temp monitoring (for battery health).
2. Network for site IP devices without using multiple main network IP addresses.
 - a. Connection for IP camera
 - b. IP connection for fuel cell (SNMP proxy and interface forwarding)





Solution = SiteBoss + Asentria Integration Expertise

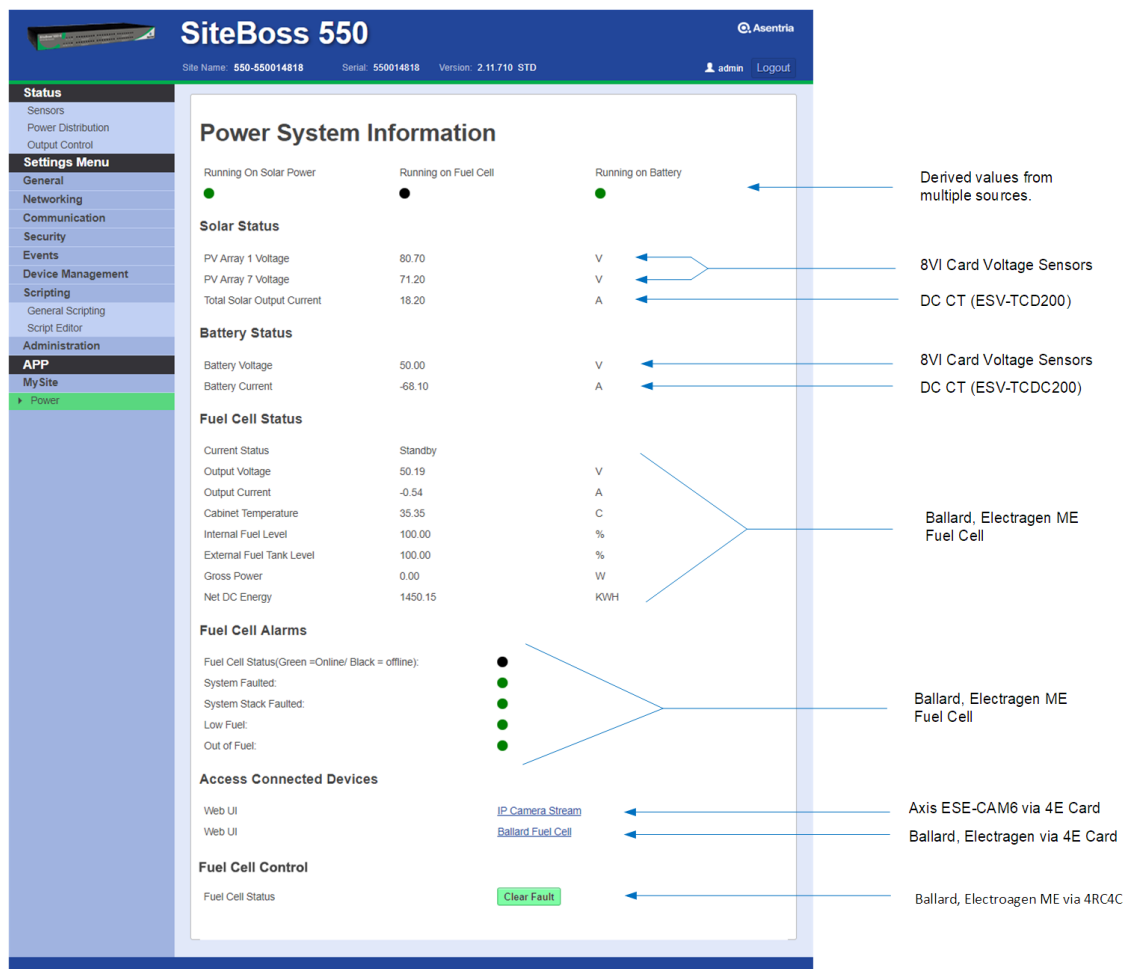
The solution was the use of Asentria's SiteBoss site automation hardware and integration expertise. In the diagram above you can see a SiteBoss example unit. Most SiteBoss units can have multiple optional expansion cards that can be chosen, depending on what equipment or sensors need to be interfaced with at a site. In this example a 4E (four Ethernet port) card, an analog input card, and a relay card were used. The fuel cell and IP cameras were connected via Ethernet ports, and the AC current sensors were connected to the analog inputs. An external relay was controlled by an onboard output relay on the SiteBoss to enable clearing of the alarm on the fuel cell. Asentria's own temperature sensor was used.

Once connected to the equipment shown in the diagram above, the Siteboss is able to gather data from and control

the attached devices. The SiteBoss also serves a separate purpose of acting as a LAN for additional IP connections/addresses for additional IP devices at the site without using ports on the site router. This allowed for the connection for remote access to an IP camera for site security, as well as giving secure remote access to the fuel cell system.

Dashboard Screenshots and Remote Access

The ideal solution was to use the SiteBoss appliance and its web interface as a means of representing all the data from the different systems in a single, easily visible place. An Asentria application engineer was on hand to help install the SiteBoss, integrate it to the various equipment, and represent the data in the dashboard within the SiteBoss web UI (shown on the next page). Asentria offers separate engineering services to customize the use of the SiteBoss in cases like this.



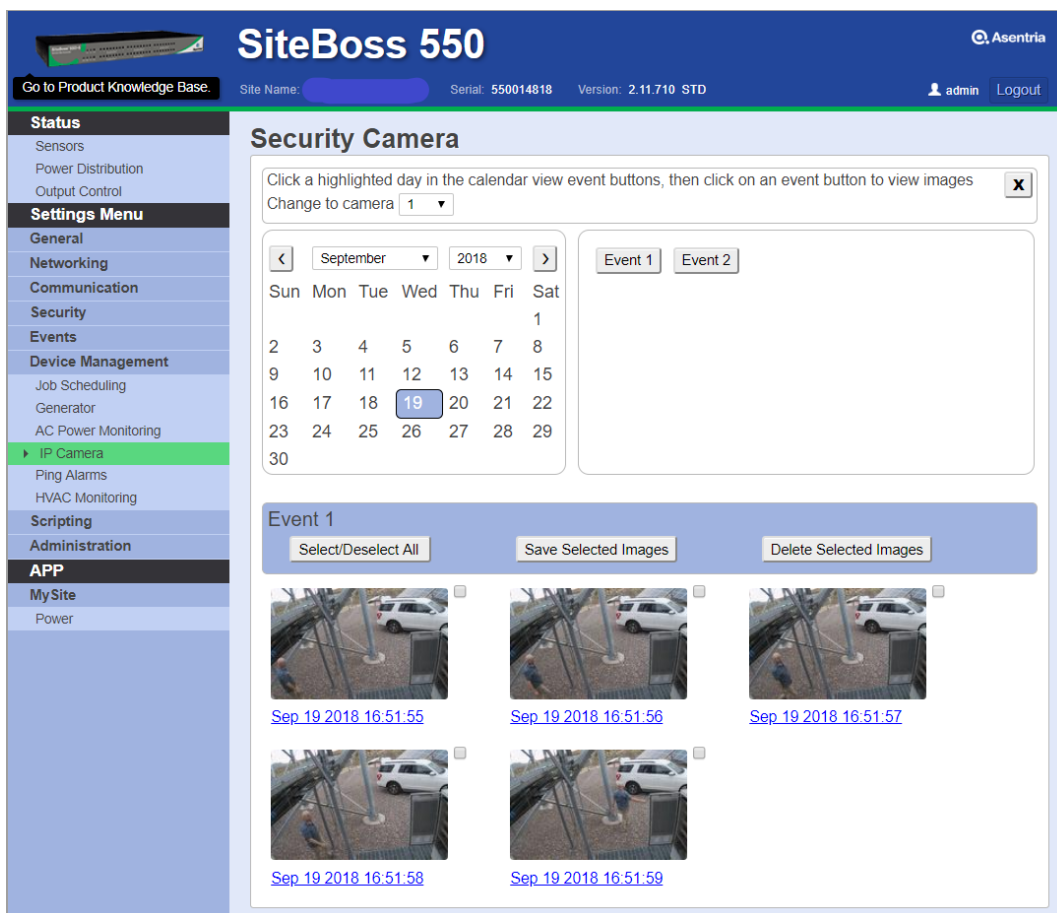
The screen capture above shows how multiple different devices at a telecom site can be represented in a single interface within the SiteBoss. In this example, the SiteBoss creates a single site dashboard representing all the desired values stated in the project objective. It is transparent to SiteBoss users that a range of physical interfaces (shown on the right margin) are being made to site equipment and sensors to create the single dashboard in the SiteBoss UI shown above. The values shown at the very top of the dashboard show the ability of the SiteBoss to create entirely

new values based on data it has gathered from various sources. In this example these new values are overall indicators of what power source the site is currently running on (solar power, fuel cell, batteries), one of the stated objectives before the project began.

At the bottom of the screen there is a button in the web interface to allow for remote relay control to reset an alarm on the fuel cell, preventing an unnecessary truck roll to trip a breaker to manually reset the alarm.

At the bottom of the dashboard screen is illustration of another function of the SiteBoss, the ability to give a user IP connectivity to devices attached to the SiteBoss unit. In this example it is showing links to the web interfaces of two connected devices, an IP camera and the Ballard smart controller.

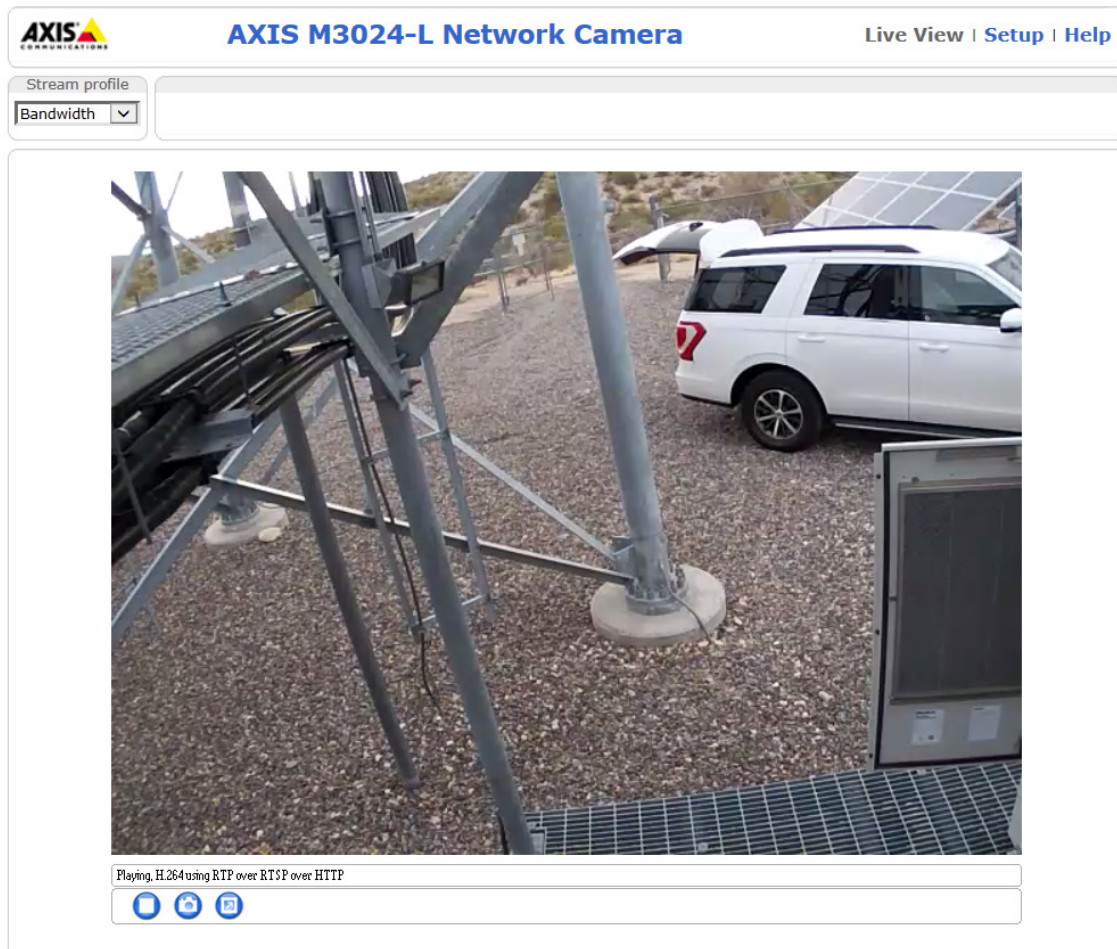
Full-sized camera images generated by the IP camera can be stored in the SiteBoss, and an alarm generated by the SiteBoss to the NOC to reduce the amount of video sent on the primary network backhaul of the site. The screen capture below shows the SiteBoss web UI with a number of thumbnail images captured from the site camera. By clicking on a thumbnail, the full-size image can be viewed.



The screenshot displays the SiteBoss 550 web interface. The top navigation bar includes the Asentria logo, site name, serial number (550014818), and version (2.11.710 STD). A left sidebar contains a 'Settings Menu' with categories like Status, Settings Menu, General, Networking, Communication, Security, Events, Device Management, Scripting, Administration, and APP. The main content area is titled 'Security Camera' and features a calendar view for September 2018. A calendar grid shows dates from 1 to 30, with the 19th highlighted. Below the calendar, there are two event buttons: 'Event 1' and 'Event 2'. Under 'Event 1', there are three buttons: 'Select/Deselect All', 'Save Selected Images', and 'Delete Selected Images'. Below these buttons are four thumbnail images of a security camera feed, each with a timestamp: 'Sep 19 2018 16:51:55', 'Sep 19 2018 16:51:56', 'Sep 19 2018 16:51:57', and 'Sep 19 2018 16:51:58'.

From within the SiteBoss dashboard shown above, it is also possible to click on a link directly to the IP camera itself. This allows for a manageable video security methodology where the NOC follows a process of waiting for motion detected by the IP camera to generate an SNMP trap (alarm). Once a trap is received at the NOC, the NOC personnel can

go directly to the SiteBoss to see thumbnail images of what triggered the alarm (as shown above). If an actual security issue is happening, NOC personnel would then click through the SiteBoss web interface to see live video stream of what is happening at the site (shown below).



```
Function ballard.clearFault() -- Function to automatically clear a fault. Trigger on a fault condition.
  for i=1,5 do -- Try 5 times to clear the fault.
    a_lib.Relay(self.relaySlot, self.relayPoint, "active", 10) -- Set fault clear to active. Close for 10 seconds
    a_lib.Sleep(11000) -- Wait 11 seconds
    if self.faultStatus == false then -- Check the fault.
      a_lib.PostAudit("Fuel Cell Fault Clear")
      break -- Fault is cleared so stop trying.
    end
  end
  a_lib.PostAudit("Failed to clear fault, technician required") -- Write audit log
end
```

Telecom Site Automation

The values shown in the SiteBoss web interface above are also available in ways that can allow for automation of processes at a site. The SiteBoss units support scripting languages, SNMP functions, and RESTful API. As one example, above is the function that underlies the “button” in the web UI to reset the alarm on the fuel cell. Instead of having a human operator go to the web UI of the SiteBoss to “click the button” to reset the alarm condition, the SiteBoss is configured to handle this problem immediately and on its own. Upon receiving the specific troublesome alarm from the fuel cell, the Siteboss has a direct “alarm action” to call the function “ballard.clearFault” shown above, triggering a physical relay to try and clear the alarm. The SiteBoss runs the routine autonomously. This is a simple example of what we refer to as telecom site automation. **In this case telecom site automation reduces the cost and delay of human intervention.**

Conclusion

The SiteBoss is a powerful and flexible telecom site automation tool. This is just one example of the type of solutions that can be created. We will write further case studies in the near future to illustrate different uses of telecom site automation.

Asentria helps create more resilient and cost-effective networks via telecom site automation. For more information on our products and the benefits of telecom site automation, visit our website at www.asentria.com

