November 2023

ENERGY MANAGEMENT FOR EDGE SITES

Andre Surcouf

Trevor Whinmill

Fabien Andrieux

Trevor Smith

ENERGY MANAGEMENT FOR EDGE SITES

AUTHORS:
Andre Surcouf
Trevor Whinmill
Fabien Andrieux
Trevor Smith

ABSTRACT

Controlling energy consumption for edge sites is becoming a more important aspect of network management. For example, for far edge sites that may be powered by renewable energy, such as solar panels, it is mission critical to reduce energy consumption at such sites to only the resources that are strictly necessary to avoid any interruption of service. Further, for large edge sites that can include several compute nodes, such nodes typically operate at all times regardless of what applications may or may not be operating on the nodes. Various techniques are proposed herein that may facilitate optimum energy management for edge sites in a network environment. Such techniques may provide for controlling power consumption at edge sites even in the absence of direct access to such sites by determining an optimal compromise between application availability and edge site power consumption without the use of specialized/bespoke hardware at edge sites.

DETAILED DESCRIPTION

Edge sites often contain several edge computes nodes that are connected to a local network (LAN). Often, these compute nodes are not fully independent and participate in conjunction with some logical compute entity such as, but not limited to, Kubernetes®, which is one popular example of such a logical entity. Kubernetes® is a registered trademark of the Linux Foundation.

Various examples provided herein are described with reference to Kubernetes-based examples to facilitate granular power management at edge sites; however, it is to be understood that the principles and apparatus discussed for various techniques of this disclosure can be implemented for other types of distributed edge runtime software/logic. Figure 1, below, illustrates an example edge site architecture.

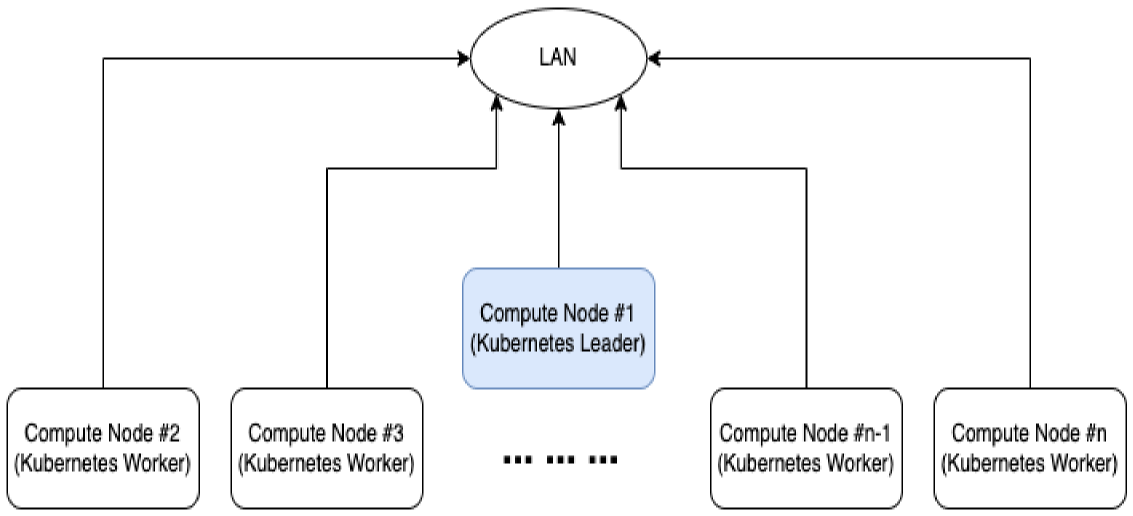

Figure 1: Example Edge Site Architecture

For the example edge site architecture as shown in Figure 1, the compute nodes are not considered to be identical. Compute Node #1 is considered to be the Kubernetes leader that will play a particular role for the energy management techniques proposed herein. In some cases, there could be more than one leader node (e.g., 3 or 5) but this does not change the logic of the proposed energy management techniques. The other compute nodes are not necessarily all identical, as each could have different hardware capabilities (unique Graphics Processing Unit (GPU) capabilities, special Input/Output (I/O) capabilities, etc.).

For the example architecture, it is to be understood that Kubernetes could be replaced by any other distributed run-time software, as long as such software makes a clear distinction between nodes logically responsible for management node(s) and controlled node(s) providing compute and more general hardware resources; however, this does not prevent a management node to be used for executing applications as do the controlled nodes.

Two different approaches are discussed herein in order to facilitate energy management and control of power consumption for edge sites, a proactive energy management mode and a controlled energy management mode. For both approaches/modes, consider various assumptions involving edge sites.

For example, the energy management approaches proposed herein assume that edge sites are not fully autonomous and are logically controlled by a control plane that can be hosted in a cloud platform and interact with multiple edge sites through, but not limited to, the open internet. The edges sites may or may not be permanently connected to their respective control planes, however, the permanent or temporary nature of the connection between an edge site and its control plane does not affect operation of the proposed approaches.

Several system components may be utilized to facilitate both the proactive and the controlled energy management modes, including:

- A power management controller;
- A hardware resources inventory for each edge site; and
- A power management agent provided for each edge.

Regarding the hardware resources inventory, such an inventory may be considered as part of the control plane. The inventory for a given edge site may represent an up-to-date inventory of the hardware resources that are available the edge site and/or an indication of occupancy/utilization of hardware resources at the edge site. When an application is deployed the control plane (based on a description of the application) can determine which hardware resources are utilized by the application. In both the proactive and the controlled energy management modes, it is assumed that each edge site is already configured (e.g., all nodes are booted and running) and are identified by the control plane to facilitate maintaining the hardware resources inventory.

Regarding the power management agent, the agent may be characterized as an edge run-time extension that is responsible for reporting current edge site power state to the control plane, as well as for managing each individual worker node power state. Thus, for the techniques proposed herein, each edge site is considered to be configured with an active power management agent that is deployed on the leader node(s) for each site in which the at least one or several leader node(s) will never be powered-off (e.g., always running).

Accordingly, the power state of each node can be externally controlled by the power management agent that is acting on behalf of the power management controller. Such control could be achieved through various mechanisms, such as compute nodes supporting wake-up-on-LAN, compute nodes being powered by an external Power over Ethernet (PoE) switch that could be externally controlled by the power management agent, and/or by the compute nodes having individual power sources that could be externally controlled by the power management agent.

During operation, when a compute node is awoken up by the power management agent, the compute node is to automatically rejoin the edge site of which it was part before being shut down. Currently, this is a Kubernetes built-in function, but such operations could also be implemented as a control plane functionality.

Consider various example details regarding the proactive energy management mode in which operations involving the proactive energy management mode are shown below in Figure 2.

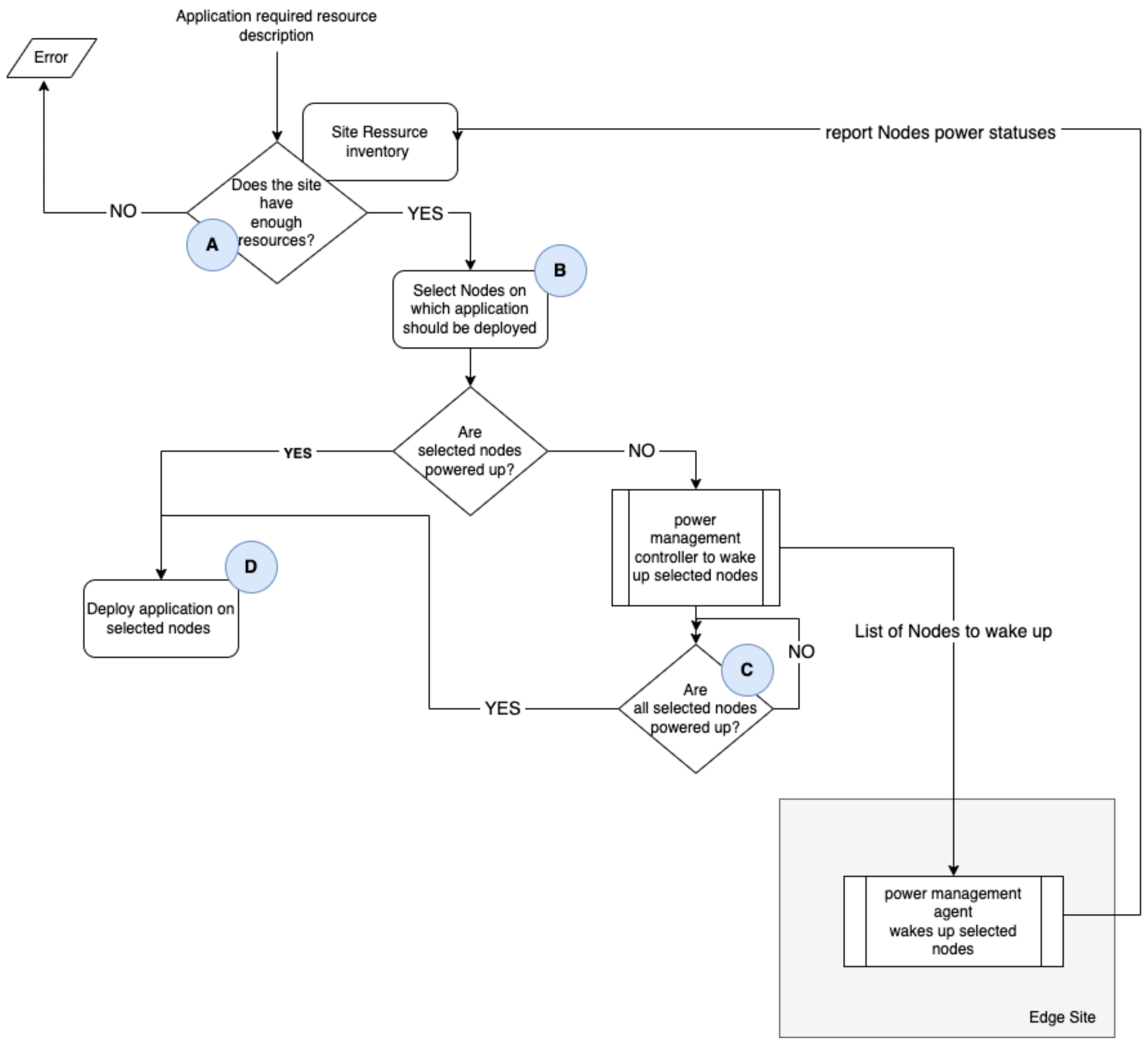
Application required resource description
Error
Site Ressurce inventory
report Nodes power statuses
Does the site have enough resources?
NO
YES
A
B
Select Nodes on which application should be deployed
Are selected nodes powered up?
YES
NO
power management controller to wake up selected nodes
D
Deploy application on selected nodes
List of Nodes to wake up
NO
C
Are all selected nodes powered up?
YES
power management agent wakes up selected nodes
Edge Site

Figure 2: Proactive Energy Management Mode Operations

In the proactive energy management mode, when an edge site is configured and is ready to execute an application, all nodes at the site, except for the leader node(s), can be shut down by the power management controller and a status regarding the shut-down can be provided to the control plane.

As illustrated in Figure 2, at Step A, the control plane can perform a test by comparing the resources that the application seeks to utilize with the resources of the targeted site. This comparison can be simply based on the site's static resources inventory (the one that is created upon site creation and initialization) or, in a more elaborate version, can be based on the remaining available resource availability for the site, which can take into account resources already consumed by other applications already deployed at the site. The site's static resources inventory can be updated each time a node is added to the site or removed from the site. A node is therefore considered as part of a site regardless of its power state.

At Step B, the control plane determines which node(s) is (are) required for the application by also comparing each node's individual resources with what the resources (node(s)) that application seeks to utilize. This comparison also could be simply based on the site's static resources inventory or could be based on the remaining available site resource availability, which takes into account what has been already consumed by applications already deployed on a node. The node static resources inventory is created when a node is added to a site and destroyed when a node is removed from the site.

If the selected nodes (or some of them) are not already powered-up (which can be identified in the hardware resource inventory) a list of the node(s) to wake up can be sent to the power management controller. The power management controller can transfer the list to the edge site power management agent (e.g., via any push/pull mechanism using any combination of temporary and/or permanent connections).

As shown at Step C, a test is performed to determine when all the selected node(s) are powered-up, based on reports from the edge site power management agent. Once all nodes are powered, the operations can move to Step D at which the application can be deployed (only) on the selected node(s) (e.g., preventing the Kubernetes scheduler from choosing other node(s)).

When an application is undeployed, a reverse process is triggered. In this case, the power management controller will carefully analyze the resources required by the remaining running applications and, if the power management controller determines that one or several nodes are no longer being utilized, the nodes will be powered down.

The advantage of this approach is that it guarantees minimum power consumption at any point in time since only required nodes will be powered up. However, a potential disadvantage to this approach is that the time required to deploy an application will be extended by the time it takes for the required nodes to boot-up and to rejoin the edge site as worker nodes.

Optionally, to avoid unnecessary power-down / power-up short cycling, when an application is undeployed, the nodes that are not used anymore may not be powered down immediately. Rather, the power management controller can first check to determine whether there are any applications in the deployment queue that could utilize a newly available node. If the application deployment queue is not empty, any new application(s) can be deployed before any other power management considerations are triggered. As an extension of this approach, an “always-on” list of nodes may be defined, which could improve application deployment time and/or to guarantee that some critical hardware resource will be always immediately available.

As another extension for the proactive mode, the power management agent can be integrated with the autoscaling capabilities of Kubernetes (i.e., horizontal pod auto-scaler) in addition to scaling the powered nodes to meet initial deployment criteria. This will allow for the ability to detect autoscaling requirements and to iteratively monitor resources in order to bring nodes up/down to meet run-time resource requirement fluctuations.

Moving to the controlled energy management mode, for the controlled energy management mode, when an edge site is configured and ready to execute an application, all nodes remain powered-up, meaning that the site will run at full power. Corresponding nodes statuses can be reported to the control plane in this mode as well.

An advantage of the controlled energy management mode is that since all nodes are running, application deployments will consume a minimum of amount of time as there is no need to reboot any node(s). However, a potential down-side to this approach is that power consumption will be maximized. As opposed to the first approach where the power management policy is fully driven by the control plane, the controlled approach provides fine-grained control to a user. As an example, a particular user could decide that a site is to run at full capacity during business hours and is to be placed into low power mode overnight. Similar considerations could be utilized for edge sites powered by solar panels.

Figure 3, below, illustrates various operations that can be utilized when a user decides to place a site in a lower power mode. For operations involving the deployment of new application(s) for a site, operations as illustrated for Figure 2 involving proactive energy management can be used to determine a current power status for the site.

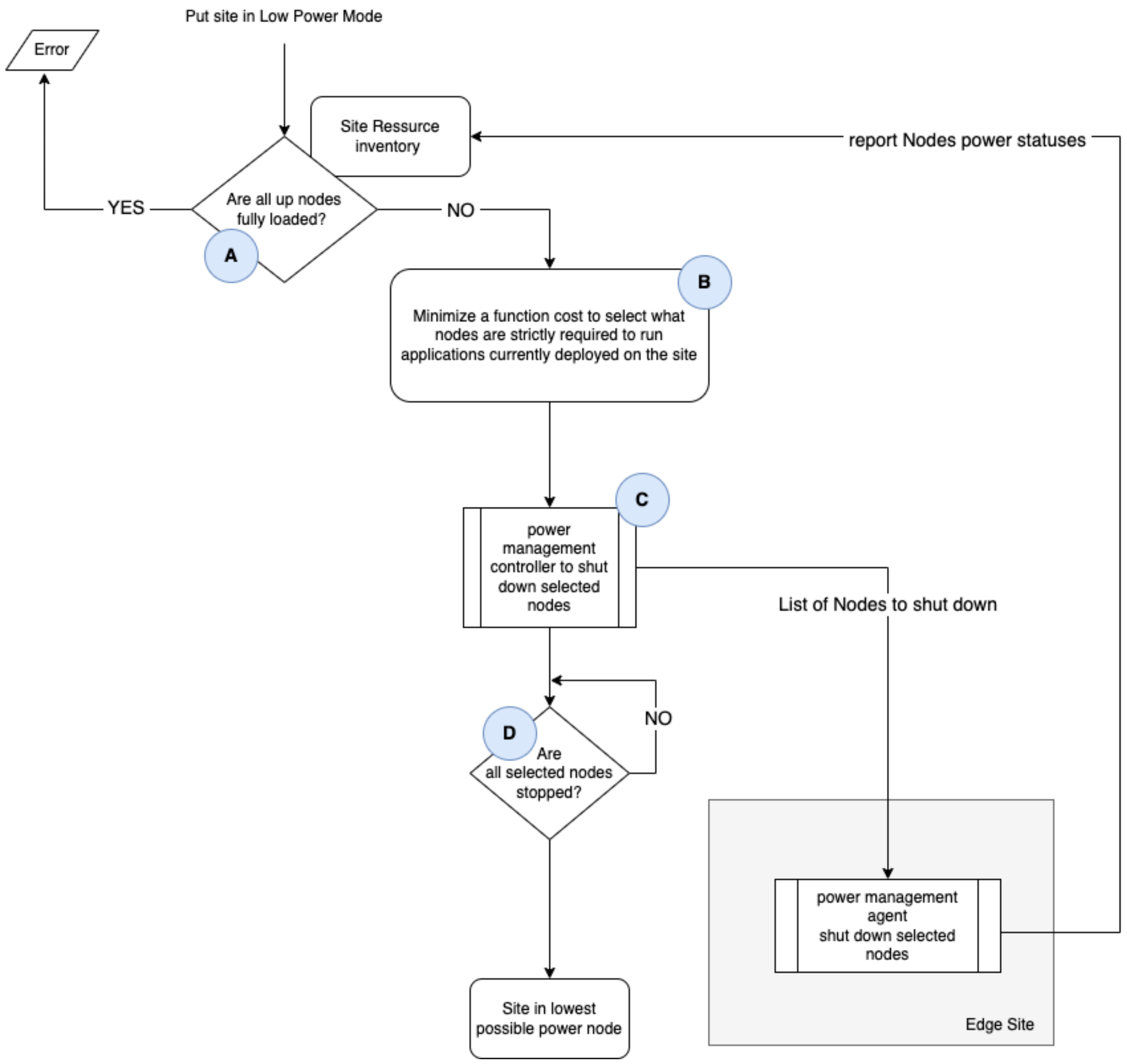
Put site in Low Power Mode
Error
Site Ressurce inventory
report Nodes power statuses
YES
Are all up nodes fully loaded?
NO
A
B
Minimize a function cost to select what nodes are strictly required to run applications currently deployed on the site
C
power management controller to shut down selected nodes
List of Nodes to shut down
NO
D
Are all selected nodes stopped?
power management agent shut down selected nodes
Edge Site
Site in lowest possible power node

Figure 3: Controlled Energy Management Mode Operations for Placing a Site in a Low Power Mode

As illustrated in Figure 3 at Step A, consider that the control plane receives the command to put a site into a low power mode, which triggers the control plane to determine if all the running nodes from the sites are fully loaded. If the nodes are fully loaded, the power down command is rejected since it is not possible to reduce the power consumption of the site considering the current set of applications running on the site.

However, as shown at Step B, if not all nodes are fully loaded there might be a possibility to shut down some nodes by redistributing the running applications on the nodes from the site. To this extent, the power management controller can try to minimize a cost function representing the power consumption of the whole site. This cost function can be roughly described as follows:

- Gather each running application's resource requirements;
- Start by creating groups of applications based on which resources the applications utilize. For instance, one group of applications might utilize only Central Processing Unit (CPU) and memory resource, whereas another group might require CPU, memory, and GPU resources, etc.;
- Associate each group of applications to a subset of nodes exposing these (and possibly only these) resources and then determine if the subset of nodes has enough resources to support the application group in question. Since not all nodes will not be the same, their power consumption can be also different. This is another input of the cost function (for instance, running two small nodes can be more costly than running a large node capable of supporting the same set of applications);
- At the end of the group creation/association process, a new distribution minimizing the function cost of these applications across the different nodes can be defined;
- As soon the new application distribution is known, applications can be moved across the nodes accordingly (e.g., based on runtime capabilities, un-deploying and redeploying each application, etc.).
    - To avoid disrupting mission critical applications, such applications can be marked as “not moveable,” in which case, nodes on which these

applications are running will be marked as “protected,” meaning that the cost function will be authorized to move applications on these nodes (assuming they have enough remaining available resources) but will not be authorized to remove “not moveable” applications from these nodes.

Moving to Step C, at the end of the above-described re-organization process, a list of candidate node(s) will be defined. This list will then be passed to the edge site power management agent, which will start to shut down the identified node(s). As shown at Step D, the power management agent will report new power state of the selected node(s) (this new status will also appear in the hardware resource inventory). As soon as all selected node(s) at the site are shut down, the site will be declared to be in a lowest possible power consumption state. In some instances, the actual power consumption for the site may also be reported.

As an extension to this approach, in some instances, a power savings target might be defined for a site (e.g., 30% of the full site power). This target can be expressed in percentage or in watts for different edge site situations and/or characteristics.

Further, to maximize power saving options (e.g., to give the power management controller more room to maneuver), some applications might be marked as “optional." In this case, to achieve a power saving target, the power management controller could decide to temporarily un-deploy one or more "optional" applications. As a further extension, a new power saving target could be defined (e.g., increased from a low target to a higher target), which could trigger the power management controller to wake-up some nodes to redeploy one or more “optional" applications that may have been evicted due to a previous, more constrained, or lower power consumption target.

Due to potential intermittent connection between an edge site and the control plane, the controlled energy management mode could be enabled only if the edge site is capable of monitoring resources in order to detect when it is to pull itself out of lower power mode, for example, as resource demand spikes due to application activity. In other words, the power management agent could have the ability to momentarily overshoot the power consumption target (e.g., can add 20% to the current target) to temporarily accommodate application demand.

In summary, techniques herein provide several approaches for solving the delicate problem of controlling power consumption of edge sites, even in the absence of direct access to edge sites. The techniques do not simply apply a brute force approach through which compute nodes are simply turned on or off; rather, the techniques proposed herein seek to determine the best possible compromise between application availability and edge site power consumption without the use of specialized/bespoke hardware operating at edge sites.