DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof. Reference herein to any social initiative (including but not limited to Diversity, Equity, and Inclusion (DEI); Community Benefits Plans (CBP); Justice 40; etc.) is made by the Author independent of any current requirement by the United States Government and does not constitute or imply endorsement, recommendation, or support by the United States Government or any agency thereof.

Final Scientific/Technical Report

TrustDER: Trusted, Private and Scalable Coordination of Distributed Energy Resources

WORK PERFORMED UNDER AGREEMENT

Agreement DE-OE0000919

Stanford University

450 Jane Stanford Way

Stanford, CA 94305-2004

Award Period of Performance: 07/01/2020 **to** 06/30/2024

(No-cost Extension Period: 07/01/2023 - 06/30/2024**)**

Submitted: 07/30/2024

Revised: 09/15/2024

PRINCIPAL INVESTIGATOR

Professor Ram Rajagopal, ramr@stanford.edu

TEAM MEMBERS (co-PIs & students)

Professor Philip Levis, pal@cs.stanford.edu

Professor Abbas El Gamal, abbas@ee.stanford.edu

Mayank Malik, mmalik@slac.stanford.edu

Lily Buechler, Sonia Martin, Obi Nnorom, Evan Laufer, Thomas Navidi, Mark Chen, Oskar Triebe, Yancheng Ou, Anthony Deglaris, Chin-Woo Tan (Project Manager)

SPONSORING PROGRAM OFFICE

U. S. Department of Energy

Office of Electricity Delivery and Energy Reliability

Via the National Energy Technology Laboratory

TABLE OF CONTENTS

Acknowledgement

This material is based upon work supported by the United States Department of Energy, National Energy Technology Laboratory (NETL), under the Fiscal Year 2020 Funding Program, Agreement Number DE-OE0000919.

Disclaimer

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

BMS: Battery Management System
BAL: Battery Abstraction Layer
BOS: Battery Operating System
CSP: Cybersecurity Plan
DER: Distributed Energy Resource
DMP: Data Management Plan
DOE: Department of Energy
EV: Electric Vehicle
IDMS: Identity Management System
IOP: Interoperability Plan
IoT: Internet of Things
LLNL: Lawrence Livermore National Lab
PMP: Project Management Plan
SLAC: SLAC National Accelerator Laboratory
SOPO: Statement of Project Objective

I. Executive Summary

In this project, the Stanford and SLAC Teams have developed a Trusted, Private and Scalable platform for coordinating Coordination of Distributed Energy Resources (TrustDER). This is a layered system that ensures private, trusted and scalable coordination and monitoring of DERs. It accommodates a variety of resources, such as solar generation, gensets and loads, with a particular focus on battery systems-based resources, as they are a transformational technology experiencing fast growth in adoption by large critical facilities. The platform can be used as standalone or added to existing aggregation systems to enable trust, privacy and resilience.

TrustDER consists of layers that address each of the shortcomings of the existing state of the art. Each layer in the platform can operate independently but provides information to the layers above it to enable a novel form of overall coordination architecture. The project consists of several tasks, with each task dedicated to the design of each layer, except Task 1 which is Project Management. For Task 1, we have separately submitted a DMP, PMP, CSP and IOP documents. Task 4, led by the LLNL team, is on Trust and Mapping IoT for Data Reliability. It was determined that the team would not pursue the LLNL subcontract as the Stanford SPO and LLNL could not agree on some legal teams. The fund that was originally allocated to LLNL was re-allocated to the Stanford team for work in virtualization (Task 2.6) and privacy (Task 6.4). The risk to the project was low as it could be mitigated by using multiple testbeds.

***Task 2**: Resource Virtualization*

This Task defined a software abstraction layer for distributed energy resources (DERs). The goal of this abstraction was to simplify the implementation of algorithms utilizing cooperation of DERs resources in a variety of use cases. Work under this task focused on virtualization of batteries and battery-solar systems. We researched, designed, implemented, deployed and evaluated a new technique for managing distributed energy resources: virtualization. Virtualization has long been used in computer systems as a way to decouple software control of resources from their physical instances. At its core, virtualization is about two key ideas: aggregation and partitioning. Aggregation means making many different resources look like a single one. For example, when a computing system takes multiple disks and makes them act like one larger disk, or multiple monitors act like one larger monitor; this is aggregation. When it divides a single disk into multiple partitions or shares a single CPU across multiple programs; this is partitioning.

We focused on battery energy storage systems and battery-solar systems for virtualization. We designed a new software abstraction, called the Battery Abstraction Layer (BAL) for controlling batteries. We demonstrated that the BAL is general enough to control both small, contained DC battery systems as well as Internet-connected consumer batteries installed in homes. The key research question that arose is how aggregate and partitioned batteries behave: when a request to discharge comes into an aggregate of 4 different batteries, how is the command distributed across them?

Similarly, when different requests come into the 3 partitions of a large battery, how does the BAL process them to ensure correct operation?

We demonstrated that the Battery Abstraction Layer allows battery owners and users to easily create new applications of energy storage. In the example topology in Figure I.2.1, a homeowner's association aggregates two batteries (A1 and A2) into a single, larger battery H. It then rents unequal partitions of H (O1 and O2) to two homeowners. Homeowner 2, who rents O2, has their own home battery installation C. They aggregate their share O2 with C, to create a larger battery O2', for them to use. Battery virtualization allows homeowners to rent arbitrary shares of a pool of batteries, and also use these rented shares to supplement their own energy storage.

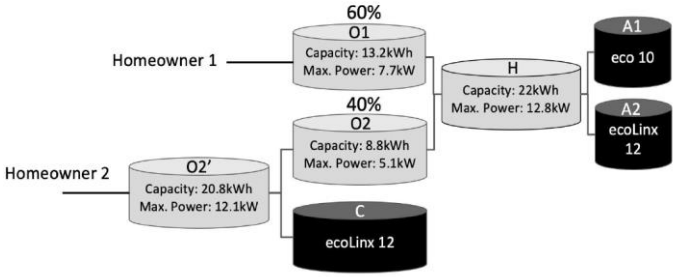

Figure I.2.1: Example topology of battery aggregation

We deployed BAL on a testbed consisting of several residential batteries as well as a lab battery setup. In one experiment, we used the virtual battery topology depicted above, demonstrating that requests to discharge O2' were distributed across O2 and C, and that requests from O2 and O1 were merged at H then distributed to A1 and A2.

We researched how to manage virtualized batteries when they are part of a battery-solar system with uncertain solar production and loads. In particular, we examined how jointly controlling a set of batteries among a set of users can lead to substantially greater cost savings than each individual using only their battery. Because these savings manifest as heavy users saving a lot by drawing on the batteries of light users, we explored hybrid schemes, in which each owner partitions their battery into a private part for their own use and a shared part for joint use. Each day, an owner can recalibrate their partitioning based on forecasts of load and solar generation. Our results show that the ability to partition virtualized batteries provides benefits in complex forecasting/control loops, in that it allows partitioning of resources for different objective functions (personal vs. joint optimization).

When evaluating the feasibility of TrustDER's mechanisms in IoT systems, we consider three metrics: computational requirements (in time and space), communication requirements, and system complexity. The first two metrics are quantitative: they involve concrete measures of properties such as computation time, RAM used, and bytes transmitted. The third metric, system complexity, is qualitative. It examines the additional complexity that incorporating the system will introduce. To evaluate whether

the complexity is feasible or acceptable, we rely on analogies in similar systems. For example, TrustDER's computational privacy requires having two non-colluding servers. To evaluate this cost, we consider use cases in which this would or would not be simple and compare with other applications that have a similar requirement.

Task 3*: Secure ID for Asset Authentication*

Identity Management Systems (IDMS) are a foundational infrastructure for interactions between entities (organizations, users, devices, and services). At a minimum, an IDMS must support the following features:

- Allow entities to authenticate
- Share authentication results with others (apps, services), and
- Maintain and protect a registry of credentials

With traditional IDMS, organization(s) store credential information for each entity, thereby creating a privileged system with disproportionate control, and a well-defined target for cyber-attacks. Centralized models for identity management face challenges due to the increasing regularity of data breaches that lead to monetary loss, compromised privacy, and reputational damage.

Currently there are several initiatives investigating alternative approaches to identity management that seek to improve the trustworthiness of the system while minimizing the risks from cyber-attacks. Secure ID is an alternative approach to identity management for grid assets.

Secure ID is blockchain-based a distributed identity management system allowing (1) identity provisioning, (2) authentication, (3) authorization, and (4) identity data sharing for IoT-enabled assets on the electricity grid.

In this project, the SLAC team focused on designing and testing *Keymaker*, a protocol for authenticating device identity managed by Secure ID.

Task 5*: Private and Safe Integration*

This task is focused on the design and evaluation of a DER cooperation scheme which allows for the aggregation of DERs without impacting network reliability. The approach is designed based on realistic assumptions regarding data availability, communication infrastructure limitations, and privacy.

To evaluate the impact of DERs on network reliability and the performance of the proposed approach, a comprehensive distribution grid simulation suite was developed. The proposed DER cooperation scheme was designed to reduce the number of grid upgrades needed to support the growing penetration of DERs. The cooperation scheme includes (i) a day ahead scheduler that calculates power injection bounds for each customer that ensures that voltage and transformer constraints are respected, and (ii)

local controllers for each customer which manage the local DERs. Results demonstrate that the algorithm can provide greater than a 10% reduction in the cost of transformer upgrades, satisfying the project metric goal for this task.

The cooperation capabilities were also extended to ramping, regulation, and black start applications. Results show a tracking error for ramping and regulation of less than 10%, which meets the goals for this task. Our analysis also investigated how stationary storage, EV chargers, and rooftop PV can assist the grid in a black start scenario and provide the grid with power during a blackout. The final portion of this task investigated testing the proposed system in a hardware-in-the-loop setup.

Task 6*: Scalable Distributed Privacy for Information and Energy Exchange*

This Task explored how virtualized batteries could be managed privately. Specifically, it examined the case in which a principal provides a partitioned battery to multiple clients. These clients each control their partitions, but wish to do so privately, such that the principal learns nothing more than their aggregate use (which it must learn, to control the battery). One complication is that the principal must be able to enforce constraints on client commands. For example, a principal needs to be able to keep a client within their energy (storage) and power (charge/discharge) limits. The principal must be able to do this without knowing the exact value a client sends, merely that it is valid, violating neither energy nor power constraints.

We researched and developed Weft, a novel cryptographic system to achieve these goals. Weft enforces constraints over power (rates) and energy (integrals) using the key observation that the additive homomorphism in prior work and a local state variable allow servers to compute and constrain secret integrals over time. To provide temporal secrecy, instead of sending commands to the device, clients send schedules of commands at fixed times. Weft can run in diverse deployment scenarios, including on resource constrained embedded devices. Weft can use three types of proofs which each minimize one of compute, memory, or network communication.

Some distributed energy resources are partitioned among many users. Sharing a communal resource in this way has many advantages: individuals can join and leave the system, resources can be put in good locations, and it allows people who cannot install resources in their home to use them. One problem that partitioned energy resources introduce is a lack of privacy. Because each user operates a share of the large resource, the controller of the resource (e.g., the battery storage provider) can see how each person is using it.

Adapting private aggregation to partitioned energy resources has three challenges. First, prior techniques have focused on single values (e.g., page load times), but distributed energy resources require verifying both an instantaneous value (power) and its integral (energy). In other words, a client request is valid only if its requested power is within allowed limits and the integral of all of its requests over time do not undercharge or overcharge its battery. Second, values are commands to an active

device, and timing of these commands can leak sensitive information about the client's usage of the resource. For example, if a client sends a request which immediately changes the aggregate power, then that request is trivially discoverable. Third, prior techniques focus on traditional client/server computing systems that have GHz of CPU and GB of RAM. Distributed energy resources, in contrast, often have embedded controllers, with MHz of CPU and kB-MB of RAM.

Weft is a secure system which addresses these three challenges. Weft enforces constraints over power (rates) and energy (integrals) using techniques from Task 6.2. Second, to provide temporal secrecy, instead of sending commands to the device, clients send schedules of commands at fixed times. Third, Weft can run in diverse deployment scenarios, including on resource constrained embedded devices. Towards this end, Weft can use three types of proofs: sorting proofs which minimizes compute, bit-splitting proofs which minimizes memory, and commitment proofs which minimize network communication. To control an energy resource for a day at 20s granularity, a client running commodity hardware needs 0.093s compute and 505 kB communication using the bit-splitting proofs, 0.027s compute and 322 kB communication using sorting proofs, and 17s compute and 17 kB communication using commitment proofs. Bit-splitting proofs reduce memory usage enough to run on memory constrained embedded devices. It takes an IoT client with 256 kB of memory using a CortexM microcontroller 4 minutes of computation time to privately control its share of an energy resource for a day at 20s granularity.

Task 7*: Use Cases*

While the other Tasks focus on designing the software and technology for trusted DER virtualization, the main goal of Task 7 was to ensure that this technology was applied in relevant situations and scenarios. Primarily, this means that virtualization needed to be employed in a manner that either improved flexibility, bolstered security or privacy, or decreased costs.

Identifying specific use cases first included surveying literature and current technology to find battery, load, and solar system configurations that could benefit from introducing virtualization. Next, we formed an industry advisory board committee to help us discover new potential use cases. This included collaborations with academic institutions as well as commercial companies such as VMware.

Ultimately, we found that virtualization used for cost minimization as well as for maintaining critical power supplies were the most important use cases. We developed specific system configurations based on these needs, and in our publications, we tested the use of virtualization on the use cases.

II. Objectives

Task 2*: Resource Virtualization*

Task 2 has three major objectives:

1. Design and implement the Battery Abstraction Layer, a software abstraction for virtualizing battery energy storage systems and battery-solar systems. Write a complete design document on the abstraction, its algorithms, and implementation considerations. Provide an open-source implementation of the Battery Abstraction Layer that demonstrates interoperability with multiple batteries.
2. Deploy the Battery Abstraction Layer on a real-world testbed to evaluate its effectiveness and operation. Demonstrate that the Battery Abstraction Layer enables new and interesting use cases of battery storage systems.
3. Extend battery virtualization to battery-solar systems, using control theory to balance and manage them in the context of unknown and varying solar charging. Explore how virtualization allows new uses of battery-solar systems, e.g., for neighborhoods to collaboratively manage EV charging load to not violate transformer limits and extend transformer lifetime.

Task 3*: Secure ID for Asset Authentication*

Utilities manage grid assets through mechanisms such as relays, supervisory control and data acquisition (SCADA), distributed control systems (DCS), meters and remote terminal units (RTUs). With new smart grid technologies, the grid is changed into an interconnected system of IoT devices communicating over a network using a variety of protocols, thereby exposing the grid to an entirely new class of cyber-attacks. Several of these attacks target device identity and compromise device integrity through clone attacks, false data injections attacks, and energy attacks. An important objective is therefore to design IDMS for grid assets that are resilient to such attacks. Since blockchains are well suited to ensuring consensus, transparency, and integrity of the transactions they store, they make a great candidate technology for developing IDMS. Blockchain offer several benefits when applied to identity management:

1. Decentralized – identity information is references in the ledger and no single entity owns or controls all device identities
2. Immutable – transactions recorded on blockchain are tamper resistant
3. Open – blockchain networks can be designed to be open, thereby allowing heterogenous devices to communicate securely over an untrusted network
4. Self-sovereign – identity information of a given device is owned and controlled by that device, and is not available to or shared with any other entity on the network

Task 5*: Private and Safe Integration*

The goal of Task 5 is to design and evaluate a DER cooperation scheme that enables the aggregation of DERs without impacting network reliability while maintaining realistic assumptions about data availability, communication infrastructure limitations, and privacy. The objective of subtask 5.1 was to evaluate the impact of DERs on network reliability and design and validate a DER cooperation scheme that can be used to reduce impacts to the distribution system. The objective for the DER coordination scheme was to achieve >10% cost savings for the grid operator compared to a scenario with no cooperation, where DER owners are performing local cost minimization. Cost savings for the grid operator can be achieved by preventing necessary upgrades to voltage regulation equipment and distribution system transformers.

Subtask 5.2 focused on extending the DER cooperation capabilities to ramping, regulation, and black start use cases. The objective for the signal tracking algorithm performance was to achieve <10% tracking error for following ramping and regulation signals. Methods for extending battery system lifetime when following regulation signals were also investigated, with a goal of extending battery lifetime by a least 10%. The ability of stationary storage and EVs to provide power during a blackout was also considered, with a goal to provide at least 4 hours of uptime.

Subtask 5.3 focused on deploying the developed coordination system on a hardware-in-the-loop (HIL) setup and evaluating algorithm performance using the performance metrics from the previous two subtasks. Specifically, the objective was to achieve greater than 10% cost savings for the consumer and to achieve signals tracking errors of <10% for ramping and regulation signals.

Task 6*: Scalable Distributed Privacy for Information and Energy Exchange*

One can formulate privacy-preserving control of partitioned energy resources as a security problem. A client wants to use a partitioned energy resource. Many clients own partitions of the resource, but it is operated by a central server. The server maintains a (possibly negative, in case of charging and discharging) minimum rate, maximum rate, and a maximum amount of the resource that the client can use. The client wants their usage to remain private; the server should learn the sum of all client's usages but not the usage of any individual client. The server wants to ensure that no client uses a rate outside their allowed bounds, more than the maximum amount, or drains their resource below zero.

To give a concrete example, consider a community battery installation which sells partitions of the battery to individuals. Each partition has associated minimum and maximum power values n and m (a rate limit) and remaining energy value e (a usage limit). In order to operate their partition of the battery, users send discharge requests to the operator. The operator checks that each request is valid by checking that the user doesn't discharge less than *n* power, more than *m* power, and that the user won't expend more than the e energy remaining in their share. The operator then discharges power equal to the sum of all user requests.

To be practical for real-world deployment, the system (and particularly the client) must be efficient. Energy systems often run on low-power embedded devices with limited compute, memory, or bandwidth (e.g., smart meters, energy system controllers). The system assumes the server is commodity hardware, with resources at least equivalent to a recent desktop. In summary, a system for privacy-preserving control of partitioned energy resources has the following goals:

1. Client privacy: the resource operator should only learn the sum of all client requests.
2. Server integrity: the resource operator should be able to detect and reject invalid client requests.
3. The system should be computationally efficient, able to run on desktop class systems and in some configurations, clients can be embedded devices.

Task 7*: Use Cases*

Task 7 has the following main objectives:

1. We would work with our industry partners on specifying and documenting uses cases, particularly those that are pertinent to critical energy infrastructure.
2. We would recruit members for an Industry Advisory Board and hold an annual industry advisory board meeting. The meeting would review and validate the approaches and outcomes for each Task, offering advice on use cases and discussing potential for commercialization and engagement opportunities.
3. We would work with industry partners to specify and document commercialization opportunities, particularly those pertaining to critical infrastructure for the Army or Navy, and electric utilities that serve military installations and communities.

III. Technical Approach

Task 2: *Resource Visualization*

Enabling innovative solutions for networked batteries requires having functional abstractions of how those batteries can be used, managed and controlled. The traditional approach to achieving this is to design abstractions for each domain or use case: there would be an interface to DER for industrial users wishing to reduce peaks, an interface for arbitrage, and an interface for microgrid operators, and every other use case.

The Battery Abstraction Layer takes a different approach: distributed energy resources have a single, common, standard interface that is general enough to use in the majority of use cases. By having a single, common interface, energy resources are not siloed into a particular use case and can be easily repurposed across them. A factory, for example, if operating well below capacity, could apply its energy storage to a larger arbitrage or demand-response market, because those applications and operators would use the same interface. Towards this end, the BAL draws inspiration from computer systems, where the operating system (OS) provides a common programming abstraction of hardware resources, while hardware manufacturers build devices with software drivers that provide that abstraction. This makes it possible for third parties to build applications on the OS which are agnostic to the underlying hardware. In line with these principles, we propose the Battery Operating System (BOS).

The key resource that BOS introduces and manages is a logical battery. There are two kinds of logical batteries: physical and virtual. A **physical battery** is what we think of today when we think of distributed energy resources. It represents physical energy storage, controlled and managed by a networked battery management system (BMS). The BMS provides information and control on the state and activity of the battery.

A **virtual battery** is the key new concept in BOS. Virtual batteries allows BOS to aggregate multiple batteries into a single, larger virtual one, or to partition the resources of a single battery into multiple virtual batteries of different sizes, which in sum are equal to the original battery. An analogy for virtual batteries is hard disk management in computer systems: multiple physical drives can be aggregated (via RAID) into a single logical disk. A single logical disk can be partitioned into multiple logical volumes that users can use like a regular hard drive. Just as a computer OS manages the virtualization of disks, systems using the BAL can manage virtual batteries, physical batteries, and what they can do.

Virtualization allows a system to allocate and manage resources according to each user's needs. There are two types of virtual batteries: aggregate and partitioned. An **aggregate battery** takes several batteries and presents them as a single, larger battery. A system may, for example, aggregate several smaller batteries as one large battery for demand response purposes. A **partitioned battery** takes a single battery and splits it into multiple smaller ones. For example, a home-owner’s association may use BAL to pool funds to acquire one large battery array, aggregate it, then partition it

for the individual homes according to their fund contribution. As shown in Figure III.2.1 – Figure III.2.2, virtualization allows systems using BAL to treat a battery the same without needing to know if it is actually one battery, part of a larger battery or many smaller batteries.

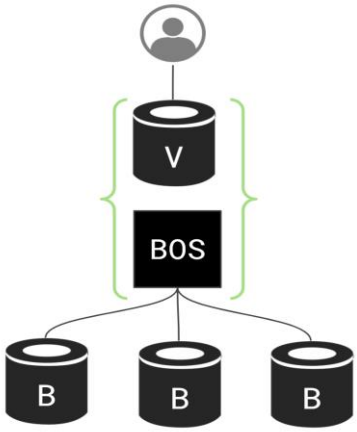

Figure III.2.1a: Aggregate Battery

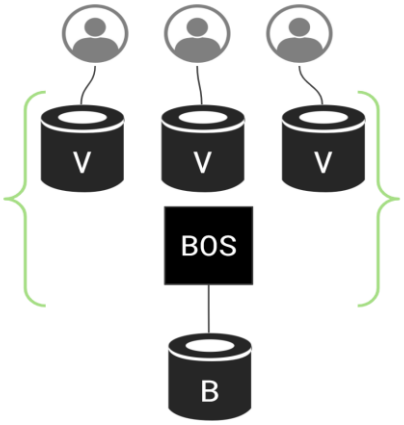

Figure III.2.1b: Partitioned Battery

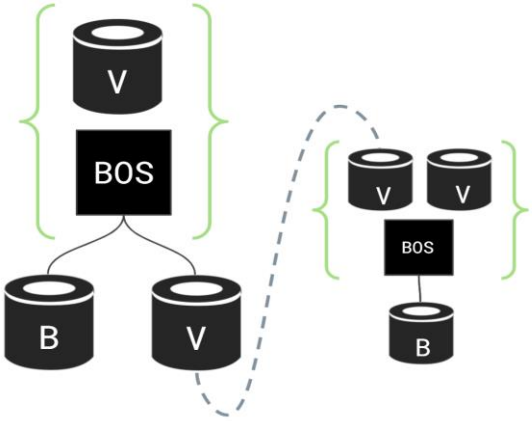

Figure III.2.2a

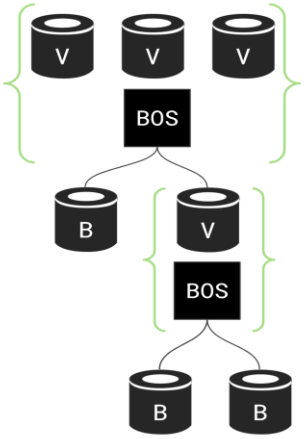

Figure III.2.2b

Figure III.2.1: Virtual batteries allow a BOS to aggregate multiple batteries into a single logical battery or partition a battery into multiple logical batteries. These source batteries B can themselves be logical batteries, or physical ones.

Figure III.2.2: Fig. III.2.2a shows an example of a networked virtual battery that allows BOS to interact with a battery using the BAL API over a network, e.g., to a logical battery on another BOS node. Fig. III.2.2b demonstrates the ability to combine the two concepts.

Aggregate Batteries

An aggregate battery takes multiple batteries (logical or physical) and composes them into a single, larger battery. Three major questions arise in aggregate batteries:

1. What current ranges and capacity should an aggregate battery have, based on its constituent batteries?
2. If constituent batteries have unbalanced states of charge, how should the aggregate battery report this?
3. How does an aggregate battery advertise its location or connection point?

We examine each in turn.

Reporting Discharge and Capacity

Consider the following example: we have two batteries, A and B, with the following properties. We use ampere-hours as the units of capacity to simplify explanation of the issues that arise.

Battery	Discharge	Capacity	State of Charge
A	60A	110Ah	100%
B	40A	40Ah	100%

Table III.2.1: Example batter configuration

If we combine these two batteries into a single, larger battery C, what values should BAL report? One simple solution is to simply sum the values.

Battery	Discharge	Capacity	State of Charge
A	60A	110Ah	100%
B	40A	40Ah	100%
C (A+B)	100A	150Ah	100%

Table III.2.2: Aggregate battery reporting a simple sum of values. The report is inaccurate: battery C cannot discharge at 100A for 1.5 hours.

This aggregate battery C reports having a maximum current of 100A and a capacity of 150Ah. This means that it has a C-Rate of 0.67 (100A/150Ah). However, if battery C is instructed to discharge at 100A, this will cause both A and B to discharge at the maximum discharge rates: A will discharge at 60A and B will discharge at 40A. After

one hour, B will fully discharge. The maximum current then drops to 60A and 50Ah remains. Therefore, reporting a discharge rate of 100A is misleading: the aggregate battery can discharge at 100A for only one hour, at which point its maximum discharge drops to 60A.

However, this discontinuity is dependent on the discharge rate. Smaller discharge rates can operate continuously over the entire charge. For example, if battery C is instructed to discharge at 30A, then it can discharge for 5 hours at 30A: battery A discharges 22A and battery B discharges 8A.

One approach to solve this problem is to report a current curve, based on capacity or some other property. For example, battery C could report several discharge curves based on a number of discharge rates. As Section 6 discusses, such complex data models are not easily supported in existing standards. Therefore, BAL takes a simpler approach, of reporting a scalar value for the maximum discharge rate.

The discharge rate for an aggregate battery is derived from the C-Rate of its constituent batteries. The C-Rate of an aggregate battery is the minimum C-Rate of its constituent batteries. Its capacity is the sum of its constituent batteries, and its maximum discharge rate is derived from the C-Rate. In our example, this means that battery C has a maximum discharge of 81A. This comes from the fact that the maximum C-Rate of battery A is 0.56 (60A/110Ah). If battery A discharges at its maximum rate, it will discharge in just under two hours. Therefore, the aggregate battery limits battery B such that it will discharge in the same amount of time, at 21A.

Battery	Discharge	Capacity	State of Charge	C-Rate
A	60A	110Ah	100%	0.56
B	40A	40Ah	100%	1
C (A+B)	81A	150Ah	100%	0.56

Table III.2.3: Aggregate battery reporting based on C-Rate. The report is accurate: battery C can discharge at 81A for 110 minutes.

Reporting with Unbalanced States of Charge

The second challenge that arises in aggregate batteries: what values should BAL report if their charges are not balanced? In the steady state and regular operation, BAL keeps the state of charges of constituent batteries balanced, because doing so allows an aggregate battery to provide a consistent current over its charge. However, when an aggregate battery is first created, the constituent batteries may not have identical states of charge. Also, if one of the constituent batteries loses network connectivity (see below), it may cease charging and discharging with the others and so have its state of charge diverge.

We consider three options:

1. Variable current: Use the minimum C-rate computation to compute the current of each battery. The advantage of this approach is that it is simple and adds no further logic. The disadvantage is that the discharge and charge currents of an aggregate battery change significantly as its charge equalizes. The state of charge of the aggregate battery is the weighted sum of the constituent batteries based on their maximum charge.
2. Offline: Take an aggregate battery offline when its state of charge is not balanced and charge or shift charge to balance it. The advantages of this approach are that it is simple and requires no logic, and the reported properties of a battery do not shift significantly over time. The disadvantage is that failures of constituent batteries can cascade into larger failures, limiting aggregation and reliability.
3. Variable charge: Fix the discharge current of the aggregate battery and derive a reported aggregate state of charge based on how long the system could sustain this discharge. This has the advantage that the discharge of an aggregate battery is constant. It has the disadvantage that the state of charge does not reflect the actual state of charge of the constituent batteries: charging the aggregate battery requires less energy than what the state of charge and capacity indicate.

The BAL uses the variable current approach. The offline approach is too fragile to network failures, which can cause cascading failures in large battery topologies. The variable charge approach leads to much greater swings and uncertainty than variable current, leading to variable current being a better choice. Consider the following example, using variable current:

Battery	Discharge	Capacity	State of Charge	C-Rate
A	60A	110Ah	9%	0.56
B	40A	40Ah	100%	1
C (A+B)	50A	150Ah	33%	1

Table III.2.4: Unbalanced aggregate battery reporting using variable current. Because A's state of charge is 9%, battery C can only provide 50A, rather than the 81A in Table III.2.3.

Battery C can report a higher C-Rate than before, because battery A is no longer limiting the discharge of the battery. However, its current is capped to 50A so that battery A can discharge its 10Ah over an hour.

With variable charge, the batteries would report a state of charge of 13% because, with a fixed maximum discharge of 81A, the aggregate battery will last longest with 40A from battery B and 41A from battery A. If 41A come from battery A, it can discharge for just

under 15 minutes (10Ah). If battery B discharges that long, it can discharge 9.75Ah, for a total of 19.75Ah, or 13% of 150Ah.

Battery	Discharge	Capacity	State of Charge	C-Rate
A	60A	110Ah	9%	0.56
B	40A	40Ah	100%	1
C (A+B)	81A	150Ah	13%	0.56

Table III.2.5: Unbalanced aggregate battery reporting using variable charge. Because A's SoC is 9%, to maintain an 81A discharge battery C can only report 13% SoC.

Because variable current is more stable (the current of the matched batteries remains stable), BAL uses variable current.

To address the uncertainty that variable current introduces, the BAL reports two sets of values for a virtual battery: the expected values and the current values. The expected values report the expected charge, discharge and capacity if the virtual battery is operating optimally. The current values report potentially lower values if constituent batteries are offline or charge is not balanced.

Reporting Location / Connection Point

The final issue that arises in aggregate batteries is how to report the location and tolerances of an aggregate battery connection point.

Location can be an important consideration when it is neceesary to keep grid elements within voltage bounds or maintain other safety limits. In addition to location, information about the connection point includes the nominal values and bounds for VAR, voltage, frequency, and wattage.

Reporting location is fundamentally in tension with virtualization: the goal of virtualization is to abstract away physical properties and break brittle assumptions on them. Location, however, is fundamentally physical and it represents a point in space. If the constituent batteries of an aggregate battery are highly distributed, they cannot be easily summarized as a single location.

We propose that the BAL reports aggregate location as the center point of all of the constituent connection points, computed with the Floyd-Warshall algorithm. This approach means that virtual batteries which aggregate geographically dispersed batteries will appear to be in the "center" of the grid topology, combined with an indication that this is a center position (i.e., the top of a connection point hierarchy).

We are currently exploring and evaluating different approaches for reporting bounds and tolerances. Our current approach involves a “down-up” query model, where values propagate down from an aggregate to its constituent batteries, then back up to report actual values.

[Floyd-Warshall algorithm: *https://dl.acm.org/doi/10.1145/367766.368168*]

Partitioned Batteries

A partitioned battery takes a single logical battery and divides it into multiple, smaller batteries. Because of normal variations and because the underlying battery may be an aggregate battery, the values of the underlying battery can change over time even in the absence of charging and discharging. When this battery is partitioned, this raises the question and policy of how these variations are reflected in the partitions. Furthermore, when some partitions request charging and others request discharging, the partitioner needs to map these requests onto the underlying battery and dynamically account for changes in charge.

Partitioning Policies

Partitioned batteries are defined in terms of fractions of the expected value of a battery. For example, if a battery has an expected maximum current of 80A, this can be partitioned into 25% and 75% as two batteries with currents of 20A and 60A. This proportional splitting is uniform across the properties. The BAL provides three policies for partitioned batteries:

1. In a **proportional** battery, the partitions are strict fractions of the underlying battery. Any variations in the values of the underlying battery are proportionally reflected in the partitions.
2. In a **tranched** battery, the partitioned batteries are placed in an ordering of tranches: there is a top (level A) tranche, a second tranche (level B), and more tranches as the battery is more finely partitioned. When values of the current values of a battery go above the expected value, these increases are first given to the highest tranche. However, when the current values of a battery go below the expected value, these reductions are first taken from the lowest tranche.
3. In a **reserved** battery, the partitions are placed in an ordering, as with a tranched battery. However, increases and decreases both affect the bottom partition first. In a reserved battery, the higher partitions have more reliable and dependable values.

Table III.2.6 shows the difference between these three partitioning policies, with capacity as the variable of interest. A single battery A is partitioned into P1, P2, and P3. P1 is the top partition and P3 is the bottom partition.

Battery	Charge	Fraction	Proportional	Tranched	Reserved
A: Expected	100				
A: Actual	90	90%			
P1	50	50%	45	50	50
P2	30	30%	27	30	30
P3	20	20%	18	10	10

Table III.2.6: If the actual charge of a partitioned battery is lower than its expected charge, the type of partition determines how this deficit appears: in both the tranched and reserved batteries the deficit is borne by the bottom battery P3.

Battery	Capacity	Fraction	Proportional	Tranched	Reserved
A: Expected	100				
A: Actual	110	110%			
P1	50	50%	55	60	50
P2	30	30%	33	30	30
P3	20	20%	22	20	30

Table III.2.7: If the actual charge of a partitioned battery is higher than its expected charge, the type of partition determines how this surplus appears. In the reserved battery the surplus is taken by the bottom battery P3, while in the tranched battery the surplus is taken by the top battery P1.

Battery	Capacity	Fraction	Maximum	Reserved
A: Expected	100			
A: Actual	110	110%		
P1	50	50%	62.5	50
P2	30	30%	37.5	35
P3	20	20%	25	25

Table III.2.8: If Surplus charge on partitioned batteries cannot go above the maximum charge of a partition. In this example, the excess 10 charge units are first applied to P3, raising it to 25. The remaining 5 surplus units are applied to P2.

Applied surpluses and deficits are bounded by the 0 and the maximum values of the partitions. For example, suppose, in Table III.2.7, the total maximum charge (capacity) is 125. P3 is 20% of this, so its maximum charge (capacity) is 25. If the actual charge of A is 110, then the surplus will be allocated so we get the results in Table III.2.8.

The tranched and reserved policies can be useful in different economic models and markets. Furthermore, these policies can compose. One can construct complex topologies of batteries, with tranches of aggregates of tranches.

Partition Accounting

A second issue that arises in partitioned batteries: partitions can request discharging and charging simultaneously. Consider, for example, the following partitioned battery A, partitioned into P1 and P2. P1 is discharging at 40A and P2 is charging at 60A.

Battery	Capacity	State of Charge	Charge	Discharge
P1	80Ah	100%		40A
P2	120Ah	16%	50A	
A	200Ah	50%	10A	

Table III.2.9: Behavior of Battery A when its two partitions (P1 and P2) request both charging and discharging. The underlying battery is the sum of the charge and discharge, while State of Charge flows from P1 to P2.

In this example, the underlying battery A charges at 10A (the sum of the charge and discharge requests). The 40A of discharge from P1 is effectively absorbed by P2's request. This means that the underlying battery only requests 10A of charging. However, the state of charge on P1 and P2 need to change to reflect this virtual energy flow from P1 to P2. Note that this change in their state of charge is entirely in software: the BAL accounts for this virtual flow. In 2 hours, P1's state of charge will decrease to 0% and it will stop discharging. This 80Ah of charge will transfer to P2. After 2 hours, P2's state of charge will be 100%, having drawn 20Ah from external charging and 80Ah from P1.

This virtual transfer of charge requires that the entity providing virtualized batteries can accordingly bill its clients. In the above example, P2's charging cost is what will pay P1

for discharging. This implies that the owner of P2 must pay the owner of P1 at least as much as P1 expected to receive from the grid. Exploring the exact billing and financial agreements necessary for these virtual charge flows is an area of future work and discussion. For example, if P1 promised to discharge as part of a day-ahead market, it is important that the owner of P1 isn't able to simply virtually transfer the charge to P2 (at no cost) and claim its responsibilities were met. One approach we are discussing is for both charges to go through the utility directly, such that a virtual transfer does involve a flow of funds through the utility directly.

Battery Abstraction Layer API

BOS provides a Battery Abstraction Layer API as a software programming interface for interacting with batteries. The API is identical for virtual and physical batteries, making them indistinguishable to software. The BAL follows three design principles:

Flexible: As the BAL defines how many different pieces of software will interact with batteries, it must provide a general and flexible API. A flexible API that allows many different use cases will be highly reusable and provide a common basis for more complex services. It should provide complete-yet-simple functionalities of a physical battery management system, which includes retrieving basic information like voltage, current, state of charge, as well as controlling the discharging/charging of a battery. This flexible API should be implementable for both physical and virtual batteries.

Simple: The API must be simple. Complex APIs are hard to implement and difficult to write software for. If the API is simple, it is also easier for it to be general, as its core abstractions do not depend on particular features or capabilities.

Atomicity: To support correct and stable services being built on top of it, the BAL must allow software to atomically (as a single, indivisible action) request multiple values at once. This is critical so that services do not try to (incorrectly) compute on values obtained at different moments in time. For example, software must be able to atomically request both current and voltage simultaneously. If the voltage and current are measured at different times, the power calculation can be incorrect. For example, suppose after the voltage is sampled there is a huge spike in draw, leading to a high current and a voltage drop. Multiplying the high current with the high (previous) voltage would indicate the power output is higher than it is.

The Battery Abstraction Layer contains the following two functions:

1. GetStatus(): retrieves a snapshot of the logical battery status, which returns a structure containing a snapshot of the following information.

- voltage,
- current,
- maximum current,
- state of charge,
- maximum charge,

- the timestamp of the snapshot, represented as an interval.

2. SetCharge()/SetDischarge(): sets the amount of current flow into (charge) or out from (discharge) the battery.

This GetStatus() function satisfies the atomicity principle, as all of those values are returned on one call of GetStatus. The timestamp interval is to tackle with different stalenesses of data from different batteries when the virtual battery is an aggregate of multiple underlying logical batteries. Each physical battery request is an atomic set of results, but virtual batteries may aggregate the results of more than one physical battery.

For physical batteries, BOS includes driver software that converts these calls into messages or commands to the physical battery, for example over a serial port or network connection to a battery management system (BMS). Virtual batteries are entirely software.The next two sections describe how virtual batteries set charge or discharge rates.

Charge / Discharge on Aggregate Batteries

BOS maps operations on virtual aggregate batteries into their constituent batteries. The charge and discharge amounts are distributed across the constituent batteries based on their contribution to the aggregate value. For example, consider the batteries in Table III.2.3 (repeated here):

Battery	Discharge	Capacity	State of Charge	C-Rate
A	60A	110Ah	100%	0.56
B	40A	40Ah	100%	1
C	81A	150Ah	100%	0.56

Table III.2.10: Aggregate battery reporting based on C-Rate. The report is accurate: battery C can discharge at 81A for 110 minutes.

Battery C's discharge of 81A consists of 60A from A and 21A from B. Therefore, if software calls SetDisharge(50) on battery C, BOS will tell battery A to discharge at 50A x (60/81) and battery B to 50A x (21/81).

Charge / Discharge on Partitioned Batteries

BOS combines change and discharge operations on partitioned batteries using a simple sum. The resulting value is passed to the underlying battery. For example, suppose we have the following partitioned battery as shown in Table III.2.11.

If software tells P1 and P2 to both discharge at 20A, BOS calls SetDischarge on the source battery with 40A. However, if software tells P1 to charge at 20A and P2 to discharge at 20A, then BOS stops charging/discharging the source. Instead, over time, BOS changes its accounting of charge for P1 and P2, shifting charge from P2 to P1.

Battery	Fraction	Capacity	Charge	Discharge
Source		200Ah	80A	100A
P1	50%	100Ah	40A	50A
P2	30%	60Ah	24A	30A
P3	20%	40Ah	16A	20A

Table III.2.11: Example proportional battery

This implies an economic exchange. E.g., P2 is discharging to earn income by providing power, while P1 is expecting to pay for the power it draws. The entity running BOS must interact with the local utility as a virtual meter, informing the utility that P1 is drawing 20A whole P2 is discharging 20A.

Battery Topology Configuration

The BOS function family `make_{battery, aggregator, splitter}()` provides the mechanism for constructing a topology of batteries that all use a common Battery Abstraction Layer (BAL).

- make_battery(name, kind, interface) -- create a new battery with the given name on the given interface. The interface can be local or remote (UART, BLE, TCP, etc.); the battery can be virtual or physical.
- make_aggregator(name, [batteries...]) -- aggregate a list of batteries into one virtual battery with the given name.
- make_splitter([(name, info)...], name) -- split a battery with the given name into n different batteries.

Example Physical Battery Driver: JBD BMS

To show how the BAL is implemented on top of a physical battery, we consider the example of a battery management system (BMS) manufactured by JBD, the SP04S020. This BMS can be controlled and queried over either a UART or BLE interface. The BMS provides a rich interface, with many controls and data.

To implement GetStatus(), our implementation reads the Basic Info register of the BMS, which contains a large set of data values, including voltage, current, state of charge,

pack cells, FET status, and maximum charge. It translates a subset of the data in this data structure into the BAL's data structure and returns the values.

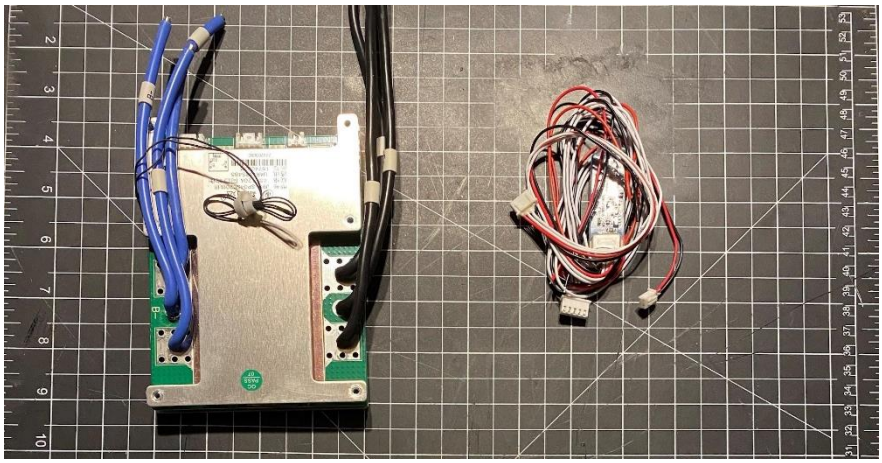

Figure III.2.3: Illustration of how BAL is implemented on top of a physical battery

To implement SetCurrent(current), our implementation controls the MOSFET on/off status of the BMS by writing to the FET status register -- if the target current is positive, turn on the discharging MOSFET and shut off the charging MOSFET, and vice versa. If the current is zero, the implementation disables the FET. The BMS does not have a current regulator. We are working to extend it to have one.

Example BOS Software

We have written two implementations of the BAL API as described above. Both implementations have documentation in the repository, explaining the software structure and how to use it.

The first implementation is an initial prototype. It operates in Python and can be found in the trustder subdirectory of the BatteryOS GitHub repository. It allows software to construct battery topologies. In addition to virtual and physical batteries, this implementation adds the abstraction of a networked battery, which is a way to control a BAL device over a network. This allows a BOS instance to construct virtual batteries from logical batteries that exist on other BOS nodes. For example, one BOS node that has three BMSes which it combines into a single aggregate battery can provide access to this aggregate battery over a network. Another BOS node can create a networked battery to take control of this aggregate battery, and partition it into multiple batteries. Networked batteries allow BOS to provide distributed battery resources.

The second, and more complete implementation, is in C. It includes the full set of BAL and Battery Operating System features, including a full implementation of the

cybersecurity mechanisms described in our Cybersecurity Design Document as well as IEC 61850 interoperability. It can be found in the bos_rewrite subdirectory of the BatteryOS GitHub repository.

Representing a BAL Device in IEC 61850

International Electrotechnical Commission (IEC) 61850 is an international standard defining communication protocols for intelligent electronic devices at electrical substations. IEC 61850 defines object models, data models, and mappings to communication protocols such as HTTP. The Battery Abstraction Layer is a software API: it defines methods and data structures that software running on a microprocessor or microcontroller can use. This section describes how the BAL API can be mapped to IEC 61850 data models: it outlines how a software service can respond to and generate IEC 61850 messages by interacting with a BAL device.

In IEC 61850, the standard battery energy storage object is a ZBAT node, defined in *IEC 61850 7-420*, Section 8.2.2. A ZBAT node has 4 Mandatory and 23 Optional data objects. This table shows how a software service can represent a BAL device as an IEC 61850 ZBAT device.

Task 3: *Secure ID for Asset Authentication*

We started our research by searching for blockchain-based identity management systems. Many of the approaches we discovered were notional and did not accompany any published work and/or code repositories. We then narrowed our search to systems that had actual technical implementation and assessed their application to grid cyber-physical network. We discovered that most of the published work required a decentralized trusted identity implying that identity proofing of devices was based on existing trusted credentials. This was a serious limitation since one of our key design criteria for the identity management system is to support self-sovereign identity meaning, only the device/asset should own and control its own identity. After a comprehensive survey of literature on blockchains and identity management, we decided that an entirely new approach was needed to support self-sovereign identity for grid assets. We developed two components, which when put together, make up the core of the Secure ID system – 1) Keymaker, an algorithm to enable self-sovereign identity for grid assets, and 2) Keychecker, an algorithm to verify that a device indeed is what they claim to be. Both algorithms were then implemented as a set of smart contracts, protocol and library functions that enable grid assets to be verified by the rest of the network.

Keymaker and Keychecker

Keymaker allows device identity to be created and identity-related data, or in our case shards, to be distributed to nodes on the network. It does so by:

- Creating multiple shards of a device identity / device key, and
- Distributing shards to adjacent nodes on the network

Keychecker allows a device identity to be socially verified by other devices on the network by:

- Enabling adjacent network nodes to combine shards

Simply speaking, Keymaker and Keychecker protocols allow self-sovereign device identity to be socially verified by other devices on the network. The cryptographic underpinnings of Keymaker include 1) Shamir Secret Sharing (SSS), and 2) Symmetric-key Encryption.

Shamir's Secret Sharing is an ideal and perfect (k , n) -threshold scheme. In such a scheme, the aim is to divide a secret S (for example, identity of a grid asset) into n pieces of data S1 , … , Sn (known as shards or shares) in such a way that:

1. Knowledge of any k or more Si pieces makes S easily computable. That is, the complete secret S can be reconstructed from any combination of k pieces of data.
2. Knowledge of any k − 1 or fewer Si pieces leaves S completely undetermined, in the sense that the possible values for S seem as likely as with knowledge of 0 pieces. That is, the secret S cannot be reconstructed with fewer than k pieces.

The essential idea of the scheme is based on Lagrange interpolation theorem, specifically that k points is enough to uniquely determine a polynomial of degree less than or equal to k − 1. For instance, 2 points are sufficient to define a line, 3 points are sufficient to define a parabola, 4 points to define a cubic curve and so forth. A (3, 5)-threshold scheme is depicted in the diagram below.

Symmetric-key encryption uses the same cryptographic keys for both the encryption of plaintext and the decryption of ciphertext. The keys may be identical, or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. Symmetric-key encryption is sometimes also referred to as Authenticated Encryption since it effectively verifies that two parties possess the same encryption key.

Secure ID contracts when executed performs the following steps:

1. Generate random key
2. Split random key into n+1 shards using SSS
3. Encrypt identity key using random key as an encryption key for symmetric encryption algorithm used is NaCl's 'secretbox', which consists of the XSalsa20 stream cipher, and a poly1305 message authentication code
4. Append each shard with resulting cipher
5. Distribute n shards to network nodes and store one shard on the originating device

This allows us to protect device identity from being shared with others over the network while also allowing the device to be authenticated by other devices on the network.

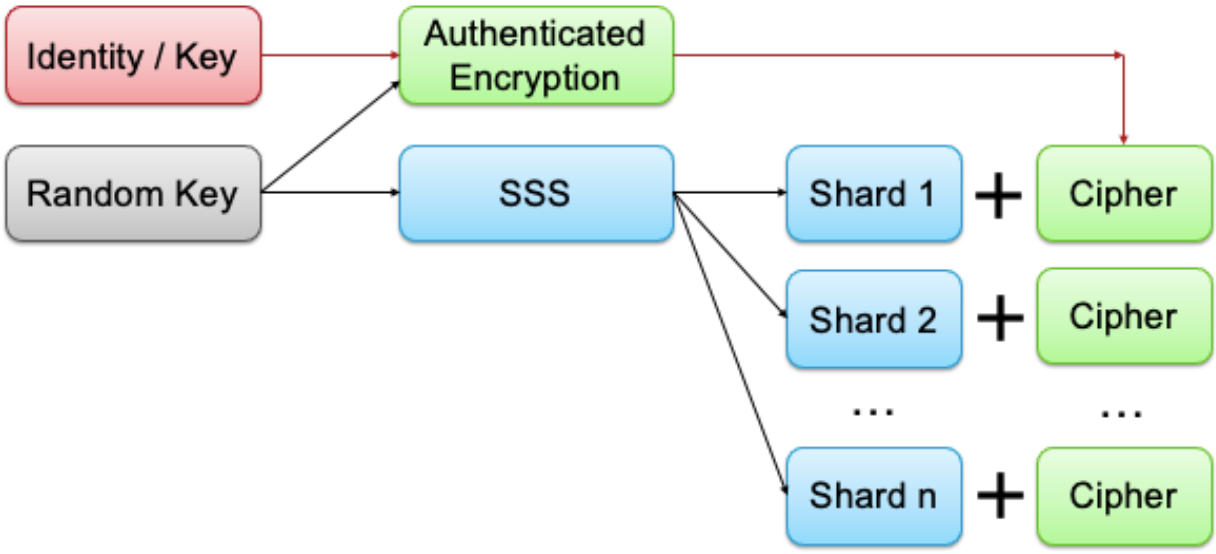

Figure III.3.1: Execution steps for secure ID contracts

Task 5: *Private and Safe Integration*

Task 5.1: Design and simulate DER cooperation for private and safe integration with learning

Learning power flow proxies

DER cooperation strategies require a power flow model to account for how power injections affect power flows and voltages within a network. However, for distribution systems, these physics-based models may not always be up-to-date or accurate and may not be available to the entity that is performing the DER cooperation. To address this, approximate network models can be learned directly from real world measurements of power flow quantities and used as a proxy for the physics-based power flow model in the DER cooperation algorithm.

We first evaluated the accuracy of different data-driven power flow models and then analyzed their performance when incorporated into grid optimization problems. The data-driven mapping is between real and reactive power injections and voltage magnitude and phases in a distribution system. The models evaluated included linear regression models, piecewise linear models, support vector regression, and several different neural network models. The results from this analysis were previously reported in the Design Document for subtask 5.1. While results showed that nonlinear models can obtain higher accuracy than linear regression models, the accuracy of the linear

approximations was not significantly worse and was determined to be sufficient for most applications. Therefore, in the rest of this work, linear data-driven power flow approximations were used.

Grid Simulation Suite

To evaluate the impact of DERs on network reliability and to verify the performance of a DER cooperation scheme, we designed a comprehensive distribution grid simulation suite including five distribution networks (Table III.5.1). The simulation suite includes five distribution networks across a range of climates, urban, suburban, and rural areas. The characteristics of these networks are listed in Table III.5.1. The simulation suite is linked to datasets for residential and commercial power consumption and PV generation and EV charging profiles. The power flow analysis of the networks is performed using OpenDSS. After intelligently combining the load profiles, DERs, and network information, the simulation suite runs a power flow study that can be used to quantify metrics on grid reliability. In our case, we evaluate metrics for the number of grid upgrades needed to sustain the quantity of DERs estimated to be present in 2050.

Name	**IEEE 123**	**Iowa**	**Downtown SF**	**Shore SF**	**Tracy**
Type	Suburban	Suburban	Urban	Urban	Suburban
Peak Month	August	July	January	January	August
Location	Sacramento, CA	Ames, IA	San Francisco, CA	San Francisco, CA	Tracy, CA
# of nodes	123	240	426	19	162
# of transformers	96	178	171	12	74
% of commercial customers	9	6	21	100	13
Peak power consumption (MW)	4.293	6.104	8.304	8.68	1.626
Average daily energy consumption (MWh)	10.607	36.624	68.746	49.824	11.211
Demand penetration %	23	23	23	23	23
EV penetration %	50	50	50	50	50
PV penetration %	64	33	45	45	64
Storage penetration %	27	14	19	19	27

Table III.5.1: Characteristics of the various simulated networks. Some networks come from reference [5.1].

The net load profiles must be assigned to each node of the network to perform the power flow simulation. The uncontrollable load profiles include a base demand profile

and rooftop PV generation. The controllable portion is determined by the operation of each local device controller under their prescribed set of storage or EV charging constraints. Therefore, the net load includes four components: a base demand profile, rooftop PV generation, EV charging, and stationary storage operation, each parameterized by a penetration percentage. The specific penetration values for each of the four components is based on the year the simulation takes place, which is chosen to be 2050 for this report. The penetration of rooftop PV generation is defined as the percent of total energy consumption that is generated by rooftop PV. EV charging penetration is defined as the percentage of cars in the network that are EVs. Storage penetration is the percentage of the average daily network energy consumption that can be stored in stationary storage. The details on how these resources are assigned to nodes in the network were described in detail in the Design Document for subtask 5.1. Each node in the simulation suite is assigned a time-of-use (TOU) electricity rate structure depending on whether it is a residential or commercial node.

Local control with autonomous operation

The main performance metric for subtask 5.1 is to achieve cost savings of >10% compared with no coordination. Here we focus on cost savings for the grid operator, which includes the prevention of voltage regulation equipment upgrades and transformer upgrades. Two metrics are used to quantify the need for these upgrades after the addition of DERs. The first metric measures the length of time a node has voltage magnitude deviations of more than ±5%. The second metric quantifies the number of overloaded transformers (greater than rated apparent power for >2 hours).

In this section, we describe a local controller (LC) scheme which is used to determine the power injection profile of the EV chargers and stationary storage units placed in the network under no network cooperation. Nodes in the network with an EV charger or stationary storage unit can measure consumption and generation in real-time and perform local computations. The LC can control the net load of the customer by setting the storage or EV charging rate, subject to constraints. The LC at each node makes its decisions based on forecasts of future uncontrollable net load. The LC algorithm is operated in a receding horizon fashion where the solution of an optimization problem is calculated every 15 minutes and the power injection setpoints are sent to the DER to operate for the next 15 minutes.

The LC optimization was previously described in detail in the Design Document for subtask 5.1. The objective minimizes the cost of energy and excessive wear on the battery. The constraints define battery capacity limits, the dynamics of the battery, and constraints on the charge capacity of each EV at the end of the charging period.

A simulation study was performed on the five networks for their peak load month to evaluate the number of grid upgrades needed by 2050 under the operation of these local controllers. Table III.5.2 shows the hours the network has a voltage violation and the number of overloaded transformers. Even the highest possible adoption of distributed storage with intelligent storage and EV charging management is insufficient

to protect the grid from increased penetration of DERs. All networks need significant upgrades to accommodate the projected penetration of DERs.

Network	Hours of voltage violation	Number of overloaded transformers
IEEE 123	5.5	19
Iowa	8	27
Downtown SF	15	32
Shore SF	1.75	5
Tracy	6	15

Table III.5.2: Results of the power flow simulation study on the five chosen networks.

DER cooperation scheme

We describe a DER cooperation scheme for reducing the number of grid upgrades required to support the growing penetration of DERs. The scheme includes two layers: (1) a global day ahead scheduler that calculates power injection bounds for each customer ensuring voltage and transformer constraint satisfaction and (2) local controllers for each customer that manage local DERs subject to the global bounds. This coordination scheme significantly reduces the need for data exchange between the two control layers and does not require knowledge of DER objectives or detailed DER data. The architecture is similar one developed in previous work [5.2]. The two layers communicate values called supply and demand bounds with each other. These values are estimated for the next day during the day-ahead scheduling phase. Then the LCs enter the operating phase in which they calculate power injection setpoints for the DERs under their control. The two phases of the algorithm are shown in Figure III.5.1.

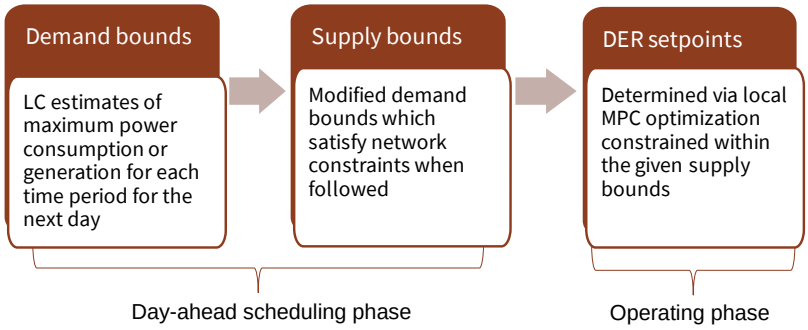

Figure III.5.1: Summaries of the key values communicated between the layers of the cooperation scheme along with the corresponding phase in which the value appears.

Each day estimated bounds on the maximum power consumption or generation for each time period and each node in a network are sent to the global controller (GC). The GC then solves an optimization to determine a new set of upper and lower power injection bounds for each LC that ensure voltage and transformer constraint satisfaction.

These modified bounds (supply bounds) are sent back to each LC. Both steps are completed within the day-ahead scheduling phase of the cooperation scheme. Upon receiving the supply bounds for the day, each LC enters an operating phase where it solves a model predictive control (MPC) optimization problem to determine the power setpoint of each DER. The local optimization minimizes its own objective and deviation from the supply bounds subject to the DER constraints. The flexible loads considered by our scheme include EV chargers and battery storage.

Global controller

To coordinate the operation of DERs across a distribution network, limits on the real power of each customer or node in the network are provided in a day-ahead schedule. These limits aim to prevent operation that will harm the distribution grid voltages or transformers. Each node will request to a global controller a range of their expected maximum and minimum real power consumption or generation for the following day, which are called demand bounds. Since, the demand bounds may not form an acceptable operating point for the network, the global controller returns modified demand bounds for each node. These modified bounds (supply bounds) represent the range of available supply of power capacity in the network. The global controller solves an optimization problem to determine the supply bounds at each node for the next day. The supply bounds are then sent to the local controllers. If each local controller can confine the power injection of their resources to their assigned supply bounds, the network constraints will be satisfied under a few conditions (e.g., the linear models from power injection to voltage and transformer power are accurate).

Local controller with cooperation

Each customer participating in the DER cooperation scheme has a local controller (LC). The LC is similar to the one described previously but can communicate with the GC once per day. In this report we consider only storage and EV charging, but other controllable devices can be included. The LC performs different operations in the day-ahead scheduling phase and in an operating phase. In the scheduling phase, customers who participate in the DER cooperation scheme provide the GC with demand bounds in a day-ahead manner or have the GC select them based on historical data. When using historical data, forecasts and scenarios of the net load for the next day are predicted for each node, which are used to calculate the demand bounds. Customers who provide their own demand bounds can calculate them using a local optimization. During the operating phase, each LC operates in a rolling horizon fashion. It solves an MPC optimization problem where the objective is to minimize local costs while remaining within the supply bounds provided by the day-ahead scheduler. This optimization problem is nearly identical to the autonomous LC optimization except only a single forecast of the load is used and the cost function has an additional penalty on the power consumed above or below the supply bounds. Its minimization allows the LCs to respect the bounds sent from the GC. The output of this optimization is the battery and EV charging power setpoint for the next 15-minute period.

Evaluation of DER cooperation scheme

The results of the DER cooperation experiments for each network are compared to results from autonomous LC operation in Table III.5.3. In addition to the two grid stability metrics, we also compare the percentage increase in electricity costs due to the DER cooperation scheme. Results show the increase in electricity costs is small at a maximum of 4% for the Downtown SF network and only 2.6% for the IEEE 123 bus network. However, the potential cost savings due to reducing the number of grid upgrades needed to prevent voltage violations and prevent transformers from overloading is significant. The total hours of time the network has a voltage violation is reduced by at least 50% for the Iowa and Tracy networks and at best 71% for the Shore SF network. Furthermore, the number of overloaded transformers in the network reduces by at least 7.4% for the Iowa network and at best 40% for the Shore SF network. When considering the cost savings due to preventing transformer upgrades alone, the DER cooperation algorithm can provide a greater than 10% benefit over no cooperation for most networks, thus, satisfying the project metric goal for this task. The benefits of reducing the need for voltage regulation upgrades further improves the cost savings for the grid operator.

Network	**Hours of voltage violation**		**Number of overloaded transformers**		**% increase in electricity costs**
	Autonomous LCs	Bounds	Autonomous LCs	Bounds	
IEEE 123	5.5	1.75	19	15	2.6
Iowa	8	4	27	25	3.1
Downtown SF	15	5	32	26	4.0
Shore SF	1.75	0.5	5	3	3.9
Tracy	6	3	15	12	3.2

Table III.5.3: Comparison of the simulation results between autonomous LCs and LCs using the bounds DER cooperation scheme.

Task 5.2: Extend cooperation capabilities to ramping, regulation and black start

Subtask 5.2 focused on extending the cooperation capabilities to ramping, regulation, and black start. A summary of the performance metrics for this subtask and the achieved results are shown in Table TIII.5.4.

Metric	**Metric goal**	**Average metric performance**
Total cost reduction	>10% compared to no coordination	11% per consumer (can increase with more storage capacity)
Ramp tracking error	<10%	2.644%
Regulation tracking error	<10%	6.132%

Uptime hours during blackout	>4 hours	>4 hours of uptime 31.4% of the time. The percentage can increase with more storage capacity.
Battery system lifetime increase	>10%	This can be achieved by skipping the lowest cost 12.85% of regulation signals, which leads to a loss in total savings of <1%. Skipping additional signals will lead to >10% battery lifetime increase.

Table III.5.4: Summary of metrics for Subtask 5.2

Signal tracking algorithm

The algorithm used for the ramp and regulation signal tracking is an adaptation of the algorithm presented in [5.3]. The algorithm was previously described in previous reports for Subtask 5.2.

Simulation Methodology

The simulations performed for subtask 5.2 use the previously described distribution grid simulation suite and the 2-layer bounds algorithm. The methodology for performing the simulations is as follows: (1) place DERs across the nodes in the distribution grid, (2) determine maximum power ramp and regulation signals that can be followed by the aggregation of DERs at each node with and without bounds, (3) run signal tracking algorithm for the aggregation of DERs within each node, (4) evaluate impacts on grid voltages and transformers, cost savings, and signal tracking error.

Four different simulation scenarios are considered: (i) Uncoordinated scenario - signal following events use the maximum power output of the DERs and occur randomly throughout the day. (ii) Worst case scenario - The maximum power signal coincides with extreme grid events, (iii) Bounded signal following scenario - the signal power is limited such that the DERs will not violate their assigned bounds and events are assigned when the DERs are idle (not performing arbitrage or grid reliability). (iv) Local cost minimization only – no signal following events. The scheduling of signal tracking events for the bounded signal following scenario is a simplification of the process used in [5.4] where we simply select time periods for events without specifying the type or value of the event. This means the time periods selected are those in which there are no other uses for the DERs.

All four scenarios were run on several networks using the placement of DERs that correspond to the year 2050. Figure F6 shows the % of nodes in the network with voltage violations and the % of transformers with overloading at some point in the simulation. As shown, signal following can cause major reliability issues in the worst-case scenario. When signal follow events are distributed randomly throughout the simulation, signal following generally will not cause significantly more issues than

autonomous local cost minimization. However, bounding the signal following and DER operation can provide significant improvement in reliability and prevent the worst-case signal following events.

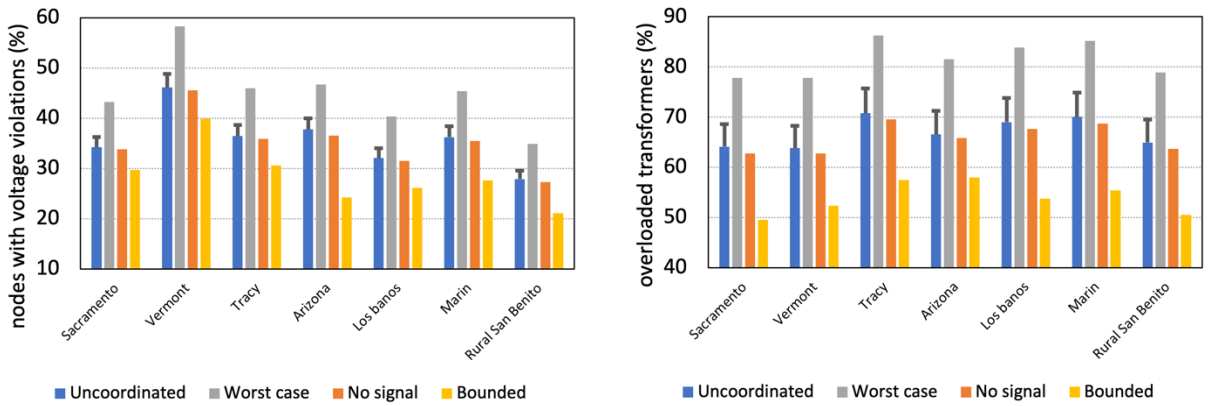

Figure III.5.2: Nodes with voltage violations and overloaded transformers under the different regulation signal scenarios.

Consumer Cost Savings

When evaluating the potential cost savings of ramp and regulation signal following, we use PJM clearing prices for ancillary services. To determine the cost savings earned through signal following, the total signal power capacity for each event is multiplied by the price. For the Sacramento network with the bounds scenario, the batteries have a capacity around 3 MWh and the daily EV charging energy is around 4.5 MWh. The total cost savings over the simulation year from participating in signal follow events was $56,064 with approximately $25,000 coming from the EV portion and the rest from the stationary storage. For consumers who own both a stationary storage and an EV charger, this accounts for approximately a 14% increase in cost savings. From results from subtask 5.1, it was found that the opportunity cost associated with missed arbitrage opportunities due following the bounds leads to a loss of cost savings of approximately 3%. Thus, the total cost savings to the consumer in this scenario is 11%, which exceeds the metric of >10% for this task.

Ramp and Regulation Signal Tracking

Next, ramping and regulation signal following accuracy are evaluated in the Sacramento network with the bounds scenario. When all signal tracking events are assumed to be alternating ramp up and ramp down events, the average tracking accuracy was 2.644%, which is less than the metric of <10% ramp signal tracking accuracy. When all signal tracking events are regulation signals, the average tracking accuracy was 6.132%,

which is less than the metrics of <10% regulation signal tracking accuracy. Figure III.5.3 shows an example regulation signal being tracking in the Sacramento network.

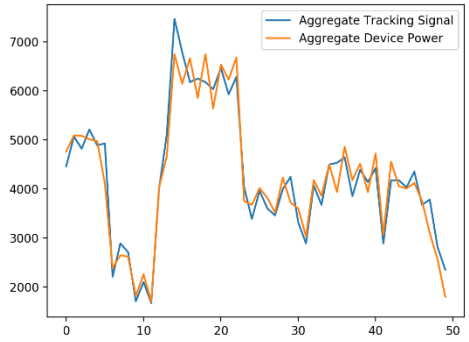

Figure III.5.3: DERs in the Sacramento network tracking a regulation signal (tracking error of 6.365%).

Uptime Hours During a Blackout

The ability of stationary storage and EVs to provide power during a blackout is limited by the total capacity of resources connected to the system at the time. We consider the projections for stationary storage capacity, EVs, and rooftop PV in 2050. When evaluating the Sacramento network, stationary storage alone is only sufficient to provide power to the grid for 1.68 hours on average. When including vehicle to grid charging, the availability of EVs must be considered. Using the EV charging data, we find that only about 15.4% of EVs are plugged in on average. Although EVs have significantly more capacity than stationary storage, they have low availability and can only provide an additional 0.78 hours of uptime during a blackout on average. PV reduces the amount of energy needed by up to 100% depending on the time of day and season. The average 4-hour window during which there is PV generation reduces the energy needed to be provided by the stationary storage and EVs by 34.2%, increasing the uptime of the stationary storage and EVs during a black out to 3.73 hours on average. Additionally, in early morning hours (12am – 5am), average EV availability is higher at 28.6% and the average power consumed is much lower at only 64.8%. Thus, the uptime during a blackout from 12am - 5am is 4.83 hours. Other hours have lower EV availability, so have fewer uptime hours. From the experiments on the Sacramento network, the uptime hours during a blackout of the stationary storage plus EVs plus solar is greater than 4 hours 31.4% of the time, which means the metric for >4 uptime hours during a blackout can be met.

Battery System Lifetime

Finally, we examine the impacts of regulation signal following on battery lifetime. Battery lifetime is generally a function of the number of cycles and the cycle depth of discharge.

If the depth of discharge is limited to 80%, the primary factor affecting the battery lifetime is the number of cycles. If we assume the lifetime of a battery ends once it reaches 20% capacity fade, then the lifetime is approximately 7000 cycles. In the Sacramento network bounds scenario from subtask 5.1 without any signal tracking, the average number of cycles per day for the stationary storage was 1.78, leading to a lifetime of 10.774 years. However, with regulation signal following, the storage units follow an average of 5.45 hours of regulation signals per day, resulting in 6.58 cycles per day and an average lifetime of only 2.92 years. Increasing the lifetime by 10% requires reducing the average cycles per day to less than 5.97, which can be done by reducing 12.85% of regulation signals. If the value of the regulation signal is independent of the number of cycles in the regulation signal, we can achieve this goal by ignoring the lowest profit 12.85% of regulation signals. The trade-off is that the battery will not be compensated for the skipped regulation signal. Using PJM price data, we calculated that a regulation signal should only be followed if the compensation is >$5/MW. This leads to a loss of regulation signal profits of only 5.94% making the total consumer savings due to regulation signal following drop to 13.17%. By ignoring the lowest profit 12.85% of regulation signals, which leads to a loss in total savings of <1%, we can increase the battery system lifetime by 10%, which meets the metric of >10% battery system lifetime increase.

Task 5.3: Implement and test DER cooperation capabilities

This section focuses on deploying the developed system on a hardware-in-the-loop (HIL) setup and evaluating algorithm performance using metrics from subtask 5.1 and 5.2. A summary of the completed performance metrics is shown in Table III.5.5.

Metric	Metric goal	Average metric performance
Total consumer cost reduction	>10% compared to no coordination	11% per consumer. Can increase with more storage capacity.
Ramp tracking error	<10%	8.99%
Regulation tracking error	<10%	9.27%

Table III.5.5: Summary of metrics for subtasks 5.1 and 5.2, applied to subtask 5.3

HL simulation setup

The HIL simulations use the distribution grid simulation suite and the 2-layer bounds algorithm developed and discussed in subtask 5.1 and published in [5.2, 5.6], and the Bounded Signal Following scenario described for subtask 5.2. A brief overview of the simulation methodology is as follows: (1) place DERs across the nodes in the

distribution grid, (2) determine maximum power ramp and regulation signals that can be followed by the aggregation of DERs located at each node with and without bounds, (3) run signal tracking algorithm for the aggregation of DERs within each node, (4) evaluate the cost savings and signal tracking error. The HIL component falls within step (3) of the simulation methodology, where a single home is replaced with the HIL system. First, the HIL controller is assigned the portion of the ramp or regulation signal to track, a baseline power profile, and its cost minimization objective function. Additionally, the HIL controller has been tuned to its associated device characteristics such as its efficiency and power operating region. The controller then runs the signal tracking algorithm for the first time-step in the given signal tracking simulation horizon and sends the power setpoint to the HIL system. The HIL system will then adjust the power setpoint for the DER to match that requested by the controller and hold that setpoint for the duration of the timestep (5 seconds). Finally, the HIL system will measure the device power injection and SOC if applicable and send the information to the controller, which will update its signal tracking algorithm and repeat for the next time-step.

The data used for the baseline device power consumption profiles are from real consumer devices from Pecan Street. The HIL controller was implemented on a Raspberry Pi, which communicated to either a smart switch for the space heater, large lights, refrigerator, or a battery inverter. The computation time for the signal tracking algorithm is <0.03 seconds and the communication delay for the smart switch response was approximately 0.5 seconds, demonstrating suitability for real time control.

Figure III.5.4 shows a sample of a ramp down signal being tracked by the space heater with a fan. Since the space heaters are discrete loads with only on/off capability, the signal tracking controller splits the ramp signal into a binary series of on and off commands. This is repeated for the other hardware devices tested in the HIL setup. The total network simulation includes 13000 other devices that follow similar commands from the signal tracking algorithm, and the aggregate of these devices and the HIL power profiles form the total network signal tracking as seen on the right plot in Figure III.5.4. The tracking error from the HIL test is compared to an equivalent scenario in the software simulation to determine the additional error due to the hardware test. This error is extrapolated to the other virtual devices in the simulation use case to get the aggregate signal tracking error. Across all hardware devices and tested signals, the average ramp signal tracking error was 8.99% compared to 2.64% in the software simulation.

For the regulation signal tracking HIL tests, the signal tracking algorithm also splits the regulation signal into a series of on/off commands to be followed by the HIL system, and the rest of the devices proceed similarly. The sources of tracking error are the same as for the ramp signal tests. Across all hardware devices and tested signals, the average regulation signal tracking error was 9.27% compared to 6.13% in the software simulation.

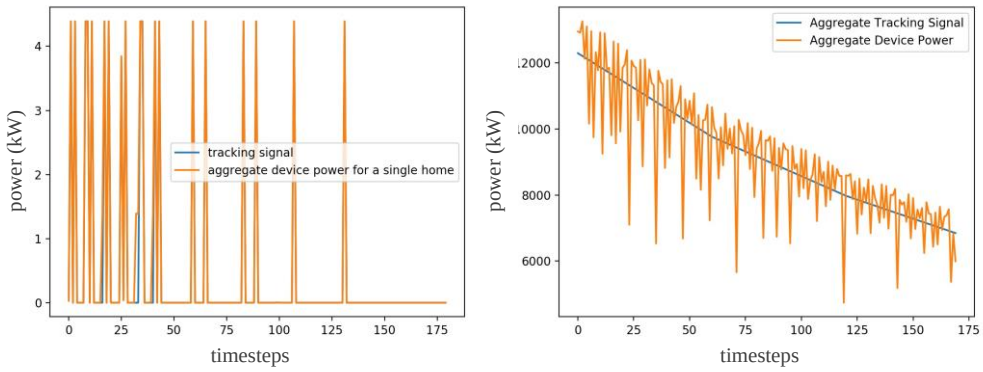

Figure III.5.4: Left – HIL system tracking signal and space heater with a fan power profile. Right – Aggregate network ramp down signal tracking.

The cost reduction for the consumers in the network is based on energy cost savings from arbitrage on the TOU electricity tariff. For energy arbitrage, the maximum cost reduction is achieved by charging all DERs to their maximum capacity before the peak price period and discharging fully during the peak price period. As long as the energy is charged and discharged over this period, the exact signal accuracy does not impact the cost savings. Thus, the signal tracking error introduced by the HIL setup does not noticeably impact the total consumer side cost reduction, which is still 11% on average.

Task 6: *Scalable Distributed Privacy for Information and Energy Exchange*

Weft follows the same security model as Prio.[1] The resource operator has a server, responsible for controlling the energy resource and sending commands to it. There is at least one additional server, operated by a third party, e.g., the Internet Security Research Group (ISRG)[2]. Weft assumes the third party does not collude with the resource operator. This is a practical assumption, as organizations such as the Internet Security Research Group provide this as a service. For example, in a deployment of secure aggregation used for COVID-19 exposure notifications, the servers were controlled by the NIH and ISRG. The clients know their own requests and the state of their partition of the energy resource, but they should not learn any information about

---

[1] *Henry Corrigan-Gibbs and Dan Boneh. Prio: Private, robust, and scalable computation of aggregate statistics. CoRR, abs/1703.06255, 2017.*

[2] *Introducing isrg prio services for privacy respecting metrics. https://www.abetterinternet.org/post/introducing-prio-services/, Nov 2020.*

other clients. The servers should learn whether each request is valid and the sum of the power across all client's requests, but no other client information. Weft assumes that the clients and servers communicate over authenticated, encrypted channels.

The clients know their own requests and the state of their partition of the energy resource, but they should not learn any information about other clients. The servers should learn whether each request is valid and the sum of the power across all client's requests, but no other client information. Weft assumes that the clients and servers communicate over authenticated, encrypted channels.

Servers are trusted for request integrity, meaning they will not modify client requests. However, servers are untrusted for request privacy, meaning they may try to learn information about client requests. This information may be from the request itself, by relating different requests, from the timing of requests, etc. Clients are trusted for neither integrity nor privacy; they may attempt to learn other client's information or submit malformed requests.

Like prior work on secure aggregation in the non-colluding third-party model, Weft uses secret sharing to preserve client privacy, following the standard share-aggregate-reveal paradigm in Prio. A client submits a power request by splitting the request into secret shares and sending one share to each server. Using the additive homomorphism of the sharing scheme, the servers aggregate the shares of different clients, hiding the client's private data in the sum. The resource operator recombines the aggregated shares to learn the true aggregate.

Standard RESTful APIs for energy resources allow clients to control a resource by setting its charge/discharge value. This value replaces the prior one. While this can be achieved with secret shares (simply cache values and reaggregate on a new value), it leaks information and therefore violates privacy. Whenever a request comes in from a client, the resource operator sees how the aggregate changes and can compute how the client's request changed.

To avoid this information leak, Weft encodes requests as schedules, an array of power values, similarly to prior work.[3] Each element in the schedule specifies the value for a time interval. The energy resource defines this interval and the length of a schedule. For example, a schedule with an interval of one minute and a schedule of one day would consist of 1,440 values. Shortly before the start of each epoch, a client sends its schedule for that epoch. The client generates a secret share of its schedule by secret sharing each value. Schedules do not leak information because they decouple changes from the timing of messages, and can be aggregated by the servers element-wise. To ensure the client request is valid, the servers must check two things: a rate constraint

[3] *NanWang, Sid Chi-Kin Chau, and Yue Zhou. Privacy-preserving energy storage sharing with blockchain and secure multi-party computation. CoRR, abs/2111.02005, 2021.*

and an integral constraint. More formally, for some minimum (possibly negative) rate n, some maximum rate m, and a maximum usage e, these constraints over a schedule S are:

1. *Rate Constraint:* $n \leq S[i] \leq m$ *for all* $0 \leq i \leq |S|$.
2. *Integral Constraint:* $0 \leq \sum_{i=0}^{j} S[i] \leq e$ *for all* $0 \leq j \leq |S|$.

The minimum rate, maximum rate, and maximum usage may all be different values for each client, and we assume that they are known to the servers and resource operator.

To ensure that client requests are valid, the system uses zero-knowledge range proofs over secret shared data and their integrals. The client constructs zero-knowledge proofs over their schedule S, which proves that S satisfies both the rate and integral constraints. They send the proofs to the servers, who jointly verify them over their secret shares. If the proofs are valid, the servers are convinced that S satisfies both constraints. If any proof is invalid, the servers reject S.

Zero-knowledge proofs

There are three techniques for the proof. The first proof is based on bit-splitting. This type of proof can be extended to support arbitrary ranges. The size of these proofs is dominated by the bits being sent, because Prio requires them to be encoded as a 32-bit value. Empirically, we find that sending 100,000 16-bit values requires ~6.4 MB of data sent from the client to each server.

The second proof is based on sorting and can decrease the amount of client to server communication when sending large power schedules with smaller values. The client first creates a second version of their schedule, by appending all values from 0 to 2^b, and sorting. The client sends the encoded version of their original schedule, and this modified version, to the servers. They also send a proof that the two schedules are equivalent, and that the second schedule is correctly sorted. The servers check these proofs and reveal the first and last value in the sorted schedule. If these values are 0 and 2^b, all the values must be in range. For this proof technique to be efficient, the bit-width of the values must be small, and the size of the schedule must be large. Empirically, we find that sending 100,000 16-bit values requires ~3.0 MB of data sent from the client to each server.

The third technique is an extension of techniques from another work called Prio+. However, this requires extra computation and communication between the servers to convert each bit back into a 32-bit value so the Prio proofs can be verified. This reduces client to server communication to ~1.9 MB but incurs ~6 MB communication cost between the two servers.

Results

Using the three proof techniques, Weft can support clients with limited computation, limited memory, or limited communication. We measure client costs on a desktop-class machine as well as a microcontroller-based embedded system. We measure server costs on only the desktop.

The embedded system has an nRF52840 microcontroller, a system-on-a-chip (SoC) integrating a 64 MHz 32-bit Cortex-M4 microcontroller with a Bluetooth 3.0/802.15.4 radio [5]. The system has 1MB of flash for code and 256kB of RAM. The desktop has an Intel i9-129000KF processor with 8 performance and 8 efficiency cores. Each performance core can run at up to 5.2GHz. The system has 30MB of shared L3 and 14MB total L2 cache. Main memory is DDR4-3200.

The primary variable we manipulate is the schedule size, in terms of number of values. Larger schedules can cover longer periods of time or have finer time resolution. A 1,440 element schedule covers a day at 1 minute granularity; an 5.760 element schedule is every 15 seconds, while 86,400 is every second.

To prove a schedule with 10,000 elements, the sorting strategy uses a peak of 2.4MB of RAM, the bit-splitting strategy uses 191kB. Figure 3b shows that commitment proofs require 524MB of RAM for the same schedule. Bit-splitting is the most RAM-efficient approach, using only 8% the RAM of sorting and 0.04% the RAM of commitment proofs. The 191kB required for bit-splitting approaches the RAM available on the nRF52840. In practice, its 256kB has to be shared between applications, the kernel, and networking stack. In our execution environment, an application is limited to 96kB of RAM. This is sufficient to prove a 4,000 element schedule, so a day-long schedule with an interval of 22 seconds. A 1,440 element schedule uses 45kB.

Peak memory has stepwise growth. For commitment proofs, this because of a requirement that the number of proofs being aggregated is a power of two. For the bit-splitting and sorting strategies, this is because libprio internally uses power of two sized Fast Fourier transforms (FFTs).Bit-splitting's bumps are smaller because the FFT for a schedule size n scales with $\sqrt{n}$ from the G-gate optimization.[4]

A Prio-based system can prove the validity of a 10,000 element schedule with the bit-splitting strategy in 0.36 seconds. Using the sorting strategy, it can prove the same schedule in 0.09 seconds. Server verification of the two Prio-based proofs strategies takes 0.085 and 0.065 seconds respectively. Proving time generally scales linearly, though there are some jumps when the FFT size goes to the next power of two.

---

[4] Dan Boneh, Elette Boyle, Henry Corrigan-Gibbs, Niv Gilboa, and Yuval Ishai. Zero-knowledge proofs on secret-shared data via fully linear pcps. Cryptology ePrint Archive, Paper 2019/188, 2019. https://eprint. iacr.org/2019/188.

Using commitment proofs (Figure 4b), the most computationally intensive approach, a client can prove the validity of its schedule in 73.59 seconds, while it takes a server 9.99 seconds to verify the proofs. Unlike bit-splitting and sorting, which scale smoothly, commitment proofs have a step function. This is because of a Bulletproofs requirement (which commitment proofs extends) that aggregated proofs are a power of two; the step function is when the number of proofs bumps to the next power of 2.

These schedules are huge: with 1m intervals, 10,000 elements is 6.94 days. For a one-day schedule with 1m intervals, a client takes 0.027 seconds to prove with bit-splitting and 0.011 seconds to prove with sorting. A server takes 0.004 seconds to verify these proofs.

A schedule of 1,440 elements takes 66s to compute for a bit-splitting client running as a Tock process on the nRF52840 microcontroller, while a maximum-sized schedule of 4,000 elements takes 4.5 minutes; as the schedule is for a day, this is an acceptable cost. A microcontroller-based embedded client has sufficient resources to generate proofs for schedules using Prior and the bit-splitting strategy.

Commitment proofs are the most communication efficient strategy, sending 41.5 KB for a schedule of 10,000 values. This is close to optimal – the values in the schedule themselves require 40 KB of data. For a 1,440 element schedule, commitment proofs send 7.1kB. The sorting strategy requires 0.66 MB for a 10,000 element schedule and 159kB for a 1,440 element schedule. The bit-splitting strategy, which can execute on an embedded microcontroller, requires the most communication. It sends 1.13MB for a 10,000 element schedule and 183kB for 1,440 elements. Sending a private schedule once a day is too expensive for LPWAN protocols such as LoRaWAN, but NB-IoT or LTE-M could support it and short-range protocols such as WiFi or Thread could easily support it.

The sorting strategy’s communication cost has a step function due to the FFT size scaling to the next power of two. Bit-splitting and commitment proofs do not see this behavior because the communication cost is dominated by the secret shares of the data they transmit (bits and values), which scale linearly with the schedule size.

Task 7: Use Cases

The overarching approach used for Task 7 encompassed three steps: 1) examining academic literature to identify potential use cases, 2) collaborating with the Task 2 team to refine use cases as they were tested, and 3) discussing with industry partners about their most relevant needs for DERs.

In the literature, the idea of a community battery or shared DERs appeared as a way for all community members to benefit from DERs [1]. BAL, our virtualization software, can readily support this use case. However, BAL’s real advantages lie in satisfying additional system needs such as flexibility and adaptability. If a collection of batteries

grows, or a battery suddenly goes offline, BAL can adapt to this new situation since it virtualizes all system components.

After implementing the virtualization software in Task 2, including demonstrating aggregation, partitioning, and dynamic partitioning use cases in a physical system, we began to refine the use cases to satisfy specific objectives. We first prioritized the objective of utilizing DERs for cost minimization of electricity bills for individual homeowners and a collective of homeowners. Once we had implemented that algorithm, we added to that objective by beginning to consider protecting neighborhood transformers using battery aggregation.

To finalize the list of use cases, we discussed with collaborators such as VMware, the US Navy, and PG&E. These groups emphasized the need for reliable, resilient, and shareable DER systems. Specifically, VMware's campus microgrid has an "islanding" mode, in which the battery must serve critical loads first and shed expendable loads when the microgrid was isolated. We met this need by developing a use case that partitions the battery into separate sections each meant for a specific type of load, sorted by priority.

Overall, for Task 7, we leveraged literature, academic, and industry sources to formulate our use cases, and we were able to iterate on the cases once we tested them in Task 2.

IV. Accomplishments and Conclusions

Task 2*: Resource Virtualization*

Our major accomplishments under Task 2 are:

- Demonstrating it is possible to virtualize battery energy storage systems by aggregating and partitioning them,
- Developing, evaluating, and testing the algorithms and policies for how aggregate and partitioned batteries behave,
- Releasing an open-source implementation of the Battery Abstraction Layer that can control both local (e.g., through a direct BMS interface) and Internet-connected batteries (e.g., through a RESTful web API), and
- Demonstrating how battery virtualization and the BAL can build new, flexible use cases for battery energy storage, including cost savings and protecting transformers from EV charging.

Together, these satisfy the objectives for the task. The Battery Abstraction Layer has been designed and implemented. The Battery Abstraction Layer was deployed on a battery testbed including both in-use residential batteries and batteries in lab settings, demonstrating new use cases for battery systems. Finally, we have demonstrated how the Battery Abstraction Layer can be integrated with solar charging, using batteries to protect local transformers from EV charging spikes. The completion of this final objective is described in a submitted manuscript, titled “Extending Transformer Lifetime with Distributed Battery Control”.

Task 3*: Secure ID for Asset Authentication*

We have developed a system architecture of Secure ID, which is a set of protocols, algorithms, distributed identity store, smart contracts, and off-chain API endpoints that together enable distributed identity management while preserving privacy of nodes participating in the distributed network. Each device that intends to have a Secure ID credential must do so by running as a node on the Secure ID’s blockchain network. As a participating node on the blockchain, the device has access to an off-chain API that allows it to invoke smart contracts that perform various identity-related operations on the network, most notably identity verification (based on Keychecker). The current work is developed on Ethereum test network with smart contracts implemented in solidity programming language. Figure IV.3.1 shows various components (system architecture) of the Secure ID system:

1. Command line interface
   The edge device interacts with Secure ID via a command-line interface that communicates with the network via a collection of off-chain API endpoints.
2. Consent and Control API
   When building a decentralized self-sovereign identity, consent must be incorporated into the system. This ensures whether identity data will remain

private or not. Through deliberately designed consent, each transaction of identity data (represented by an API invocation request) will only be executed with the edge devices' consent. This creates full device autonomy–in which they will have full control of where and how their identity data is shared or exchanged. The consent and control API offers basic identity management system operations such as registration, deregistration, and verification. Consent and control layer is designed to demonstrate how fundamental identity operations can be executed while maintaining identity sovereignty. The API layer is not designed to comprehensively support all identity-related operations.

3. Serverless functions
   API requests from Consent and Control API are dispatched to a layer a serverless functions that act as proxy for smart contract functions on the Secure ID blockchain.
4. Smart Contracts
   Identity verification, registration, and deregistration are implemented as smart contracts. Smart contract function layer is a code-based implementation of the Keymaker algorithm.
5. Event Log DB
   All events in the off-chain system are logged in an Event Log DynamoDB for troubleshooting and forensic purposes.

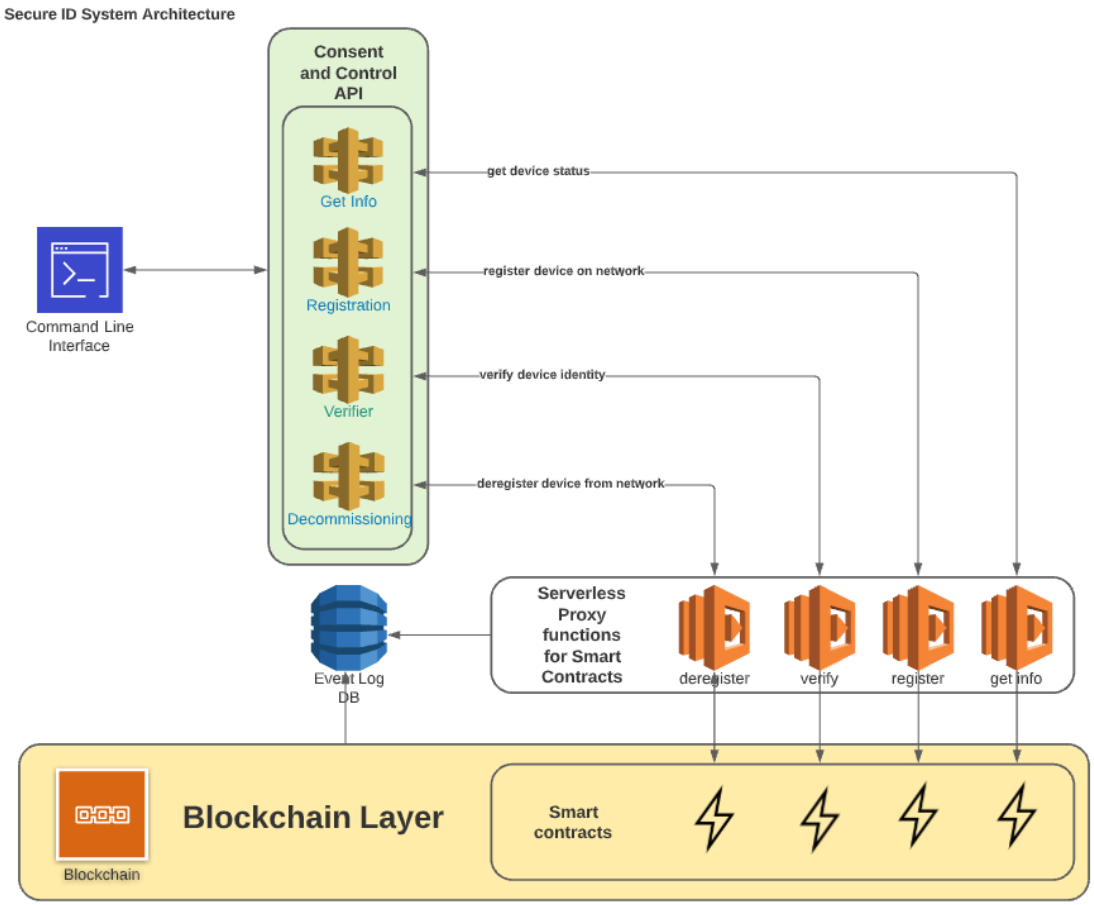

Figure IV.3.1: Secure ID System Architecture

Task 3.4: Lab testing of the Proof-of-Concept

We implemented the above architecture using AWS cloud resources and a Raspberry Pi microcontroller. We deployed Keymaker and Keychecker smart contracts on 4x Elastic Cloud Compute (EC2) instance with each instance emulating a DER asset alongside a Raspberry Pi 4 microcontroller that acted as a proxy for Sonnen Battery installed in SLAC Gismo Lab. Each of these nodes ran as a node on a Hyperledger Fabric (HLF) Test Network. 3 of the 5 nodes ran as verifiers. One node served as an orderer enabling “ordering service” of HLF, which literally orders transactions into blocks so the peer nodes can then validate and commit it to their ledgers. One Certificate Authorities (CA) node to connect the whole network via certificate's chain of trust. Each of the peer nodes had its own Couch DB instance for data persistence. The composition of the HLF testnet is listed below:

Deployed Hyperledger Fabric (HLF) Testnet

- 5 nodes (1x Raspberry Pi 4, 4x EC2 Instances)
- 3 verifiers (3x EC2 Instances)
- 1 orderer node (1x EC2 Instance)
- 1 CA node (1x EC2 Instance)
- 5 Couch DB Instances (1 for each of the 5 nodes)

The test comprised of running the following Smart Contracts in the environment detailed above:

Contract Name	Purpose	Number of Runs
Provision_Identity	Created the identity key-pair without any association to a given device	5 (one identity key pair for each peer on the HLF network)
Keymaker	Assigns an identity key-pair to a given device and performs key sharding using the Keymaker protocol	5 (one identity key pair associated with each peer on the HLF network)
Distribute	Shares 4 shards per key across the network	5 (shards distributed for each peer on the HLF network)
Keychecker	Collects 3 shards per verification request and performs identity verification using the Keychecker protocol	300 (60 verification runs for each peer on the HLF network)

Table IV.3.1: Environment for lab testing of the proof-of-concept of secure ID

Running 300 executions of the Keychecker smart contract resulted in several notable outcomes. First and foremost, the smart contract was able to accurately verify the identity of each device involved in the executions, without any instances of false positives or false negatives. This suggests that the contract is highly reliable and effective at its intended purpose. Additionally, the executions were completed quickly and efficiently, with minimal delay or error. We observed an average execution time of 2.67 seconds for the Keycheker protocol and an average latency of 5.48 seconds indicating that the smart contract is well-designed. Given this was meant to serve as a proof-of-concept for the Secure ID system, we did not run stress tests to demonstrate how the system performs as we scale up the number of device. Overall, the results of these executions demonstrate the feature-completeness of the Secure ID system and how it may be used to in real-world environments allowing DERs and utility-owned grid assets to communicate securely over a decentralized and untrusted network.

Task 5*: Private and Safe Integration*

Subtask 5.1 focused on the design and validation of a DER cooperation scheme which ensures network reliability. To evaluate the impact of DERs on network reliability and validate the proposed coordination scheme, a grid simulation suite was designed. This grid simulation suite combines distribution system simulation in OpenDSS with datasets for residential/commercial power consumption, PV generation, and EV charging profiles. This simulation suite allows us to evaluate metrics for the number of grid upgrades needed to sustain future quantities of DERs under different scenarios.

A DER coordination scheme was designed to reduce the number of grid upgrades needed to support the growing penetration of DERs. The coordination scheme incorporates a global day ahead scheduler which ensures that voltage and transformer constraints are respected, and local controllers for managing local DERs. Data-driven proxies of the power network model were used in the DER coordination scheme. The proposed coordination approach was compared to a scenario with no coordination, showing a greater than 10% benefit in terms of the avoided costs of transformer upgrades.

Subtask 5.2 focused on extending the coordination capabilities to ramping, regulation and black start applications. A signal tracking algorithm was designed for ramping and regulation. For ramping signals, the average signal tracking accuracy of the proposed approach was 2.6%, and for regulation signals the accuracy was 6.1%, exceeding the task goals of <10% signal tracking accuracy. Results also show an average cost savings for the consumer of 11% from signal following (which can improve with additional storage capacity). For black start capabilities, we investigated how stationary storage, EV chargers and rooftop PV can provide the grid with power during a blackout. We find that the uptime hours during a blackout of the stationary storage, EVs, and PV is greater than 4 hours 31.4% of the time, which means the project metric for >4 uptime hours during a blackout can be met. Finally, we examined the impacts of regulation signal following on battery lifetime. Results show that battery system lifetime can

improve by 10% by ignoring a certain fraction of regulation signals which provide the least profit.

Subtask 5.3 focused on testing the proposed system on a hardware-in-the-loop (HIL) setup using the same performance metrics from subtask 5.1 and 5.2. The HIL controller was implemented on a Raspberry Pi, which communicates with different hardware devices. Across all hardware devices and tested signals, the average ramp signal tracking error was 8.99% compared to 2.64% in software simulation, and the average regulation signal tracking error was 9.27% compared to 6.13% in simulation. In both cases, results satisfy the performance metric objectives for this task.

Task 6*: Scalable Distributed Privacy for Information and Energy Exchange*

Weft is a cryptosystem that provides distributed privacy for a shared battery energy system. It builds on the ideas in Task 2, providing privacy mechanisms for a partitioned battery. Using modern cryptographic techniques, Weft is able to provide strong privacy guarantees to clients. Using Weft, the controller of a partitioned battery learns that each client's request is valid, the sum of their requests (the actual power output of the battery), and the sum of their battery partitions (the actual energy capacity of the battery).

This privacy has limitations; for example, if the battery is discharging at its maximum rate, the owner knows that every partition is discharging at its maximum rate. Similarly, if the battery is fully charged, the owner knows every partition is fully charged. This information leakage is fundamentally unavoidable, as the owner must generate commands to its own battery. Therefore, Weft meets the Client Privacy objective as stated in the Objectives section for Task 6.

Using zero knowledge proofs, Weft is able to verify that values provided by clients are valid and correct without learning their values. It meets the Server Integrity objective.

Weft demonstrates how very lightweight, modern cryptographic mechanisms can be used to provide private and sound control of a shared battery resource. The privacy component of Weft – additive secret shares – is trivial computationally. Its overhead is entirely from its need for proofs across those private values. Weft meets the computational efficiency objective.

Task 7*: Use Cases*

The accomplishments for Task 7 included compiling a list of relevant use cases for DERs and forming an industry advisory board. The industry advisory board had members from various universities as well as a variety of companies: VMware, Hitachi, AutoGrid, Siemens, and Nevermore Security. We hosted an advisory board meeting in which we discussed the most pertinent problems for DER owners. Some takeaways were that flexibility and privacy are of utmost importance to users. If DERs are shared

among homeowners, it is crucial that specific homeowners' data remains secure and private.

Throughout the work in Task 7, we came to two main conclusions about the nature of DER operations in the real world. One conclusion stems from our collaboration with VMware, whose Palo Alto campus contains two 1 MWh batteries, a solar photovoltaic installation, and the ability to operate independently as a microgrid. Given the campus' various needs of resiliency, 24/7 carbon-free power, and electricity bill reduction, our battery abstraction layer (BAL) virtualization software has the potential to help the company. In spring of 2023, we visited the campus to learn more about the battery and microgrid operation. One major takeaway from the visit was that when the microgrid was installed, VMware was promised that it would come with software that would maintain operation autonomously. However, in reality, a VMware employee must monitor the system most of the time and reset it after a fault. In addition, the batteries often discharge during the day and are not utilized to provide carbon free power, which would best happen by charging via the solar array during the daytime and discharging in the evening. This conclusion, that physical systems are plagued by reliability issues, is something that we have focused on in our software implementation.

In addition to industrial needs, we have also identified a second conclusion from the unique needs of military installations. During the project, we met with members of the Department of Defense and the Navy to discuss how the Stanford team could assist with their DER development needs. From this meeting, we learned that the military bases are committed to decarbonizing but lack the hardware and software expertise to install microgrids and associated components themselves. They are also deeply committed to using DERs, and especially batteries, for resiliency both on stationary installations and for expeditionary forces. We learned that this focus on resiliency is key: even more so than emissions or cost minimization, military installations need backup systems to run during a grid outage.

In summary, we have identified that resiliency and reliability are key concerns for both industry and military partners. Virtualization software can be a critical component in resiliency efforts. Companies and the military generally will need to use their batteries for other cost-saving methods instead of just keeping a battery fully charged waiting for a grid outage. Thus, using virtualization to first aggregate a set of batteries and then create partitions for resiliency and cost-savings is a useful commercialization pathway.

To meet the objectives of the subtasks of Task 7, we first identified the resiliency and reliability use cases for our various potential test sites, including VMware and the Naval Installation (7.1). For 7.2, we formed an industry advisory board and hosted a meeting. Lastly, for 7.3, we worked with the electric utility company and the Navy to identify commercialization opportunities. The papers resulting from Task 2 were also outputs of the work completed in Task 7.3.

Our future work includes continuing to implement the use cases in Task 2, with a focus on transformer protection. Our use cases have the potential to be a foundation for many

additional DER implementations in smart grids that include electric vehicles, distribution grid resiliency, and flexible loads.

APPENDIX A: Product or Technology Production

Task 2: *Resource Virtualization*

Publications

https://sing.stanford.edu/site/publications/100 (Software defined grid energy storage)

Software code

https://github.com/Stanford-New-Energy-Systems/BatteryOS

Task 3: *Secure ID for Asset Authentication*

Project Resources

We have created a repository of additional resources to share code, documentation, and demo of the Secure ID system.

https://github.com/orgs/Secure-Identity/repositories

Item	URL
Github Organization for Secure-ID	https://github.com/orgs/Secure-Identity/repositories
Hyperledger Fabric Network v1 (single DB instance for all peers)	https://github.com/Secure-Identity/HyperledgerFabricNetwork
Hyperledger Fabric Network v2 (separate CouchDB instances for each peer)	https://github.com/Secure-Identity/HLFNetwork-V2
Implementation of Keymaker protocol in Go Lang	https://github.com/Secure-Identity/keymaker
Handoff documents including architecture diagrams, videos, project poster, presentations, and technical documentation	https://github.com/Secure-Identity/TrustDER-handoff
Overview of Secure ID (video)	https://www.youtube.com/watch?v=9XgmZtdlSBQ

Task 5: *Private and Safe Integration*

Publications

Navidi, T., El Gamal, A., & Rajagopal, R. (2023). Coordinating distributed energy resources for reliability can significantly reduce future distribution grid upgrades and peak load. *Joule*, 7(8), 1769-1792.

Navidi, T., Gamal, A. E., & Rajagopal, R. (2023). Coordination of DERs for Grid Reliability via Day-ahead Demand-Supply Power Bounds. arXiv preprint arXiv:2307.00188.

Navidi, T. (2023). Coordination of Distributed Energy Resources for Distribution Grid Reliability (Doctoral dissertation, Stanford University).

Task 6*: Scalable Distributed Privacy for Information and Energy Exchange*

Publications

https://sing.stanford.edu/site/publications/104 (Privacy-Preserving Control of Partitioned Energy Resources)

Software code

https://github.com/emlaufer/battery-privacy

Task 7*: Use Cases*

Publications

Sonia Martin, Obi Nnorom, Oskar Triebe, Liana Patel, Philip Levis, and Ram Rajagopal. 2022. Software defined battery-solar systems: poster abstract. In Proceedings of the 9th *ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (BuildSys '22). Association for Computing Machinery, New York, NY, USA, 291–292. https://doi.org/10.1145/3563357.3567750

Sonia Martin, Nicholas Mosier, Obi Nnorom, Yancheng Ou, Liana Patel, Oskar Triebe, Gustavo Cezar, Philip Levis, and Ram Rajagopal. 2022. Software defined grid energy storage. In Proceedings of the 9th *ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation* (BuildSys '22). Association for Computing Machinery, New York, NY, USA, 218–227. https://doi.org/10.1145/3563357.3564082

REFERENCES

Task 5: *Private and Safe Integration*

[5.1] Li, H., Wert, J. L., Birchfield, A. B., Overbye, T. J., San Roman, T. G., Domingo, C. M., ... & Palmintier, B. (2020). Building highly detailed synthetic electric grid data sets for combined transmission and distribution systems. *IEEE Open Access Journal of Power and Energy*, 7, 478-488.

[5.2] Navidi, T., Leblanc, C., El Gamal, A., & Rajagopal, R. (2020, November). DER information unaware coordination via day-ahead dynamic power bounds. In 2020 *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids* (SmartGridComm) (pp. 1-6). IEEE.

[5.3] Bernstein, A., Bouman, N. J., & Le Boudec, J. Y. (2017, December). Real-time control of an ensemble of heterogeneous resources. In 2017 *IEEE 56th Annual Conference on Decision and Control* (CDC) (pp. 1401-1406).

[5.4] Navidi, T., El Gamal, A., & Rajagopal, R. (2019, August). Co-optimizing energy, grid services and voltage support in networks with distributed storage. In 2019 *IEEE Power & Energy Society General Meeting* (PESGM) (pp. 1-5).

[5.5] Millner, A. (2010, September). Modeling lithium ion battery degradation in electric vehicles. In 2010 *IEEE Conference on Innovative Technologies for an Efficient and Reliable Electricity Supply* (pp. 349-356).

[5.6] Navidi, T., El Gamal, A., & Rajagopal, R. (2023). Coordinating distributed energy resources for reliability can significantly reduce future distribution grid upgrades and peak load. *Joule*, 7(8), 1769-1792.