APPROVED

Version 1.2

March 29, 2021

Blockchain to Record Private Key Properties in DER Equipment

Abstract

Blockchain Work Group specification, defining use cases, analyzing threats, developing governance structures and providing technical requirements for using blockchain technology to secure connected device credentials used in energy communications.

License Agreement and Copyright Notice

Attribution

This specification was developed by the SunSpec Alliance Blockchain Workgroup. This workgroup includes representatives from the following companies:

See list of contributors in Appendix A.

Contact Information

SunSpec Alliance
4040 Moorpark Avenue, Suite 110
San Jose, CA 95117
info@sunspec.org

Revision History

Revision	Date	Reason
1.0	21-12-2020	Adopted Specification. Public version.
1.1	16-02-2021	Remove references to SunSpec as the host of the Governing Body.
1.2	29-03-2021	Added statement to About the SunSpec Specification Process to indicate that APPROVED specifications are considered to be Adopted Specifications according to the SunSpec Membership Agreement and Bylaws.

About the SunSpec Alliance

The SunSpec Alliance is a trade alliance of developers, manufacturers, operators and service providers, pursuing open information standards for the Distributed Energy industry. SunSpec Alliance standards address operational aspects of PV, storage and Distributed Energy power plants on the smart grid—including residential, commercial, and utility-scale systems— thus reducing cost, promoting innovation, and accelerating industry growth.

Global leaders from Asia, Europe, and North America are members of the SunSpec Alliance. Membership is open to corporations, non-profits, and individuals. For more information about the SunSpec Alliance, or to download SunSpec specifications at no charge, please visit **www.sunspec.org**.

About the SunSpec Specification Process

SunSpec Alliance specifications are developed by SunSpec member companies seeking to establish industry standards for mutual benefit. Any SunSpec Alliance member can propose a technical work item. Given sufficient interest and time to participate, and barring significant objections, a workgroup is formed. Workgroups meet regularly to advance the agenda of the initiative.

The typical output of a workgroup is a SunSpec interoperability specification. SunSpec interoperability specifications are considered to be normative, meaning that there is a matter of conformance required to support interoperability. The revision and associated process of managing these documents is tightly controlled. Other SunSpec documents developed by workgroups are informative, and provide recommendations regarding best practices, but are not a matter of conformance. Informative documents can be revised more freely and frequently to improve the quality and quantity of information provided.

SunSpec interoperability specifications follow this lifecycle pattern of **DRAFT**, **TEST**, **APPROVED** and **SUPERSEDED**. **DRAFT** specifications are early works in process and are purely informative. **TEST** specifications are advanced works in process that are subject to modification pending feedback from implementors. **APPROVED** specifications are normative and represent the state of the art. **SUPERCEDED** specifications are also normative but are not state of the art. **APPROVED** specifications are backward compatible with **SUPERCEDED** specifications.

APPROVED specifications are considered **Adopted Specifications** according to the SunSpec Alliance Membership Agreement and Bylaws.

For more information or to download a SunSpec Alliance specification, go to http://sunspec.org/about-sunspec-specifications/.

Table of Contents

1 Executive Summary 6

2 References 8

3 Definitions and Abbreviations 10

3.1 Definitions 10
3.2 Abbreviations 11

4 Introduction 12

5 Use Cases 13

5.1 Introduction 13
5.2 Actors 14
5.3 Main Use Case 17
5.3.1 Current State-of-the-Art 17
5.3.2 Problems with Current Solution 18
5.3.3 New High-Level Architecture 18
5.3.4 Main Use Case Description 19
5.3.5 Performance 21
5.4 Detailed Use Cases and Activities 22
5.4.1 Audit Creation 22
5.4.2 Manufacturing and Key Provisioning 25
5.4.3 Certificate Generation 26
5.4.4 Communication Security 29
5.4.5 Key Life Cycle Tracking 30
5.5 Supply Chain and Distribution Aspects 32
5.5.1 Secure Component provisioned by Manufacturer 33
5.5.2 Secure Component provisioned by Service Provider 33
5.5.3 Secure Component provisioned by OEM 34
5.5.4 Component Integration by OEM 35
5.5.5 Key Provisioning by Applicance at Manufacturing 35
5.5.6 Key Provisioning by Service Provider at Manufacturing 36
5.5.7 Key Provisioning by Service Provider at Installation 37
5.6 Use Case and Activity level Security Threats 38
5.6.1 Nation State Coercion 38

5.6.2 Malicious Accessors 38
5.6.3 Unrecorded Process Change 39
5.6.4 Incorrect Audits 39
5.7 Requirements and Recommendations 39
5.7.1 Eco-System Requirements 39
5.7.2 System Architecture Requirements 40
5.7.3 System Performance Requirments 40

6 Blockchain 42

6.1 Blockchain Architecture 42
6.1.1 Permissionless blockchain 42
6.1.2 Permissioned blockchain 43
6.1.3 Comparison: Permissioned vs. Permission-Less 44
6.2 Blockchain Transactions 45
6.2.1 Fixed Transactions 45
6.2.2 Smart Contract 45
6.2.3 Comparison: Fixed Transactions vs. Smart Contract 45
6.3 Blockchain-level Security Threats 46
6.3.1 What to Protect - Main System Assets 46
6.3.2 Who are the “black hats” – Attacker profiles 47
6.3.3 Attack Scenarios 47
6.4 Requirements and Recommendations 49
6.4.1 Blockchain Architecture 49
6.4.2 Consensus Protocol 49
6.4.3 Blockchain Transactions 49

7 Governance 50

7.1 Governance Domains 50
7.1.1 System Setup 50
7.1.2 Operations 50
7.1.3 System Updates 50
7.2 Governance Mechanisms 50
7.2.1 Ad Hoc 51
7.2.2 Governing Entity 51
7.2.3 Contractual 51
7.2.4 Decision Implementation 51
7.3 Governance-Level Threats 52
7.3.1 Shared Interest Threats 52

7.3.2 Government Coercion 52
7.3.3 Company Infiltration 52
7.4 Consensus Node Governance 53
7.4.1 Maximum consensus nodes 53
7.4.2 Minimum consensus nodes 53
7.5 Requirements and Recommendations 53
7.5.1 Governance Domains 53
7.5.2 Governance Mechanisms 54

8 Data Model 55

8.1 Introduction 55
8.2 Confidentiality Considerations 55
8.3 Data Elements 56
8.3.1 Key Data Element 56
8.3.2 Process Data Element 56
8.3.3 Assessment 57
8.3.4 Entity 58
8.3.5 Data Model 58
8.4 API 59
8.5 Requirements and Recommendations 60

9 Roadmap for Implementation 61

9.1 A Single System 61
9.2 Governing Body 61
9.3 Remaining Specification Work 61
9.4 Implementation and Launch 62
9.5 Regulations and Certifications 62

10 Conclusion 63

APPENDIX A (Informative): List of Contributors 64

1 Executive Summary

Future power grids leverage Distributed Energy Resources (DERs), but the current industry practices and regulation are not sufficient to ensure that such critical resources are effectively protected from cybersecurity threats. A particular problem is that the communication keys (cryptographic private keys) used in DER devices can have very different levels of security and protection, but such important information is not easily available to communication endpoints such as IEEE 2030.5 utility servers. To address this problem, SunSpec Alliance created the SunSpec Blockchain Work Group and hosted its first meeting on October 29, 2019. In the following meetings the work group discussed how a blockchain-based key registry for DER devices can provide an easily accessible repository for querying security-critical information about DER devices and their keys, while at the same time provide high availability and strong integrity protection for the stored information. The present document is the final report of this work group.

The work group followed a typical standard creation process that starts with defining use cases and security threats before defining technological features and requirements. The main use case considered consists of manufacturers of hardware, key generators that create unique cryptographic keys, DER clients that store and use the keys, security assessors that perform process audits, DER servers that communicate with DER clients, and certificate authorities that sign certificates based on the cryptographic keys. Manufacturers utilize key generators to create private keys and provision them into DER clients. DER clients implement mechanisms to protect the keys. Assessors evaluate this process and store the evaluation on the blockchain. DER servers and certificate authorities access this information on the blockchain to make trust decisions and sign certificates. Section 5 describes multiple variations on this main use case.

The work group evaluated performance requirements for the main use case and determined that the blockchain should be able to address the needs of all DER device globally, which requires throughput of a billion transactions per year. The work group considered many security threats and malicious actors, including nation state actors with large resources and almost infinite patience. These actors have multiple attack vectors at their disposal, including coercion of companies in their jurisdiction. Security threats are described in sections 5.6, 6.3, and 7.3.

Given the potential threats, the work group settled on a permissioned architecture using a Byzantine Fault Tolerant (BFT) consensus algorithm. The work group elected to specify a consortium format as the governing body of the blockchain and imposed requirements on composition of the governing body's board of directors to address foreign collusion attacks. The work group also elected to specify that foreign companies cannot operate one third or more of the consensus nodes that validate transactions on the blockchain.

The work group developed a high-level data model to store private key security information on the blockchain (Section 8) as well as a high-level application programming interface (API) format for reading from and writing

to the blockchain. Remaining work, such as tightening blockchain specifications and software development, is left to the future governing body of the blockchain.

2 References

[1] Wivity Inc.: "Linen: Blockchain Key Registry for Distributed Energy Resources", White Paper, 2020.

[2] "UML Actor", https://www.uml-diagrams.org/use-case-actor.html

[3] GlobalPlatform, "Introduction to Secure Elements", May 2018.

[4] ISO/IEC 11889-1:2015, "Information technology — Trusted platform module library — Part 1: Architecture".

[5] ISO/IEC 15408-1:2009, "Information technology — Security techniques — Evaluation criteria for IT security — Part 1: Introduction and general model".

[6] IEEE 2030.5-2013, "IEEE Adoption of Smart Energy Profile 2.0 Application Protocol Standard", 31.10.2013.

[7] NIST FIPS PUB 140-2, "Security Requirements for Cryptographic Modules", 25.05.2001.

[8] Executive Order on Securing the United States Bulk-Power System, https://www.whitehouse.gov/presidential-actions/executive-order-securing-united-states-bulk-power-system/

[9] Maofan Yin, Dahlia Malkhi, Michael K Reiter, Guy Golan Gueta, and Ittai Abraham. 2018. Hotstuff: BFT consensus in the lens of blockchain. arXiv preprint arXiv:1803.05069 (2018).

[10] Loi Luu, Yaron Velner, Jason Teutsch, and Prateek Saxena. 2017. Smartpool: Practical decentralized pooled mining. In 26th {USENIX} Security Symposium ({USENIX} Security 17). 1409–1426.

[11] IEEE. 2018. IEEE 1547-2018 - IEEE Standard for Interconnection and Interoperability of Distributed Energy Resources with Associated Electric Power Systems Interfaces. (2018). https://standards.ieee.org/standard/1547-2018.html.

[12] Miguel Castro, Barbara Liskov, et al. 1999. Practical Byzantine fault tolerance. In OSDI, Vol. 99. 173–186.

[13] Shehar Bano, Alberto Sonnino, Mustafa Al-Bassam, Sarah Azouvi, Patrick McCorry, Sarah Meiklejohn, and George Danezis. 2017. Consensus in the age of blockchains. arXiv preprint arXiv:1711.03936 (2017).

[14] Gavin Wood et al. 2014. Ethereum: A secure decentralised generalised transaction ledger. Ethereum project yellow paper 151, 2014 (2014), 1–32.

[15] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine generals problem. ACM Transactions on Programming Languages and Systems (TOPLAS) 4, 3 (1982), 382–401

3 Definitions and Abbreviations

3.1 Definitions

Within the present document, the following definitions are used.

Assessor	An individual, who as an independent subject matter expert, evaluates and assesses systems and processes as determined by the scope of an audit. The purpose is to provide an objective measurement rather than to express an opinion.
Blockchain	A blockchain-based key registry for DER devices, as described in [1], which is specified in detail in the present document. The term Blockchain (with a capital B) is used within the present document to include all elements belonging to such blockchain implementation, including consensus nodes (also called validators), API Servers to read from and write to the blockchain, and governance mechanisms. The Blockchain is governed by a legal entity (see Governing Body actor definition in Section 5.2). The Blockchain is a system and the Governing Body is a legal entity.
Key Generation	Defines the necessary steps to create a private/public key pair, including using a random number generator and then provisioning the created keys into the (secure) environment, which will store them.
Secure Element	A tamper-resistant platform (typically a single chip secure microcontroller) capable of securely hosting applications and their confidential and cryptographic data (for example cryptographic keys) in accordance with the rules and security requirements set by well-identified trusted authorities. There are different form factors of SEs: embedded and integrated SEs, SIM/UICC, smart microSD as well as smart cards. Different form factors are necessary to address the requirements of different business implementations and market needs. [3]
Trusted Platform Module	A component which enables trust in computing platforms, compliant to ISO/IEC 11889-1 [4]. A TPM is physically isolated from the rest of the processing system and is often a separated hardware module.

3.2 Abbreviations

Within the present document, the following abbreviations are used.

API	Application Programming Interface
BFT	Byzantine Fault Tolerant
CA	Certificate Authority
CRL	Certificate Revocation List
CSR	Certificate Signing Request
DApp	Decentralized Application
DER	Distributed Energy Resource
ID	Identifier
OCSP	Online Certificate Status Protocol
ODM	Original Device Manufacturer
OEM	Original Equipment Manufacturer
PoS	Proof-of-Stake
PoW	Proof-of-Work
SE	Secure Element
TLS	Transport Layer Security
TPM	Trusted Platform Module

4 Introduction

Distributed Energy Resources (DERs) are expected to play an important role in generating power for the grid. While this development helps the transition towards clean energy, it also opens up new threats and risks. If power-exporting DER devices can be compromised by malicious actors, the consequences can be severe both economically and in terms of public safety. Thus, it is especially important that exporting DER devices are sufficiently protected.

Many DER device requirements mandate that all DER devices have private/public key pairs and communication to them happens only over protected channels like TLS connections. However, the certification practices used to enforce such requirements, such as IEEE 2030.5-based California Rule 21, do not capture the fact that DER devices may support very different security mechanisms including various means of key generation, provisioning, storage, access control and physical attack hardening. While this information may in principle be available in manufacturer factsheets, the provided data is not programmatically accessible, easily comparable or integrity-protected which makes automated trust decisions by communication endpoints very difficult in practice.

To address this problem SunSpec Alliance created the SunSpec Blockchain Work Group with the charter to create requirements and specifications for using blockchain technology to secure connected device credentials used in power grid communications. The main idea is to complement the current device certification infrastructures and security evaluation practices with blockchain technology. The blockchain functions as an integrity-protected and immutable registry that allows vendors to record easily-comparable information about registered DER devices, their key credentials, and their security properties. Energy grid communication endpoints can query this information and perform safe and automated trust decisions. The goal of the work group is to create a report that describes specifications for a decentralized application (DApp) and requirements for a blockchain that runs the DApp. The present document is this report. More background information on this system can be found in the SunSpec Blockchain Work Group Position Paper [1].

5 Use Cases

5.1 Introduction

The definition of use cases is an important and valuable step when capturing requirements. Use cases capture the main objectives of a system. A use case driven specification development process allows one to detect and analyze top-level building blocks of a system in order to identify the main actors and determine how such actors are going to interact with the system.

An actor specifies a role played by an entity external to the system, that interacts with the system. Section 5.2 will introduce the main actors.

A use case can be described as a list of actions which defines the interactions between an actor and the system in order to achieve a specific individual goal or execute a specific activity. A use case may include events, which describe an occurrence or a sudden change in the system's status. Actors can be a human or an external system. While Section 5.3 defines the main use case, individual activities will be covered in Section 5.4.

Section 5.5 will look deeper into different supply chain and distribution scenarios.

Analyzing system vulnerabilities can be a second valuable step in capturing requirements. Threat modeling allows one to identify weaknesses or strengths within the system, and to develop and to specify mitigation strategies. Threat models allow one to systematically analyze possible attackers and their intentions, and to understand the possibility and relevance of attacks and attack surfaces. Use case and activity level security threats will be introduced in Section 5.6.

In the present document, use cases and activities will be described and analyzed for the following reasons:

- To derive high-level technical and non-technical requirements.
- To understand dependencies from external stakeholders and from the external environment.
- To scope the technical architecture.
- To describe the main activities.

Section 5.7 will summarize requirements and recommendations derived from the use case analysis.

Once the details and constraints of the use case(s) are understood, the underlying pieces of the technical and non-technical architecture will be specified. The technical architecture will include the definition of building blocks, and their interfaces.

Technical requirements will describe functional as well as non-functional requirements, as detailed in Section 6. Non-technical requirements will address governance aspects, as detailed in Section 7.

5.2 Actors

In UML an actor is a behavioral classifier which specifies a role played by an external entity that interacts with a system (e.g., by exchanging signals and data). The term "role" is used informally to group users that require specific services from the system, modeled within associated use cases. When an external entity interacts with the system, it plays the role of a specific actor. Such an entity may play several different roles. In turn, a specific role may be played by one or more entities. [2]

In the present document, the aforementioned UML system is a blockchain-based key registry for DER devices ("Blockchain") as described in [1]. The term Blockchain (with a capital B) includes all elements belonging to a blockchain implementation, including consensus nodes, API Servers to read from and write to the blockchain, and governance mechanisms.

NOTE: For the purpose of the use case modeling, the legal entity implementing the governance mechanisms (i.e. the Governing Body) is regarded as an entity external to the Blockchain/system.

In the smart energy space, different actors are involved in securing Distributed Energy Resources (DER) devices. They will be used in the main use case description and the different activities.

The main actors are shown in Figure 1 and described in detail in the following.

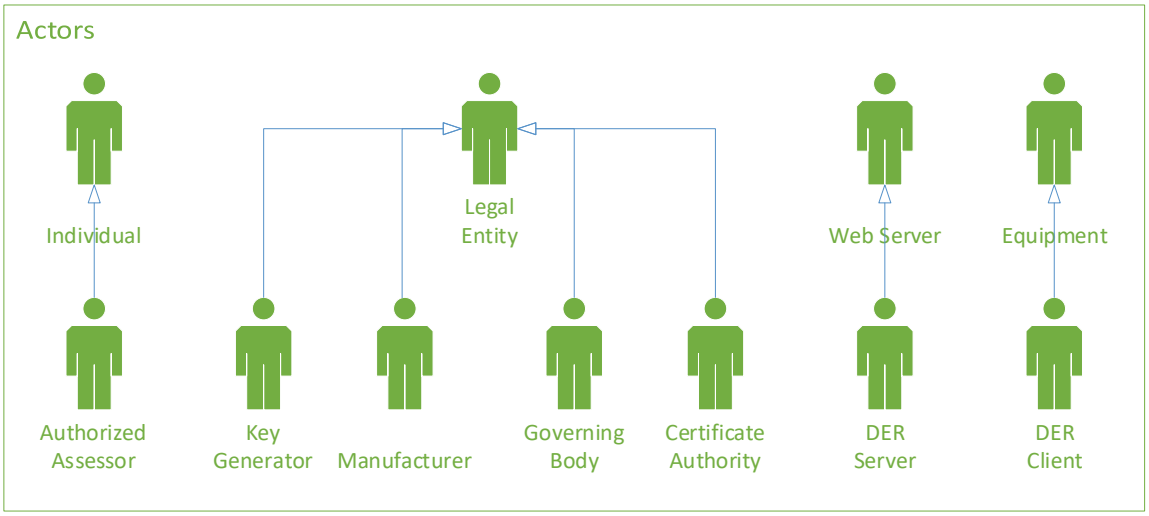

Figure 1: Actors

Manufacturer

The Manufacturer is a legal entity which is responsible for manufacturing, distributing, integrating, or installing a device, a module, or a component.

The Manufacturer is responsible for registering the address of the generated private/public key pair on the Blockchain once it has been provisioned onto the equipment. The Manufacturer is also responsible for recording the change in equipment ownership when sold by the Manufacturer.

NOTE: The definition of the public key address is out-of-scope for the present document.

DER Client

The DER Client is equipment in the Distributed Energy Resource space. It can be a solar inverter, a gateway, or an aggregator communicating with a DER utility server or another Service Provider backend. The DER Client is a specific piece of equipment that requests and accepts communications with an endpoint.

While individual equipment may be mass produced, it is expected that each device has its own unique private/public key pair which is registered on the Blockchain. Equipment may have other unique identifiers, such as a serial number, MAC addresses, etc., which allow the identification of a device or a networking/communication interface.

Manufacturing, installation, ownership transfer (reselling), and decommissioning are important stages within the DER Client's life cycle.

Key Generator

The Key Generator is a legal entity which is responsible for creating public/private key pairs and transferring (i.e. provisioning) them into the environment within the equipment that is responsible for storing and protecting them. A Key Generator can also be a Manufacturer in some scenarios, or an entity hired by a Manufacturer in other scenarios.

The quality of the key creation process mainly depends on the entropy of the randomness source that drives the key creation process. Low entropy sources reduce the effect key length of the private key.

Key creation can happen outside or inside the equipment's secure environment. In the case of key creation in a secure element, the private key will not leave the secure environment. Key provisioning is then part of the key creation process and not a distinctive step.

If the key creation happens in an external, remote, or offline appliance, the created key pair will be provisioned (i.e. stored) into a (secure) environment within the equipment. This provisioning step will then be considered part of the key creation process within the context of this specification unless otherwise stated. Key storage and protection are not considered part of the Key Generator role. Therefore, Key Generators create and provision keys, but are not responsible for storing or protecting them after the keys are provisioned.

Authorized Assessor

An Authorized Assessor is an individual who, as an independent cybersecurity subject matter expert, evaluates and assesses the key creation and key provisioning processes as well as the equipment's hardware and software environment in which the private key is stored and used.

The Authorized Assessor will need to be independent and free of any conflicts of interest with the Manufacturer or any entity creating and provisioning keys.

The Authorized Assessor will be authorized by the Governing Body of the Blockchain to conduct audits and to make them available on the Blockchain.

Certificate Authority

A certificate authority (CA) is a legal entity which issues digital X.509 certificates, for example those specified for communications by IEEE 2030.5 [6]. The issued certificate certifies the ownership of the public key by the named subject in the certificate (public key certificate). Additionally, the certificate contains identity information on the subject and how the certificate should be used. Certificates issued from the CA can be used in a DER Client or DER Server e.g. to establish a TLS session with certificate-based authentication, as specified by IEEE 2030.5 [6].

A CA can use information available on the Blockchain to validate or amend information about the key or the named subject. A CA will then read key data from the Blockchain and use this information to issue a digital X.509 certificate.

Such issued certificates provide additional important information that can aid in authorization decisions that leverage existing certificate-based security mechanisms, e.g.- TLS with mutual authentication.

Governing Body

The Governing Body is a legal entity responsible for governing the Blockchain. Within the use case definitions, the Governing Body is not considered part of the Blockchain.

The Governing Body has the sole responsibility and authority to make binding decisions for the Blockchain by establishing rules and processes. The details of the Blockchain are defined in more detail in Section 6. The structure and governing principles of the Governing Body are defined in more detail in Section 7.

DER Server

The DER Server is a Web Server endpoint specific to DER communication. The DER Server communicates with DER Clients, which allows the DER Server to control DER Client operations. When

setting up a secure connection, the DER Server needs to understand the security properties of a peer device's private key to authorize operations and to trust the DER Client's messaging.

These actors might be complemented by additional actors, further described in more detailed use cases or activity descriptions.

5.3 Main Use Case

5.3.1 Current State-of-the-Art

Since Distributed Energy Resources (DERs) are expected to play an important role in generating power for the grid it is important that the deployed DER devices are sufficiently protected against malicious actors. The current de-facto solution mandates that all DER devices have private/public key pairs and communicate only over protected channels like TLS connections that use certificates for (mutual) authentication.

This state-of-the-art certificate-based use case is shown in Figure 1.

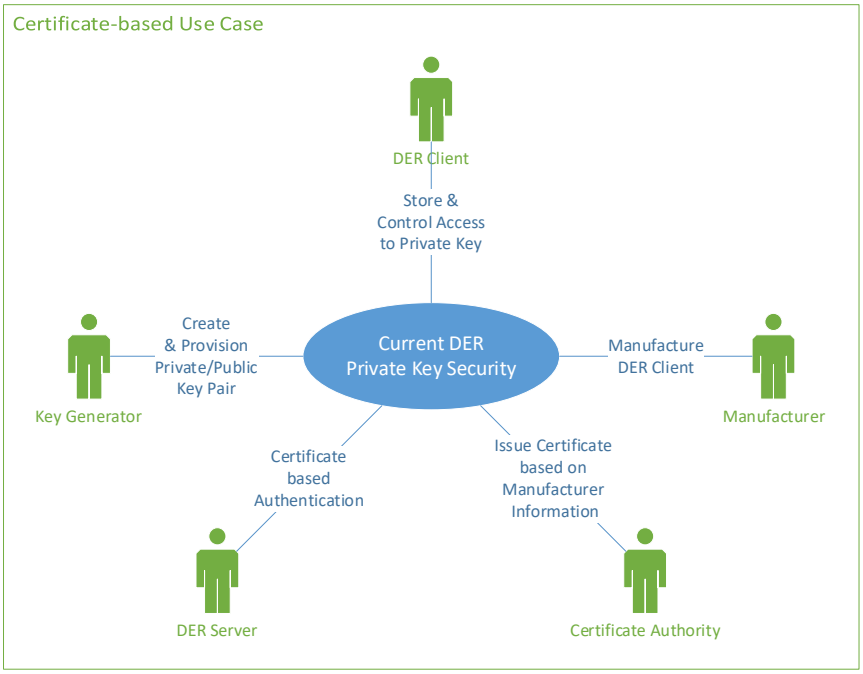

Figure 1:State-of-the-Art Certificate based Use Case

The certificate-based authentication consists of the following basic steps:

1. The Key Generator creates and provisions a private/public key pair.
2. The DER Client stores the key pair and provides controlled access to the private key.
3. The Manufacturer manufactures the DER Client.
4. The certificate authority issues a certificate based on the information provided in the certificate request.

5. The DER Server receives the issued certificate as part of the TLS handshake and establishes a connection to the DER Client if it validates the TLS session setup.

5.3.2 Problems with Current Solution

The current practice does not capture the fact that there are different means of key creation, provisioning, storage, and access control which result in vastly different security properties of the private key used to setup secure TLS sessions. As described in [1], different DER devices may implement very different security mechanisms, and the level of protection that the device provides is not captured by current device certificate practices. Therefore, the security level is not accurately conveyed to the communication endpoint (e.g. the DER Server) that needs to decide whether to trust the device or not.

The main differentiating factors of DER device security are [1]:

- **Key creation and provisioning**. Some DER devices use keys for TLS that are created on the device by a cryptographic random number generator that derives strong randomness from the physical environment. Other devices use keys that are derived from limited entropy or are provisioned to the device in plaintext.
- **Key storage and access**. Some DER devices store keys in a processing environment based on dedicated memory or memory encryption that is strongly isolated from the rest of the device and only specific code running on the device can access the private key. Other devices store the key in the main memory of the device and the private part of the key is accessible to all code running on the device.
- **Manufacturing process**. Some DER devices may be manufactured using processes that are documented in great detail and evaluated by accredited auditing authorities (e.g., Common Criteria's Security Evaluation Level [5] or FIPS 140-2 [7]), while other devices are manufactured using processes that are not documented or reviewed. In some cases, the manufacturing process of a DER device can be outsourced to another company or different country and the company that is considered the Manufacturer of the device has limited visibility to the actual manufacturing process of the device.

The fact that a DER device has a key pair that has been certified by a trusted certification authority (CA) does not capture such important differences that determine how trustworthy the DER device is.

5.3.3 New High-Level Architecture

The present document describes a new solution [1] to address the limitations in current industry practices. Specifically, the solution addresses the security aspects of private keys that are the foundation of security mechanisms like TLS by complementing current device certificate infrastructures and security evaluation practices with blockchain technology.

A blockchain is a decentralized data structure where transactions are organized into blocks. Data can be written into blocks and read back. The blockchain is integrity-protected and immutable due to an underlying consensus

mechanism. The consensus mechanism, achieved and enforced from consensus nodes using a specified consensus protocol, ensures that any data written on the chain cannot be later modified and that all system participants have the same view of the latest state of the chain. [1]

A high-level system architecture view of a blockchain is shown in Figure 2. Detailed discussions on the different consensus mechanisms are discussed in Section 6.1.

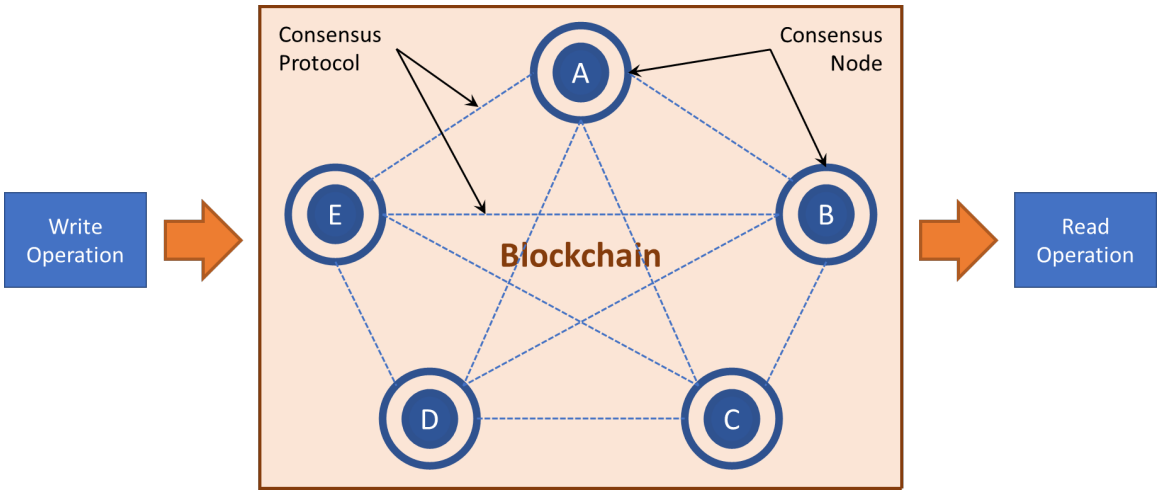

Figure 2: High-Level System Architecture

The Blockchain functions as an integrity-protected and immutable registry that allows vendors to record easily comparable information about registered DER device keys and their security properties. The present report promotes the adoption of already known good practices across the chip and DER device manufacturing industry by making a specific DER device's security practices accessible in a transparent way. These elements are combined into a complete solution that can be leveraged for secure DER device deployments.

5.3.4 Main Use Case Description

The main use case addressed within the present document increases transparency to the security properties of private keys provisioned within distributed energy resource (DER) devices, as shown in Figure 3.

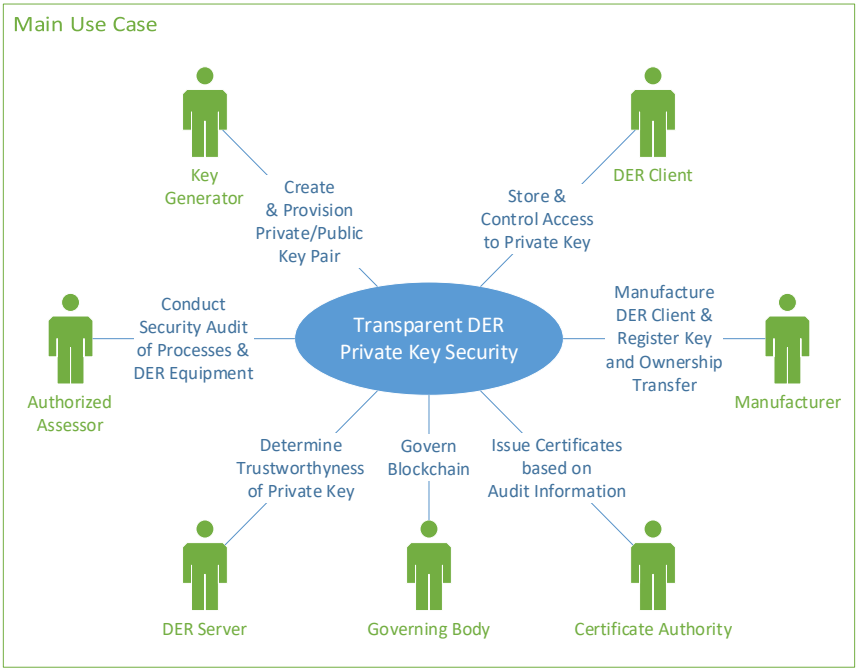

Figure 3: Main Use Case

The main use case consists of the following actors and their interaction:

1. An Authorized Assessor conducts a security audit of (a) the key generation process, (b) the private key's storage and control access mechanisms within the DER Client and (c) the exposure of the private key through the DER Client's manufacturing supply chain. Audit results are stored on the Blockchain.
2. The Key Generator creates and provisions a private/public key pair into the DER Client. The Key Generator processes were previously audited.
3. The DER Client stores the key pair and provides controlled access to the private key. Storage and access mechanisms were previously audited.
4. The Manufacturer manufactures the DER Client. The manufacturing process was previously audited.
5. The Manufacturer will register the key of a manufactured DER Client on the Blockchain. The registration will link the generated key pair to (a) the audited process used to create and provision the keys in step 2, (b) the audited DER Client process for storing and protecting the key in step 3, and (c) information from the Manufacturer in step 4. Any further ownership transfer can be recorded as well.
6. The certificate authority issues a certificate for a public key based on the information provided in the certificate request, which is verified and amended using information about the key on the Blockchain.
7. The Governing Body governs the Blockchain, which keeps integrity protected records of all audit results, processes, Manufacturers, and key information. The Governing Body gives authorized entities read and write access to the Blockchain.
8. The DER Server receives the issued certificate as part of the TLS handshake with the DER Client. It retrieves information associated with the certificate from the Blockchain and validates the TLS session

setup after determining that the trustworthiness of the private key meets the DER Server's minimum requirements.

5.3.5 Performance

Throughput measures how many transactions can be processed within a certain amount of time. Bitcoin processes a few transactions per second. Visa processes thousands of transactions per second. Throughput requirements for the DER industry depend on how the industry is defined. Table 1 compares throughput requirements assuming one device per transaction.

Type of Device	Units per Year	Transactions per Second
US solar installations	1,000,000	≈0.03
Worldwide solar installations	10,000,000	≈0.3
Worldwide vehicle sales	100,000,000	≈3
Worldwide connected devices	10,000,000,000	≈300

Table 1: Throughput Requirements

The first anticipated customer segment for the Blockchain is California solar installations, but it is anticipated that the Blockchain's addressable market will expand to encompass all devices that draw a significant amount of power as these devices could all be potentially incorporated into demand/response systems. Such devices include electric vehicles, thermostats, HVAC systems, and large screen electronics. The size of this device category is expected to be in the billions per year. This growth in addressable market over time implies the blockchain need only process less than 1 transaction per second at first to address solar installations but be able to grow to hundreds of transactions per second when the market expands to other types of distributed energy resource devices.

Latency measures how quickly a transaction can be finalized after the transaction is submitted to be processed by the system. Some separate the concept of latency into two measurements: *latency*, the time it takes to put a transaction on a blockchain, and *finality*, the time the transaction requester must wait to be sure the transaction has been accepted (e.g.- in Bitcoin, the chain that contains the transaction is the one that miners choose to use going forward). They will be treated the same because the present document's main use case requires finality.

The Blockchain's use case with perhaps the highest sensitivity to latency is key provisioning. Often keys are provisioned on the manufacturing floor which does not lend much time to wait for a transaction to be processed. However, if transactions can be batched together and be decoupled from the manufacturing synchronization this could reduce latency requirements. Therefore, the present document does not mention a latency requirement

The following sections will provide details of the different actors involved, as well as further breakdown into individual use cases and activities.

5.4 Detailed Use Cases and Activities

The main use case can be broken down into a set of detailed use cases. In addition to the use case description, simple activity diagrams are provided to capture dynamic aspects.

5.4.1 Audit Creation

An audit assessment allows for an independent examination of an entity's security process or implementation. The Authorized Assessor will provide an objective assessment of pre-determined aspects and properties of the audited entity that are relevant to the scope of the audit. With respect to the security of private keys in DER equipment, the following aspects are core to the audit:

1. Process to create the private/public key pair.
2. Process to provision the created private/public key pair into the DER equipment.
3. Mechanisms to protect access to the private key within the final DER equipment.

NOTE: In the case of hardware secure environments like TPMs or Secure Elements (SE), the private/public key pair can be created on the same hardware where the key is stored. Hence, no separate provisioning step is needed. This information will be recorded in the audit covering the key provisioning part.

NOTE: The assessment of the first two steps are referred to as the Process Assessment within this document. The assessment of the 3rd step is referred to as the Component Assessment.

There are four relevant actors in this use case- the Key Generator, the DER Client, the Authorized Assessor, and the Governing Body. The audit creation use case is shown in Figure 4.

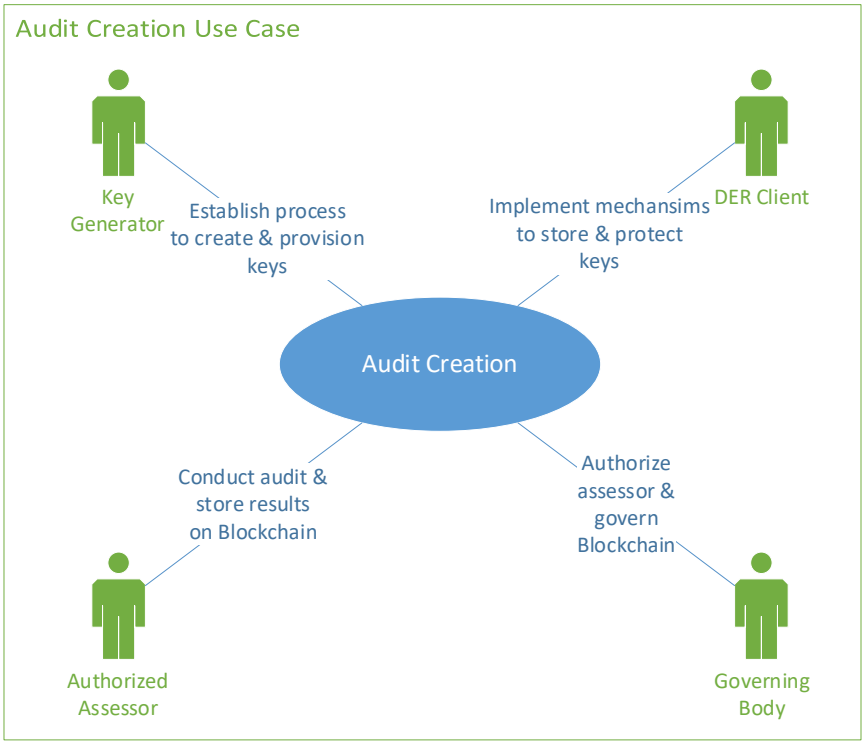

Figure 4: Audit Creation Use Case Diagram

The Key Generator can be an OEM, a semiconductor vendor, or a service provider like a distributor or SaaS provider. The Key Generator designs a key creation process that specifies what kind of hardware and/or software creates the key.

The Key Generator also designs a process to transfer the created key pair to the DER Client that stores the key. This process is typically called key provisioning. A key can be provisioned before, during, or after the DER Client has been manufactured.

The DER Client contains the hardware and software that receives the created keys and securely stores them. Access, specifically to the private key, is restricted to only authorized users. Access control is not only limited to the key itself, but also to the security functions that use the private key as part of the operation, such as signing or encryption/decryption. The DER Client employs hardware and/or software mechanisms to protect the key.

The Authorized Assessor first gets authorized by the Governing Body. The authorization process validates that the Authorized Assessor has the necessary competencies and subject matter expertise within the audit areas, is free of conflicts of interest, and possesses documents proving the Authorized Assessor's adherence to auditing principles. To obtain authorization, an Authorized Assessor submits a request for review by the Governing Body. If the request meets the requirements, the Authorized Assessor is approved. This high-level activity flow is shown in Figure 5.

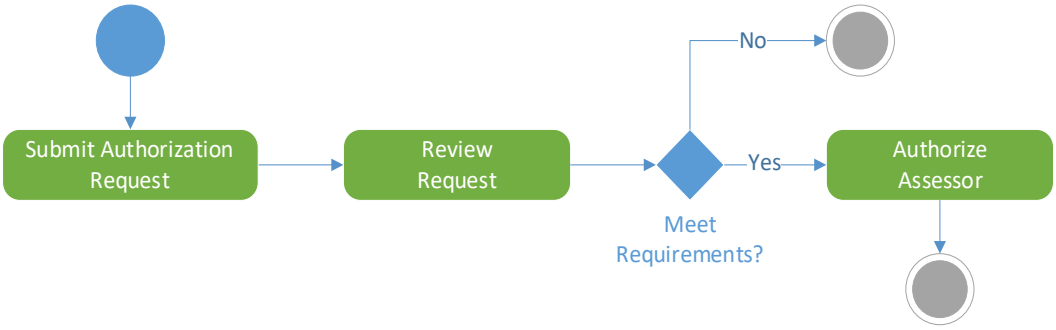

Figure 5: Assessor Authorization Activity Flow

The criteria an Authorized Assessor must meet to get authorized are defined from the Governing Body. Technical expertise will need to be kept in sync with advances in the security domain and developments in cyber-security threats. The Authorized Assessor's qualifications are regularly re-assessed as defined by the Governing Body.

The Authorized Assessor reviews (a) the key creation and key provisioning process and (b) the key protection and access control. The assessment is done against standards and requirements that will be specified in separate documents.

Finally, the Authorized Assessor uploads the results of the audit assessment- including the identity of the Authorized Assessor, Key Generator, and Manufacturer- to the Blockchain. An audit will be attached with a unique identifier (Audit ID). Details of the assessment are introduced in Section 8. The Governing Body can establish processes to review and approve (parts of) a report before it can be uploaded to the Blockchain. The Governing Body can create a program management structure overseeing the Authorized Assessor. The high-level activity flow is shown in Figure 6.

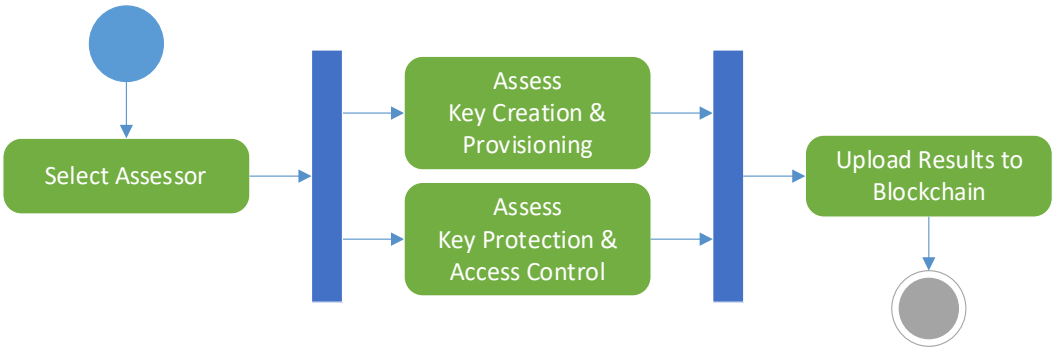

Figure 6: Audit Assessment Activity Flow

In case re-evaluations are needed at regular intervals, the results from these re-assessments are put on the Blockchain, referring to the original Audit ID.

NOTE: The assessment holds much more information about a key than the current practice of certificate extensions.

5.4.2 Manufacturing and Key Provisioning

When the keys are created in mass quantity during manufacturing, each private key has a public key pair. Every manufactured DER Client will be provisioned with a unique key pair to store and protect. A separate key provisioning step is not needed if the key is created within the DER Client's (secure) environment that stores the key. Key creation and key protection will use the process and implementation previously audited. There are three relevant actors in this use case- the Key Generator, the DER Client, and the Manufacturer. The manufacturing and key provisioning use case is shown in Figure 7.

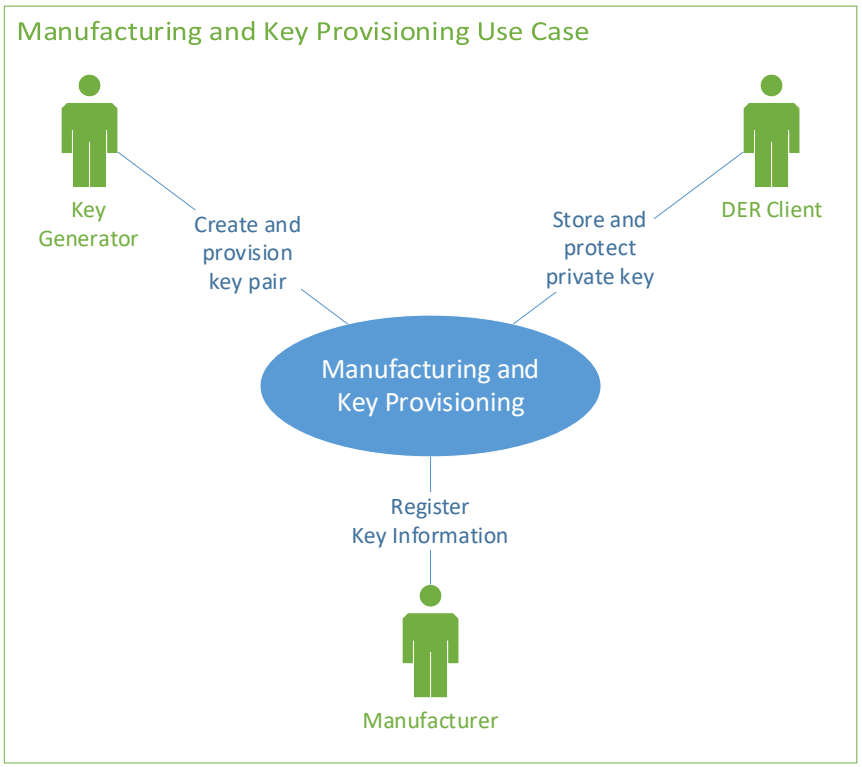

Figure 7: Manufacturing and Key Provisioning Use Case Diagram

The Key Generator can be an OEM, a semiconductor vendor, or a service provider like a distributor or SaaS provider. The key creation process that specifies what kind of hardware and/or software creates the key was previously assessed by an Authorized Assessor, as described in Section 5.4.1. A key pair, consisting of a private and corresponding public key, will be created following this process.

NOTE: The hash of the public key can be used as the basis of the key identifier on the Blockchain.

If the key pair is created outside the DER Client, the pair will be provisioned into the manufactured DER Client. The key provisioning process was previously assessed by an Authorized Assessor.

The manufactured DER Client stores and protects the generated key pair in hardware and/or software. The detailed implementation was previously assessed by an Authorized Assessor. The DER Client has access to the private key in order to perform needed cryptographic computations.

NOTE: A DER Client can cryptographically prove to peer devices that it owns the private key belonging to the respective public key.

The Manufacturer is responsible for uploading the key information to the Blockchain. This links the key information, identified through the key identifier, to the audited process. The high-level activity flow is shown in Figure 8.

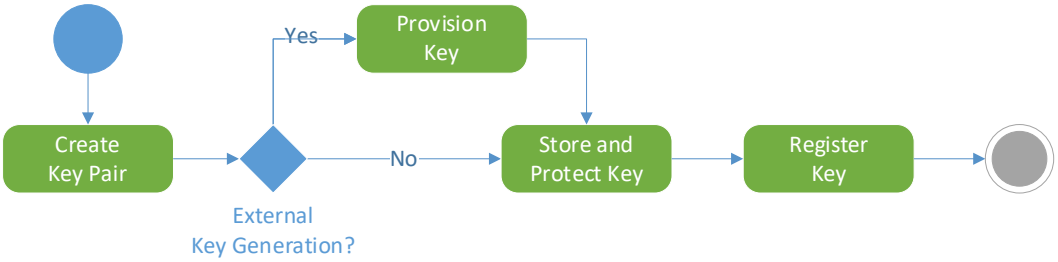

Figure 8: Key Generation and Provisioning Activity Flow

This use case covers different scenarios for the manufacturing and key provisioning processes. The main scenarios are given below:

- Keys are created within a SE or similar hardware protected secure environment. Generated private keys never leave the protected environment. The key generation can be executed by the original SE Manufacturer or by a distributor within the supply chain. The SEs are then integrated into DER equipment, e.g. as modules.
- Keys are created on the manufacturing floor of the DER Client and then provisioned into the manufactured equipment at the time of manufacturing, e.g. at the time of first system boot up. Key provisioning can happen offline.
- Keys are created remotely and provisioned into the manufactured equipment at the time of installation. This requires dedicated steps for the installer and/or can require connectivity to an online provisioning service.

There are many different manufacturing and supply chain scenarios. These are described in more detailed in Section 5.5.

5.4.3 Certificate Generation

The certificate is a data structure that contains the public key of the subject (e.g. the DER Client), the requested attributes, and additional information signed by the private key of a trusted Certificate Authority (CA). The

primary security function of a certificate is that it provides a secure mapping between the public key of the subject and subject attributes like name, model, and domain. Any third party who knows the public key of the CA (or its root CA in the case of an intermediate CA) and trusts the (root) CA can verify the correctness of this mapping by verifying the certificate. [1]

The CA will need to correctly validate all the attributes before issuing the certificate. Otherwise the mapping between the public key and these attributes cannot be trusted. The Blockchain allows the CA to validate the attributes independently of the received CSR as described in this use case. There are three relevant actors in this use case- the DER Client, the Manufacturer, and the Certificate Authority, as shown in Figure 9.

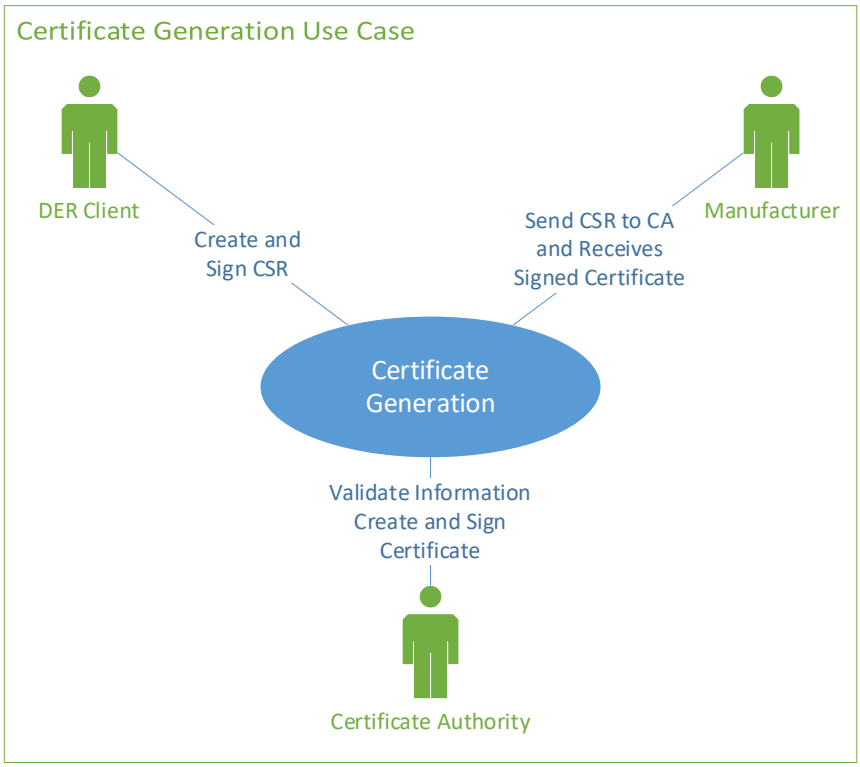

Figure 9: Certificate Generation Use Case Diagram

At some point between the time of manufacturing and the time of installation, a certificate needs to be created for the provisioned private/public key pair. A certificate signing request (CSR) is created by the DER Client after it is provisioned with the private/public key pair. The CSR will contain the public key and all additional information and certificate properties that will be included in the certificate. The CSR is signed by the provisioned private key.

NOTE: A valid extension of this use case can involve the Key Generator as the creator of the CSR. If the CSR is created prior to the provisioning of the private/public key pair, possibly even prior to the manufacture of the DER Client, the certificate is provisioned together with the key pair.

The Manufacturer will send the CSR to the relevant signing CA. The selection of the CA is outside the scope of this document. The selection of the signing CA can be dependent on where the DER Client is installed as different policies can exist regarding trusted root CAs and signing CAs.

NOTE: A valid extension of this use case can involve a DER Client with multiple private/public key pairs for different communication peers. The signing CA for a certificate used to establish a connection to a DER Server can be different than the signing CA for a certificate used to connect to the Manufacturer's backend.

NOTE: A valid extension of this use case can involve the installer initiating the issuance of a certificate.

The certificate authority receives the CSR and validates its signature using the included public key. The CA will then derive the key identifier (Key ID) from the CSR and query the Blockchain for cybersecurity related information related to the key. The CA assesses all available security related information, like the Key Generator, the audit assessment etc., prior issuing the certificate. The CA inserts relevant information into the certificate, possibly using certificate extensions, and signs the certificate with the CA's private key.

The issued certificate is sent back to the Manufacturer together with the signing trust chain (intermediate certificates). The Manufacturer then provisions the certificate into the device. The high-level activity flow is shown in Figure 10.

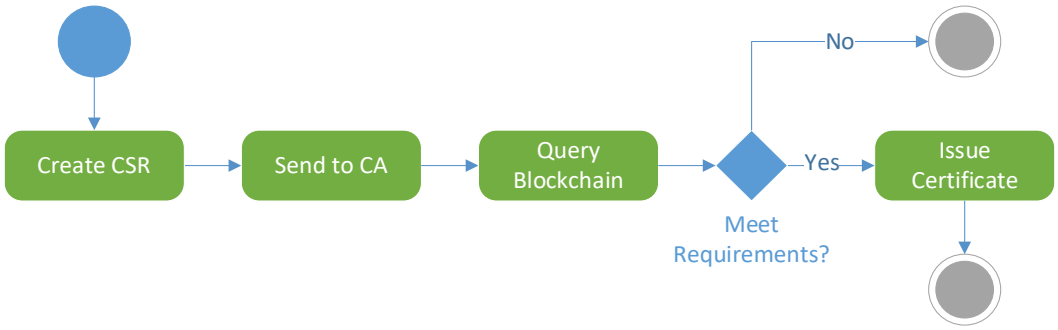

Figure 10: Certificate Generation Activity Flow

The requirements for allowing the issuance of a certificate are outside the scope of this specification. These requirements will depend on how the certificates are going to be used, e.g. part of TLS session authentication, and which communication they are going to authenticate, e.g. IEEE 2030.5. Requirements may also depend on the type of DER Client. A DER Client connecting an aggregator may need to meet higher requirements, than a single inverter.

5.4.4 Communication Security

Modern secure communication between two remote peers is established using Transport Layer Security (TLS). In many cases, like IEEE 2030.5, the establishment of a TLS session requires mutual authentication using X.509v3 public key certificates.

A DER Client will connect to a DER Server, such as a utility server, to receive a command or retrieve status information using the IEEE 2030.5 protocol. When setting up a secure TLS session, the DER Client and DER Server will perform mutual authentication using X.509 public key certificates for authentication as defined in the TLS protocol. The DER Server and DER Client have to be provisioned with certificates, as described in the use case within Section 5.4.3. The DER Client receives the DER Server's certificate and can validate the certificate's signature and trust chain. The DER Server receives the DER Client's certificate and can then validate the certificate's signature and trust chain. These validations do not provide information on the trustworthiness of the DER Client or DER Server's cryptographic environment. For example, the DER Server will not be able to distinguish a DER Client whose private key is stored in plain text readable to everybody, from a DER Client whose private key is securely contained within a hardware protected environment like a SE or TPM.

The Blockchain described within the present document allows DER Servers to independently validate the cyber-security properties of the DER Client, as described within this use case. There are three relevant actors in this use case- the Certificate Authority, the DER Client, and the DER Server, as shown in Figure 11.

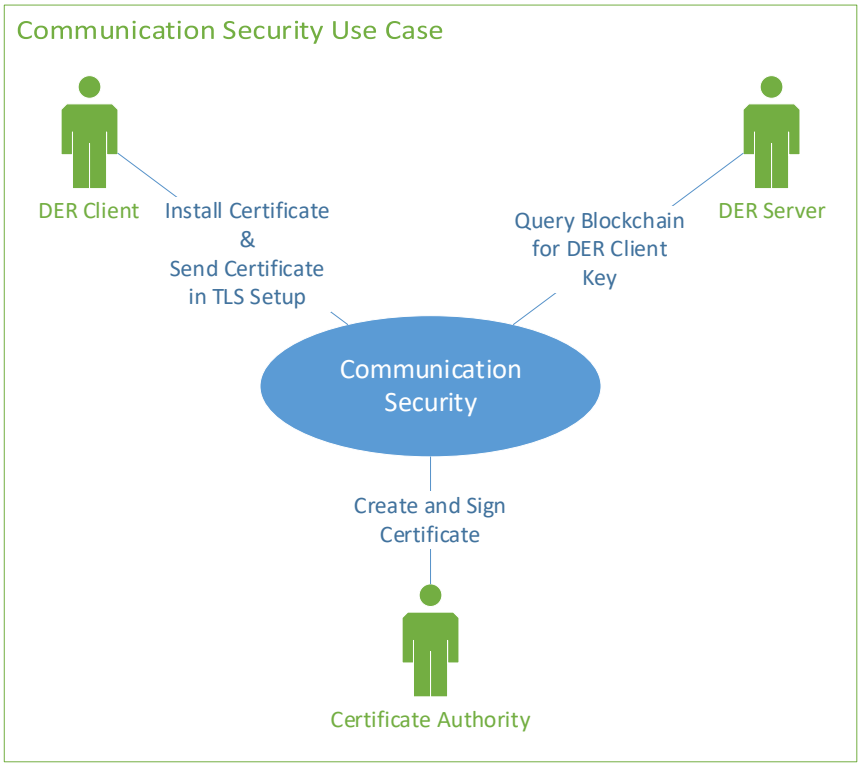

Figure 11: Communication Security Use Case Diagram

A traditional certificate authority (CA) creates and signs an X.509 public key certificate that will be provisioned into the DER Client. The certificate contains the DER Client's public key and additional certificate (policy) parameters. The standard certificate contains no information regarding the key creation, key provisioning, or protection of the private key on the device.

When the DER Client establishes a connection to the DER Server to execute commands in compliance with a protocol such as IEEE 2030.5, the DER Server will receive a copy of the DER Client's certificate. The DER Server can validate the certificate's signature and further extract the DER Client's key identifier. The DER Server can then query the Blockchain using the extracted key identifier to receive additional cyber-security information. This allows the DER Server to authorize the DER Client for specific operations or reject the communication request.

NOTE: The role of the DER Server can be extended to an aggregation server, or any other server that provides a service to the DER Client and relies on correct data and operations from the DER Client.

NOTE: The available information on the Blockchain can go beyond details on the security of a DER Client's private key. Additional details are beyond the scope of the present document.

Blockchain queries are complementary to OCSP and CRL. These protocols only allow the revocation of an issued certificate. In querying the Blockchain the DER Server or DER Client receives much richer information about the security properties of the peer. Details are defined in Section 8. The high-level activity flow is shown in Figure 12.

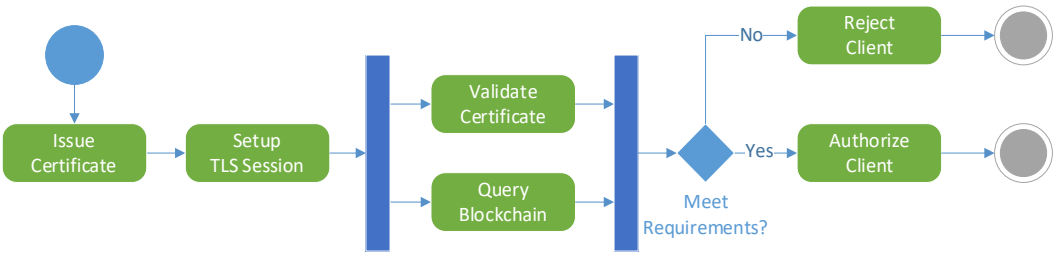

Figure 12: Communication Security Activity Flow

The requirements which need to be met to accept a certificate are outside the scope of this specification. These requirements will depend on how the certificates are going to be used. Requirements may also depend on the type of DER Client or the kind of access the DER Client is seeking, e.g. privileged vs. non-privileged access.

5.4.5 Key Life Cycle Tracking

Manufacturing a device is typically a complex process involving multiple vendors in a supply chain. After manufacturing, a device may undergo several steps within a distribution chain before finally being installed and put into operation. After installation, a device may be sold on a secondary market before finally being decommissioned.

One of these ownership transitions from a Manufacturer to a distributor is described within the key life cycle tracking use case. Other ownership transfers can be described accordingly. There are two relevant actors in this use case- the Manufacturer (as instance A and B) and the DER Client, as shown in Figure 13.

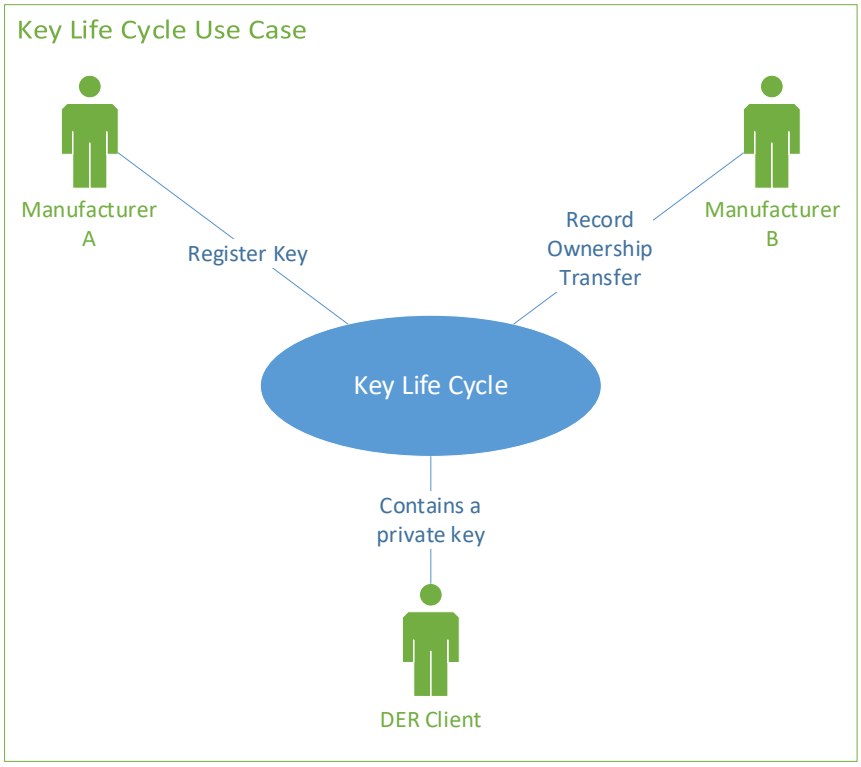

Figure 13: Key Life Cycle Use Case Diagram

The key life cycle use case describes how the ownership of a DER Client and its included private key can be tracked over the lifetime of the DER Client. The Manufacturer A initially records the key information on the Blockchain. Later, when Manufacturer A ships the device containing the key to another Manufacturer B (e.g. a distributor), Manufacturer A sends a record of ownership transfer to the Blockchain. Manufacturer B may need to accept the transfer. This makes the ownership of the private key transparent. Section 5.5 describes different life cycle scenarios originating from supply and distribution chains.

A Manufacturer can also record that a device has reached end of life and has been decommissioned. This end-of-life record on the Blockchain enables the prevention of the private key's misuse in the future, as a DER Server with this knowledge can deny service to a device that uses the decommissioned private key.

The life-cycle information is recorded whenever a change of the recorded information is required. This ensures that always the latest information relevant for a specific key is available on the Blockchain. The high-level activity flow is shown in Figure 14.

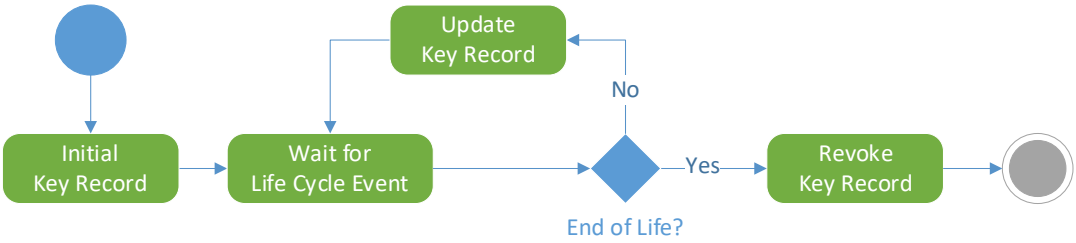

Figure 14: Life Cycle Activity Diagram

Like an end-of-life event, a key record can be revoked when the private key has been compromised. Key leakage can be result of various attacks and can scale from individual devices, such as a physical attack on a device, up to a family of devices due to a successful supply chain attack. Therefore, compromised keys will need to be revoked in a timely manner.

5.5 Supply Chain and Distribution Aspects

The manufacture of a DER Client can involve a multi-stage supply chain, which includes multiple suppliers, and distributors. After manufacturing, the DER Client can go through a product distribution chain before being installed. This section will describe the most important aspects of the supply chain and distribution.

The following specializations of the generic Manufacturer role, defined in Section 5.2, can be defined:

Component Manufacturer

Manufacturer of a hardware component used in the manufacture of a DER Client.

Secure Component Manufacturer

Manufacturer of a hardware component, like a Secure Element (SE), which creates a secure execution environment.

Secure Component Service Provider

Service Provider that acquires the basic secure hardware component from a secure component Manufacturer and then implements additional secure applications on it.

OEM Manufacturer

Manufacturer of the DER Client.

Key Generation Service

An online web service that creates and provisions keys. The key generation service can also handle the key registration with the Blockchain if the key generation service also is tied to a key storage and protection scheme. Provisioning can happen at the time of manufacturing or installation.

Key Generation Appliance

An offline hardware appliance that creates and provisions keys during the manufacturing process. The key generation appliance can also handle the key registration with the Blockchain if it is tied to a key storage and protection scheme.

Installer

Entity installing the DER Client.

5.5.1 Secure Component provisioned by Manufacturer

In this scenario, a Secure Component Manufacturer handles all relevant aspects, like manufacturing, key creation, key provisioning, and key registration in compliance to a previous component and process audit commissioned by the Secure Component Manufacturer. An OEM Manufacturer acquires the provisioned secure component and integrates it into its final product. The Secure Component Manufacturer will record the ownership transfer of the already registered key. Finally, the installer deploys the manufactured DER Client and the Manufacturer records the final ownership transfer to the consumer. These steps are shown in Figure 15.

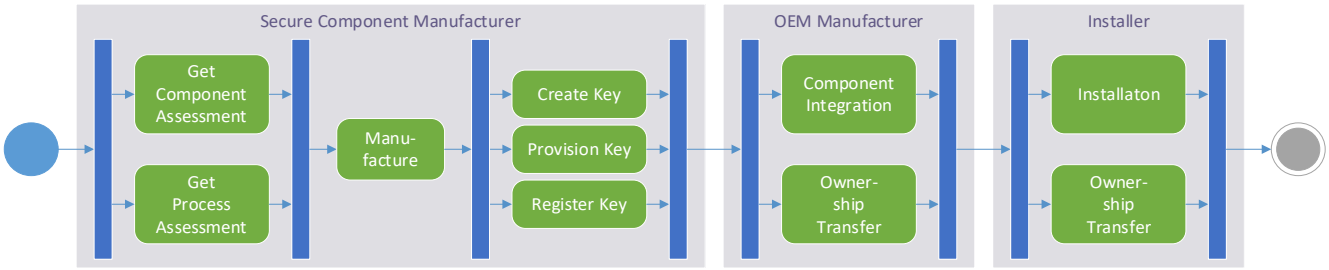

Figure 15: Secure Component provisioned by Manufacturer

In the case where the OEM Manufacturer does not directly source from the Secure Component Manufacturer a chain of distributors may be involved. Distributors may be required to record ownership transfers on the Blockchain. Similar distribution may exist for the DER Client prior to being deployed by the installer.

5.5.2 Secure Component provisioned by Service Provider

In this scenario, a Secure Component Manufacturer is only responsible for manufacturing the component in compliance with a previous component audit commissioned by the Secure Component Manufacturer. All other relevant aspects, like key creation, key provisioning, and key registration are managed by a Secure Component

Service Provider in compliance to a previous process audit commissioned by the Secure Component Service Provider. An OEM Manufacturer acquires the provisioned secure component and integrates it into its final product. The Secure Component Service Provider will record the ownership transfer of the already registered key. Finally, the installer deploys the manufactured DER Client and the Manufacturer records the final ownership transfer. These steps are shown in Figure 16.

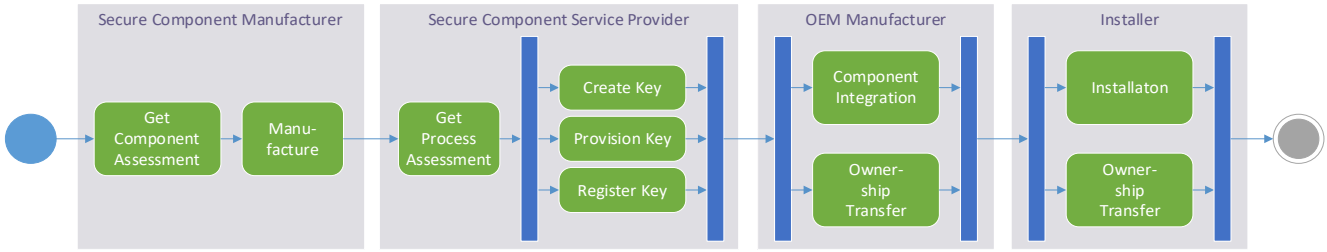

Figure 16: Secure Component provisioned by Service Provider

In the case where the OEM Manufacturer does not directly source from the Secure Component Service Provider, a chain of distributors may be involved. Distributors may be required to record ownership transfers on the Blockchain. Similar distribution may exist for the DER Client prior to being deployed by the installer.

5.5.3 Secure Component provisioned by OEM

In this scenario, a Secure Component Manufacturer is only responsible for manufacturing the component in compliance with a previous component audit commissioned by the Secure Component Manufacturer. All other relevant aspects, like key creation, key provisioning, and key registration are managed by the OEM Manufacturer in compliance with a previous process audit commissioned by the OEM Manufacturer. The OEM Manufacturer integrates the component into its final product. Finally, the installer deploys the manufactured DER Client and the OEM Manufacturer completes the final ownership transfer. These steps are shown in Figure 17.

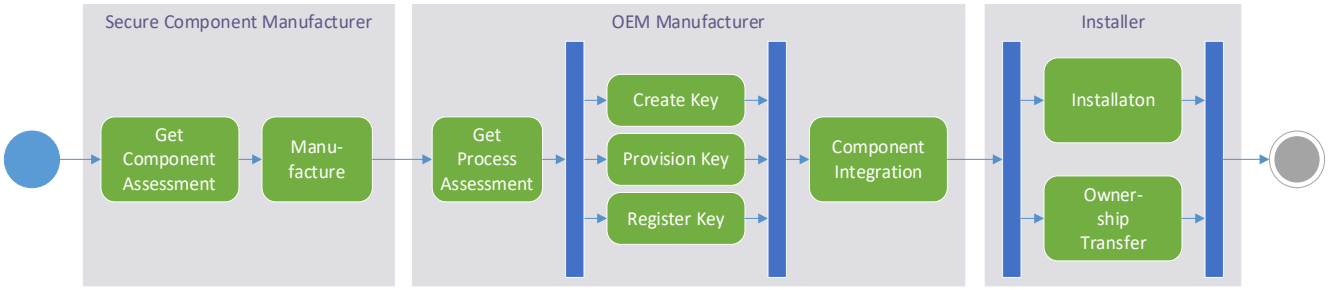

Figure 17: Secure Component provisioned by OEM

In the case where the OEM Manufacturer does not directly source from the Secure Component Manufacturer, a chain of distributors may be involved. Similar distribution may exist for the DER Client prior to being deployed by the installer. Distributors of the DER Client containing a registered key may be required to record the ownership transfer on the Blockchain.

5.5.4 Component Integration by OEM

In this scenario, an OEM Manufacturer integrates components acquired from a component Manufacturer. The OEM Manufacturer handles component integration and all relevant aspects, like key creation, key provisioning, and key registration, in compliance with a previous component and process audit. Finally, the installer deploys the manufactured DER Client after the OEM Manufacturer records the final ownership transfer. These steps are shown in Figure 18.

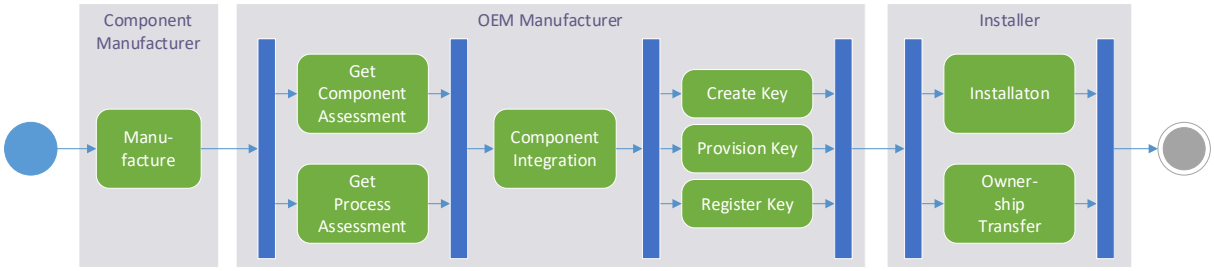

Figure 18: Component Integration by OEM

In the case where the OEM Manufacturer does not directly source from the Component Manufacturer, a chain of distributors may be involved. Similar distribution may exist for the DER Client prior being deployed by the installer. Distributors of the DER Client containing a registered key may be required to record the ownership transfer on the Blockchain.

5.5.5 Key Provisioning by Applicance at Manufacturing

In this scenario, an OEM Manufacturer integrates components acquired from a component Manufacturer. The OEM Manufacturer handles component integration in compliance to a previous component audit commissioned by the OEM Manufacturer.

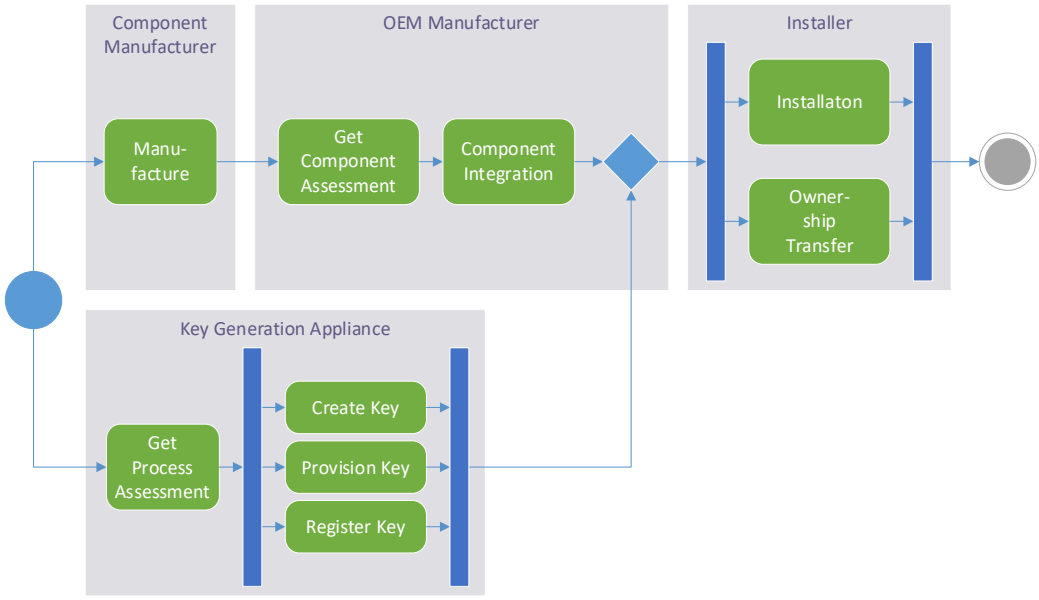

Figure 19: Key Provisioning by Appliance at Manufacturing

The OEM Manufacturer uses an offline Appliance to handle key creation, key provisioning, and key registration, in compliance to a process audit commissioned by the Appliance vendor. In the last step, the installer will deploy the manufactured DER Client, including recording the final ownership transfer to the consumer. These steps are shown in **Error! Reference source not found.**.

In the case where the OEM Manufacturer does not directly source from the Component Manufacturer, a chain of distributors may be involved. Similar distribution may exist for the DER Client prior to being deployed by the installer. Distributors of the DER Client containing a registered key, may be required to record the ownership transfer on the Blockchain.

5.5.6 Key Provisioning by Service Provider at Manufacturing

In this scenario, an OEM Manufacturer integrates components acquired from a component Manufacturer. The OEM Manufacturer handles component integration in compliance with a previous component audit commissioned by the OEM Manufacturer. The OEM Manufacturer uses an online Key Provisioning Service to handle key creation and key provisioning in compliance with a process audit commissioned by the Key Provisioning Service. Key registration is performed by the OEM Manufacturer after key provisioning. In the last step, the installer will deploy the manufactured DER Client, including recording the final ownership transfer to the consumer. These steps are shown in Figure 20.

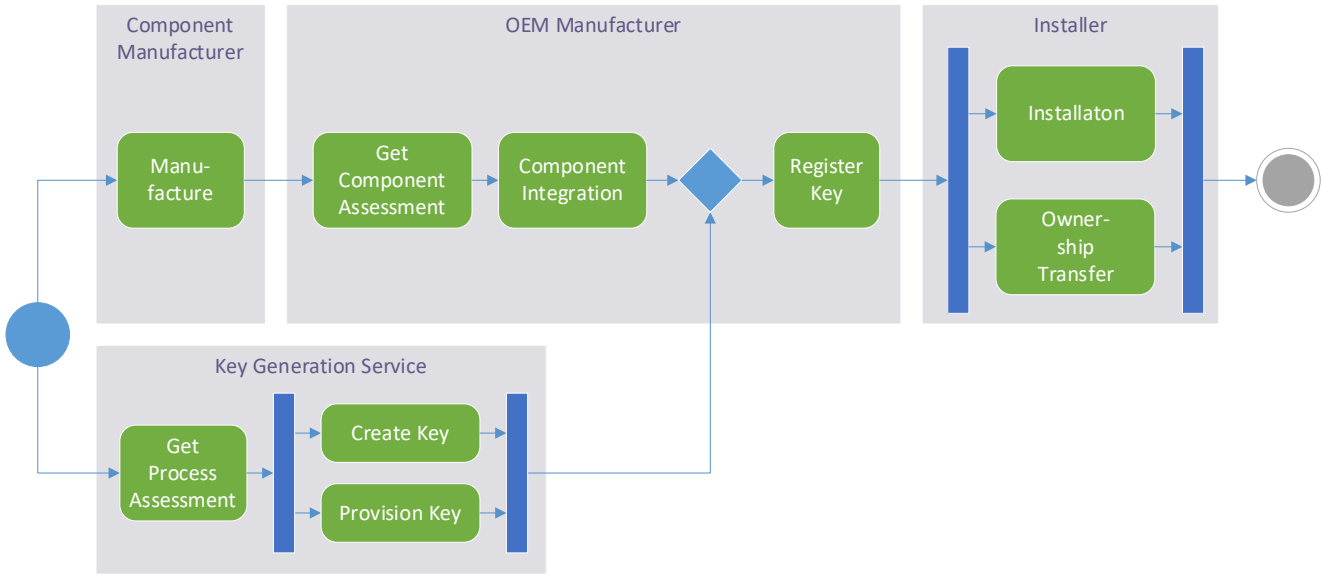

Figure 20: Key Provisioning by Service Provider at Manufacturing

In the case where the OEM Manufacturer does not directly source from the Component Manufacturer, a chain of distributors may be involved. Similar distribution may exist for the DER Client prior to being deployed by the installer. Distributors of the DER Client containing a registered key may be required to record the ownership transfer on the Blockchain.

5.5.7 Key Provisioning by Service Provider at Installation

In this scenario, an OEM Manufacturer integrates components acquired from a component Manufacturer. The OEM Manufacturer handles component integration, in compliance to a previous component audit commissioned by the OEM Manufacturer. In the final step, the installer will deploy the manufactured DER Client. The installer uses an online Key Provisioning Service provided by the OEM Manufacturer (e.g.- a mobile app) to handle key creation, key provisioning, and key registration, in compliance with a process audit commissioned by the OEM Manufacturer. These steps are shown in Figure 21.

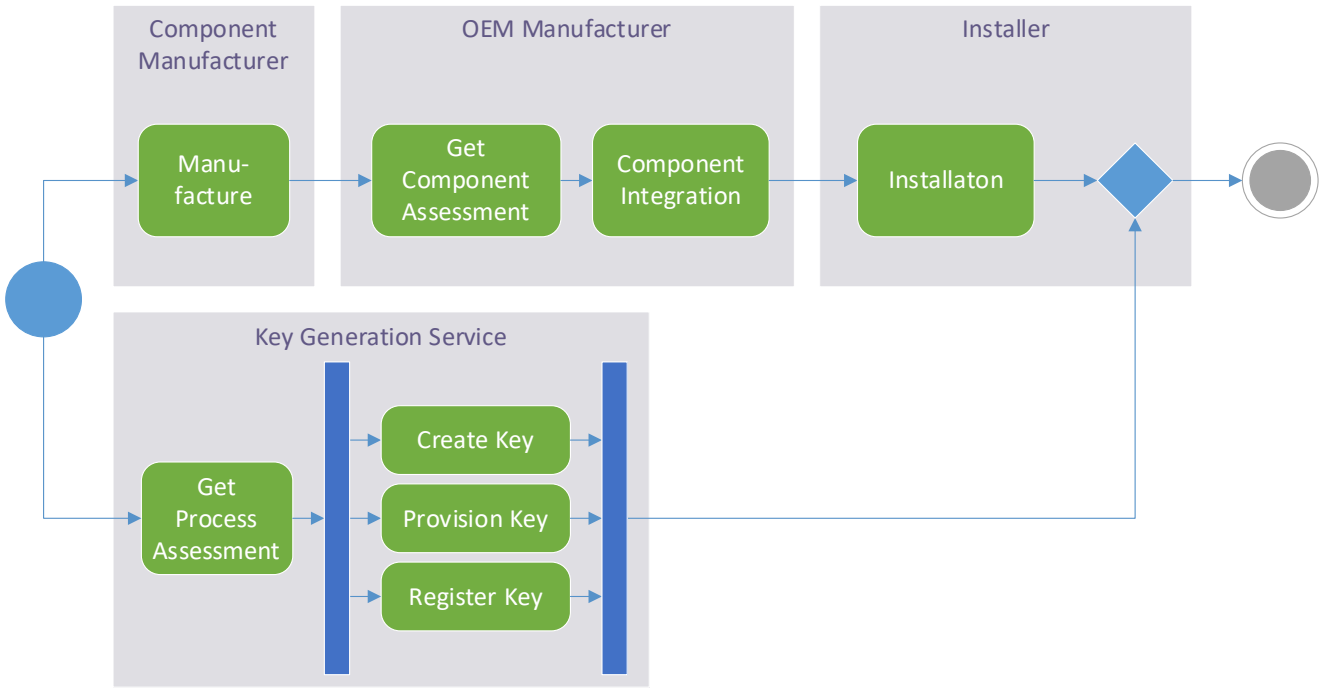

Figure 21: Key Provisioning by Service Provider at Installation

In case the OEM Manufacturer does not directly source from the Component Manufacturer, a chain of distributors may be involved. Similar distribution may exist for the DER Client prior to being deployed by the installer.

5.6 Use Case and Activity level Security Threats

The uses cases considered in this section allow storage and retrieval of security information on the many ways DER Clients can be secured, but there are known security threats to these use cases. How the Blockchain is implemented and governed can significantly reduce these threats.

5.6.1 Nation State Coercion

Actors may reside in foreign jurisdictions and come under pressure from authorities to become malicious actors in the Blockchain. Careful design of the data model (Section 8) and assessment procedures can reveal this risk so that regulators can create requirements that mitigate this threat. For example, regulators can require that all process owners are domestic companies and all processes are located domestically.

5.6.2 Malicious Accessors

The root of trust in the aforementioned use cases is the Authorized Accessors. If an Authorized Accessor ignores security issues or falsely reports information, the information on the Blockchain cannot be trusted. Existing security certification programs that critical industries like military, government, and banking rely on such as FIPS,

Common Criteria, and PCI also use accessors as their root of trust so this threat is well understood and is mitigated by certification programs for auditors.

Within such certification programs, auditors are required to seek regular re-authorizations, demonstrate continued training activities, and participate interlaboratory comparison programs. Additionally, the Governing Body can employ a dedicated certification program manager, overseeing all authorized auditors and their work. To prevent nation state coercion on authorized accessors, requirements may be put in place to limit jurisdictions under which accessors can operate.

5.6.3 Unrecorded Process Change

Process audits are updated on a regular basis to ensure Accessor audits accurately reflect reality. However, it is possible that a process owner may change the process in between assessments to circumvent security requirements. For example, a Key Generator may switch the process used to generate a private key yet use the same process ID as the accessed process. Or, a manufacturer may swap the component used to protect keys to a different component that is much less secure.

This threat is also well-understood in the security industry and is mitigated by unannounced audits and product sampling along various points of the supply chain. It can also be mitigated by requiring process owners to be third-party trusted Manufacturers. For example, a Secure Component Manufacturer in a domestic jurisdiction may supply components to foreign manufacturers in a way that takes the process ownership out of the hands of the foreign manufacturer. Such an arrangement is described in Section 5.5.1.

5.6.4 Incorrect Audits

A malicious actor may attempt to deceive an Authorized Accessor by providing a process to audit that is different than the process that is used in production. This threat is also well understood in the security industry and can be mitigated with careful assessment process design.

5.7 Requirements and Recommendations

From the use cases described in the previous sections, a set of high-level requirements and recommendations can be derived. These provide a framework for the technical and governance analysis in subsequent sections.

5.7.1 Eco-System Requirements

The following ecosystem requirements are made:

Requirement 1 The Blockchain shall be designed for the benefit of a nation state (e.g. USA) or a group of nation states (e.g. the European Union).

Requirement 2 The Blockchain shall prevent collusion from foreign nation states.

Requirement 3 The Blockchain should allow the inclusion of foreign markets. This may imply that each nation state or group of nation states will operate their own instance of the Blockchain.

Requirement 4 The Blockchain shall support the main use cases described in Section 5.3, and the detailed use cases described in Section 5.4.

Requirement 5 The Blockchain shall support the supply chain and distribution aspects described in Section 5.5.

5.7.2 System Architecture Requirements

The following system architecture requirements are made:

Requirement 6 The Blockchain shall store a globally unique entry for every generated public/private key pair registered on the Blockchain.

Requirement 7 The Blockchain shall store audit assessments for the creation and provisioning of private/public key pairs as well as the mechanisms to protect access to them. The assessment shall include a reference to the Authorized Assessor who completed the assessment.

Requirement 8 Every public/private key pair entry on the Blockchain shall include a reference to the applicable audit assessment.

Requirement 9 The Blockchain shall allow the recording of updates to previously stored records.

Requirement 10 The Blockchain shall provide an API interface for relevant actors to store transactions (write) on the chain.

Requirement 11 The Blockchain shall provide an API interface for relevant actors to read information from the chain. Read operations shall return the most up-to-date data.

Detailed technical requirements are found in subsequent sections.

5.7.3 System Performance Requirments

The following system performance requirements are made:

Requirement 12 The implementation of the Blockchain, compliant with requirements and recommendations in the present document, shall have a throughput capability to service all DER devices globally.

Requirement 13 The Blockchain shall support a throughput target of 1,000,000,000 transactions/year.

Requirement 14 The Blockchain may achieve the maximum throughput in stages as market adoption grows. No specific date is defined as per when the total market needs to be met.

Requirement 15 The Blockchain shall support batch processing of key registrations from the Manufacturer.

NOTE: The Blockchain will need to specify a maximum latency for completing an individual write or read operation, under regular operation conditions, to allow Manufacturers to better integrate the key registration into their manufacturing processes. The present document does not specify a maximum latency.

6 Blockchain

Just as there are many different kinds of databases there are also many different implementations of blockchain. The use cases described in Section 5 will be used to inform how the Blockchain should be designed. This section describes the high-level architectural implementation choices available, discusses the pros and cons of the choices, and lists implementation requirements for the present document's use cases. Section 6.1 focuses on participation and consensus implementation and Section 6.2 focuses on transaction implementation.

6.1 Blockchain Architecture

There are two main types of blockchains- permissionless and permissioned. The choice of which one to use has implications for security, performance, and governance.

6.1.1 Permissionless blockchain

Permissionless systems, also called "fully decentralized" systems, like Bitcoin and Ethereum are completely free of trusted authorities. Anyone is free to participate in the consensus process without permission, hence the name "permissionless". The most widely used permissionless consensus scheme is Proof-of-Work (PoW). There are also other permissionless consensus schemes like Proof-of-Stake (PoS) [3]. For illustration purposes we discuss Proof of Work.

PoW combines computationally difficult puzzles and economic incentives to achieve consensus. Thousands of miners contribute to the consensus process by computing hashes and the miner that finds a suitable hash first can propose the next block of the chain. This process does not provide immediate consensus, because sometimes more than one miner can find suitable hashes roughly at the same time, and as a result the chain forks. But thanks to economic incentives that are implemented as block rewards one chain branch eventually becomes longest and all the system participants can safely trust what is recorded on that chain. Because of this eventual consensus mechanism, permissionless consensus schemes like PoW are slow. In Bitcoin, for example, transaction finalization takes up to one hour.

The security of PoW systems relies on several trust assumptions. Probably the best-known trust assumption is that the majority of the mining power is not malicious. Otherwise an adversary that controls the majority could launch a 51% attack, where the adversary creates a long fork in the chain, and thus violates chain integrity.

Although systems like Bitcoin have thousands of consensus participants (miners), the mining power is concentrated in a small number of large mining pools. Most Bitcoin miners decide to join a pool due to economic incentives that make individual mining non-profitable. According to a recent study [10], a majority of the mining power is controlled by a couple of pools and there have been moments when a single pool (Ghash.io) has controlled

the majority of the mining power. Accordingly, although existing permissionless systems like Bitcoin may seem highly decentralized, in some ways such systems provide a low degree of decentralization. Whether it is possible to build a permissionless system that remains highly decentralized is currently an open research question.

Despite this centralization due to mining pools, permissionless systems can be viewed as highly resistant to attack because of economic incentives. A 51% attack on the system would trigger a loss of trust in the system, which may trigger a currency sell-off that results in a sharp reduction in the value of the Bitcoin held by the miners. This is the reason why the largest mining pools often artificially constrain their computing power- it maintains the value of their cryptocurrency assets.

The PoW consensus algorithm used by Bitcoin and Ethereum uses large amounts of energy. In recent years Bitcoin mining has consumed over 50 Twh of power. As a result, new permissionless consensus algorithms have entered the market such as Proof of Authority. These new algorithms have not had enough time on the market to validate their security properties.

Permissionless blockchains typically have loose governance by miners. In Bitcoin each minor decides on its own, or as part of its pool, whether to upgrade to new proposed software. This has caused issues when there is not clear consensus, such as in 2017 when many miners did not agree with a proposal to change the block size. The result was a hard fork (splitting a single system into two separate blockchain systems) that created Bitcoin Cash.

6.1.2 Permissioned blockchain

Permissioned chains rely on a trusted authority that can appoint a set of consensus nodes. These consensus nodes could be each operated by a different company, public institution or other reputable entity. Because one needs permission from the trusted authority to become a new consensus node, these systems are called “permissioned” or “distributed” (as opposed to decentralized) systems.

In a setting where a relatively small number of pre-defined consensus nodes exist, consensus can be achieved efficiently and securely using a so-called Byzantine-fault tolerant (BFT) consensus protocol. The term Byzantine-fault means that some of the nodes can deviate from the correct protocol execution in arbitrary ways, which covers node compromise by an external attacker, malicious inside administrator and other similar attacks.

BFT consensus protocols have been studied for decades. A classical theoretical work by Lamport et al. from the early 1980’s shows that there are solutions to the BFT consensus problem only when there are less than 1/3 faulty nodes [11]. This is why any BFT consensus protocol can tolerate at maximum 1/3 compromised nodes. Another classical work in this field is an efficient and formally-proven BFT consensus protocol called Practical BFT (PBFT) by Castro et al. [12]. This protocol is commonly used in modern permissioned blockchain systems. Recently, there has been significant research into more optimized BFT consensus protocols that can have slightly better scalability or other improvements. The research community has peer-reviewed widely available public

proofs of schemes such as Practical BFT and HotStuff and determined that these schemes indeed have this tolerance. There have been many new BFT-based schemes introduced in the past few years that modify the BFT protocol to increase performance, but many of these schemes either do not have a published mathematical proof or have not gone through a peer review process.

The trust assumptions of BFT protocols, and thus permissioned blockchains, are simple and well-understood. To violate consensus, the adversary needs to compromise at least 1/3 of the consensus nodes. For example, when 30 consensus nodes are used, the adversary needs to gain control of 10 different organizations. If each organization protects their consensus node appropriately, this is a difficult task for an external adversary. The other possible way to violate consensus is to have at least 1/3 of the organizations collude. If the organizations are chosen carefully and they are reputable ones, this is unlikely to happen. Importantly, in permissioned chains the consensus nodes can be chosen based on the security needs of the application such that they are generally trusted by the users of the blockchain. Permissioned blockchains are significantly faster than permissionless systems. A typical transaction finalization takes 1-2 seconds, instead of one hour in Bitcoin [13]. The operation of permissioned chains is also more efficient, as expensive mining (hashing) is not required.

6.1.3 Comparison: Permissioned vs. Permission-Less

Table 2 summarizes the aforementioned differences between permissionless and permissioned systems.

- Performance: Characteristics such as transaction throughput and latency
- Energy use: Amount of energy required to operate the blockchain
- Updates: Process used to update the blockchain software
- Trust model: Amount of trust required in the decision making of the blockchain
- Complexity: Amount of resources and coordination needed to run the blockchain
- Node incentives: How consensus node operators are incentivized to operate a node

Category	Permissioned BFT	Permissionless Proof of work
Performance	High	Low
Energy Use	Efficient	Expensive
Updates	Controlled	Miner Dependent
Trust Model	Several Known	Many Unknown
Complexity	Several Servers	Unbounded
Node Incentives	Usually Transaction Fees	Usually Mining

Table 2: Permissioned vs. Permissionless Blockchains

6.2 Blockchain Transactions

6.2.1 Fixed Transactions

Bitcoin uses a fixed data format to store transaction data. The format can only be used for sending Bitcoin from one address to another or rewarding Bitcoin for mining.

The benefit of fixed transactions is higher security once the blockchain code is finalized, because it is not possible to introduce potentially harmful features once the system is operational. However, if there is a problem with the blockchain code that is found during operations it requires all consensus nodes to update their software with new code that contains a fix. Another implication of requiring fixed transaction formats is that for the present document's use case existing Layer 1 code must be rewritten to implement the aforementioned uses cases since existing fixed transaction blockchains are designed for different use cases. Existing blockchain projects such as Hyperledger Fabric and Hyperledger Besu cannot be used if a fixed transaction format is desired.

6.2.2 Smart Contract

A smart contract [14] is a program that is executed by the consensus participants and its execution results are stored on the blockchain as new transactions. In other words, smart contracts allow blockchains to run arbitrary applications rather than a single specific application (such as digital currency in the case of Bitcoin).

Smart contracts enable the implementation of new types of applications that benefit from the advantages of blockchain technology. Compared to traditional contracts and financial applications, smart contracts can enable better business automation (a smart contract can be executed automatically by the blockchain system when its execution conditions are met), increased transparency (the business logic and transaction correctness can be verified by anyone from the chain), high availability (the contract's execution cannot be prevented by a single contract participant), and better privacy (business partners can enter contracts without using or revealing their real identities).

Blockchains such as Ethereum and Hyperledger Fabric use smart contracts to store data on the chain. The Layer 1 software implements a virtual machine that runs the instructions contained in the smart contract. To store data on the blockchain one must write a smart contract that takes a data input and creates the transaction that gets stored on the chain. Smart contracts can be created and added to the blockchain at any time.

6.2.3 Comparison: Fixed Transactions vs. Smart Contract

Table 3 compares the use of fixed transactions to smart contracts for the contemplated use case.

- Additional Features: Amount of work it takes to implement new features once the blockchain is operational.

- Defect Risk: Risk of problems being discovered and exploited. It is important to note that permissioned systems may institute processes that mitigate risk.
- Leverage Established Technology: Blockchain developers' freedom to use software that has already been tested and hardened in the market.
- Upgrade complexity: How hard it is to make changes to the system.

Criteria	Fixed Transactions	Smart Contracts
Additional Features	Hard	Easy
Defect Risk	Medium	High
Leverage Established Technology	No	Yes
Upgrade Complexity	High	Medium

Table 3: Fixed Transactions vs. Smart Contracts

6.3 Blockchain-level Security Threats

Blockchain was developed as a technology for increasing trust and security. As an example, by binding subsequent transactions ("blocks") by hash-values of their predecessors, blockchain delivers a strong method for assuring integrity of its data as a primary security goal.

But, of course, data and information stored in such a distributed ledger are of high value and therefore can be an interesting target for attackers. In the following we have listed possible threats for the Blockchain proposed in the present document. Possible threats are those that attack a corresponding vulnerability that exists in the system under consideration. For calculating risk factors for a specific threat one needs the probability of an incident to take place and its possible impact. Both factors are hard to estimate and so we will focus on threats and not on risk values within the present document.

6.3.1 What to Protect - Main System Assets

First of all, we have to clarify which assets of the system under consideration are most valuable and therefore may act as targets for attackers. A system which records private key information of a critical infrastructure – which includes DERs – acts as an anchor of trust for all involved stakeholders. Stakeholders must be sure that

- all stored information concerning private keys of DERs is correct and unaltered,
- information which is added to the system is not altered during any transactions, and
- only authorized organizations and persons can get access to the system and all authorized users can get access when it is needed.

With these protection targets in mind it is easy to define the most important assets of the Blockchain: first of all, the integrity of the private key related data stored in the blockchain. Secondly, the integrity and availability of

transaction processes adding new information blocks to the system. And last but not least, secure authentication and availability of access to nodes in the block chain.

6.3.2 Who are the "black hats" – Attacker profiles

As mentioned above the Blockchain is part of a critical infrastructure. That means that the availability of the service delivered by the critical infrastructure is a prerequisite for other relevant systems. In the case of energy and DERs the impact of the lack of this resource is obvious: collapse of IT-systems, of medical care, of public transportation – to name only a few.

Therefore we must be aware that the most important malicious actors are the nation state attacker and big terrorist organizations – for them the incentive of carrying out attacks on DERs as part of the energy infrastructure is very high and they have the required financial, technological and organizational resources.

One other important possible attacker is the malicious insider – the so called "bad admin". The incentive may not be a blackout, as described above, but often financial goals or just revenge. To be clear: in most cases it is very difficult to realize effective technical protection measures against insider actions – so in the following, countermeasures against such bad admin actions are not pursued further. In general, the best measures are satisfied and loyal employees on the one hand and on the other hand proper detection measures of conspicuous activities which are communicated within the whole organization. The latter follows the guiding principle of "no security by obscurity" – if a measure can be circumvented by an attacker simply by knowing that it exists, the measure is worth nothing. So it is better to tell everyone that relevant activities are monitored – this increases trust in your organization and can also serve as a deterrent.

6.3.3 Attack Scenarios

There are many possible attack scenarios concerning a blockchain system, but many of them are more theoretical without proof of concepts or real-life attacks based on these procedures. Additionally, we consider the proposed blockchain implementation as a permissioned blockchain to select relevant attacks.

(A1) Hostile takeover of 1/3 of consensus nodes

A hostile takeover is a severe threat against blockchain's data integrity, because an attacker can prevent or manipulate authorized transactions or may be able to compromise existing data such as adding duplicates. But the effort for the attacker is high: for a permissioned blockchain he needs to takeover at least 1/3 of all consensus nodes, which was shown for comparable systems based on BFT consensus protocols by Lamport et al [15] in the early 1980's.

Protection Measures

First of all, a good and effective governance of the Blockchain by the Governing Body is required. This means that the correct choice of organizations operating consensus nodes is of vital importance. Technical and organizational policies and guidelines are needed, which are obligatory for each consensus node. And there must exist rules concerning the admissibility of audits and the withdrawal of approval in case of violation of guidelines.

On the technical level, consensus nodes must be protected by state-of-the-art network and endpoint security measures. Network security measures include but are not limited to correctly configured firewalls, intrusion detection and prevention systems and proper network segmentation. Endpoint security means hardened operating systems, regularly performed vulnerability scans and further measures.

(A2) Exploitation of Smart Contract vulnerabilities

Smart contracts are distributed applications bringing business logic to a blockchain system. Smart Contracts are pieces of software, therefore may contain bugs and vulnerabilities which can be exploited by an attacker. These may serve as APTs (Advanced Persistent Threats), malware which is not observed during normal operation but can be initiated at any time in future.

Protection Measures

For mitigation of this risk, it is crucial that software developing organizations are trustworthy and have a Secure SDLC (Secure Software Development Lifecycle) in place. This includes software development on the basis of common security guidelines and pair programming (four-eyes principle). Every Smart contract has to be penetration tested and code-reviewed by a third, independent expert.

Operation of such software has to be accompanied by extensive logging of all security relevant parameters and processes. The logged values have to be monitored automatically – including proper alarm mechanisms - and there must be carried out audits of all logs – not only findings - in an at least daily manner.

Further attack scenarios

There are further attack scenarios with lower probability or impact, which are not discussed here in detail: DoS (Denial-of-Service) attacks on 1/3 (or more) consensus nodes, DoS-attacks on individual consensus nodes, Sybill attacks misleading users of the system, attacks on blockchain protocols, and supply chain attacks.

As this Blockchain is a tool of trust for an important part of our energy supply system, it is of high importance to develop it on the basis of best security practices. This covers not only the above described

threats and measures, but also the security awareness of all stakeholders within this system – during its whole lifecycle: planning, realization, and operation.

6.4 Requirements and Recommendations

6.4.1 Blockchain Architecture

The following system performance requirements are made:

Requirement 16 The Blockchain shall implement a permissioned system.

6.4.2 Consensus Protocol

Requirement 17 The Blockchain should adopt a BFT-based consensus protocol with a public proof that has been peer-reviewed by the relevant research community.

Requirement 18 The Blockchain should adopt a consensus protocol that has been in use for some time, with no recent problems exposed.

6.4.3 Blockchain Transactions

Requirement 19 The Blockchain should use smart contracts to implement the use cases described in Sections 5.3 and 5.4.

Requirement 20 The Blockchain should be based on a public open-source project.

7 Governance

The previous section describes the Blockchain architecture. This section discusses how the Blockchain is operated and governed.

7.1 Governance Domains

Blockchain governance defines the process of how decisions are made regarding setting up, operating, and updating a blockchain system.

7.1.1 System Setup

The tasks in system setup include designing and finalizing the technical specification for the system, defining the processes used in the system, and specifying any requirements for operating the system. For example, a blockchain governing body (the Governing Body in the present document) needs to define the process for incorporating a new consensus node operator. Or, the governing body needs to set the security requirements an entity must meet in order to host a consensus node. The creation of the present document can be viewed as a system setup task. Other setup tasks include defining system roles and responsibilities as well as the mechanism for authorization and revocation of such roles and responsibilities.

7.1.2 Operations

Once the system is set up, governance is also responsible for managing the blockchain according to the specifications and descriptions defined during system setup. A blockchain's governance may make decisions such as who is permitted to host a consensus node, managing growth of consensus nodes, determining which smart contracts are allowed to run, policing bad actors, and changing documents defined during system setup, for example to increase security requirements.

7.1.3 System Updates

All highly utilized software systems need updates from time to time. Blockchain governance determines how these software updates happen. For example, governance may determine when an updated is needed, when a new software version is ready to deploy, and how the new software is deployed.

7.2 Governance Mechanisms

There are several different types of governance mechanisms used in blockchain systems.

7.2.1 Ad Hoc

Ad hoc governance is what Bitcoin has, as described in Section 6.1.1. Consensus node operators decide for themselves whether they want to upgrade or not. If there is disagreement the blockchain forks.

7.2.2 Governing Entity

Some blockchain systems are governed by a legal entity, as is described in Section 5.2. This legal entity can be an existing company that acts as the linchpin of the ecosystem, such as Walmart for a supply chain blockchain implementation. The existing entity can also be a government agency (e.g.- the government of Venezuela).

Sometimes a consortium is a blockchain's governing body. Consortiums have been around for decades and are the entity of choice for creating industry technology standards such as Wi-Fi and Bluetooth. Blockchains can also be governed by a joint venture between participating companies.

When there is a legal entity governing a blockchain, there are two levels of governance. At the highest level there is the entity's board of directors which, in corporate law, has the ultimate decision-making power for the entity. The governing layer below the board of directors is the consensus node operators. Since consensus nodes are mostly operated by companies that are approved by, yet independent from, the blockchain's governing entity there is often governance amongst the consensus nodes as well. The process of approving a transaction to be added to the chain is a governance process, albeit one that is often embedded in the blockchain's software (the consensus protocol). The governing entity may require consensus node operators to contractually follow any decisions made by the board of directors as a condition of hosting a consensus node.

7.2.3 Contractual

A blockchain can be governed by a contract between all the companies involved with the blockchain. No new entity needs to be created, but the contract needs to be amended each time a new member takes on governing responsibility.

7.2.4 Decision Implementation

Once a governance decision is made the decision must be implemented on the blockchain. This usually involves some method of cryptographic signing with private keys so that the blockchain software recognizes the authority of the new code that implements the decision. This signing can be done by a single private key controlled by a trusted party (e.g.- a consortium), multiple signatures from a predefined minimum number of board members, or multiple signatures from a predefined minimum number of consensus nodes.

7.3 Governance-Level Threats

Attacks on governance are often overlooked and must be addressed with governance requirements.

7.3.1 Shared Interest Threats

In many ecosystems there are multiple industry segments that often have differing interests. For example, in cybersecurity regulation governments desire to mandate back doors so criminal surveillance is possible, whereas industry solution providers do not like back doors because it reduces the value of the security solution to end customers. Composition of a blockchain's governing entity can affect the ability of one industry segment to enact decisions that benefit it at the detriment of others. If 90% of a blockchain's board of directors comes from one industry segment it makes such one-sided decisions rather easy and ultimately reduces the value of the system to the whole ecosystem.

7.3.2 Government Coercion

As detailed in Section 6.3 the energy grid of any nation is a high value target for attackers. Unfortunately, building a permissioned blockchain system that protects all nations to the highest degree is not possible. To protect Country A from collusion attacks spearheaded by a collection of other countries, most of the consensus nodes should be under the jurisdiction of Country A. However, if this is done then it is a straightforward path for Country A to launch an attack on Country B that also relies on the system for security.

Conversely, requiring the consensus nodes in a system to be within the borders of a single country may not be an optimal solution for high availability either, especially since manufacturing happens all over the world.

Initially the Blockchain may only be used in the California DER market, specifically by California utilities and other communication endpoints. However, over time usage may expand to the rest of the United States and eventually the world. If the United States desires to have ultimate protection against collusion by foreign nations, the Blockchain could require that all consensus nodes be controlled by US-based entities hosted in a US jurisdiction. However, this obviously sacrifices geographic diversity and reduces the value of the system for the rest of the world. A compromise position is to allow up to one-third of consensus nodes to be controlled by a foreign entity or located outside US jurisdiction. A position that treats all nations equally is to specify that no nation can control one third or more of the consensus nodes. It is quite possible that there must be multiple blockchain governance systems to meet the security needs of all nations.

7.3.3 Company Infiltration

A company is comprised of a collection of individuals. If a company contributes to governance of the Blockchain attacking the company's employees this is a security threat, especially if multiple companies responsible for governance are attacked through their employees. Attackers can steal the authorization credentials of key

employees (e.g.- spear phishing). Attackers can influence key employees through methods such as bribery. Attackers can implant malicious personnel in key roles. This is a well-understood security threat. Mitigation for this threat encompasses requirements on access control as well as a large governing body so that the number of individuals that need to be compromised increases.

7.4 Consensus Node Governance

7.4.1 Maximum consensus nodes

Decisions on governance have very real implications for a permissioned blockchain. As mentioned in Section 6.1.2, permissioned blockchains that use BFT consensus become compromised when one-third or more of the consensus nodes are malicious. One may think to increase the number of consensus nodes so attackers must compromise many nodes to successfully one-third attack, but BFT consensus complexity increases exponentially with the number of consensus nodes and there is a corresponding performance reduction. Practical BFT is shown to deteriorate significantly above 30 consensus nodes. In a lab environment, HotStuff (a modification of BFT consensus) has been shown to be effective with up to 100 consensus nodes [9]. Given the threats listed in Section 7.3, blockchain governing bodies must be careful in choosing the composition of consensus node operators.

7.4.2 Minimum consensus nodes

As mentioned above, BFT consensus tolerates up to one-third of consensus nodes being malicious. This means that a BFT blockchain with three or fewer consensus nodes is actually a less secure system than a centralized system with only one server because an attacker only needs to compromise one of the three available consensus nodes to compromise the system. A blockchain must have at least four consensus nodes for the blockchain to be more robust than a centralized system because at that number an attacker would need to compromise two malicious nodes to compromise the whole system. With seven consensus nodes, an attacker would need to compromise three nodes.

7.5 Requirements and Recommendations

7.5.1 Governance Domains

Requirement 21 The Governing Body of the Blockchain shall be a consortium.

NOTE: The present document does not address governance on the consensus node level. This decision is left to the Governing Body.

NOTE: The present document also does not address requirements for implementing consortium decisions in the system software. This is also left to the Governing Body.

7.5.2 Governance Mechanisms

Requirement 22 To ensure security, the production Blockchain shall operate with a minimum of seven consensus nodes.

Requirement 23 Based on Requirement 2 above, to protect against foreign collusion attacks against the implementing nation state yet at the same time allow for geographic diversity, the number of consensus nodes in the production Blockchain operated by non-nation state companies shall be less than one-third of the total number of consensus nodes.

Requirement 24 To ensure there are not siloed systems for each country, the Blockchain should provide a path for other countries to adopt the Blockchain for their own domestic entities.

Requirement 25 The number of consensus nodes for the production Blockchain operated by an industry segment shall be less than one-third of the total number of consensus nodes. The definition of industry segment is left to the Governing Body. Examples of industry segments the Governing Body should consider are utilities, inverter manufacturers, auto manufacturers, service providers, and chip vendors.

Requirement 26 The number of governing board members held by non-nation state companies shall be less than the minimum number of members required to pass a motion (usually majority).

Requirement 27 The number of governing board members held by any industry segment shall be less than the minimum number of members required to pass a motion (usually majority).

NOTE: The Blockchain will need to be protected against external one-third attacks. Therefore, consensus node operators will adhere to security requirements that will be defined by the Governing Body. The present document does not specify specific security requirements.

8 Data Model

8.1 Introduction

The Blockchain is an append and read application. Data gets appended to the Blockchain when private keys are provisioned, and data is read by communication endpoints when a private key is used to establish a secure communication channel. Blockchain technology does not delete data- all data on the Blockchain stays on the Blockchain. Blockchain technology does not update data on the chain, but instead adds new transactions that have the same effect as updating data and keeping a record of all changes.

The data on the Blockchain is security-related information on a private key, not the private key itself. The private key is represented by a statistically unique address that is based on a hash of the private key's public key. On the Blockchain, a private key address is associated with the provisioning process that created and stored the private key. The provisioning process is also represented by a unique ID on the Blockchain. When a private key is created, the provisioning process puts the address of the private key on the Blockchain. The key creation and provisioning process is audited by authorized accessors. These accessors put the resulting audit information on the Blockchain so that every private key on the Blockchain is associated with information that describes how it was created and protected. Authorized Accessors are the only entities that can put provisioning process security information on the Blockchain. Accessors are also uniquely identified elements on the blockchain.

8.2 Confidentiality Considerations

The Blockchain data is available to all consensus node operators and any other entity that is permitted to hold the data. The most straightforward way to use a blockchain is to have security information about each provisioned private key readily available on the blockchain. However, this model leaks information. For example, it could reveal the number of devices a Manufacturer produces. To protect against such information leaks a few possible techniques can be used. The first technique is that the provisioning process does not publish separate information for each device, but rather information per device type, class or batch. This technique is used by the Blockchain's data design by associating private keys to provisioning processes, not Manufacturers. The second technique is that the ability to read data stored on the blockchain is limited to authorized entities only. This authorization is controlled by the Governing Body. The third technique is to complement the consensus nodes with SGX enclaves that can process transactions privately and include them in encrypted format into created blocks. The fourth possible technique is to store cryptographic commitments of data on-chain and demonstrate its correctness using zero-knowledge proofs. The present document tasks the Governing Body with identifying sensitive information in the data model and protecting it appropriately.

8.3 Data Elements

This section describes the data elements the blockchain will store, the parameters associated with each element, and how the elements interrelate (the data model). This section is not a software specification and is meant to give a high-level description of what the data model looks like. A specification is left for future work. The application programming interfaces (APIs) for the Blockchain are described in Section 8.4.

8.3.1 Key Data Element

A key data element represents a single private key. Its details are shown in Table 4.

Parameter	Value	Description
key_id	string	Unique ID based on a hash of the public key
process_id	string	Unique ID of the process that created the private key
status	enum	Status of the key such as Active, Revoked, EOL, etc.

Table 4: Key Data Element

Each key's unique identifier is a string of characters based on the hash of the private key's public key. The generation of the *key_id* is beyond the scope of the present document.

Each key is also associated with the process that created and provisioned the key, and defines how the key is stored and protected. A key data element only stores information that is specific to an individual key. Currently the status field is the only information specified. Other information that may be added in the future could be time of key creation, but these additional parameters are left for future work.

8.3.2 Process Data Element

The process data element contains parameters that describe the process under which an individual key has been generated and protected on the equipment. It links the Process Assessment and the Component Assessment to the owner of the process. Every key data element links to one process data element. A process data element is typically referenced by many key data elements, e.g. all equipment belonging to the same product family manufactured on the same manufacturing line. The process data element details are shown in Table 5.

Parameter	Value	Description
process_id	string	Unique ID of the process
owner_id	string	Unique ID of the entity that owns the process
key_generation_id	string	Unique ID of the key generation assessment, which includes key generation and key provisioning (Process Assessment).
key_protection_id	string	Unique ID of the key protection assessment, which includes key storage, protection, and access control (Component Assessment).

Table 5: Provisioning Process Data Element

Every key created by the process has the same creation mechanism, key format, key transfer mechanism, storage location, and protection mechanism. If any of these characteristics changes then another process ID needs to be created. A process is additionally be associated with an entity (manufacturer, distributor, service provider, etc.) that owns the process. A process is audited regularly to confirm the process characteristics.

8.3.3 Assessment

The preponderance of data is stored in the assessment data element. The assessment data element describes the security characteristics of the provisioning process element. Its details are shown in Table 6.

Parameter	Value	Description
assessment_id	string	Unique ID of the audit
assessor_id	string	Unique ID of the accessor
past_assessment_id	string	Original process audit. If this value is not empty it means this audit as a renewal of a previous audit and other parameters may be empty.
assessment_result	array	Assessment results. See following tables for details.
completion_date	integer	Date assessment was completed
approval_date	integer	Date assessment was approved by Governing Body

Table 6: Assessment Data Element

The key generation and key protection assessments both have their own unique structure contained within the *assessement_result* field. Key generation is described in Table 7 and key protection is described in Table 8.

Parameter	Value	Description
cryptographic_strength	integer	strength in bits, e.g. 128
reference	string	SHA-256 hash of any data that is stored off-chain so the off-chain data can be validated
key_generation_entity	enum	The type of hardware that generates the key, such Onboard SE, Onboard processor, Offline appliance, Online appliance, etc.
entropy_bits	integer	number of true random bits, e.g. 64
entropy_source	enum	how entropy was created
location	string	Location the key was created: ISO 3166-1 country code + (ZIP code)
certifications	hex string	Bit field that records security certifications held by the provisioning process such as FIPS, Common Criteria, etc.

Table 7: Key Generation Assessment Element

Parameter	Value	Description
key_storage	enum	key storage mechanism such as secure hardware, secure memory, unprotected memory, etc.

Parameter	Value	Description
access_control	enum	how access to the key is controlled, e.g.- hardware, privileged software, non-privileged software, none
device_integrity	enum	Secure boot (HW protected), secure boot (SW protected), no protection
certifications	hex string	Bit field that records security certifications held by the provisioning process such as FIPS, Common Criteria, etc.

Table 8: Key Protection Assessment Element

There are many assessment elements per process element as assessments need to be repeated or renewed periodically. Renewal assessment elements need not store characteristics of the process. Instead, renewal assessment elements may only store the ID of the original audit element that contains all the characteristics and the date of the audit. This limits the amount of data stored on the blockchain.

NOTE: An alternative implementation is to move assessment information from the audit element into the process element, so that each audit merely confirms the information in the process element.

8.3.4 Entity

An entity is any independent Actor involved in the Blockchain, including Authorized Accessor and Manufacturer (an entity that owns a provisioning process). Its details are shown in Table 9.

Parameter	Value	Description
entity_id	string	statistically unique address of the private key used by the entity, similar to a wallet address in Bitcoin
type	enum	the nature of the entity, such as company, individual, government body, etc.;
name	string	Name of the entity
rlid	string	reference to a TBD internationally accepted entity ID
role	hex string	Role the entity plays in the ecosystem (accessor, process owner, manufacturer, etc.). An entity can have multiple roles.
role_certification	string	reference to a certification for the role(s) given by the consortium

Table 9: Entity Data Element

Entities have tax IDs, so they are not devices or servers.

8.3.5 Data Model

Figure 22 illustrates how the above data elements interrelate. Arrows describe one to many relationships, where the arrow points from “many” to “one”. A provisioning process owner (entity element) can have many provisioning process elements. A provisioning process element can have many key elements. A provisioning process element can have many audit elements. An accessor entity element can also have many audit elements.

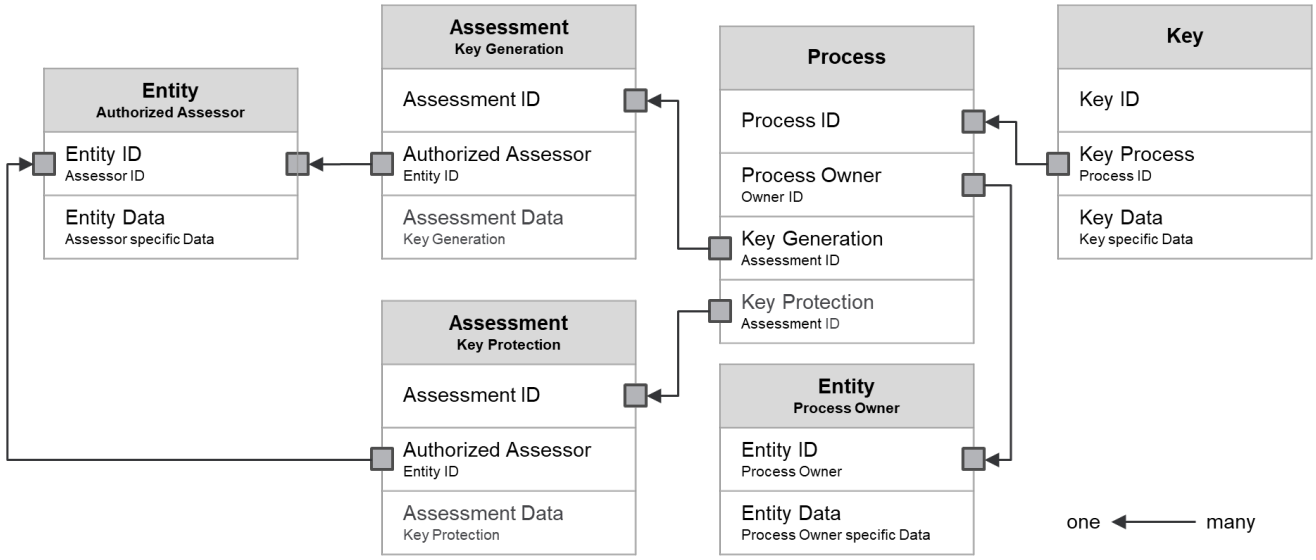

Figure 22: Data Model

To get information on a private key from the private key's public key one would hash the public key to create the address, look up the address on the Blockchain, look up the process element that created the address, and finally look up the latest audit element of the process. One could also obtain information on the accessor and process owner if desired.

8.4 API

The Blockchain can be accessed by an application programmer interface (API) so DER Servers can read information and authorized entities can write to the system. The API calls are shown in Table 10. This table is not a specification and is meant to give a high-level description of the API calls. A specification is left for future work.

API Call	Input	Output	Description
create_process	Process Owner (entity_id)	-	Create a process
create_assessment	Authorized Assessor (entity_id), Process (process_id), Assessment Result	-	Record an assessment for a specific process
create_key	Key Identifier (key_id), Process (process_id), Key Data	-	Record a key
update_key	Key Identifier (key_id), Key Data	-	Update key data for a specific key
get_key_info	Key Identifier (key_id)	Key Data, Assessment Data	Gets information on a key

Table 10: API Calls

The APIs are described as follows:

create_process:	Used by a provisioning process application or an Authorized Assessor to create a new provisioning process.
create_assessment:	Used by an Authorized Assessor to record an assessment for key generation or key protection.
create_key:	Used by a provisioning process to record a private key's address on the blockchain.
update_key:	Used by authorized parties to record status updates on a private key.
get_key_info:	Used to obtain security information on a private key.

8.5 Requirements and Recommendations

Requirement 28 The Blockchain should implement the data elements described in Sections 8.3.1, 8.3.2, 8.3.3, and 8.3.4, and the data model as described in Section 8.3.5.

Requirement 29 The Blockchain should provide the APIs as described in Section 8.4

9 Roadmap for Implementation

9.1 A Single System

The present document is a SunSpec open standard and is free for anyone to use. However, for the United States, and possibly the rest of the world, it may be counterproductive to have multiple implementations of the Blockchain competing for market share. For example, secure element manufacturers already manage their own proprietary private key certification records. It would be easier for them to only support one additional key creation API rather than several. Furthermore, as is detailed in Section 9.3 and 9.4 below, additional work is necessary to realize a production implementation and this additional work should be coordinated by a single governing body so there aren't differing APIs that would make integration more time consuming and error-prone.

9.2 Governing Body

One of the next steps to implement a production system is to designate a consortium that will be the Governing Body for the Blockchain, as is required in Section 7.5.1. The Governing Body could be an existing consortium that already governs a blockchain, an existing consortium that currently does not govern a blockchain, or a new consortium created for the specific purpose of governing the Blockchain. The Governing Body must ensure it follows the requirements and recommendations put forth above and design new processes and operations to continue the work described in the present document. This effort will clarify governance and maintenance details such as:

- Governing bylaws;
- Voting approval requirements for decisions (e.g.- majority or supermajority);
- The procedure for issuing permissions and certifications (node operators, Authorized Assessors, data users);
- Change control and change implementation.

9.3 Remaining Specification Work

The present document is just a first step to defining a full specification for the Blockchain. Tasks that remain unfinished include:

- Solidify the data elements and APIs;
- Further define Blockchain requirements such as consensus protocols and security practices;

- Choose an existing blockchain platform to build upon or write the specification for building a platform from scratch.

9.4 Implementation and Launch

The other major task is to implement the Blockchain in accordance with specifications and requirements. This requires resources for software development, test, pilots, scaling, security audits, consensus node hosting, and network set up. The Governing Body will need to create the proper incentives so that entities are willing to contribute these resources.

9.5 Regulations and Certifications

As of the date of this writing the California Public Utility Commission is in the process of determining cybersecurity requirements for its Rule 21 tariff. The impetus for the effort behind the present document came out of work group proceedings in the SunSpec Cybersecurity Work Group, whose recommendations will inform CPUC cybersecurity deliberations. The Blockchain can be viewed as a candidate for solving a potential requirement of the CPUC- ensuring the trustworthiness of cryptographic private keys.

On the federal level, Recommendation 4.6 of the U.S. Cybersecurity Solarium Commission's final report [11] states: "Congress should direct the U.S. Government to develop and implement an information and communications technology industrial base strategy to ensure more trusted supply chains and the availability of critical information and communication technologies". Provisioning and protection of private keys is a critical component of protecting communication technology supply chains and the system described in this document is in sync with this proposed strategy. Furthermore, the Whitehouse Executive Order on Securing the United States Bulk-Power System issued on May 1, 2020 puts restrictions on equipment that could be influenced by foreign equipment. Transparency with regards to private key creation and protection could solve the problem this executive order attempts to address.

10 Conclusion

The distributed energy industry has a number of characteristics that taken together necessitate a new tool for managing security. It has a highly diverse supply chain that consists of hundreds of manufacturers and service providers located all over the world. There is a wide range of power generation capacity and price points that require a flexible approach to security requirements. It will most certainly become a target of highly resourced nation state attackers. The Blockchain described in the present document does not solve all the industries cybersecurity challenges, but it does fill the holes in current cybersecurity regulations with regards to the integrity of private keys in a way that is informed by the aforementioned industry characteristics.

While the present document focusses on the protection of private key in DER equipment, the Blockchain can also cover key protection needs within other industries and market verticals. Additionally, the Blockchain allows the tamper resistant recording of other critical information, including transactional data, enabling new applications besides key protection transparency.

The present document encompasses the input and contributions of multiple key stakeholders in distributed energy, and these stakeholders are committed to executing the next step which is commercialization of the Blockchain.

APPENDIX A (Informative): List of Contributors

Arutyunov, Roman	Xage Security
Bosch, Jeff	OutBack Power, an EnerSys Company
Brakensiek, Jörg (Editor)	Wivity Inc.
Calò, Pietro	Fimer SpA
Erhart, David	Stem Inc.
Hanke, Ingo	SMA Solar Technology AG
Irwan, Susanto	Xage Security
Jose, Abraham	PG&E Corporation
Kostiainen, Kari	Wivity Inc.
Kuppuswamy, Sathishkumar	PG&E Corporation
Lum, Gordon	Kitu System Inc.
Tom, Alfred	Wivity Inc.
White, Bob	Fimer SpA