Distributed Systems — Power actions in a smart grid

Bart Wiegmans & Hessel van Apeldoorn

October 23, 2014

1 Context/background

Since the beginning of the 21st century the roles of energy suppliers has changed. In the beginning the suppliers only delivered energy to their customers for a fixed tariff, which the energy suppliers produced when there was a demand for it. Since the emergence of natural energy production from sources such as the sun and wind, the energy suppliers have to deal with a production of energy even if there is no demand for it. This results in a problem where the energy supplier gets stuck with an overproduction of energy. This problem in combination with several other factors led to the development of the Smart Grid. One of the features of a smart grid is an automatic load balancing based on production of energy. For example by shutting down or starting up a refrigerator in order to deal with the imbalance between production and consumption of energy on the grid. An extra functionality it offers in combination with the balancing function here above is to stimulate balancing of the grid by offering discounts on the energy price in case of overproduction or higher prices in case of energy shortage. By doing this local energy users, as for example server farms, can decide to reduce or increase its usage and so saving money and reducing the imbalance on the grid.

A second change in technology since the beginning of the 21st century is the shift from local computing to cloud computing. Today more and more services and software are placed in the cloud. The cloud is a general name for a large network of servers connected with each other through the Internet. By connecting a large group of server the storage, computing power and other resources of individual servers can be combined to process more complex task or balance multiple tasks over the servers that are idle at that moment.

A smart grid also has to be present in a data center. The data center itself receives a steady stream of power that it has to split among the nodes in its grid. nodes are in general servers in the data center. A server may sometimes require less power and may sometimes require more. A server should always receive its required amount of power. In general, servers are not turned off and thus in total there should always be enough power for all servers.

2 State of the Art

Advanced metering infrastructure Multicast

3 Problem statement

as more and more computing take place on servers it becomes interesting if tasks are cost efficient. are the costs lower than the potential revenue generated by it.

4 Relation to Distributed Systems

Dynamic host discovery, server can be added or removed. Needs to be a broker, so a leader inside the system. Server must be ensured that they actually can bid and use energy, so reliable channels to the broker are important.

5 Solution details

In our project several algorithms and techniques have been used. In this section we try to give a better inside in where and how these techniques were used.

5.1 IP multicast

IP multicast is the most important communication technique in our project. IP multicast is used for sending messages within a group. It works as follows in our project: A process creates a message which it marks as a message to be sent by multicast. The message is then put into the output queue of this process. When the sender thread of this process has time to sent the message, it encodes the message and sends the message to the queue of the group to which the process belongs. Every client of this group can now grab and decode the message that has been sent.

These are the steps necessary for basic multicasting. Our multicast should also be ordered and reliable. To achieve this we give a sequence number to each message that we want to sent. This gives us a way to check in what order we should receive messages. Furthermore, we introduce two new components: a resend buffer and a resend requester. The resend buffer stores two types of message: all sent (unicast or multicast) messages and all (sent or received) multicast messages. The resend requester gathers all messages that we expected to receive, but haven't received yet. Practically this means all messages that we expect to receive but can't find in the resend buffer. In the case of a multicast message, send the sender a request to resend the message. If the sender is dead, send a request to the entire group. In the case of a unicast message, send the sender a request. In case the sender has died, the receiver acts like the message has never existed.

There are a few applications for multicast in our project:

- **Sending of heartbeats:** The broker sends out heartbeats to all members within a group using multicast. When a member receives a heartbeatmessage, he will send back an acknowledgement message to the broker. This acknowledgement message is only send to the broker. As such, the technique used to send back a message is unicast instead of multicast.

- **Removing a member:** Whenever a member is non-responsive (e.g. when no acknowledgements of heartbeats are received), a message is sent to all members of the group to inform that this member is leaving the group. These messages are sent using multicast.

- **Setting the energy price:** A broker sends a message with the energy price to all its client through multicast.

5.2 Bully algorithm

main solution algorithms IP multicast bully algorithm

6 Results

technical implementation Fault tolerance