Software Defined Grid Energy Storage

Sonia Martin, Nicholas Mosier, Obi Nnorom Jr., Yancheng Ou, Liana Patel, Oskar Triebe
Gustavo Cezar, Philip Levis and Ram Rajagopal*
Stanford University
Stanford, CA, USA
{soniamartin,nmosier,obdk,ou2,lianapat,triebe,gcezar,ramr}@stanford.edu,pal@cs.stanford.edu

ABSTRACT

Today, consumer battery installations are isolated, physical devices. Virtual power plants (VPPs) allow consumer devices to aggregate for grid services, but they are are vertically integrated, vendor controlled systems (e.g., Tesla's VPP). Consumer batteries are therefore unable to participate in energy markets or other grid services outside what their vendor provides.

We describe a software system that provides software control of multiple, networked battery energy storage systems in the electric grid. The system introduces two new ideas that enable flexible and dependable management of energy storage. The first is a virtual battery, which can either partition a battery or aggregate multiple batteries. The second is a reservation-based API which allows asynchronous control of batteries to meet contractual guarantees in a safe and dependable manner.

Virtual batteries and a reservation-based API address the unique challenges of achieving high and efficient utilization of energy storage systems, including heterogeneity of battery systems such as varying C-rates, participation in energy markets, utility bill management systems, community resource sharing, and reliability. Using a testbed comprised of sonnen Inc. storage units installed in several homes and a lab, we demonstrate that virtualized batteries can seamlessly replace physical batteries, flexibly manage energy storage resources, isolate multiple clients using a shared battery, and create new energy storage applications.

CCS CONCEPTS

• **Hardware → Batteries**; • **Networks → Logical / virtual topologies**; *Cyber-physical networks.*

KEYWORDS

battery energy storage systems, virtualization

ACM Reference Format:
Sonia Martin, Nicholas Mosier, Obi Nnorom Jr., Yancheng Ou, Liana Patel, Oskar Triebe, Gustavo Cezar, Philip Levis and Ram Rajagopal. 2022. Software Defined Grid Energy Storage. In *The 9th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation (BuildSys '22), November 9–10, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3563357.3564082

*Author list is alphabetical students then alphabetical faculty and staff.

1 INTRODUCTION

Batteries are a fundamental and necessary component of any zero-carbon power strategy. While renewable sources such as solar and wind provide zero-carbon power, they are intermittent and their production profile does not match demand. This leads to a phenomenon called the "duck curve", in which solar power reduces daytime fossil fuel demand, but evening energy usage is still primarily carbon-based [7, 11]. Renewable power reduces the total *energy* required from fossil fuels, but does not significantly reduce the peak power required [33]. Time shifting renewable energy to nighttime or peak periods is required for carbon-free power. Without storage, solar farms cannot reduce the carbon footprint of buildings at night [10, 26]. Energy storage systems, however, cost much more than a comparably sized renewable generator. A 6kW solar panel system that generates 60kWh of energy per day costs $6,000-9,000, while a 13.5kWh Tesla Powerwall costs $12,000. Moving to a carbon free power future thus requires efficiently using energy storage.

Until recently, consumer storage and solar could not participate in large-scale U.S. energy markets. Recent regulations, however, have lifted this restriction. In September 2020, Federal Energy Regulatory Commission (FERC) Order 2222 ruled that consumer solar and energy storage can now participate in regional wholesale energy markets if they aggregate into larger resources [15, 16]. Under FERC 2222, someone can combine 1,000 Tesla Powerwalls to offer 5MW of power. In fact, combining all of the installed 200,000 Powerwall [31] units in the United States [13] totals 2.7GWh, which surpasses the 2.6GWh of industrial large-scale battery installations [25].

Unfortunately, today there is no common API or interface to aggregate these energy storage resources. Consumer systems are controlled through a vertically integrated stack from batteries to cloud APIs and operated as explicit, physical devices. Current technology limits consumers to aggregating their energy only with those who own a similar brand of energy storage resources. For example, the Tesla Virtual Power Plant is a vertically integrated system of Tesla products only. Recent work such as AutoShare has shown there are significant benefits when communities can partition pooled energy resources [23], but this is impossible with APIs today. At the same time, industrial-scale systems are controlled through IEC [9], SunSpec [2], or proprietary protocols.

An API capable of facilitating community pooled resources has two requirements: partitioning and aggregation. Partitioning allows for many use cases, such as ensuring that a portion of a battery is set aside for specific purposes, while the remaining portion of the battery can be used freely, completely isolated from the reserved portion. Aggregating batteries is also beneficial for consumers. Today, aggregation is the only way consumers with excess energy can participate in energy markets. Current technology provides some, but not all, of these functionalities.

This paper proposes a solution to seamless partitioning and aggregation by managing networked batteries similar to a computing resource. By decoupling the abstraction of a logical unit of energy storage from a particular physical battery, partitioning and aggregation become agnostic of system size or chemistry. Much like a logical disk can be a physical disk, a partition of a physical disk, or an aggregate of several physical disks (e.g., RAID), a logical battery can be a battery, a partition of a battery, or an aggregate of several batteries.

There are three major challenges to virtualizing energy storage in this way: defining an interface, ensuring reliability, and enabling scalability. The interface challenge is that an API must transparently present a uniform abstraction for logical batteries while flexibly supporting energy storage system policies. This API must be able to partition storage among multiple users [23] and aggregate both heterogeneous [8] as well as distributed [16] resources. The reliability challenge is due to consumer storage units connecting through home Internet connections; the system must be dependable despite disconnection. Finally, due to the scales involved (consumer devices produce kW but large buildings and wholesale markets deal in MW), the API and system must scale to hundreds of batteries and complex topologies.

We address the challenge of defining an interface by proposing the abstraction of a *virtual battery*, called the Battery Abstraction Layer (BAL). Virtual batteries can be either partitions, which split a single battery into multiple, smaller batteries, or aggregates, which combine multiple batteries into a single, larger one. Virtualizing real battery storage systems requires answering novel research questions, such as how to calculate aggregate battery ratings and how battery changes are reflected in partitions. We show that simple answers, such as adding values or proportionally sharing changes, are either incorrect or violate use case requirements.

Charge/discharge leases address the reliability challenge by allowing control systems to rely on predictable behavior despite network disconnection. A request to charge or discharge a virtual battery has a time and a duration. This API is simple and general: it supports a wide range of use cases and can control many battery systems, including cloud-based RESTful APIs, direct-connected battery management systems (BMSes) and IEC 61850-based systems.

Energy systems deal with timescales drastically longer than computing. Physical battery storage systems introduce delays of up to tens of seconds. As energy storage is a geographically local resource (one cannot transfer power from Texas to Illinois), network latencies are in the tens of milliseconds. Using asynchronous I/O over commodity Internet connections therefore allows virtual batteries to easily scale to support wide and deep topologies.

We have implemented battery virtualization in a Linux software platform. Internal driver implementations map BAL to physical batteries or networked batteries, which the system can virtualize and re-export as networked batteries. We present results for an energy testbed consisting of grid-connected energy storage units installed in homes and an energy research lab. Using several battery topologies, we present testbed experiments that show that BAL can implement energy storage use cases that are not possible with current APIs.

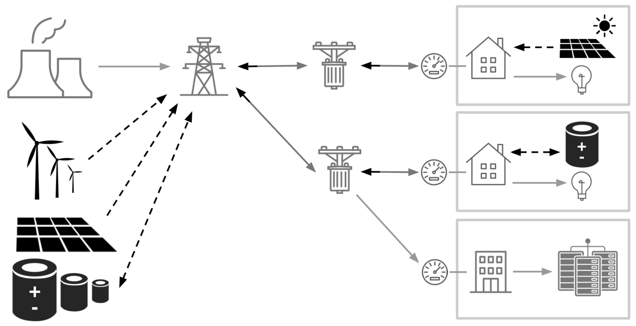

Figure 1: Today, intermittent renewable energy causes uncertainty, while DERs located at both transmission and distribution levels can cause bidirectional flows of energy.

2 MOTIVATION

In the last decade, the introduction of distributed energy resources (DERs) have moved the electricity grid away from a centralized model. Figure 1 shows the grid today, in which consumers now generate and store energy with solar and battery installations, such as sonnen ecos [28, 29] and Tesla Powerwalls [31]. Clean energy penetration has increased, but sources such as wind and solar are volatile and driven by weather, not human planning. Recently, California was able to generate 97.60% of its power carbon-free in the middle of the afternoon [6], but only 33.09% on average. Individuals and companies are now looking to energy storage systems to ensure round-the-clock carbon-free power.

Residential energy storage installations are vertically integrated systems for storing solar energy and reducing home loads. To a utility, their power draw at a meter is indistinguishable from other home loads. Motivated by recent regulatory changes [16], such consumer energy storage units can now participate in energy markets, if aggregated. However, users are restricted with whom they can aggregate their energy. With current technology, an owner of a sonnen battery could not aggregate their energy with a group of homeowners that all own Tesla Powerwalls. As a result, consumers with less popular brands of energy storage resources have fewer opportunities to participate in these energy markets. While prior work such as AutoShare has explored how *centralized* energy storage units can be shared among many users [23], the ability to aggregate *distributed* energy resources remains unsolved. Besides providing consumers with the opportunity to participate in wholesale energy markets, such a capability also enables new use cases for energy storage.

In the rest of this section, we describe four example uses of distributed energy storage that are not possible with existing storage systems. We explain the technical details around a homeowner's association and owners, but they represent much broader use cases. We use these four examples to motivate a set of requirements for a new, programmable abstraction.

2.1 Sharing a Centralized Installation

Figure 2 shows the first use case, in which a homeowner's association (HOA) has invested in a pool of solar and storage systems and

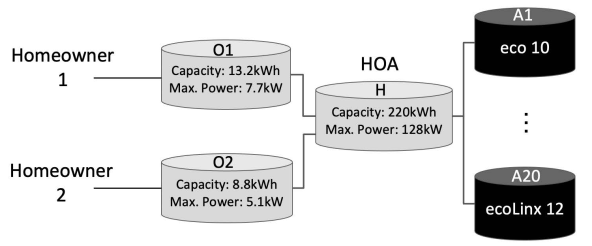

Figure 2: A homeowner's association purchases a pool of solar/battery systems and sells fractional shares to individual homeowners. To each homeowner, the battery share behaves the same as a physical battery they control.

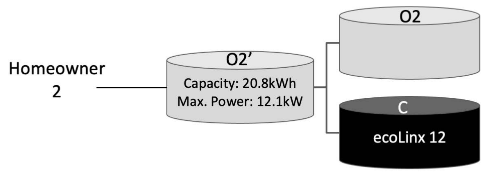

Figure 3: Homeowner 2, needing more energy, supplements their share O2 from the HOA with by installing battery C in their home and combining their two resources.

rents portions of it out to homeowners. Similar to the community solar system proposed in AutoShare [23], the HOA purchases an array of battery systems A1-A20 and combines them into one large pool H. It then sells or rents fractions of H to homeowners. As shown in the figure, Homeowner 1 and Homeowner 2 rent O1 and O2 from the HOA's battery system, respectively. Over time, the HOA can increase or decrease the aggregate battery power and capacity of the pooled resource, H, by adding or removing physical batteries. Conversely, fractional battery shares can be re-allocated to accommodate new homeowners without changing the underlying physical battery resources. As a result, the fractional battery shares and physical batteries in the HOA system can scale independently of each other. As the batteries in the installation degrade with age, this reduction is spread evenly across the shares.

2.2 Aggregating Distributed Batteries

Figure 3 shows a use case in which Homeowner 2 decides they need more storage to reduce their dependency on the grid during high-cost peak hours. Unfortunately, the HOA has already rented all of its capacity, so the homeowner decides to supplement their HOA share by installing a unit C in their home. They combine C and O2 into a larger energy store, O2'. Although it consists of distributed batteries that are behind different meters, the homeowner is able to manage them as a single energy store. Today, this kind of aggregation is not possible: the home unit has its own management system, while the HOA share has another.

2.3 Selling Storage Capacity

Figure 4 shows a use case in which the homeowner decides to sell a fraction of their storage to a regional aggregator. This could happen, for example, when their evening energy use drops significantly. They set aside 20% of their storage for their own use and sell the

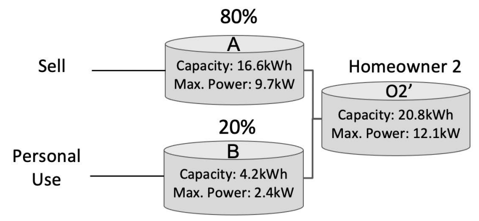

Figure 4: Homeowner 2, no longer needing all of their available storage, decides to rent a large fraction of it to someone else. They keep a small fraction for their own use (e.g., to shave their evening power draw).

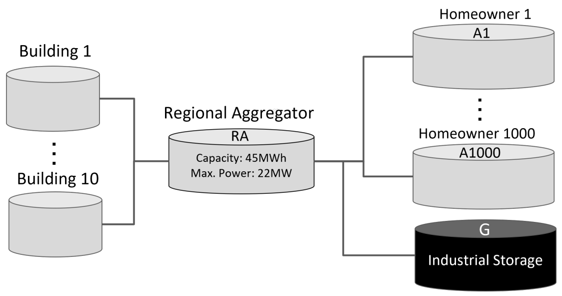

Figure 5: A regional energy aggregator rents hundreds of homeowner batteries, combining them with an industrial storage system. It sells portions of this large storage pool to companies seeking to store carbon-free energy.

remaining 80% to a regional aggregator within the wholesale energy market, according to FERC 2222 [16]. Because energy markets require participants to contract in advance to provide or consume a specific amount of energy, the homeowner needs to isolate their own use from their contractual obligation to the aggregator. Thus, partitioning the aggregated storage unit O2' into units A and B helps Homeowner 2 isolate the energy storage associated with a contractual obligation so other uses, such as load shaving, do not violate the contract. Users who desire privacy and use the battery for their own purposes do not need to register with their billing entity; registration is only required for users that desire economic compensation for their grid services.

2.4 DERs for Large Buildings

Figure 5 shows how a regional energy aggregator can combine storage from many homeowners into a grid-scale resource offering tens of MW. This storage resource can also include an industrial installation. At the scale of tens of MW, the regional aggregator can sell or rent this storage to large office buildings, datacenters, or companies committed to decarbonization. Because distributed energy storage systems are not perfectly reliable, shares of this large resource can have differing levels of dependability and be priced accordingly. A company that is committed to decarbonization may purchase a dependable energy store to ensure that it can match its loads, while another that is simply interested in shaving loads might pay less for a less dependable one.

2.5 Problem Statement

Based on the above use cases, we derive the following requirements for a software energy storage management system:

Transparent: a client should not be able to tell whether a battery is physical, a portion of a battery, or made up of a pool of multiple batteries. The power ratings a battery reports, for example, should sustain over its entire energy capacity.

General: physical batteries come from many vendors: they have different APIs and physical interfaces. Some are controlled over RESTful cloud APIs, while others require direct wired connections. The system should support all of them and integrate them into a distributed energy storage system.

Flexible: the system must be able to support a wide range of use cases, ranging from communal pooling of resources to participation in large-scale energy markets. The system must support the different topologies and policies these use cases introduce.

Dependable: because consumer batteries are connected through unreliable home networks, the system must provide an API which behaves in a dependable and predictable way in the presence of disconnections.

Scalable: to be able to participate in energy markets, the system must be able to manage hundreds of batteries without adding significant latencies. It should be able to support topologies that are hundreds of batteries wide as well as topologies that are tens of batteries deep.

3 VIRTUALIZED ENERGY STORAGE

This section proposes *virtual batteries* for energy storage allocation and management. Virtual batteries decouple energy storage and power delivery from physical batteries. This section explores how a software system presents the power and energy values of virtual batteries to meet the requirements in Section 2.5.

3.1 Logical Batteries

The overarching concept in programmable energy storage is a logical battery. A logical battery can be either physical or virtual: it is a transparent abstraction for both types of energy storage. Logical batteries report two values: their power in watts (W) and their energy capacity in watt·hours (Wh). The reported power is the battery's nominal power that it can maintain over its entire capacity.

3.2 Virtual Batteries

There are two types of virtual batteries: aggregate and partitioned. Aggregate batteries take multiple logical batteries and merge them into a single, larger one. Partitioned batteries take a single logical battery and split it into multiple, isolated, smaller ones. Isolation is critical for a partition to behave as a separate, physical battery. Because both aggregates and partitions are logical batteries, they can compose into large and complex topologies. Topology cycles are prohibited by enforcing a temporal order on battery creation. An aggregate, once created, cannot change its constituent batteries, so forming a cycle would mean that a child is older than its parent. In these complex topologies, the creation of virtual batteries does not strictly increase or decrease battery aging due to the nonlinear nature of cell degradation.

Battery	Power	Capacity	Max. C-rate
eco 7.5	3.6kW	7.5kWh	0.48
eco 10	7.0kW	10.0kWh	0.70
Aggregate	8.4kW	17.5kWh	0.48

Table 1: When constituent batteries begin with the same state of charge, the aggregate battery reports 8.4kW (17.5kWh · 0.48/h) as its maximum power because the lowest C-rate of its constituent batteries is 0.48. When discharging at maximum power, eco 7.5 and eco 10 will provide 3.6kW and 4.8kW, both operating at a C-rate of 0.48.

The primary question that arises in virtual batteries is what power and energy values they report. The next two subsections answer this question and propose algorithms and policies for aggregate and partitioned batteries.

3.3 Aggregate Batteries

An aggregate battery combines multiple logical batteries into a single larger battery. A strawman solution to reporting power and energy is to sum the values of the constituent batteries. While this holds for energy, reporting the simple sum of power, however, violates the property that an aggregate battery behaves like a physical battery, delivering its reported power over its entire capacity. For example, consider a sonnen eco 7.5 and eco 10, which provide 3.6kW and 7.0kW power, respectively. When these two batteries are aggregated, they can provide 10.6kW maximum power. However, if the aggregate delivers 10.6kW, the eco 10 drains in 85 minutes. 2.4kWh still remains in the eco 7.5 and power drops to 3.6kW: the advertised power is not valid for the entire capacity.

The key insight behind aggregate batteries is that maintaining nominal power over capacity means that constituent batteries must fractionally drain at the same rate. To explain how this works, we first assume batteries have equal states of charge (%).

Energy storage systems have a metric, *C-rate*, which defines the fraction of the battery that could be charged or discharged in an hour at maximum power. For example, a battery that discharges from 100% to 0% in 40 minutes has a C-rate of 1.5.

The energy of an aggregate battery is given by the sum of its constituent batteries. The maximum C-rate of an aggregate battery is the lowest maximum C-rate of its constituent batteries. An aggregate battery calculates its nominal power as its capacity times its maximum C-rate. Table 1 shows an example of how rate-limiting the aggregate battery to its slowest constituent ensures that it can deliver the reported power over its entire capacity.

Computing power from the lowest C-rate means that aggregate batteries which need to maximize power output should be composed of batteries with similar C-rates. This is similar to how in RAID arrays throughput can be limited by the slowest disk. Among the leading home energy storage systems today, the maximum C-rate values vary from 0.37 to 0.80 [14, 17, 24, 27–29, 31]. While these C-rates are all below 1, a large virtual battery with a C-rate of 0.37 is useful as it can fully discharge during the worst 3 hours of the 6-hour ramp/peak period that often occurs daily from 3-9PM.

3.3.1 Variable States of Charge. The example in Table 1 assumes that the two batteries have the same state of charge (SOC), or fractions of charge. If constituent batteries have different SOCs, the aggregate battery may have to report a lower power since otherwise some batteries might drain first. Table 2 shows an example.

We introduce a novel extension of C-rate for aggregate batteries to handle state of charge, called *effective C-rate*. The effective C-rate of a battery reports how long it will take to drain or fill the battery. Effective C-rate is defined as nominal power divided by *current* charge. The effective discharge C-rate of a battery is equal to its discharge C-rate at 100% SOC, while the same is true for effective charge C-rate at 0% SOC.

The effective C-rate of an aggregate battery is the minimum of the effective C-rates of its constituent batteries, as shown in Table 2. The *effective power* of an aggregate is its current capacity times its effective C-rate. When constituent batteries have equal SOCs, effective power is equal to nominal power.

Battery	Power	Capacity	Stored	*eff.* C-rate
eco 7.5	3.600kW	7.5kWh	0.75kWh	4.8
eco 10	7.000kW	10.0kWh	10.00kWh	0.7
Aggregate	7.525kW	17.5kWh	10.75kWh	0.7

Table 2: When constituent batteries begin with different SOC values, aggregate batteries calculate power using effective C-rate. Given the new "Stored" column values, if the aggregate tries to deliver 8.4kW (from Table 1) the eco 7.5 will drain first. Thus, the aggregate power is now limited by the eco 10's C-rate of 0.7; the effective C-rate then calculates the aggregate power as 7.525kW (0.7/h · 10.75kWh).

3.4 Partitioned Batteries

A partitioned battery takes a single logical battery and splits it into multiple smaller, independent batteries. The sum of the partition powers must be less than or equal to the source battery power, and the sum of the partition energies must be less than or equal to the source battery energy. Power and energy can be allocated differently, however. For example, one can partition a battery into a small, high power battery and a large, low-power one; therefore, partitions can have a higher maximum C-rate than their source. Table 3 shows two example partitionings of the aggregate battery in Table 1.

Battery	Power	Capacity	Max. C-rate
Aggregate	8.4kW	17.5kWh	0.48
Partition A1	6.3kW	13.1kWh	0.48
Partition A2	2.1kW	4.4kWh	0.48
Partition B1	2.4kW	11.5kWh	0.21
Partition B2	6.0kW	6.0kWh	1.00

Table 3: Two different partitioning options (A and B) of the aggregate battery in Table 1. A splits both energy and power 3:1 between the two partitions, which have the same C-rate as the aggregate. B constructs a large, low-power battery B1 (C-rate of 0.21) and a small, high-power B2 (C-rate of 1.0).

Two questions arise in partitioned batteries. First, what happens to partitions when the source battery changes from its expected values? Second, what happens if one partition receives a request to charge and another receives a request to discharge? We address these questions in the next two subsections.

3.4.1 Policies. Networked batteries can become unavailable due to network failures. If one of the constituent batteries becomes unavailable, the aggregate battery must report correspondingly reduced power and energy. When an administrator creates a partitioned battery, they define the partitions and set a partitioning policy. The partitioning policy defines what happens when a source battery's reported values differ from the expected ones. There are three policies, motivated by the use cases in Section 2: proportional, tranched, and reserved.

Proportional: Changes to the source battery are split proportionally across the partitions. For example, if the aggregate battery in Table 3 suddenly dropped in power to 6kW, then partitions A1 and A2 would report powers of 4.5kW and 1.5kW, respectively, while B1 would report 1.7kW and B2 would report 4.3kW. The homeowner's association use case in Figure 2 uses this policy: changes are shared evenly across owners.

Reserved: In the reserved policy, partitions are ordered from high to low. If the reported values are lower or higher than the expected values, these changes are first applied to the lowest partition, then the next lowest, until finally only the highest tranche remains. The reserved policy is used when batteries are being paid for time shifting/storage. For example, the homeowner uses this policy in Figure 4 to provide a reliable energy store to the regional aggregator, who might want to both charge and discharge.

Tranched: In the tranched policy, partitions are ordered from high to low, just as in reserved. If the reported values are lower than the expected values, they are taken from the lowest tranche first, just as in reserved. If the reported values are higher than the expected values, however, they are first added to the highest tranche, then the next highest, with the limit that a partition cannot go over its nominal (maximum) value. The partitions sold to companies in Figure 5 are an example of the tranched policy, where batteries are paid for energy delivery: an office building is happy to have some extra energy, and lower-cost tranches are less reliable.

Resetting a partitioned battery's expected values to its currently observed ones is an explicit operation, as it may involve financial exchanges (e.g., someone suddenly lost 2MWh of energy). The timing of this operation is dependent on the contract and so is outside the scope of this paper.

3.4.2 Billing and Virtual Flows. The final design question for virtual batteries is what happens when some partitions discharge while others charge. The aggregate behavior is clear: the source battery should discharge or charge based on the sum of the requests. Suppose the owner in Figure 4 decides to charge their battery B at 2kW because solar power is available while the regional aggregator decides to discharge battery A at 3kW. While the source battery discharges at 1kW, the amount of energy allocated to the partitions follows their requests: battery B charges at 2kW while battery A drains at 3kW. There will be a "virtual" flow from A into B which must be accounted for.

Virtual flows can allow bad actors to break promises and be paid for doing nothing. For example, suppose a person enters a contract to discharge a partition P1 at 10kW for 2 hours, from 5-7PM. At 5PM, the person tells P1 to discharge and the virtual battery records this for the utility. However, the person also tells P2 to charge at 10kW, such that the source battery does not discharge. An energy meter cannot distinguish this virtual flow from P1, thus reducing load behind the meter by 10kW.

Virtual batteries solve this problem by requiring batteries behind meters that participate in energy markets to register with the billing entity (e.g., utility). In the above example, the utility sees no net power at the meter, but from P1's discharge it can calculate that there is a 10kW load and bill for it. This is a requirement today for energy systems that discharge into the grid; however, discharging distributed virtual batteries may affect someone else's metering. The billing system therefore needs to add these discharges back to their physical meter and subtract them from the meter of the controlling customer. A customer may still create virtual flows as a form of arbitrage: if the cost of power is lower than the amount they were paid to discharge, they can purchase and resell power at a small profit. They are only able to do this, however, when they own and reserve energy resources.

4 BATTERY ABSTRACTION LAYER (BAL)

This section describes the Battery Abstraction Layer (BAL), a programming API for energy storage. BAL is designed to present the abstraction of a logical battery in a fashion that handles network behavior and latency. Since current APIs are imperative, battery controllers execute commands when they receive them. For example, using IEC 681507-420 [20] to control a battery, software can issue a command to set the target power output. Starting and stopping power output can be set on a delay, but this behaves as if the command is issued when the delay ends.

Combining an imperative interface with the realities of power electronics introduces delays and inaccuracies, as shown in Figure 6. This figure shows what happens when a sonnen eco 10 battery is sent a command through a RESTful cloud API to charge at 2.5kW. The battery's inverter firmware introduces a 30 second delay before the battery starts charging. When the battery is discharged, the observed delay is even longer, 50 seconds, and when discharging begins it slowly ramps up over more than 10 seconds.

These imperative APIs have two major drawbacks. First, as they were designed for reliable grid control networks, they require unbroken connectivity. Commands, once issued, continue until a new command is received. discharging until told to do otherwise. Second, they do not consider the temporal behavior of battery firmware. Figure 6 shows that a battery can wait tens of seconds before responding to a command. Firmware often has good reasons for this delay; for example, sensing circuits to assess safety (this explains why discharging ramps up slowly over 10 seconds), but client operations should not need to factor in this delay.

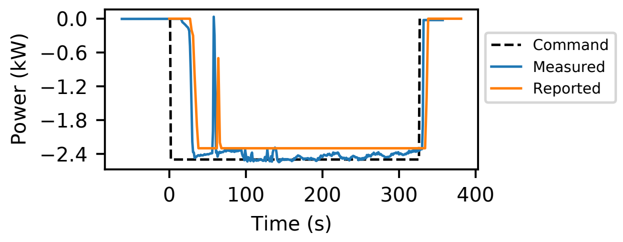

Figure 6: Charging and discharging a commercial energy storage system. The hardware can introduce significant (tens of seconds) delays. This motivates the need for a declarative API that can handle and adjust for these details.

In the current implementation of BAL, there is a hardware requirement that the battery power electronics contain a power output setpoint controller, whether inverter-based control or otherwise. Furthermore, the battery API must be able to perform the actions described in the following subsections: querying the status of the battery and controlling power output. While the former is standard on most BMSes and APIs, the latter requirement may not be found on all commercial systems.

Finally, commercial batteries have a variety of BMSes and APIs. We propose a uniform yet simple abstraction layer for batteries, called the *Battery Abstraction Layer* (BAL), with a lease-based API.

4.1 Querying

BAL's `get_status` function fetches and returns the status of a logical battery. It returns the maximum charge, present charge, maximum discharge current, maximum charge current, and the present current flow (charge is negative, discharge is positive). All values are returned as one atomic sample regardless of the manufacturer of the underlying battery; while this is a simple API, it differs from existing ones, which return values in separate messages or RESTful operations. This meets the generality requirement.

For BMSes connected through the network or other slow responding protocols, fetching the battery status can involve a significant delay. To mask this delay, `get_status` implements a caching mechanism. Each BAL instance has an associated maximum staleness. Before cached data expires, BAL pre-fetches data from the underlying battery so that it always has more recent data than the maximum staleness. A call to `get_status` returns, in addition to battery data, the time that data battery was sampled.

If a caller requires fresher data than what BAL may have, it can request `get_status` with a maximum staleness. If the cached data is older, BAL fetches the battery status from the underlying battery. Because this fetch can introduce a blocking delay, BAL instances should be configured to have a maximum staleness close to what callers require. Setting the maximum staleness to 0 means each call to `get_status` calls down to the leaves of the battery hierarchy, fetching data and processing it through partitions and aggregates. Specifying a maximum staleness ensures that virtual batteries made of deep and complex topologies can accurately maintain their battery status. Providing flexible staleness values and reporting timestamps meets the flexibility and scalability requirements.

When virtual batteries form a hierarchy, each node of the hierarchy has its own maximum and current staleness. The fetch time of each partition is equal to the fetch time of the source battery. The fetch time of each aggregate is equal to the oldest fetch time of its constituent batteries.

4.2 Control

The function `schedule_set_current` controls how a battery charges or discharges. It takes three parameters:

```
schedule_set_current(current, start, stop)
```

The `current` parameter is in Amps and specifies whether to charge (negative) or discharge (positive). The `start` and `stop` parameters indicate when the operation starts and completes.

The start and stop parameters allow control systems to schedule charging and discharging such that the operations will execute correctly even if there is a network disconnection. Furthermore, this API allows a lease-like approach to a battery resource. A client can extend an existing charge or discharge into the future by calling `schedule_set_current` with a later stop time. There can only be one charge/discharge lease active at any time, and newer operations replace older ones. When a call for a new lease arrives, the system truncates or removes any leases it overlaps with. Providing leases to charge/discharge the battery meets the dependability requirement.

As an example, if the user asks the battery to discharge at 1kW from 6:00PM to 7:00PM, and then asks the battery to discharge at 2kW from 6:30PM to 6:45PM, this second call will split the first lease into two separate ones (6:00-6:30 and 6:45-7:00). The battery will discharge at 1kW from 6:00PM to 6:30PM, at 2kW from 6:30PM to 6:45PM, and at 1kW from 6:45PM to 7:00PM.

When a client makes a call on an aggregate battery, BAL splits the operation across the constituent batteries so they have the same effective C-rate. When a client makes a call on a partition, BAL merges the lease in with those of other partitions to compute the leases for the source battery. A client does not see this underlying activity and instead observes their battery behaving as a physical battery behaves. Thus, BOS meets the transparency requirement.

A client is not assured that a requested lease will execute until it receives an explicit acceptance notification. For example, if a client requests a lease on an aggregate battery but one of the constituent batteries cannot be reached, the aggregate may delay accepting it until it can contact the constituent. If the constituent is disconnected long enough, the BAL implementation might consider it failed, reduce the battery values accordingly, and send updates to clients indicating the new values.

5 IMPLEMENTATION

This section describes our prototype implementation of energy storage virtualization and the battery abstraction layer in the Battery Operating System. The Battery Operating System (BOS) is a new software platform we have developed for managing and controlling battery-based energy storage. Battery *drivers* implement the BAL and map it to the underlying operations of the corresponding battery. For example, a driver for a battery managed by another BOS instance maps BAL calls to RPC calls over TLS, a driver for a UART-based battery management system maps BAL calls to a BMS's serial protocol, and a driver for a commercial storage unit controlled through the cloud maps BAL to RESTful API calls.

BOS is implemented in Linux. There are five supported drivers: a JBD BMS [21] for DC batteries over a serial port, a JBD BMS for DC batteries over Bluetooth, sonnen batteries over RESTful HTTP calls, remote BOS batteries with RPC, and IEC 61850 ZBAT/ZINV devices. Supporting this variety of battery interfaces shows that BAL meets the generality requirement. The JBD BMSes export a binary protocol over a UART and Bluetooth for reading and writing firmware state. The protocol supports reading the entire state of the battery in a single command. We use the JBD drivers in a 12V DC testbed; we omit these results for lack of space.

The sonnen driver makes BAL operations as sonnen web API calls. It places the battery in manual mode and requests dynamic values from it. The sonnen API does not provide maximum discharge and charge current values: the driver fills these in based on the type of battery. Telling a battery to charge or discharge involves setting its `setpoint` resource. The IEC 61850 driver is built on top of `libiec61850`, an open-source implementation of the IEC MMS/GOOSE protocols over TCP/TLS [18].

6 EVALUATION

In this section, we evaluate BAL according to the requirements outlined in 2.5: transparency, generality, flexibility, dependability, and scalability. We first describe the experimental methodology, then we evaluate BAL with a series of topologies, each with a different configuration of sonnen battery storage units and virtual batteries. With each topology, we show plots confirming that the output of the sonnen units correctly tracks the BAL command. We examine how virtualized batteries can correct for delays in batteries and transparently replace physical batteries.

6.1 Methodology

This section describes the experimental setup used to evaluate battery virtualization and BAL, consisting of sonnen systems connected to the grid in a lab setting and residential homes as well as an emulation setup to measure scalability.

The testbed has sonnen ecoLinx 12 units installed in critical loads subpanels in multiple suburban homes. All of the testbed homes have a solar system connected to the subpanel. The testbed also has a single sonnen eco 10 in an energy lab which is connected to an entire simulated home. The devices downstream of the battery include a 5.6kW solar system, an electric vehicle level 2 charging station, and multiple outlets for appliance loads.

All sonnen units have a Li-Fe-PO_4 chemistry, a 100% depth of discharge (DoD), and a nominal power rating of 7kW in grid-tied mode. Internally, each system is comprised of multiple 2kWh modules. In both systems, a BAL driver uses the sonnen API.

The sonnen API reports charge and discharge power as well as state of charge. To have a separate, ground truth measurement, all of the sonnen installations have an eGauge Systems EG4115 as an external energy metering system. The eGauge uses clamp-on current transformers [12] to measure current: these are small inductive coils that sense the alternating current. Figure 7 shows the lab and home setups.

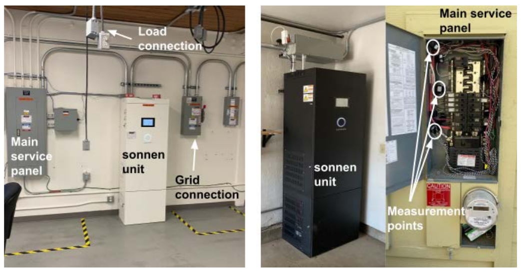

Figure 7: Photos of installed sonnen units. Left is the lab setup while right shows the home setups.

Our testbed has too few batteries to evaluate BAL's scalability, so we emulated software batteries with added latency. To provide realistic battery latency values, we used the ZMap tool [30] to measure latency distributions of five networked batteries in the Bay Area. Each emulated battery was randomly assigned one of the five distributions from which to take random latency samples.

6.2 Full Topology

Figure 8 demonstrates the flexibility of BAL by encompassing all four use cases in Section 2. The HOA in Section 2.1 could use proportional partitions to evenly divide the physical battery among homeowners. A homeowner in Section 2.3 could use reserved partitions to partition their battery, keeping the highest tranche to prevent SOC deviations from affecting a personal use partition. The regional aggregator in Section 2.4 could use tranched partitions, shielding high tranche large buildings from changes in the underlying physical batteries.

With the transparency provided by the system, the user controlling a logical battery at each node of the topology cannot deduce whether the logical battery is physical or virtual. Consequently, BAL allows HOAs to provide renewable energy to homeowners while simultaneously permitting aggregators to aggregate kW-scale energy resources to MW-scales that can be used by companies to meet their 24/7 carbon-free initiatives.

6.3 Policies

To examine how the partitioning policies described in Section 3.4.1 allow virtual batteries to meet contractual obligations, we examine a subset of the full topology, shown in Figure 9.

The advertised SOC values change when battery A2 has a sudden change in SOC. This could happen, for example, if the owner of A2 inadvertently changes the battery named A2, or if A2 fails. We begin with the O1 and O2 partitions each at an expected value of 60% SOC, and we look at how those SOC values change when A2's SOC jumps to 80% or drops to 0%.

Table 4 displays the observed behavior. In the proportional policy, the change is split across O1 and O2. In the tranched policy, a drop in energy is taken from the lower tranche, O2, while an increase is given to O1 until it spills over to O2. In the reserved policy, a drop in energy is taken from O2 and an increase fills O2 first.

Although not shown in the paper, to verify that charge/discharge leases make BAL systems dependable in the presence of system failures, we configured a logical battery partition to discharge and then forcibly killed the battery client. After the lease expired, the battery stopped discharging.

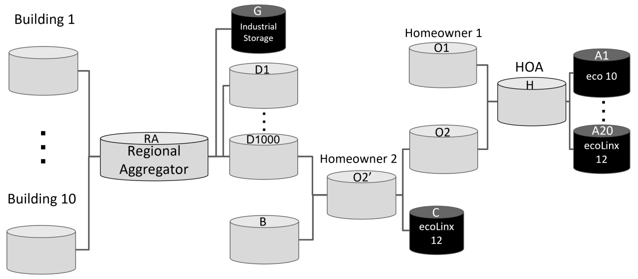

Figure 8: The full topology of the use cases in Section 2. Batteries purchased by an HOA become part of an energy store used by large-scale consumers such as office buildings through several levels of aggregation and partitioning.

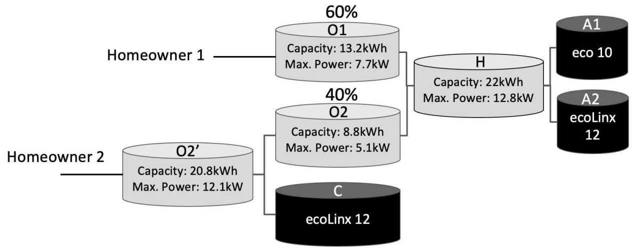

Figure 9: Multilayer battery topology diagram extending the topology in Figure 2. Partition O2 is made available to an energy aggregator, who aggregates it with a ecoLinx 12.

6.4 Charging and Discharging

The flexibility of BAL supports differing topologies depending on users' needs. In this section, we show how commands to partitions are split across constituent batteries: the partitions are decoupled from the underlying physical resources. We examine subsets of the topology in Figure 9 to show that this conservation of power holds. Figure 10 displays the power values of physical batteries A1 and A2 after commands are sent to virtual batteries H and O1. Figure 11 shows how charge/discharge commands from logical batteries propagate through the topology in Figure 9. The plot confirms that the physical batteries (A1, A2, and C) only respond to the commands from their respective child batteries.

This topology shows a particularly salient use case. A homeowner could purchase multiple batteries from different manufacturers, pool them into a single larger battery, set aside some of the capacity for reducing their power bill, and then sell the rest to someone else for aggregation. BAL's generality and breadth of battery drivers ensures that an individual is not tied to a specific manufacturer brand when multiple batteries are pooled.

6.5 Scalability

The use case described in Section 2.4 describes a scenario in which an aggregator aggregates hundreds of logical batteries together

Policy	Battery	A2 ↓ 0.0	A2 ↑ .80
Proportional	O1	0.49	0.89
	O2	0.49	0.89
Tranched	O1	0.60	1.00
	O2	0.32	0.73
Reserved	O1	0.60	0.82
	O2	0.32	1.00

Table 4: This table shows SOC values of partition O1 and O2 (each starting at 60% SOC) after a drop and increase in battery A2 SOC according to the specific policies.

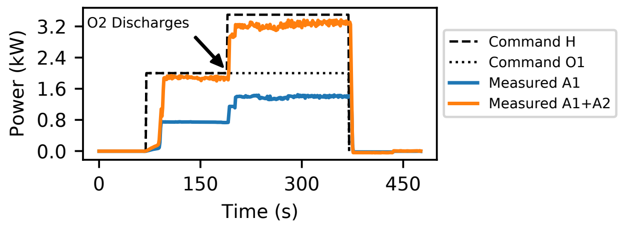

Figure 10: Partition O1 discharges at 2kW for 5 minutes. After 2 minutes, partition O2 discharges at 1.5kW for 3 minutes. The combined measured outputs of the physical batteries corresponds to the aggregated virtual battery's command, and both physical batteries respond to the change in discharge power from the aggregate command.

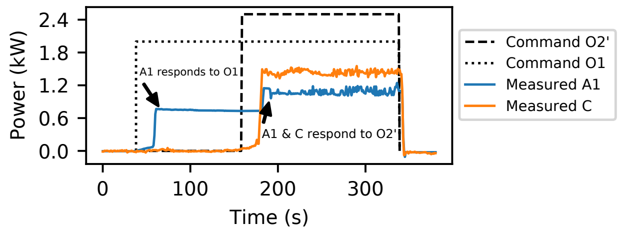

Figure 11: Measured power of A1 and C are shown with the commands from O1 and O2' overlaid. While only a fraction of A1 contributes to powering O2, both A1 and C respond to the command from O2'. C only discharges when O2' sends a command.

before selling partitions to companies. Therefore, a BOS system must be able to scalably aggregate. For aggregations of six hundred or fewer batteries, the network latency observed is more than half of the total latency. At approximately a width of six hundred batteries, the BOS processing latency starts to contribute to more than half of the total latency. However, the processing and network delays are small in comparison to the time scales of the battery cloud APIs shown in Figure 6. The system latency follows an approximately linear trend as the topology depth increased.

7 DISCUSSION

Virtualization is a common technique in computing systems: operating systems virtualize disks, cores, memory, and the network. Energy storage systems differ from these other resources in that they encompass both energy as a quantity (like storage or memory) and power as a rate (like networks). Furthermore, these two are linked: changes in the state of charge can alter effective power. Virtualized energy storage therefore requires new policies and algorithms. We first examine prior work in virtualized energy and then highlight the contributions of BAL.

7.1 Related Work

The term "virtual battery" often refers to shifting power demand and creating "virtual" storage without any physical batteries [1]. Instead, we propose virtual batteries as the aggregation and partitioning of true physical batteries. Various related works discuss engineering solutions to battery aggregation while others explore the concepts of sharing virtualized energy resources. For example, software defined batteries [3] combine heterogeneous technologies to prioritize battery performance while Han et al. examine how firmware can improve performance [19]. RAIBA examines how a flexible interconnect can dynamically combine batteries behind an inverter to provide consistent power [8]; BAL can use batteries with this technology, but it manages AC, not DC power.

AutoShare and vSolar examine how a centralized residential solar and storage system can share energy between users [22, 23]. AutoShare requires all users to follow a battery charging policy based on their loads. Virtual batteries, in contrast, have no such restrictions, allowing clients to implement their own charging and discharging policies. The concept of a virtual battery can be extended to a virtual power plant (VPP), such as Toshiba's VPP [32], which creates an illusion of storage by reducing loads through demand response and other interventions.

7.2 Contributions and Future Work

The evaluations in Section 6 show how the Battery Abstraction Layer meets the challenge of effective power dependency on state of charge. BAL allows software systems to manage energy storage through virtualization, enabling new applications and use cases, such as purchasing multiple batteries, selling use of a fraction of them to a third party, and time shifting storage for 24/7 carbon-free power. Battery virtualization meets the five requirements set out in Section 2: it provides a transparent API to physical and virtual batteries, is general enough to abstract a wide range of physical batteries, is flexible enough for a wide range of applications, operates dependably in the presence of network disconnections, and can scale to both wide and deep topologies.

The Battery Abstraction Layer implementations in this paper look solely at batteries as pooled resources, which illuminates two design drawbacks. First, BAL focuses on guaranteeing a charge or discharge current over a system's entire charge, which prioritizes reliability over performance. Exploring how techniques for heterogeneous systems such as software defined batteries [3] and RAIBA [8] could improve aggregate performance is an area of future work. Second, balancing effective C-rates in multiple batteries leads to an imbalance in battery utilization. Currently, BAL does not contain a degradation balancing mechanism, which is vital to reduce battery waste; this is also an area of future improvement.

Lastly, future work can incorporate ideas from other types of energy systems. Helios [4] provides control of solar arrays by controlling the power output of of the system, an idea that can be used with virtual batteries. vPeak [5] describes how a novel control algorithm using machine learning can be applied to volunteer energy resources. A future area of work is to apply this algorithm to logical batteries after they have been partitioned or aggregated using BAL.

8 CONCLUSION

For the past 100 years, the electric grid has been a highly centralized and centrally managed system. Distributed solar power and batteries have shifted this paradigm. Consumer battery storage systems have the potential to accelerate the transition to carbon-free power by time-shifting renewable energy to match energy demand. Today, however, there are no abstractions or software systems to manage these at scale. This paper proposes virtual batteries and the Battery Abstraction Layer (BAL) as abstractions for distributed energy storage management. Virtual batteries support flexible topologies through aggregation and partitioning, enabling novel use cases. A prototype BAL implementation controls a variety of battery systems and testbed experiments on consumer storage units in homes, demonstrating use of the abstractions. Virtual batteries enable the control and management of distributed battery resources, supporting the transition to carbon-free power for individuals, office buildings and even large-scale power consumers.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Electricity under Award Number DE-OE0000919.

REFERENCES

[1] Anup Agarwal, Jinghan Sun, Shadi Noghabi, Srinivasan Iyengar, Anirudh Badam, Ranveer Chandra, Srinivasan Seshan, and Shivkumar Kalyanaraman. Redesigning data centers for renewable energy. In *Proceedings of the Twentieth ACM Workshop on Hot Topics in Networks*, HotNets '21, page 45–52, New York, NY, USA, 2021. Association for Computing Machinery.
[2] SunSpec Alliance. Sunspec specifications. https://sunspec.org/specifications/, 2022. Accessed: 2022-04-19.
[3] Anirudh Badam, Ranveer Chandra, Jon Dutra, Anthony Ferrese, Steve Hodges, Pan Hu, Julia Meinershagen, Thomas Moscibroda, Bodhi Priyantha, and Evangelia Skiani. Software defined batteries. In *Proceedings of the 25th Symposium on Operating Systems Principles*, pages 215–229, 2015.
[4] Noman Bashir, David Irwin, and Prashant Shenoy. Helios: A programmable software-defined solar module. In *Proceedings of the 5th Conference on Systems for Built Environments*, BuildSys '18, page 63–72, New York, NY, USA, 2018. Association for Computing Machinery.
[5] Phuthipong Bovornkeeratiroj, John Wamburu, David Irwin, and Prashant Shenoy. Vpeak: Exploiting volunteer energy resources for flexible peak shaving. In *Proceedings of the 8th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*, BuildSys '21, page 121–130, New York, NY, USA, 2021. Association for Computing Machinery.
[6] California Independent System Operator (CAISO). California iso hits all-time peak of more than 97% renewables. https://www.caiso.com/Documents/California-ISO-Hits-All-Time-Peak-of-More-Than-97-Percent-Renewables.pdf, 2022. Accessed: 2022-04-19.
[7] California ISO. What the duck curve tells us about managing a green grid. https://www.caiso.com/documents/flexibleresourceshelprenewables_fastfacts.pdf, 2016. Accessed: 2022-04-19.
[8] Tzi-cker Chiueh, Mao-Cheng Huang, Kai-Cheung Juang, Shih-Hao Liang, and Welkin Ling. Virtualizing energy storage management using {RAIBA}. In *2018 USENIX Annual Technical Conference (USENIX ATC 18)*, pages 187–198, 2018.
[9] International Electrotechnical Commission. Iec 61850:2022 ser series, communication networks and systems for power utility automation - all parts. https://webstore.iec.ch/publication/6028, 2022. Accessed: 2022-04-19.
[10] Paul Denholm, Erik Ela, Brendan Kirby, and Michael Milligan. Role of energy storage with renewable electricity generation. Technical Report NREL/TP-6A2-47187, National Renewable Energy Lab. (NREL), 1 2010.
[11] Paul Denholm, Matthew O'Connell, Gregory Brinkman, and Jennie Jorgenson. Overgeneration from solar energy in california. a field guide to the duck chart. Technical Report NREL/TP-6A20-65023, National Renewable Energy Lab. (NREL), 11 2015.
[12] eGauge Systems. eGauge Core Residential Specifications. https://www.egauge.net/media/support/docs/eg4115-datasheet.pdf. Accessed: 2022-04-19.
[13] Electrek. Tesla has doubled number of powerwalls installed to 200,000 over this opast year. https://electrek.co/2021/05/26/tesla-doubled-powerwalls-installed-over-last-year, 2021. Accessed: 2022-04-19.
[14] Enphase. Enphase Encharge 3 Datasheet. https://support.enphase.com/s/article/Enphase-Encharge-3-Datasheet, 2021. Accessed: 2022-04-19.
[15] Federal Energy Regulatory Commission . Docket No. RM18-9-000; Order No. 2222. https://www.ferc.gov/sites/default/files/2020-09/E-1_0.pdf, 09 2020. Accessed: 2022-04-19.
[16] Federal Energy Regulatory Commission . Docket No. RM18-9-002; Order No. 2222-A. https://cms.ferc.gov/media/e-1-rm18-9-002, 03 2021. Accessed: 2022-04-19.
[17] Generac. Generac PWRcell. https://www.generac.com/generaccorporate/media/library/content/all-products/clean-energy/pwrcell_battery_specsheet_2.pdf?ext=.pdf, 2020. Accessed: 2022-04-19.
[18] MZ Automation GmbH. libiec61850, the open-source library for the iec 61850 protocols. https://github.com/mz-automation/libiec61850, 2022. Accessed: 2022-04-19.
[19] Weiji Han, Torsten Wik, Anton Kersten, Guangzhong Dong, and Changfu Zou. Next-generation battery management systems: dynamic reconfiguration. *IEEE Industrial Electronics Magazine*, 14(4):20–31, 2020.
[20] International Electrotechnical Commission. *Communication networks and systems for power utility automation - Part 7-420: Basic communication structure - Distributed energy resources and distribution automation logical nodes*, 2021.
[21] Jiabaida. JBD-SP04S020. http://en.jiabaida.com/home-pp2-info-id-47-catId-26.html. Accessed: 2022-04-19.
[22] Stephen Lee, Prashant Shenoy, Krithi Ramamritham, and David Irwin. Vsolar: Virtualizing community solar and storage for energy sharing. In *Proceedings of the Ninth International Conference on Future Energy Systems*, e-Energy '18, page 178–182, New York, NY, USA, 2018. Association for Computing Machinery.
[23] Stephen Lee, Prashant Shenoy, Krithi Ramamritham, and David Irwin. Autoshare: Virtual community solar and storage for energy sharing. *Energy Informatics*, 4(1):1–24, 2021.
[24] LG Chem. The LG ESS. https://www.lg.com/us/business/download/resources/BT00002151/180830_LG_ESS_Datasheet.pdf. Accessed: 2022-04-19.
[25] Energy Storage News. Utility-scale energy storage in us tripled in 2021 while growth of other clean energy stalled. https://www.energy-storage.news/utility-scale-energy-storage-in-us-tripled-in-2021-while-growth-of-other-clean-energy-stalled/, 2022. Accessed: 2022-04-19.
[26] Government of the United Kingdom. London pioneers first 'virtual power station'. https://www.gov.uk/government/news/london-pioneers-first-virtual-power-station, 2021. Accessed: 2022-04-19.
[27] Panasonic. EVAC-105 / EVDC-105 Energy Storage. https://ftp.panasonic.com/batterystorage/datasheet/evervolt_datasheet.pdf, 2021. Accessed: 2022-04-19.
[28] sonnen. Tech specs - sonnen eco gen 3.1. https://cdn-sonnen-media.s3.amazonaws.com/0024e065-f9f6-4c5d-bd80-b654ff2567cf-en-download, 2021. Accessed: 2022-04-19.
[29] sonnen. Tech specs - sonnen ecolinx 1.5. https://cdn-sonnen-media.s3.amazonaws.com/d0c1c592-edd9-4e2a-96f6-079e0e164a6f-en-download, 2021. Accessed: 2022-04-19.
[30] The ZMap Team. The zmap project. https://zmap.io, 2020. Accessed: 2022-04-19.
[31] Tesla. Powerwall. https://www.tesla.com/sites/default/files/pdfs/powerwall/Powerwall%202_AC_Datasheet_en_northamerica.pdf, 06 2019. Accessed: 2022-04-19.
[32] Toshiba. VPP (Virtual Power Plant). https://www.global.toshiba/ww/products-solutions/renewable-energy/products-technical-services/vpp.html, 2022. Accessed: 2022-4-17.
[33] Alexandra Von Meier. *Electric Power Systems: A Conceptual Introduction*. Wiley-IEEE Press, 09 2013.