SANDIA REPORT
SAND2019-1490
Unlimited Release
Printed February 2019

Recommendations for Trust and Encryption in DER Interoperability Standards

James Obert, Patricia Cordeiro, Jay Johnson, Gordon Lum, Tom Tansy, Max Pala, Ronald Ih

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

SAND2019-1490
Printed February 2019
Unlimited Release

Recommendations for Trust and Encryption in DER Interoperability Standards

James Obert, Computer Systems Security Analysis R&D
Patricia Cordeiro, Mission Analytics Solutions
Jay Johnson, Renewable and Distributed Systems Integration
Sandia National Laboratories
Albuquerque, New Mexico 87185

Gordon Lum
Kitu Systems
San Diego, California 92111

Tom Tansy
SunSpec Alliance
San Jose, California 95117

Max Pala, Ronald Ih
Cable Labs
Louisville, Colorado 80027

Abstract

Recently developed Distributed Energy Resource (DER) interoperability standards include communication and cybersecurity requirements. In 2018, the US national interconnection standard, IEEE 1547, was revised to require DER to include a SunSpec Modbus, IEEE 2030.5 (Smart Energy Profile, SEP 2.0), or IEEE 1815 (DNP3) communication interface but does not include any normative overarching cybersecurity requirements. IEEE 2030.5 and associated implementation requirements for California, known as the California Smart Inverter Profile (CSIP), prescribe the greatest security features—including encryption, authentication, and key management requirements. SunSpec Modbus and IEEE 1815 security requirements are not as comprehensive, leading to implementation questions throughout the industry. Further, while the security features in IEEE 2030.5 are commonly used in computing platforms, there are still questions of how well the technologies will scale in highly-distributed, computationally-limited inverter environments. In this paper, (a) the elements of IEEE 2030.5 encryption, authentication, and key management guidelines are analyzed, (b) potential scalability gaps are identified, and (c) alternative technologies are explored for possible inclusion in DER interoperability or cybersecurity standards.

ACKNOWLEDGMENTS

This work was funded by the Department of Energy (DOE) Energy Efficiency and Renewable Energy (EERE) Solar Energy Technologies Office (SETO).

TABLE OF CONTENTS

1. Overview 10
2. DER Communication Protocols 12
    2.1. IEC 61850 12
    2.2. IEEE 1815 13
    2.3. SunSpec Modbus 13
    2.4. IEEE 2030.5 14
3. Cryptography and Trust 16
    3.1. Cryptographic Features of IEEE 2030.5 16
        3.1.1. CSIP implementation of IEEE 2030.5 PKI 17
        3.1.2. Authenticated Encryption 18
    3.2. Implementation with Embedded Systems 20
4. Gap Analysis 22
    4.1. IEEE 2030.5 Certificate Validity and Revocation 22
        4.1.1. Reasons for Using CRLs or OCSP 22
        4.1.2. Reasons for Not Using CRLs or OCSP 23
    4.2. Brute Force Attacks 24
    4.3. Enabling Encrypted Traffic Inspection 25
    4.4. DER Physical Security 26
5. Trust Alternatives 28
    5.1. Mobile Trusted Module 29
    5.2. Mobile Device Management Software 31
    5.3. Per-Application VPN 31
    5.4. ARM TrustZone 32
    5.5. Post-Quantum Crypto 33
    5.6. Distributed Trust Models using Blockchain PKI 34
        5.6.1. Blockchain Basics 34
        5.6.2. Blockchain Types 36
        5.6.3. Blockchain Consensus Algorithms 37
        5.6.4. Blockchain Limitations and Vulnerabilities 38
        5.6.5. Blockchain Used as a PKI 39
6. Recommendations for DER Trust and Encryption 41
    6.1. Secure Application Enabled Devices 41
    6.2. Legacy Device Support 41
    6.3. Device Secure Key Storage 42
    6.4. Certificate Maintenance 43
    6.5. Legacy Device Trust 43
    6.6. Physical Security 44
7. Conclusions 45

Appendix A 46
    A.1 Encryption Methods 46
        A.1.1. Symmetric Cryptography 46
        A.1.2 Asymmetric Cryptography 47
        A.1.3 Enabling Symmetric Cryptography via Asymmetric Cryptography 51

A.2 Digital Signing and Verification 52
A.2.1 RSA Signatures 52
A.2.2 Elliptic Curve Signatures 53
A.3 Public Key Distribution 54

FIGURES

Figure 1. Utility-to-DER communications options per CA Rule 21 and CSIP. 15
Figure 2: IEEE 2030.5 cipher suite components. 16
Figure 3. Protocol translator approach to DER communications. 21
Figure 4. Two-part Tunnel for TLS/SSL Interception. 26
Figure 5. Techniques for defending against various attacks on DER trust. 29
Figure 6: MTM Architecture 30
Figure 7: TrustZone Trusted Execution Environment. 32
Figure 8: Sequence of blocks. 35
Figure 9: Block elements. 35
Figure 10: Openchain Blockchain PKI. 40
Figure 11: Symmetric Encryption/Decryption. 47
Figure 12: Asymmetric Encryption/Decryption. 48
Figure 13: RSA Encryption/Decryption. 49
Figure 14: Elliptic Curve Operations. 50
Figure 15: ECC Encryption/Decryption. 51
Figure 16: Asymmetric Crypto Enables Symmetric. 52
Figure 17: RSA Signature/Verification. 53
Figure 18: Elliptic Curve Digital Signature Algorithm. 54
Figure 19: Public Key Infrastructure. 55

TABLES

Table 1: Trust and Cryptography Features in Common DER Communication Protocols. 12

NOMENCLATURE

AAA	Availability, Authorization, and Accounting
AC	Alternating Current
ACL	Access Control List
AES	Advanced Encryption Standard
ANSI	American National Standard Institute
BITW	Bump-in-the-Wire
BPS	Bulk Power System
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
CA	Certificate Authority
CBC	Cipher Block Chaining
CIA	Confidentiality, Integrity, and Availability
CIGRE	International Council on Large Electric Systems
CIP	Critical Infrastructure Protection
CPU	Central Processing Unit
CPUC	California Public Utilities Commission
CRC	Cyclic Redundancy Checking
CSIP	Common Smart Inverter Profile
DC	Direct Current
DCS	Distributed Control System
DDoS	Distributed Denial of Service
DER	Distributed Energy Resource(s)
DERMS	Distributed Energy Resource Management System
DES	Data Encryption Standard
DHS	Department of Homeland Security
DNP3	Distributed Network Protocol
DOE	Department of Energy
DoS	Denial of Service
DR	Demand Response
DSA	Digital Signature Algorithm
DSS	Digital Signature Standard
EAP	Extensible Authentication Protocol
ECC	Elliptic Curve Cryptography
ECC	Error Correcting Code
EO	Executive Order
EPRI	Electric Power Research Institute
EPS	Electric Power System
EPU	Electric Power Utility
ERO	Electric Reliability Organization
ESS	Energy Storage System
EXI	Efficient XML Interchange
FEMS	Facility Energy Management System
FERC	Federal Energy Regulatory Commission
FIPS	Federal Information Processing Standard

GCM	Galois/Counter Mode
GDOI	Group Domain of Interpretation
GOOSE	Generic Object Oriented Substation Event
HAN	Home Area Network
HTTPS	Hypertext Transfer Protocol Security
ICS-CERT	Industrial Control System Cyber Emergency Response Team
ICT	Information and Communication Technologies
IDS	Intrusion Detection System
IED	Intelligent Electronic Device
IoT	Internet of Things
IOU	Investor-Owned Utility
IPS	Intrusion Prevention System
IPS	Internet Protocol Suite
IPSec	Internet Protocol Security
ISO	International Organization for Standardization
LICs	Logical Interface Categories
MESA	Modular Energy Storage Architecture
MMS	Manufacturing Message Specification
NAT	Network Address Translation
NERC	North American Electric Reliability Corporation
NIST	National Institute of Standards and Technology
NSM	Network and System Management
OE	Office of Electric Delivery and Energy Reliability
OSGP	Open Smart Grid Protocol
OSI	Open System Interconnection
PCC	Point of Common Coupling
PCS	Power Conditioning System
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
PLC	Programmable Logic Controller
PPD	Presidential Policy Directive
PUC	Public Utilities Commission
PV	Photovoltaic
RAID	Redundant Array of Independent Disks
RBAC	Role-Based Access Control
REP	Retail Energy Providers
RFC	Request for Comments
Risk	Financial loss, disruption or damage to the reputation of an organization from some sort of failure of its information technology systems
RMP	Risk Management Process
Root-CA	Root Certificate Authority
RTCP	Real-time Transport Control Protocol
SCADA	Supervisory Control and Data Acquisition
SIWG	Smart Inverter Working Group
SNL	Sandia National Laboratories
SRTP	Secure Real-time Transport Protocol

SSL	Secure Socket Layer
Threat	Possibility of a malicious attempt to damage or disrupt a computer network or system
TLS	Transport Layer Security
UDP	User Datagram Protocol
VEN	Virtual End Node
VPN	Virtual Private Network
VTN	Virtual Top Node
Vulnerability	A weakness which allows an attacker to reduce a system's information assurance
WEP	Wired Equivalent Privacy
XML	Extensible Markup Language

1. OVERVIEW

Customer-owned Distributed Energy Resources (DER), such as photovoltaic (PV) inverters and energy storage systems (ESS), are being deployed rapidly in the US[1]. This growth is expected to continue—driven by a combination of low prices and regulations, e.g., renewable portfolio standards and public utility commission mandates. For instance, in California, new homes after January 1, 2020 will be required to be built with solar systems, due to standards adopted by the California Energy Commission[2].

These new DER devices are equipped with a range of physical and logical communication mechanisms to report performance data to owners and enable grid operator control and monitoring. These features provide greater flexibility for grid operators to run the power system efficiently but it expands the attack surface of the power system as well. Interoperable and dispatchable DER—as a relatively new player in the Internet of Things (IoT)—provides a range of opportunities and risks for the smart grid.

Unfortunately, IoT connectivity often outpaces security implementations, resulting in regular news of data breaches, distributed denial of service attacks, and other malicious activities from millions of relatively weakly protected, internet-connected devices. These attacks may be motivated by fraud, influence, profit, or pure disruption. Power system cyber resilience must be improved by increasing the difficulty of launching these types of attacks.

Worldwide, DER communication requirements and protocols are diverse. In the US, IEEE 1547-2018 requires newly-interconnected DER to communicate IEEE 1815, IEEE 2030.5, or SunSpec Modbus. The Smart Inverter Working Group (SIWG) led the multi-phased process to update California Electric Rule 21; one of the revisions to the interconnection standard described in the California Public Utilities Commission (CPUC) Decision 16-06-052 ordered the CA Investor-Owned Utilities (IOUs) to implement inverter communication protocols, with IEEE 2030.5 specified as the default protocol[3]. IEC 61850 is common in Europe and the rest of the world. OpenADR is under consideration as the default DER communication protocol in other countries such as Japan[4]. With this range of protocol options, there is a need to work toward a common set of data-in-flight cybersecurity requirements for DER communications and look at what future technologies could be implemented to better secure DER equipment and the power system as a whole.

In this report, since California is an early adopter of communications-enabled DER equipment, we focus on the benefits and challenges derived from IEEE 2030.5 implementation. Based on this analysis, recommendations for improvements to trust and encryption in DER communication networks are provided. The first set of recommendations are intended for near-term consideration and include modifications to the DER cybersecurity implementation strategy in California:

[1] R. Margolis, D. Feldman, D. Boff, "Q4 2016/Q1 2017 Solar Industry Update," NREL/PR-6A20-68425, April 25, 2017.
[2] M. Chediak, P. Gopal, B. Eckhouse, "California Becomes First State to Order Solar on New Homes," Bloomberg, May 9, 2018.
[3] CPUC, Decision 16-06-052, June 23, 2016, URL: http://docs.cpuc.ca.gov/publisheddocs/published/g000/m164/k376/164376491.pdf, accessed 6-29-18.
[4] OpenADR Press Release, "OpenADR Alliance To Extend its Automated Demand Response Standard to Help Utilities Manage Their Distributed Energy Resources" June 20, 2017.

1. Create an ecosystem that supports certificate revocation lists and establish a certificate policy to ensure all certificate authorities (CAs) have good security procedures.
2. Support updates for stronger encryption algorithms.
3. Clarify and standardize the interface between the DER network, aggregator DMZ, aggregator IEEE 2030.5 server/client, utility DMZ, and utility IEEE 2030.5 server such that locations where encryption are required are fully defined.
4. Add security requirements to the IEEE 2030.5 standard pertaining to aggregation servers.
5. Provide the option to dynamically re-provision DER devices such that non-expiring keys are still renewed after a certain period of time.
6. Improve physical and data-at-rest security so that keys cannot be extracted from DER equipment.

The second set of recommendations focuses on investigating longer-term trust and encryption alternatives for DER communication systems:

1. Investigate decentralized trust technologies, including blockchain.
2. Secure device keys using hardware security mechanisms like Mobile Trusted Module.
3. Prevent modification to DER firmware through TrustZone or other hardware-backed security features.
4. Prepare for a post-quantum computing environment by investigating new encryption algorithms.
5. Explore new cybersecurity features such as per-application virtual private networks.

2. DER COMMUNICATION PROTOCOLS

Revised in 2018, IEEE Std. 1547 defines interconnection and interoperability requirements for grid-interconnected DER. This standard enumerates three protocols as options for DER interfaces, but leaves the security requirements to be defined by the individual communication protocols. At this time, there are no overarching security requirements for DER interoperability.

Most of the solar industry's efforts in securing DER has focused on protecting data-in-transport, i.e., creating a secure data path between communicating devices. A summary of protocol-specific requirements is presented in Table 1. More information on these protocols are presented in "Cyber Security Primer for DER Vendors, Aggregators, and Grid Operators"[5]. In short, until October 2018, Modbus did not include any encryption, node authentication or key management features and therefore contemporary SunSpec Modbus implementations of the standard lack over-the-wire security. IEC 61850, IEEE 1815, and IEEE 2030.5 all require or permit Transport Layer Security (TLS) Encryption and X.509 certificates for node authentication. There are multiple differences in key management practices between protocols. The remainder of this section briefly highlights the cryptographic and trust requirements of several DER communication protocols.

Table 1: Trust and Cryptography Features in Common DER Communication Protocols.

Protocol/Security Standard	**Encryption**	**Node Authentication**	**Certificate/Key Management Notes**
IEC 61850/ IEC 62351	IEC 62351-3 requires TLS	X.509 Digital Certificates	IEC 62351-9 covers generating, distributing, revoking, and handling public-key and symmetric keys for groups (GDOI) but does not define the type of keys or cryptography
IEEE 1815/ DNP3-SA	VPNs and IPSec are recommended. TLS is optional. Multiple TLS cipher suites are permitted, but TLS_RSA_WITH_AES_128_SHA shall be supported at minimum.	X.509 Digital Certificates	IEEE 1815-2012 allows pre-shared keys but also includes methods for symmetric and asymmetric cryptography.
SunSpec Modbus	None	None	None
IEEE 2030.5/ CSIP	IEEE 2030.5 requires TLS. AES-128 in the Counter with Cipher Block Chaining – Message Authentication Code Mode shall be supported	X.509 Digital Certificates	IEEE 2030.5 requires key management by a public key infrastructure which shall use Ephemeral Elliptic Curve Diffie–Hellman key exchange with Elliptic Curve Digital Signature Algorithm signatures (ECDHE_ECDSA)

2.1. IEC 61850

The International Electrotechnical Commission (IEC) 61850 protocol provides end-to-end communications for power systems. IEC 61850 actually covers three communication protocols: MMS (Manufacturing Message Specification), GOOSE (Generic Object Oriented Substation

[5] C. Lai, N. Jacobs, S. Hossain-McKenzie, C. Carter, P. Cordeiro, I. Onunkwo, J. Johnson, "Cyber Security Primer for DER Vendors, Aggregators, and Grid Operators," Sandia Technical Report, SAND2017-13113, Dec 2017.

Event), and SMV (Sampled Measured Values). The cybersecurity features for IEC 61850 communications are defined in IEC 62351. TLS encryption, key management principles, and node authentication are specified, though specific cipher suites are not. IEC 62351-3 describes the TLS encryption requirements and X.509 authentication certificates for TCP/IP traffic[6], IEC 61351-4 covers the encryption and authentication requirements for MMS[7], and IEC 62351-6 standardizes VLAN use for GOOSE communications[8].

2.2. IEEE 1815

IEEE 1815 Standard for Electric Power Systems Communications-Distributed Network Protocol (DNP3) is recognized as a widely used protocol which started with no security features. Updates in IEEE 1815-2012 included a number of security upgrades commonly referred to as *DNP3 Secure Authentication (DNP3-SA)* which included encryption options and X.509 certificates, based on the IEC 62351-5 security standard. An analysis of DNP3-SA security features was previously conducted by researchers at the University of Oxford.[9]

2.3. SunSpec Modbus

Modbus is a serial communications protocol originally published by Modicon (now Schneider Electric) in 1979 for use with its programmable logic controllers[10]. Modbus has become a *de facto* standard communication protocol and is now a commonly available means of connecting industrial electronic devices. Modbus includes both a Remote Terminal Unit (RTU) variant that uses RS-485 and at Modbus/TCP variant that uses TCP/IP networks. Modbus RTU is the most common implementation available for Modbus but Modbus/TCP also widely used in the DER industry.

Recognizing both the popularity of Modbus and the lack of standardized information models, in 2009 the SunSpec Alliance embarked on a project to define standard information models for DER devices (e.g. inverters, meters, batteries). The result of this effort is the open SunSpec Modbus standard that is incorporated into IEEE 1547-2018 and adopted by California to provide grid-support functions defined in CPUC Electric Rule 21 requirements[11].

Modbus has traditionally had no encryption requirements whatsoever and has relied on bump-in-the-wire technologies for add-on security. In October 2018, the Modbus Organization announced the Modbus/TCP Security specification. This specification calls out TLS 1.2 and X.509 v3 certificates as building blocks of its over-the-wire security scheme. SunSpec Modbus implementations are compatible with the Modbus/TCP Security specification.

[6] IEC 62351-3, "Power systems management and associated information exchange - Data and communications security - Part 3: Communication network and system security - Profiles including TCP/IP," 2014.
[7] IEC 62351-4, "Power systems management and associated information exchange - Data and communications security - Part 4: Profiles including MMS," 2007.
[8] IEC 62351-6, "Power systems management and associated information exchange - Data and communications security - Part 6: Security for IEC 61850," 2007.
[9] C. Cremers, M. Dehnel-Wild, K. Milner, "Secure Authentication in the Grid: A formal analysis of DNP3: SAv5," European Symposium on Research in Computer Security, Aug 2017.
[10] Modbus Organization, Inc., "Modbus FAQ," URL: http://modbus.org/faq.php/, accessed 01-29-2019.
[11] SunSpec Alliance, "SunSpec Specifications & Information Models," URL: https://sunspec.org/download/, accessed: 6-29-18.

2.4. IEEE 2030.5

IEEE 2030.5 is an approved interoperability standard listed in the NIST/SGIP Catalog of Standards[12] for communications for IoT devices like energy sensors, smart light bulbs, solar inverters, and electric vehicles. The CPUC has been phasing in new interoperability requirements in the California interconnection standard, Electric Rule 21, *Generating Facility Interconnections.* As part of this process the California investor owned utilities (IOUs) established IEEE 2030.5 (Smart Energy Profile 2.0, SEP2) as the default DER communications protocol originating from the IOUs. The Common Smart Inverter Profile (CSIP) was developed to foster interoperability between IOUs and inverters or the services managing those inverters.[13] California's requirements and guidelines regarding IEEE 2030.5 are viewed as a generic solution that other states will likely follow. As stated in the CSIP Implementation Guide, IEEE 2030.5 applies to communications between the utility and DER systems through connections via an aggregator or direct connections. In direct systems without an aggregator entity, either the inverter or a separate control unit may be the IEEE 2030.5 client. These three cases are shown in Figure 1, wherein the DER is composed of a communication module and the power converter, i.e., an inverter or rectifier.

IEEE 2030.5 is an application protocol that uses HTTPS transport over TCP/IP. It can run on any physical and MAC layer that support the transport of TCP/IP. IEEE 2030.5 uses a RESTful client/server model with a defined XML schema. The typical usage scenario is for the server to expose available resources that it hosts using URL's, and for the client to GET those resources, or PUT/POST its information to those resources. From a security perspective, the server wants to securely expose the resources it hosts to clients that are authorized to receive the information, and the client wants to make sure that it is securely sending its information to the correct server. In Section 3, IEEE 2030.5 encryption, authentication, and key management guidelines for transport layer security are analyzed in more detail.

The Common Smart Inverter Profile (CSIP) defines the operation of IEEE 2030.5 in the California Rule 21 use case. In this environment, the client device is an Aggregator or a DER device. It is expected that utilities will enroll or registers DER devices for approved CSIP operation. Enrollment will probably involve the following steps:

1. Establishment of a contractual arrangement between the Utility and the DER Device owner.
2. Provisioning of the DER Device with the Utility Server's connection credentials (e.g. DNS name, IP address, ports, etc.) and security credentials (e.g. server certificate, server public key, etc.)
3. Provisioning of the DER Device information (e.g. DER client certificate, serial number, etc.) onto the Utility Server

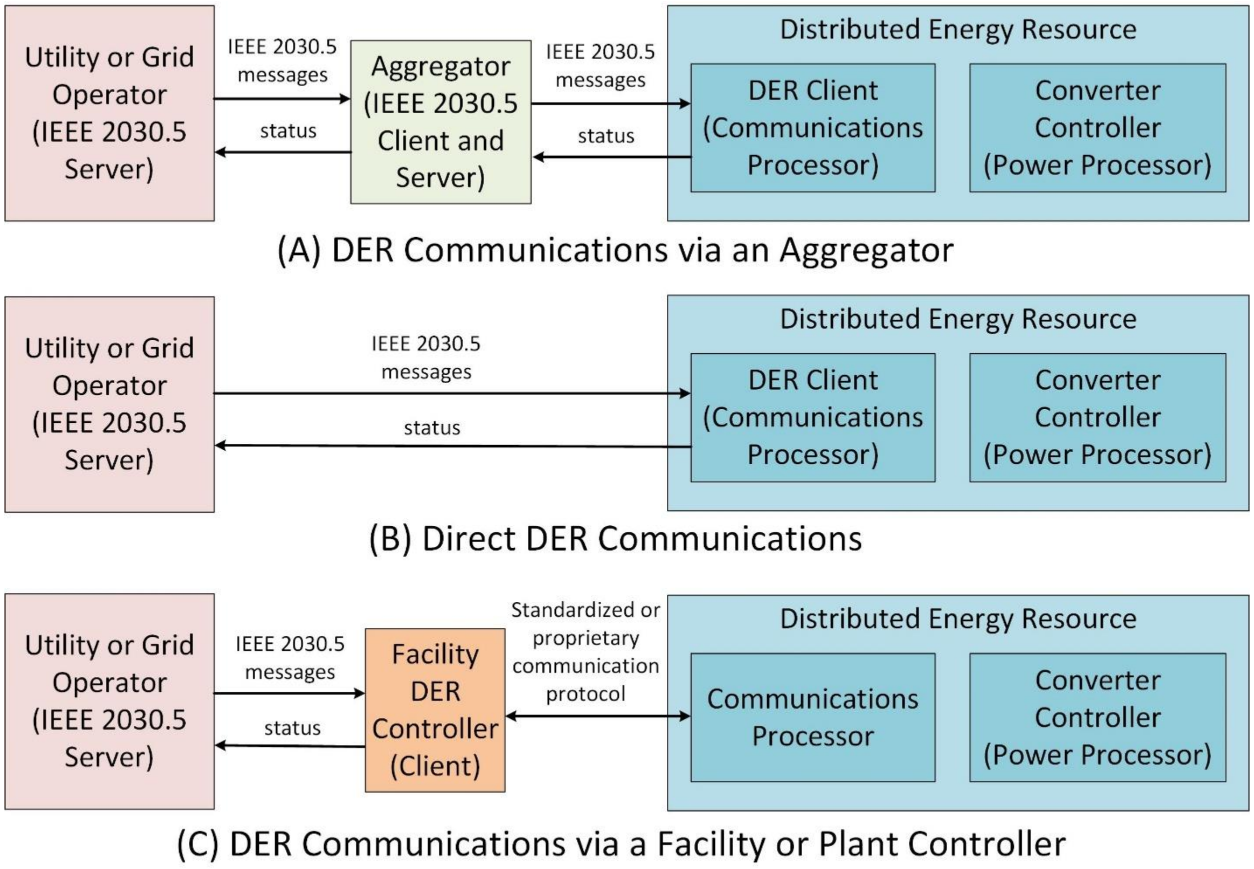
Once enrolled, the DER client device connects with the Utility Server (possibly via an aggregator) to perform CSIP/Rule 21/IEEE 1547 operations, e.g.:

1. Utility Server sending DER Controls to the DER device:
    a. Autonomous functions: volt-var, volt-watt, freq-watt, etc.

[12] SEPA Catalog of Standards, URL: https://sepapower.org/knowledge/catalog-of-standards/catalog-of-standards-complete-list-of-entries/, accessed 12-19-18.

[13] California Smart Inverter Implementation Working Group, "IEEE 2030.5 Implementation Guide for Smart Inverters," Common Smart Inverter Profile V2.0, March, 2018.

   b. Immediate controls: active power curtailment, fixed reactive power, fixed power factor
   c. Protection settings: High/Low Voltage Ride-Through, High/Low Frequency Ride-Through
2. DER Client sending information to the Utility
   a. DER nameplate ratings and settings
   b. DER alarms and status
   c. DER measurements (W, Var, V, A, PF, Wh, Hz, etc.)


Utility or Grid Operator (IEEE 2030.5 Server)
IEEE 2030.5 messages
status
Aggregator (IEEE 2030.5 Client and Server)
IEEE 2030.5 messages
status
Distributed Energy Resource
DER Client (Communications Processor)
Converter Controller (Power Processor)
(A) DER Communications via an Aggregator
Utility or Grid Operator (IEEE 2030.5 Server)
IEEE 2030.5 messages
status
Distributed Energy Resource
DER Client (Communications Processor)
Converter Controller (Power Processor)
(B) Direct DER Communications
Utility or Grid Operator (IEEE 2030.5 Server)
IEEE 2030.5 messages
status
Facility DER Controller (Client)
Standardized or proprietary communication protocol
Distributed Energy Resource
Communications Processor
Converter Controller (Power Processor)
(C) DER Communications via a Facility or Plant Controller
Figure 1. Utility-to-DER communications options per CA Rule 21 and CSIP.

3. CRYPTOGRAPHY AND TRUST

According to The Handbook of Applied Cryptography[14], “cryptography is the study of mathematical techniques related to aspects of information security”. In particular, cryptographic methods assure the confidentiality, integrity and authentication of information services. The mathematical process of encrypting data provides confidentiality, rendering data unintelligible to all, except with possession of the proper decryption key. The process of tagging data with its hash value or cryptographic hash value enables verification of data’s integrity. A mathematical signature created with a private key can verify the originating entity, authenticating the source of the information. Methods of encrypting data and providing trust between entities (i.e., performing authentication processes) are discussed in Appendix A.

3.1. Cryptographic Features of IEEE 2030.5

In IEEE 2030.5, the cipher suite TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8 is specified for aggregators and DER clients unless otherwise indicated by utility interconnection obligations. The string, ‘TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8,’ gives the concatenated parameters to be used in configuring the necessary transport layer security parameters for the communication transactions[15]. Per CSIP, Transport Layer Security Version 1.2 (TLS 1.2) must be used to create an encrypted tunnel between network points with certificates to authenticate the end devices establishing the tunnel. Transport layer security protects data-in-flight, i.e., in transit between the two end devices. The TLS 1.2 protocol allows for application profile standards to specify choices of various parameters for achieving authentication, encryption and integrity[16,17] of exchanged information, and the particular cipher suite selection dictates the key length, tag length, and the encryption/authentication algorithms to be used by the devices and certificates. The parameters of TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8, shown in Figure 2, are described in detail in Section 3.1.2.

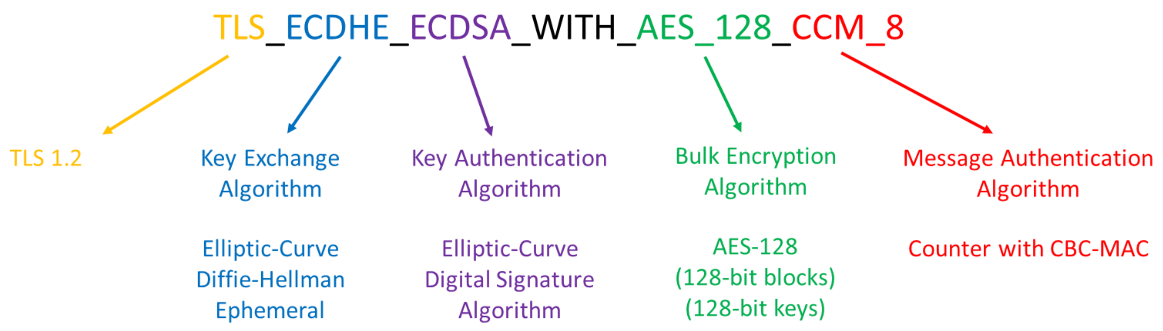

TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8
TLS 1.2
Key Exchange Algorithm
Elliptic-Curve Diffie-Hellman Ephemeral
Key Authentication Algorithm
Elliptic-Curve Digital Signature Algorithm
Bulk Encryption Algorithm
AES-128 (128-bit blocks) (128-bit keys)
Message Authentication Algorithm
Counter with CBC-MAC

Figure 2: IEEE 2030.5 cipher suite components.

In brief, the cipher suite components are:

[14] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, 1997.
[15] https://docs.microsoft.com/en-us/windows/desktop/secauthn/cipher-suites-in-schannel
[16] Mozilla, “Transport Layer Security,” URL: https://developer.mozilla.org/en-US/docs/Web/Security/Transport_Layer_Security, accessed 6-29-18.
[17] Internet Assigned Numbers Authority (IANA), TLS Cipher Suites, URL: https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml#tls-parameters-4

1. **TLS** –TLS version 1.2 is specified, but, notably, the IEEE 2030.5 cipher suite complies with the recently approved RFC for TLS version 1.3.
2. **ECDHE** – The key exchange algorithm is Elliptic Curve Cryptography (ECC) with the P-256 (also known as secp256r1 or prime256v1) curve using the Diffie-Hellman (DH) key agreement algorithm with Ephemeral (E) keys. The use of ephemeral keys provides for Perfect Forward Secrecy.
3. **ECDSA** – The signature algorithm is the Elliptic Curve Digital Signature Algorithm
4. **AES_128** – The bulk (symmetric) traffic encryption algorithm is the Advanced Encryption Standard (AES) using 128-bit keys. 128-bits complies with NSA Suite B requirements at the secret level. AES is run in Counter mode with Cipher Block Chaining (CBC).
5. **CCM_8** – Counter mode with CBC-MAC (CCM) is an Authenticated Encryption with Additional Data (AEAD) algorithm that combines encryption and authentication in one process, utilizing the final block of ciphertext as the Message Authentication Code (MAC)

3.1.1. CSIP implementation of IEEE 2030.5 PKI

In the following sections, trust establishment and maintenance according to IEEE 2030.5 CSIP V2.1 are discussed. CSIP is a guide to assist manufacturers, DER operators, system integrators and DER aggregators in implementing IEEE 2030.5 for DER communication networks in California. For DER devices, CSIP uses the IEEE 2030.5 PKI model as the default PKI model. Under special agreement with the utility, an aggregator may use an alternate PKI model, but it is envisioned that the vast majority of CSIP devices will be governed by the IEEE 2030.5 PKI model.

The IEEE 2030.5 PKI model contains a Root Certificate Authority (Root-CA) that acts as a trust anchor for the PKI. All devices in the PKI have a copy of the Root-CA public key provisioned into the device using a secure out-of-band process. The Root-CA public key is used to validate the certificate chains of communicating devices as part of the TLS handshake process.

The IEEE 2030.5 PKI model supports up to a certificate chain depth of 3 in the PKI hierarchy. All DER devices must be able to support any of the 3 options listed below. The maintainer of the PKI will choose which option to used based on its CA management practices. The available options are:

- Depth 1: Root-CA → Device
  - The Root CA directly signs device certificate.
- Depth 2: Root-CA → MICA → Device
  - The Root CA authorizes a **M**anufacturer **I**ssuing **CA** to sign device certificates.
- Depth 3: Root-CA → MCA → MICA → Device
  - The Root CA authorizes a **M**anufacturer **CA** to authorize a Manufacturer Issuing CA to sign device certificates.

CSIP requires all communicating devices, both clients and servers, to have an IEEE device certificate that chains back to the Root-CA using one of the three options listed above. Only devices certified to be compliant with CSIP are authorized to request for and obtain a valid device certificate from the Root-CA or one of its subordinate CAs. Because a DER entity's certificate is used to vouch for the entity's identity, the utility is trusting the authenticity of the signature or chain of signatures on the certificate presented by the DER.

3.1.1.1. Device Identification

Node authentication is achieved by using X.509 v3 embedded *digital certificates*. The purpose of these certificates is to provide a means to verify the device identity. According to the CSIP, all communicating devices must have an IEEE 2030.5 compliant, X.509 v3 device certificate that chains back to the Root Certificate Authority (Root-CA). Each X.509 v3 DER-encoded certificate will have an associated SHA256 fingerprint. Truncated versions of the certificate fingerprint are used for device identification. There are two types of the truncated fingerprints:

- *Long Form Device Identifier (LFDI)* which is a fingerprint that is left-truncated to 160 bits and globally unique with an entropy of 2^{160}.
- *Short Form Device Identifier (SFDI)* which is a fingerprint left-truncated to 36 bits and expressed at 11 decimal base-10 digits. The entropy of the SFDI is 2^{36} and is not great enough for globally unique identification purposes, but is sufficient for identification within a LAN or local site domains.

3.1.1.2. Authentication

Client-server authentication is achieved using a TLS mutual authentication handshake. The TLS handshake entails first exchanging X.509 certificates, verifying the integrity of the received certificate, checking that the certificate chains back to the Root-CA, and, finally, verifying the certificate contents conforms to the requirements of IEEE 2030.5. If authentication fails, each party issues a TLS Alert and the communications connection is closed. In most TLS handshakes, the certificate would be inspected to verify that it is not expired, but this is not required in CSIP.

3.1.1.3. Authorization

Each server maintains an authorization list of client LFDIs. If a client is contained in the authorization list, it is permitted to communicate with the server. During the TLS handshake, the contents of the client's certificate is read and the LFDI is calculated. If the calculated LFDI is not found in the authorization list, the server returns an HTTP 404 and the transaction is terminated.

3.1.1.4. Access Control

A server maintains Access Control Lists (ACL) which controls which clients have access to server resources. A client device that is in the ACL is authorized to access specific resources on the server. Access policies must be drafted which dictate what types of access (read, write, control) clients have for specific resources. IEEE 2030.5 does not require ACLs.

3.1.2. Authenticated Encryption

In addition to verifying client and server identity, messages in IEEE 2030.5 are also authenticated. However, instead of gluing together one algorithm for encryption and another for authentication, the two operations are combined, conserving operational overhead as well as preventing certain types of attacks[18]. The following section briefly reviews the protocols chosen for IEEE 2030.5 authenticated encryption including the key exchange mechanism.

[18]M. Bellare, P. Rogaway, D. Wagner, "A Conventional Authenticated-Encryption Mode", 2013. URL: https://csrc.nist.gov/csrc/media/projects/block-cipher-techniques/documents/bcm/proposed-modes/eax/eax-spec.pdf

3.1.2.1. Key Exchange

Prior to the start of traffic encryption, the end devices must agree on the traffic encryption key to be used. Certificates from a CA provide the basis of trusted information for the establishment of keys. The key exchange algorithm for IEEE 2030.5 is Ephemeral Elliptic Curve Diffie-Hellman key exchange with Elliptic Curve Digital Signature Algorithm signatures (ECDHE_ECDSA)[19]. Ephemeral key exchange provides forward secrecy in that the compromise of a single session key will not compromise the data of other sessions. In the IEEE 2030.5 context, this means that even if the DER device's private key is compromised, the attacker cannot decrypt messages from past session.

In the TLS handshake previously described, the server and client have exchanged and verified certificates including the necessary elliptic curve key exchange parameters. Next, the server and client each complete the ECDH computation generating a shared secret that is then used to derive the traffic-encryption key (see appendix A for details).

3.1.2.2. Authenticated Encryption

Authenticated Encryption (AE) is a form of encryption which simultaneously provides confidentiality, integrity, and authenticity assurances on the data. These attributes are provided under a single, easy-to-use and implement, algorithm. The traffic encryption algorithm is specified as Advanced Encryption Standard (AES) in the Counter with Cipher Block Chaining–Message Authentication Code (Counter with CBC-MAC) Mode, with a 128-bit key length and an eight-octet authentication tag length (AES_128_CCM_8). AES is the industry- and government-standard symmetric key (both parties must know the same key) encryption algorithm. Though the AES algorithm can utilize 198- and 256-bit keys, brute force attacks on 128-bit keys are not yet considered feasible on conventional computers, therefore 128-bit keys are used in constrained devices. The particular cipher modes, i.e. counter mode, cipher block chaining (CBC) mode and message authentication code (MAC) mode each provide a desired attribute.

In AES counter mode the cryptographic routines for encryption and decryption are identical whether implemented in hardware or software, which serves to reduce the footprint of the implementation in constrained devices. Cipher block chaining (CBC), where each resulting block is combined in digital logic with the previous result, provides resistance against cryptanalysis. That is, cipher block chaining prevents patterns in the input data from still being evident in the AES output data, e.g., "Electronic Codebook (ECB) Penguin"[20]. Message Authentication Code (MAC), in this case CBC-MAC, provides a data-dependent tag which can be verified upon receipt to ensure the integrity of the transmitted data. In the specified TLS cipher suite, the tag is truncated from 16 octets to eight for constrained bandwidths and message sizes. Properly combined, AES Counter with CBC-MAC (AES-CCM) provides authenticated encryption with a single cryptographic algorithm.

Note that CSIP calls out TLS version 1.2 that was developed in 2013. In 2018, TLS version 1.3 was approved as an RFC[21]. TLS 1.3 has made many changes to improve security, including:

[19] Blake-Wilson, et al., RFC 4492, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)," 2006.
[20] F. Valsorda, 10 Nov 2013. URL: https://blog.filippo.io/the-ecb-penguin/
[21] RFC 8446, The Transport Layer Security (TLS) Protocol Version 1.3, Aug 2018.

- Removing all insecure algorithms of TLS 1.2
- Approving algorithms are all authenticated encryption with associated data (AEAD) algorithms
- Approving algorithms all provide Perfect Forward Secrecy
- Providing baseline support for both ECC and RSA

The cipher suite in IEEE 2030.5 is compliant to TLS version 1.3.

3.2. Implementation with Embedded Systems

Communication-enabled DER controls allow advanced grid functionality which enable distributed generators to support and protect the power system. The enabling technologies of secure communications, such as microprocessors, communication protocols, cryptographic primitives, are subject to constraints such as power, size, throughput rates, and cost.

An embedded system may well have multiple specialized processors for the numerous tasks involved, i.e., status and signaling with the power electronics, upstream and downstream communications over main and backup physical layers, and hardware or software cryptographic support. These depend on communication rates, upstream and downstream polling or interrupt rates, and analog and digital input/output response times, as well as internal factors such as processor speed with respect to state machine or embedded operating system functions.

The capability of the processor can limit or enable the type of encryption to be performed, and should be factored into the design rather than treated as an add-on. The cryptographic processing demand can be met by either software or hardware primitives, with tradeoffs such as adaptability, ease of use, speed, code size and memory use. With either hardware or software, modules are readily available for implementing the standard algorithms.

IEEE 2030.5 selected a cipher suite that, although it has been accepted for use by other parties as a useful constrained applications solution, does not yet exist as a fully integrated reference design in most existing off-the-shelf development systems or software libraries. Research efforts are underway exploring how to ease entry into securing DER under the 2030.5 specifications.

In the 2010s, DER communications were commonly established to monitor power production and DER operations without stringent or standardized cybersecurity practices. Therefore, many available DER products lack the capacity for security upgrades via simple firmware changes, due to limitations, for example, of memory and program storage.[22] In such cases, protocol translators can convert communications with higher security features to less secure protocols that only exist in proximity with or internal to the DER equipment. Ideally, these gateways will be located as close to the DER as possible, and some have proposed "bolt-on" translators that convert IEEE 2030.5 with the associated encryption and authentication to Modbus[23] (see Figure 3).

[22] NOTE: Firmware upgrades are currently managed out of band in IEEE 2030.5 networks.

[23] B. Seal, et al., "Final Report for CSI RD&D Solicitation #4 Standard Communication Interface and Certification Test Program for Smart Inverters," June 2016.

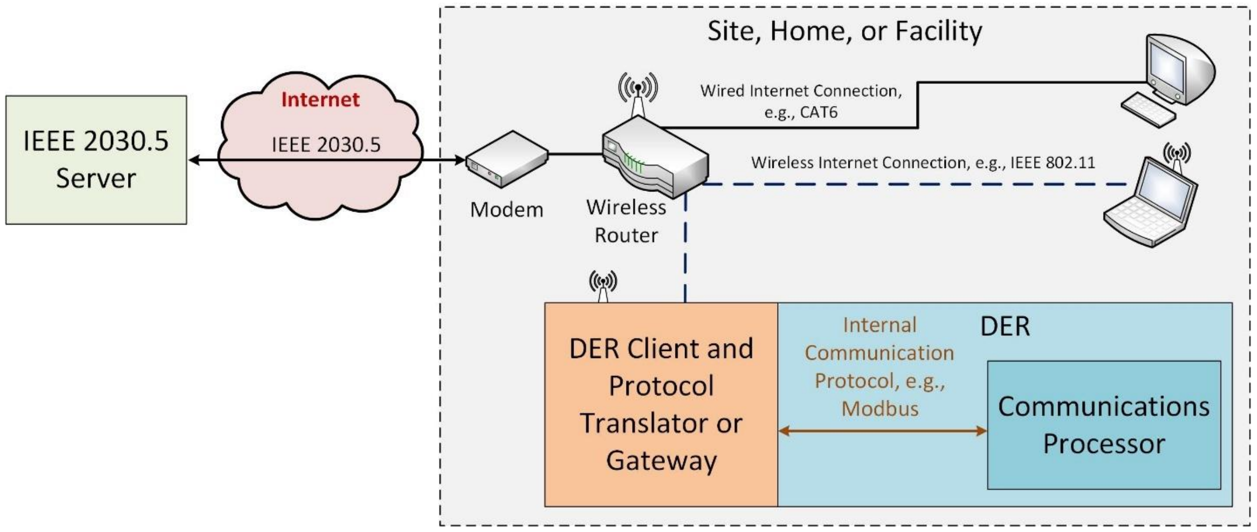

Site, Home, or Facility
IEEE 2030.5 Server
Internet
IEEE 2030.5
Modem
Wireless Router
Wired Internet Connection, e.g., CAT6
Wireless Internet Connection, e.g., IEEE 802.11
DER Client and Protocol Translator or Gateway
Internal Communication Protocol, e.g., Modbus
DER
Communications Processor

Figure 3. Protocol translator approach to DER communications.

4. GAP ANALYSIS

The security features in IEEE 2030.5 are commonly used in computing platforms, however, there are still questions of how well the technologies will scale in highly-distributed, computationally-limited DER environments. In this section, potential security gaps are identified. These include:

- No Certificate Policy defining security procedures, policies and practices for the ecosystem
- Non-expiring certificates and no certificate revocation methods
- No method to update the cryptographic algorithms for the lifetime of the DER devices
- Possibly weak or poorly implemented TLS interception techniques
- No physical security requirements
- Unclear requirements for aggregator IEEE 2030.5 servers

4.1. IEEE 2030.5 Certificate Validity and Revocation

The IEEE 2030.5/CSIP PKI is different than other PKI's in the use of non-expiring certificates and the explicit prohibition of CRL and OCSP. Once issued, a device certificate has an indefinite lifetime, so it is always valid. It is expected that device manufacturers use best practices to secure and protect the device's private key. If the device's private key is compromised, the device certificate cannot be revoked. There are PKI systems that provide for certificate revocation using Certificate Revocation Lists (CRLs) or Online Certificate Status Protocol (OCSP), but neither is used by IEEE 2030.5. In fact, IEEE 2030.5 prohibits their use. This means that Certificate Authorities (CA) shall not maintain CRLs or run OSCP servers, and clients and servers shall not check for CRLs or OCSP servers to verify certificate validity. This does not preclude servers or clients from maintaining or obtaining their own list of blacklisted or whitelisted devices if the operator prefers. As with most security discussions, IEEE 2030.5's decision to mandate against using CRLs or OCSP includes tradeoffs.

4.1.1. Reasons for Using CRLs or OCSP

Certificates may be compromised before they expire, so it is good to have a way to revoke the certificate to mark that device as not-trusted. OCSP stapling also fixes most of the issues with OCSP server access/availability by having the server cache its OCSP status response and provide it to any clients requesting the server's certificate status. If more than one grid operator and/or aggregators are communicating with the equipment for various grid operations, detection of a poorly behaving DER should be reported to all the grid operators to indicate the equipment is compromised and should not be trusted any longer.

The lack of a robust Certificate Policy (CP) and Revocation process can result in a number of management issues for an ecosystem. The CP not only defines the security requirements around the structure and storage of signing keys, it also defines the policies around what to do if a key/certificate are compromised. Revocation is not just simply about issuing a centralized master list of revoked certificates, the CP defines the conditions under which a revocation can occur and the review and approval processes around that event.

Consequently, the use of disconnected black/white lists operated by independent operators can lead to arbitrary processes resulting in fragmentation and uneven enforcement of the ecosystem. A blacklist is essentially a local CRL but without the process, rigor and verification behind its issuance. To understand the potential impact we need to realize that a certificate is not the device, but an identifier for the device similar to the relationship between a person and their passport. In both cases, the device certificate or a person's passport are used to gain access to something. So, just like a person's passport, an improperly stored certificate and private key can be stolen and used by unauthorized parties to gain access. With fragmented blacklists and policies, it is possible for a stolen device certificate that was disallowed in one region to be used in a different region to gain access because it was only locally blacklisted.

Not having a defined and well-managed means of revocation has some potentially larger consequences for Manufacturing Issuing CAs (MICAs)—a CA that issues device certificates during the manufacturing process—and Manufacturer's CA (MCAs)—an Intermediate CA operated by a specific manufacturer designed to issue MICAs. This is because MCAs and MICAs are able to generate and sign certificates for the ecosystem. It is like operating your own passport printing press. Since there is no revocation, if someone operating an MCA/MICA is not abiding by the requirements of the ecosystem, there is really no effective way to stop them from continuing to issue certificates. If a company operating an MCA goes out of business, there is no way to recall or otherwise disable that MCA.

Finally, similar to passports, certificate identities should have a limited life. There are cases where a full or large scale revocation are not practical or necessary as in the case with obsolete hardware. If there is an updated specification that makes improvements, having certificates with limited validity allows legacy systems to age out. In the case of hardware that may have a very long operational life, allowing the option to either let them expire, or to renew them provides flexibility in management of elements within the ecosystem.

Having a CP that enables revocation and defines its respective procedures eliminates fragmentation, applies revocation procedures uniformly, and provides a centralized list that is signed by the CA on behalf of the ecosystem owners to cryptographically prove authenticity of the list. The purpose of revocation is not to arbitrarily exclude devices, but to provide a mechanism to uniformly enforce governance of the ecosystem's integrity and security. Revocation is not exercised at the whim of the CA provider. Through the CP and its contract with the ecosystem, the CA only revokes certificates at the request of the owners of the ecosystem and completion of due process.

4.1.2. Reasons for Not Using CRLs or OCSP

In the CSIP use case, servers only communicate with approved aggregators or approved DER client devices. The utility will enroll a device (i.e. add to the utility server whitelist) after entering a contractual relationship with the end user. After enrollment, the utility provides the connection info (e.g. IP Address, DNS Name, port, SFDI (certificate hash) to provision onto the end user's DER. This provisioning effectively adds the utility server's certificate to the DER clients whitelist. Since both the server and client maintain whitelists of allowed certificates, there may not be a need for a CRL or OCSP blacklist.

CRLs and OCSP are typically hosted on the issuing CA's servers on the internet. There is no guarantee of internet access for IEEE 2030.5 devices, and therefore, there is no reliable networking infrastructure for CRL or OCSP. For instance, if there is no access to the CRL or OCSP, the device has two options:

- Hard-Fail – Reject the certificate if CRL or OCSP cannot be verified. This will have a major impact of availability and can cause service disruptions.
- Soft-Fail – Accept the certificate if CRL or OCSP cannot be verified. This significantly weakens the benefit of revocation as hackers can easily bypass revocation by blocking access to the CRL or OCSP servers.

Since IEEE 2030.5 devices are expected to be used world-wide, slow access or no access to the CRL or OCSP servers can significantly degrade service. CRL and OCSP servers also provide a single point of failure and a large target for hackers. Certificate expiration, CRL, and OCSP all assume the DER has the correct time to do these checks. This implies the DER has access to a secure time server. If not, then hackers can spoof a time server to bypass all the CRL and OCSP protections. We now need a secure and authenticated time server to make CRL and OCSP work correctly.

Furthermore, there is no clear indication of who is responsible for reporting compromised certificates. Many devices are "head-less" and are expected to operate with no user intervention. The owner may not even know if his device's certificate has been compromised. Likewise, it is unclear how the CA would determine if a certificate has been compromised. If a server owner detects bad behavior from a device, it will probably black-list that device. At that point the server owner could inform the CA that the device has been compromised, but this would need to be defined.

A procedure would need to be created to systematically report and address a revoked certificate. This procedure would include the following:

- Methods for reporting a suspicious certificate.
- Methods to validate the report.
- Appeals process for revocation.
- Legal and financial responsibility for certificate compromise.
- Device behavior when the certificate is revoked. Some options are to disable or disconnect the communications interface of the device, disable grid-support functions, or, possibly "brick" the equipment to prevent malicious actions.

4.2. Brute Force Attacks

New attacks will emerge against previously trusted algorithms. Brute force attacks become possible by discovering new weakness in a particular implementation or as computational efficiency increases. The solar industry must recognize that cryptography may not be static over the life of a DER installation. This is evidenced by the fact that no white-box AES key obfuscation has survived the test of time.[24] The IEEE 2030.5 ECC P-256 and SHA-256 algorithms for keys and signatures may be vulnerable to factorization attacks since the devices will be in service for 25+ years and no other algorithms can be used for the DER, servers, intermediate CAs and root CA.

[24] L. Goubin, P Paillier, M. Rivain, J. Wany, "How to Reveal the Secrets of an Obscure White-Box Implementation," URL: https://eprint.iacr.org/2018/098.pdf, accessed 6/25/2018.

Additionally, while it is uncertain when quantum computing will mature to the point of breaking long-standing encryption technologies, it will significantly disrupt business-as-normal security approaches when it does. Quantum computing will easily solve the factorization and discrete log one-way math problems that currently protect asymmetric encryption[25], i.e., current methods of asymmetric cryptography will be broken. Quantum computing will not so easily break symmetric encryption nor the one-way math of hashes. Post-quantum cryptography will require larger hash and key sizes. Eventually new way to exchange symmetric keys will also be required, e.g., quantum key distribution. Researchers are currently seeking standardize post-quantum cryptography algorithms, in which both conventional and quantum computers can perform quantum resistant cryptography[26].

4.3. Enabling Encrypted Traffic Inspection

Utilities and aggregators may wish to inspect incoming and outgoing network traffic, but often portions of the traffic is encrypted, e.g., IEEE 2030.5 traffic. This prevents the utility or aggregator from scanning the data for threats. For example, a utility user may inadvertently download a malicious file while on an https-secured website, or malware itself may employ encrypted tunnels to communicate with their command and control centers[27], and these encrypted malicious payloads will not be discovered. In order to protect against nefarious content, utilities and other entities communicating with aggregated or individual DERs using TLS may require inspection of the DER traffic prior to reaching its internal endpoints called SSL/TLS interception. There exist hardware security devices and software security applications that provide a mechanism for intercepting and decrypting encrypted network traffic for both legitimate and illegitimate purposes[28].

Security gateways and anti-virus software perform the legitimate service of intercepting and inspecting traffic by splitting a secure tunnel into two parts, as shown in Figure 4. The end-users and devices configured by an enclave will hold a special "intercept" certificate from the enclave CA, while the enclave server holds all the normally available CA certificates. When users and devices within the enclave connect to a server outside of the enclave, there are then two halves to the authentication-encryption processes. The first is between the user and an enclave server, authenticated and encrypted with the enclave's special certificate. The second piece is between the enclave server and the external server, authenticated and encrypted with the normally available certificates. While passing through the enclave server, traffic in both directions can be decrypted by the enclave server to be inspected for malicious content.

[25] T. Güneysu, V. Lyubashevsky, T. Pöppelmann, Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems. In: Prouff E., Schaumont P. (eds) Cryptographic Hardware and Embedded Systems – CHES 2012. CHES 2012. Lecture Notes in Computer Science, vol. 7428. Springer, Berlin, Heidelberg.
[26] NIST, Post-Quantum Cryptography, URL: https://csrc.nist.gov/Projects/Post-Quantum-Cryptography
[27] R. Arandjelovic, "How to Deal with the Blind Spots in Your Security Created by Encrypted Traffic," Info Security, 7 Mar 2016, URL: https://www.infosecurity-magazine.com/blogs/deal-blind-spots-security-created/
[28] "TLS Interception and SSL Inspection," 20 Mar 2017, URL: https://tlseminar.github.io/tls-interception/

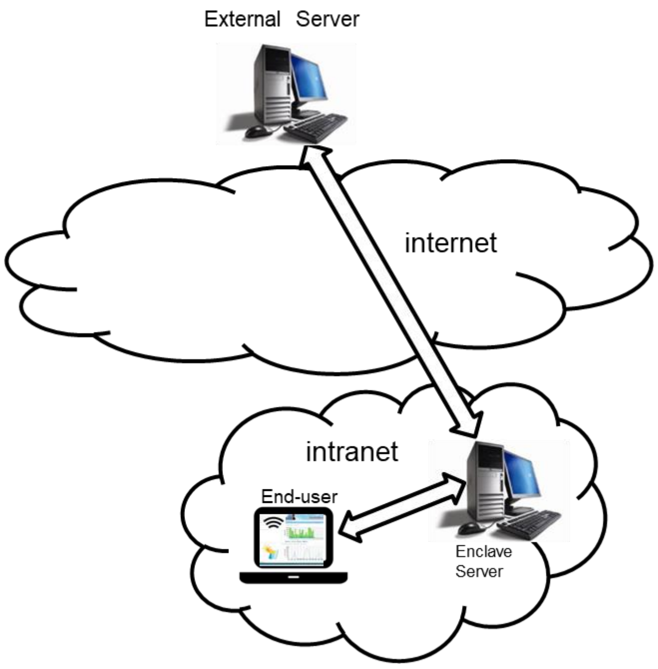


Figure 4. Two-part Tunnel for TLS/SSL Interception.

Though there are sound reasons for inspecting traffic, research has shown that even legitimate implementations of TLS interception can be harmful if improperly implemented or managed[29]. Issues may inadvertently introduce security vulnerabilities for example through broken certificate chains or weak ciphers. Therefore, tradeoffs should be weighed when considering security architectures requiring TLS/SSL interception. Utilities and aggregators need to be particularly careful when terminating encryption at the DMZ (reverse proxy, Web Application Firewall (WAF), etc.) or creating a means of sharing the server key with the DMZ. At this time, there is some ambiguity in the interfaces between the DER network, utility DMZ, and utility server, especially with regards to where the IEEE 2030.5 keys are located.

4.4. DER Physical Security

Physical security has many implications for data-in-flight security. For instance, if private keys can be extracted, data-in-flight security will be compromised. As an example of the risk, in a recent challenge, none of the 90 submitted DRM and mobile payment systems with AES software and obfuscated hard-wired keys implementations resisted key extraction.[30] Therefore, there is a clear need to improve these systems to prevent the loss of communication trust. Fortunately, recent penetration testing research from EPRI using a Cable Labs' reference architecture has shown possible improvements to DER key extraction resistance.[31]

[29] Censys, The FREAK Attack, 3 Mar 2015, URL: https://censys.io/blog/freak

[30] L. Goubin, P. Paillier, M. Rivain, J. Wang, "How to Reveal the Secrets of an Obscure White-Box Implementation," IACR Cryptology ePrint Archive, 98, 2018.

[31] "Test Report as part of the Distributed Energy Resources (DER) Embedded Security Assessment," Southwest Research Institute Report, 26 March 2018.

Additionally, in many cases the DER natively communicates plaintext Modbus which exposes DER communications and potential security weaknesses. Often this concern is addressed through the use of physical security because these communications only are exchanged within the DER equipment housing or over short distances between a protocol translator and the DER. Most—if not all—Modbus communications are going to occur within a few feet of the DER equipment, and, arguably, presents no additional security risk because for an adversary to manipulate the Modbus operations, they would need physical access to the device. If an adversary had this level of physical access, they could just as easily manipulate the DER AC or DC disconnect to impact grid and communication operations. This said, there are no requirements on the permitted physical or logical separation between a gateway/protocol translator and the DER, so the potential for less secure implementations is possible.

5. TRUST ALTERNATIVES

In this section, alternative trust technologies are explored for possible inclusion in DER interoperability or cybersecurity standards, including hardware-based root-of-trust techniques. Some security models include lifecycle considerations as opposed to abandoning devices in the field. Inverters and other DER are small, compute and data storage-limited devices, which are widely distributed on the edge of the electrical grid. Tablets and smart phones are also compute and data storage limited, and the Trusted Computing (TC) solutions developed for these devices can potentially be leveraged for trust management in smart inverters. The prevailing TC architectures available for smart phones and tablets which can be potentially leveraged include:

- Secure Elements, Trusted Platform Modules (TPMs), and Mobile Trusted Module (MTM) Standard[32]
- Mobile Device Management (MDM)[33]
- Per-App Virtual Private Network (VPN)[34]
- ARM TrustZone[35]
- Post-quantum computing cryptography
- Blockchain PKI

To understand the importance of key storage, it is necessary to understand the importance of the private keys. Each certificate is unique and has a public key that is associated with a unique private key. The public/private key association proves (through cryptographic operations) that the device presenting the certificate is the proper owner of that certificate. Using the passport analogy again, the holograms prove a passport is genuine and was issued by the US State Department. The picture proves that the passport is yours. Similarly, the CA signature in the certificate proves the certificate is genuine and was issued by an authorized CA, while the private key proves the certificate is owned by the entity presenting it. As a result, having someone steal a private key is like having a way to seamlessly swap out the picture in a passport. It allows someone to steal an identity to gain access.

With that in mind, there is often the perception that ARM TrustZone is secure storage when it is not. TrustZone is indeed an important part of securing devices, but it is not secure storage in the sense of protecting private keys. The purpose of TrustZone is to prevent unauthorized tampering of executable code over the network. It provides a Trusted Execution Environment (TEE) where firmware cannot be arbitrarily changed by anyone remotely and prevents large-scale code tampering over the network. It sections off part of the memory so that only a part of the logic that is controlled is provided access. However, this memory is not hardened against various electronic, side channel or physical attacks so private keys stored they can be extracted relatively easily by someone who has physical access to the device. As mentioned earlier, once the private key is stolen, the identity and access is stolen with it.

Secure elements and TPMs on the other hand are designed specifically with key storage and cryptography in mind. These devices typically have limited programmability and are meant to

[32] "Mobile Trusted Module 2.0 Use Cases", Trusted Computing Group, 2011.
[33] G. Blokdyk, "Mobile Device Management MDM Standard Requirements", April 9, 2018.
[34] "Per-App VPN Setup Guide", Microsoft Press, 2016.
[35] "Building a Secure System using TrustZone Technology", ARM, 2013.

operate alongside a main processor or controller. These devices are specialized for cryptographic operations and providing hardened storage for private keys and can be thought of as crypto co-processors. Even in the event of a physical attack where the chip package is decapped/unencapsulated and the silicon die is micro-probed, there are special metal layers that prevent easy access to the memory cells and can even destroy the chip if tampering is detected.

As such, it should be noted that TrustZone and Secure elements are complementary to each other, not competitive. A TrustZone-enabled microcontroller can be used in concert with a secure element to provide strong security over its executable code as well as its cryptographic identity.

In the figure below, a security hardened smart inverter containing post-quantum, per-application VPN, post-quantum crypto algorithm, secure key management and MDM components is illustrated.

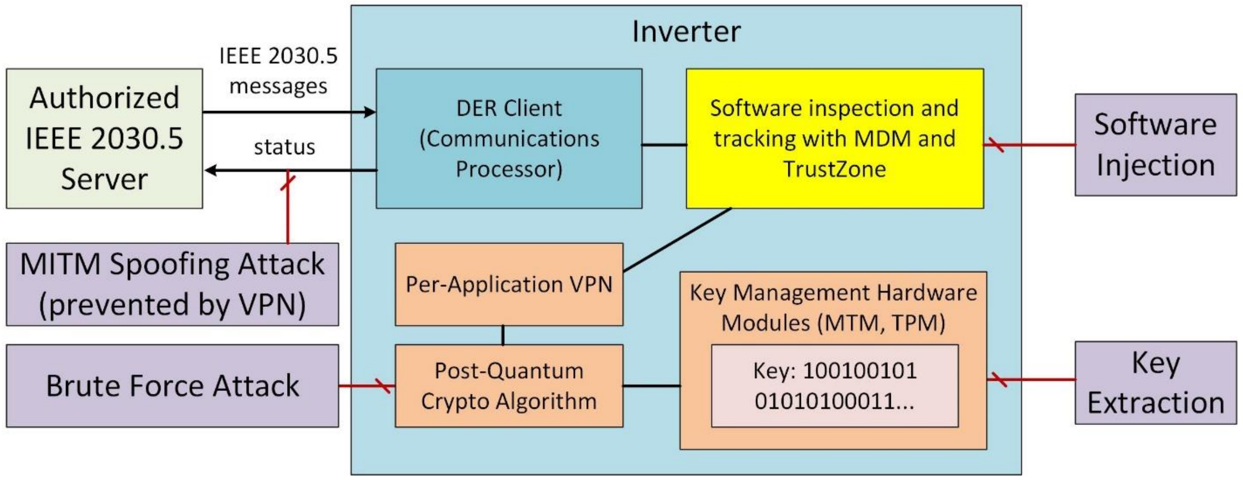

Inverter
IEEE 2030.5 messages
status
Authorized IEEE 2030.5 Server
DER Client (Communications Processor)
Software inspection and tracking with MDM and TrustZone
Software Injection
MITM Spoofing Attack (prevented by VPN)
Per-Application VPN
Key Management Hardware Modules (MTM, TPM)
Key: 100100101 01010100011...
Brute Force Attack
Post-Quantum Crypto Algorithm
Key Extraction

Figure 5. Techniques for defending against various attacks on DER trust.

Authorized IEEE 2030.5 encrypted messages are bi-directionally transferred to and from the inverter communications processor for parsing and other types of processing. The security equipped smart inverter is able to ensure adequate data integrity, client authentication and data privacy is enforced such that potential attacks such as brute force defeat of encryption, side-channel, key-extraction, software injection and man-in-the middle attacks are thwarted. The following sections describe the trust enhancing components (MDM, Per-Application VPN, MTM and TrustZone) in more detail.

5.1. Mobile Trusted Module

While data-in-flight is critical to ensure trusted communications between grid operators and DER, it is also essential to securely store keys inside the devices. If an adversary is capable of extracting the keys, it is possible the adversary will use a device's private keys to masquerade as a legitimate network node. A masquerading node can potentially launch insider attacks on the network causing grave damage.

The Trusted Computing Group (TCG) Mobile Trusted Module (MTM) use-cases and standards have been written to guide subsequent technical requirements and specification work within the

TCG Mobile Phone Working Group (MPWG).[36] The MTM is a security element and a newly approved TCG specification for use in mobile and embedded devices. Many embedded devices and mobile phones are subject to regulatory approval, which requires the enforcement of integrity protection and secure boot. A secure boot sequence measures the boot process and aborts any non-approved state transition. The MTM specification supports implementation of the MTM as a functionality implementation in hardware. This makes it possible for device manufacturers to add the MTM as an add-on to already deployed, proprietary security solutions. Reference architectures implemented by MPWG vendors consider the support of several parallel MTM instances in the same device. Some are discretionary and exposed to user applications while others are device manufacturer mandatory access controls.[37]

An MTM is generally a dedicated hardware-based trust-anchor and is composed of Root of Trust Storage (RTS) and Root of Trust Reporting (RTR) elements comparable to a Trusted Platform Module (TPM). An MTM supports a subset of TPM commands in support of local verification and mobile device functionality. The MTM architecture for a smart inverter is described in the Figure 6.

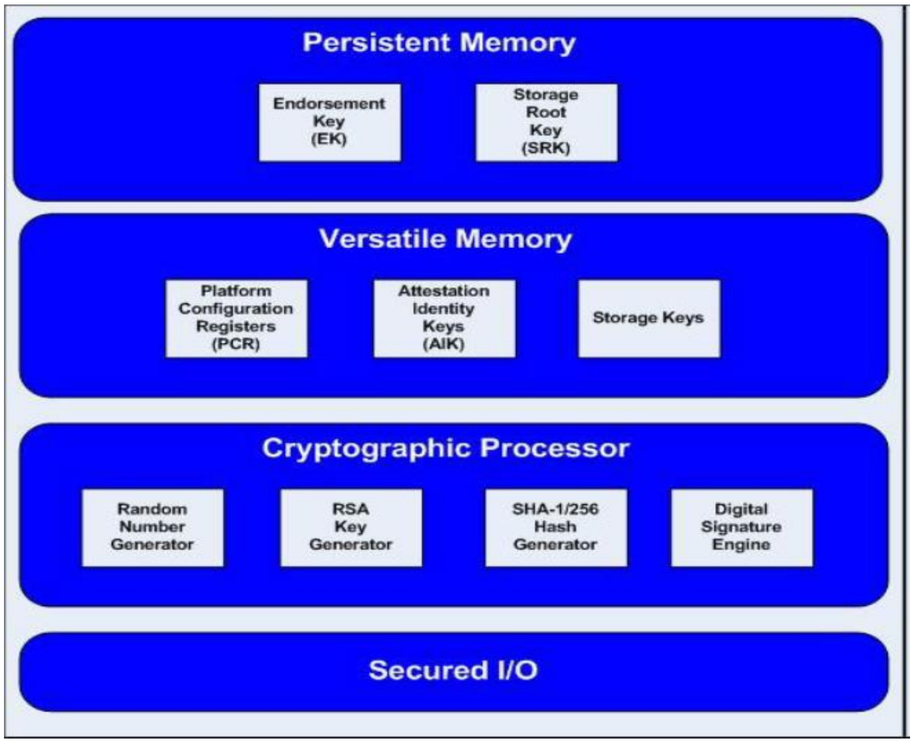


Figure 6: MTM Architecture

- The ***cryptographic processor*** implements bus encryption protecting data and instructions traveling on the system bus. Parties that have access to the system bus are not able to view clear text instructions and data due to the employment of encryption and hashing. Within the crypto processor there is an encryption-decryption signature engine that is used to create signed content for data to be placed on the bus. There is also an RSA key generator

[36] "TCG Mobile Reference Architecture Specification v. 1.0, 2009.
[37] "TCG Mobile Trusted Module, Specification v. 1.0", revision 1, 12 June 2007.

and a random number generator used for public key encryption as well as a SHA-1 hash generator for data integrity checking of the data transmitted over the bus. Future versions of the MTM will include SHA-256, and elliptic curve cryptography.

- The ***Persistent memory*** contains an endorsement key (EK) and a storage root key (SRK). The EK is a permanent encryption key pair configured at the time of manufacture. The EK private key is used by MTM software to encrypt and sign data for transmission to parties that use the corresponding public key to verify the identity of the MTM. The SRK is used to protect MTM keys created by applications, and is generated when ownership of a MTM is assumed. Upon change of MTM ownership, it is common practice to clear the MTM and generate a new SRK.
- *The* ***Versatile memory*** contains platform configuration registers (PCR), attestation identity keys (AIK) and storage keys. The PCRs contain security relevant metrics whose elements are used in forming attestation certificates. The PCRs contain specific information regarding the state of a system and applications In order to verify that systems and applications have not been tampered with, attestation certificates containing PCR information are issued by a MTM to authorized remote systems and users. Remote attestation is usually combined with public-key encryption such that the information sent can only be read by the parties presenting and requested the attestation.

5.2. Mobile Device Management Software

Mobile Device Management (MDM) is a key business area in the mobile security arena. MDM is defined as software, or more accurately a suite of software, that is used by an enterprise to manage its mobile devices. MDM software performs functions such as: hardware inventory creation and tracking, security policy enforcement, software distribution, and more. An example of how this would be useful would be a security policy that requires a device that is part of an MDM solution to have a minimum software version, if the device is detected with a lesser version it could be disallowed from accessing grid resources or a new firmware version could be automatically pushed to the inverter. An advantage to many MDM implementations is that the solutions work across many or even all mobile platforms (Android, Linux, etc.). It should be noted that many MDM implementations have a VPN component, which can be used to secure all IP traffic leaving a device.

5.3. Per-Application VPN

In per-application VPN, a device creates a VPN connection when a specific communications function is required, e.g., providing a status update to a control station. This poses a problem and a security risk in scenarios when a device is possibly mixing various classes of usage where data from running network communicating processes must be kept separate. The per-app VPN concept addresses this issue by segregating the applications on a device, and each application is run in its own per-app VPN tunnel.

Although per-app VPN technology is available in several OSs, it is still up to vendors to adopt the technology and create implementations for various applications. This implementation has begun, with MDM vendors being the first to offer products in this realm. Potentially these same vendors working in the mobile device realm, could also develop per-app VPN technologies for the various smart inverters. An end-goal for inverter manufacturers would be to develop downloadable smart inverter applications for their platforms, and allow that application to be run in a per-app VPN.

5.4. ARM TrustZone

ARM TrustZone is implemented in Cortex-A, 1176 , Apple A7, and ARM-M series processors to partition all of the system-on-chip (SoC) hardware and software resources so that they exist in either the secure world or normal-world. Hardware logic present in the TrustZone-enabled AMBA3 AXI™ bus fabric ensures that no secure-world resources can be accessed by the normal world components. A design that places the sensitive resources in the secure world, and implements robust software running on the secure processor cores, protects against possible attacks. The TrustZone hardware architecture contains extensions that have been implemented in the ARM processor cores enabling a single physical processor core to safely and efficiently execute code from both the normal and secure-worlds in a time-sliced fashion as shown in Figure 7.

The TrustZone architecture shown below allows critical trusted applications to be run within a Trusted Execution Environment (TEE) and afforded access through the TEE API. Commercially available TEE compliant applications are available through GlobalPlatform providers, and custom-built TEE applications can be built using ARM provided references and toolkits. Included in the TEE is a monitoring component that continually monitors and enforces TC behaviors.

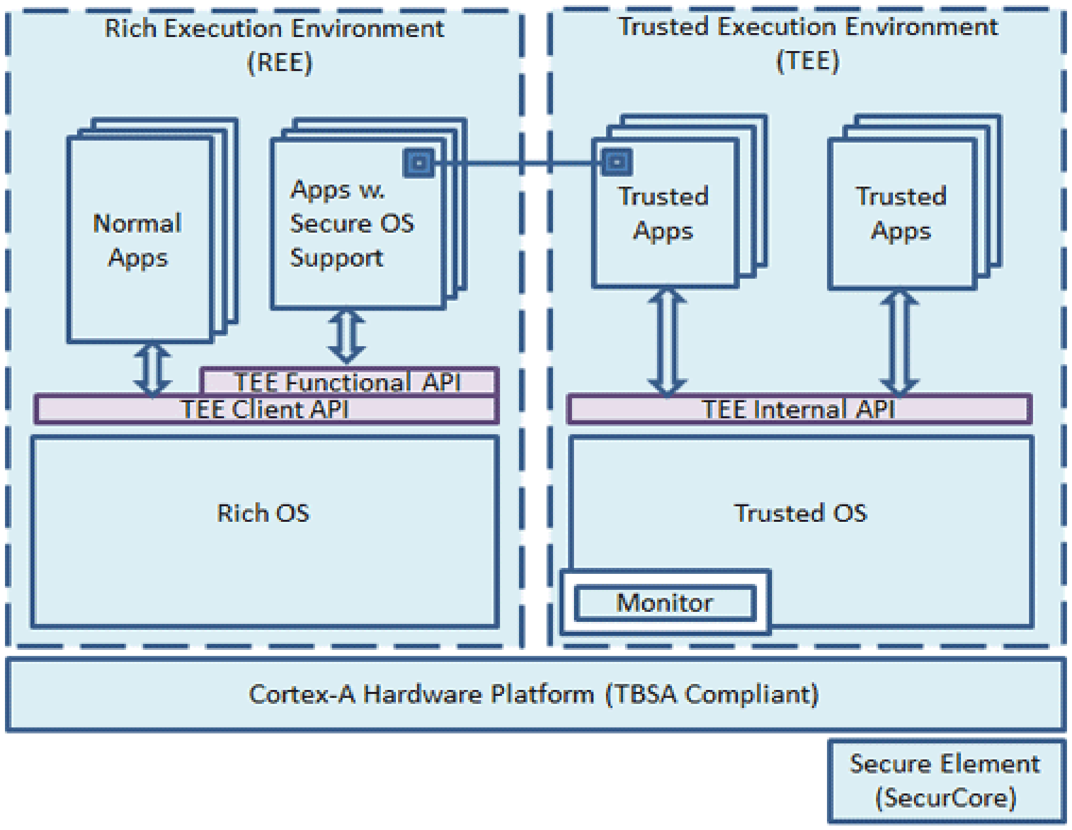

Rich Execution Environment (REE)
Trusted Execution Environment (TEE)
Normal Apps
Apps w. Secure OS Support
Trusted Apps
Trusted Apps
TEE Functional API
TEE Client API
TEE Internal API
Rich OS
Trusted OS
Monitor
Cortex-A Hardware Platform (TBSA Compliant)
Secure Element (SecurCore)

Figure 7: TrustZone Trusted Execution Environment.[38]

[38] ARM, "Building a Secure System using TrustZone Technology," 2013.

As related to secure inverter communications, inverter communication channels can be separated and designated according to data sensitivity levels. A separate communication processor could be designated for each communications channel. To ensure that cross-channel security breaches do not occur, each communication processor execute as separate secure processes with the TEE while standard inverter functions execute with the REE.

5.5. Post-Quantum Crypto

The quantum computing threat to cryptographic one-way functions used heavily in key-exchange protocols has been discussed in Section 4.2. The following section provides a brief overview of efforts underway to standardize quantum-safe digital signatures and key establishment, including discussion of efficiency for uses in embedded systems.

Post-quantum crypto systems gaining the most attention in recent years have been based on lattices, error correcting codes, hash functions and multivariate equations. The long-known code-based McEliece crypto system which was largely passed over for many years is of renewed interest as it is immune to certain known quantum-enabled attacks[39]. It requires, however, much larger key sizes than other algorithms. For 80-bit security McEliece required a key size of 520,047 bits compared to a 1024-bit key for RSA, where the number of bits of security is an indication of how much work is believed to be required to break a cryptographic algorithm with respect to the types of known attacks against the algorithm[40]. For example, symmetric key size in bits will indicate security bits (most of the time), but attacks against asymmetric cryptography may run faster than brute-force attacks, so achieving 128-bit security, for example, requires a (possibly much) larger key. Today, 128-bit security is considered secure against conventional computing brute-force attacks.

In 2012 a lattice system was optimized with "moderate signature and key sizes as well as performance suitable for embedded and hardware systems"[41]. Subsequent advances in security measures for lattice schemes are seen in the NIST submissions for digital signatures and key establishment from the CRYSTALS team[42].

In addition to quantum resistance, PQCrypto submissions include proposals using key encapsulation mechanisms (KEM) intended to improve upon existing key distribution schemes. Current systems that use public key cryptography to encrypt shorter symmetric keys must pad the symmetric key prior to encrypting, with RSA or ECC for example, to prevent attacks against the short payload of the public key encryption scheme. The dilemma is that there are also known

[39] R J. McEliece, "A Public-Key Cryptosystem Based On Algebraic Coding Theory" DSN Progress Report, 44, 114–116, 1978.
[40] NIST Special Publication 800-57 Part 1, Revision 4, "Recommendation for Key Management, Part 1: General," January 2016.
[41] T. Güneysu, V. Lyubashevsky, T. Pöppelmann, "Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems," In: E. Prouff, P. Schaumont (eds) Cryptographic Hardware and Embedded Systems – CHES 2012. CHES 2012. Lecture Notes in Computer Science, vol 7428. Springer, Berlin, Heidelberg, 2012.
[42] "CRYSTALS, Cryptographic Suite for Algebraic Lattices" accessed 1/28/19, URL: https://pq-crystals.org/index.shtml

padding attacks[43]. KEM solves this problem. Rather than encrypting and sending the shorter symmetric key, a longer random number is encrypted and sent, and then both sides use this value in an agreed upon key derivation function (KDF) to determine a shared symmetric key. In this way, no padding is needed, and no proof of padding security is needed.

Research continues with various potential quantum-safe schemes toward the goal of jointly achieving security and efficiency. As described in recent articles[44], too simple a scheme can be broken by classical computing; too difficult a scheme is deemed inefficient for realistic uses. NIST recently completed the first round of a call[45] for algorithms that are both quantum-resistant and realizable on conventional computers in order to replace the existing signature and keying schemes.

5.6. Distributed Trust Models using Blockchain PKI

As opposed to using a centralized certificate authority, it is possible to utilize a blockchain PKI to track transactions. It is not a replacement for authentication. The process of converting an established centralized trust model to a decentralized model would require a complete overhaul of the trust chain and major revisions to the standards, but it is technically possible. The basic idea of a blockchain is quite simple: it is a shared, replicated log file (sometimes called a ledger). The entries are sequential and time-stamped. A one-way function produces a short sequence of bits that is dependent on all of the items that are placed in the log. The one-way mathematical function ensures that it would be extremely difficult to produce a different log with the same output. The output of the one-way function is an abbreviation for the log itself, and when adding new entries the function uses its current value and the contents of every new entry to compute a new output. Those parties that maintain the log, publish the log and the output value so that independent parties are able to verify correspondence. It should be noted that blockchain has primarily been utilized to measure the validity of a node's currency balance in a cryptocurrency network. With respect to PKI, the notion of currency balance is replaced with the concept of a node's trust level in examples given below.

5.6.1. Blockchain Basics

The typical blockchain consists of a series of blocks that consists of a historical transaction record list in a public ledger form. In Figure 8, an example blockchain is illustrated where each block is linked by a block hash within the block header. The block hash used for linking each block together is referred to as a *parent block*. There are other hashes stored in a block which are referred to as *uncle blocks* and are children of the parent block. The *genesis block* (block *i*) does not have a parent block.

[43] RSA, "Key Encapsulation: A New Scheme for Public-Key Encryption," XML Security Working Group F2F, May 2009.
[44] N. Wolchover, "The Tricky Encryption That Could Stump Quantum Computers," WIRED, 15 Sept 2015. URL: https://www.wired.com/2015/09/tricky-encryption-stump-quantum-computers/
[45] NIST CSRC, "Post-Quantum Cryptography, Post-Quantum Cryptography Standardization" accessed 1/28/19, URL: https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization

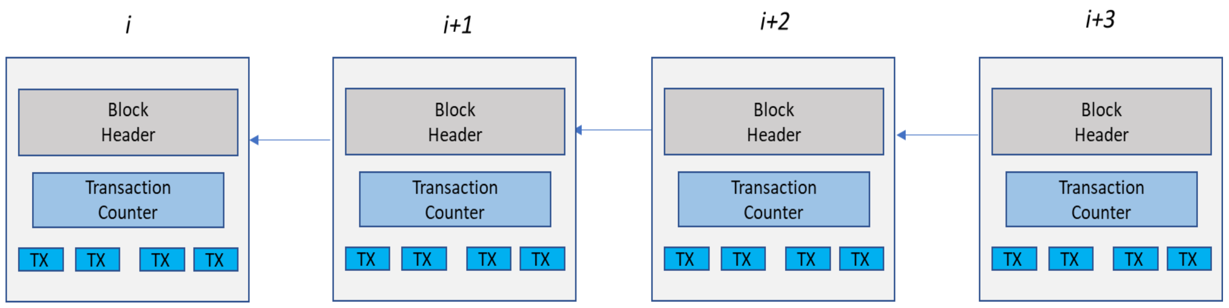


Figure 8: Sequence of blocks.

Figure 9 below shows the basic components of a block which consists of a *block header* and *block body*. The block header components include:

- ***Parent block hash*** which is a 256-bit hash linked to a preceding block.
- ***nBits*** is the target threshold of a valid block hash.
- ***Timestamp*** is the current time as seconds in UTC (universal time since January 1, 1970).
- ***Merkle tree root hash*** is the hash value calculated for all the transactions in a block.
- ***Nonce*** is a four byte field that generally starts at zero or a randomly generated value and increases as each hash is calculated.
- ***Block version*** is used to determine the set of block validation rules to use.

The block body consists of a *transaction counter* and a set of individual *transactions* with the maximum number of transactions limited by the block and individual transaction sizes. Authentication of transactions is achieved using asymmetric cryptography and the use of digital signatures is employed in high-threat circumstances. The advantages of using blockchain are derived from its ability to support *party anonymity*, *persistency*, *decentralization* and *auditability*:

- *Anonymity* for each party is provided through the use of generated addresses that mask the identity of each party, yet allow the identity of the party to be verified. Parties perform trusted and encrypted transactions using private-public keys linked to their generated addresses. The transactional data associated with each party is publicly visible and thus absolute privacy in this respect is not maintained.

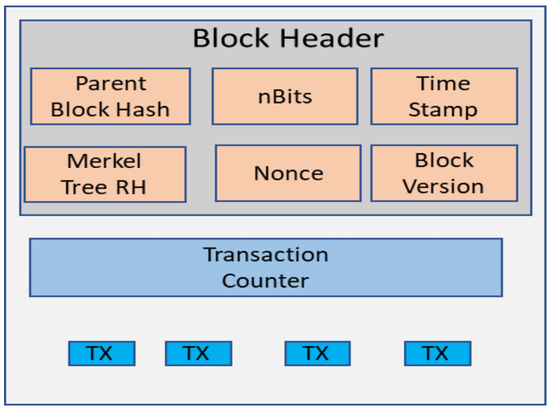


Figure 9: Block elements.

- *Persistency* is supported by the fact that transactions are validated continually, and blockchain is designed to permanently store transactions after inclusion in the blockchain. The inherent transactional structure of blockchain ensures that rapid identification of invalid transactions within individual blocks is maintained.
- *Decentralization* is afforded in blockchain through the use of consensus algorithms. The consensus algorithms allow for the validation of transactions in a decentralized manner which alleviates the performance bottlenecks often experienced when using centralized trusted agency servers. Additionally, not using centralized trusted third-party servers reduces overall infrastructure costs and naturally promotes efficient cooperative, distributed computing practices.
- *Auditability* is supported through the use of transaction linking. When transactions are written to the blockchain, the current state of the transaction is recorded in the transaction entry. For example, in the case of cryptocurrency, a specific state is represented by a user's currency account balance, and an unspent transaction is linked to a spent transaction. When currency is spent an unspent transaction will transition from an unspent to spent state with an associated balance reduction.

5.6.2. Blockchain Types

Blockchains are configured as either private, public or consortium blockchains. Private blockchains allow only those nodes originating from one specific organization to participate in the consensus process and is controlled by a single organization. Public blockchains expose all records to the public which allows all parties to participate in the consensus process. In a consortium blockchain only specific nodes are allowed to engage in the consensus process while other nodes are able to access the blockchain, but not participate in the consensus process itself. In terms of record immutability, the public blockchain is superior. Because of the distributed storage arrangement of records on numerous public nodes, it is very difficult for a malicious party to tamper with transaction records. A centrally managed private blockchain where transaction records are managed by one organization is more easily targeted.

In terms of transaction efficiency, the public blockchain is less efficient due to the propagation delay associated with sending transactions and blocks to a large number of nodes. Transaction throughput and latency limitations are major drawbacks in a public blockchain, and high transaction throughput blockchain systems are generally of either the consortium or private variety due to the reduced number of validating and storage nodes. The chief advantage of public blockchain is its openness and ability to attract a large number of users and communities. Consortium blockchain being more centrally managed and with a relatively high transaction throughput is gaining ground within the business community. Ethereum [46] and Hyperledger [47] are two open source projects currently developing tools and frameworks for business consortium blockchain application.

[46] Consortium chain development." URL: https://github.com/ethereum/wiki/wiki/Consortium-Chain-Development
[47] "Hyperledger project," 2018. URL: https://www.hyperledger.org/

5.6.3. Blockchain Consensus Algorithms

It is required in blockchain that all ledgers on distributed nodes be identical, and various consensus protocols are used to ensure ledger consistency. Since there is no central node to perform this function, individual nodes in the blockchain must come to a consensus during transaction processing. A few of the more common consensus protocols that can be applied to PKI are discussed below:

The *Proof of work (PoW)* consensus algorithm is based on a decentralized network where a node is selected to record all transactions, and each *miner* node (those participating in the consensus process) creates a hash value using the block header nonce. The block header nonce is modified periodically and a new hash value is calculated each time. Consensus criteria are met, when the calculated hash value is less than or equal to a given target value. Once a node's calculated hash reaches the target value, the block is broadcasted to the other mining nodes. The other miners verify the hash for correctness. If the transmitted block hash is verified as correct, the other miners append the new block to their stored blockchains. In some situations, suitable hashes are found simultaneously by different miners, and various branches of validated blocks are subsequently stored by all mining nodes in the blockchain network. In such cases the longest block branch stored in the blockchain is deemed the most reliable set of blocks, and the shorter block branches are then abandoned.

The *Proof of Stake (PoS)* consensus algorithm in the case of PKI would base the validity of node on the level of trust the node has obtained. In situations where there are nodes that have obtained a high level of trust, there is the drawback that these nodes will dominate the blockchain network. With such high-trust nodes, the blockchain could consist almost exclusively of blocks created by a high-trust node. To counter the potential domination of high-trust nodes, block generator selection strategies such as randomized selection of block generators, consideration of a node's age or consideration of the lowest hash value in combination with the PoS are employed.

Delegated Proof of Stake (DPoS) consensus protocol differs from PoS in that PoS uses a direct democratic process while DPoS uses a representative democratic process. In DPoS delegates are elected to generate and validate blocks. The result is that a smaller number nodes are responsible for validating each block, and block confirmation time and resources are greatly conserved over PoS. An additional benefit derived in DPoS is that delegates are able to adjust block size and block intervals. Delegate nodes that consistently under-perform or lose trust are not re-elected.

The *Ripple*[48] consensus algorithm uses trusted subnetworks within the overall blockchain network to designate selected server nodes. All other nodes are designated client nodes and are used only for transaction processing, excluding block validation and storage. In Ripple, each server node has an Unique Node List (UNL) which is used by the server to determine the validity of a transaction. When the trusted subnetwork servers receive a transaction for validation, each individual server searches its UNL lists to locate the initiating node's identity. If 80% of the nodes in the trusted subnetwork agree on the validity of the transaction, the transaction is added to the ledger.

[48] D. Schwartz, N. Youngs, and A. Britto, "The ripple protocol consensus algorithm," *Ripple Labs Inc White Paper*, vol. 5, 2014.

Tendermint[49] is a byzantine consensus algorithm where a new block is determined in each round, and a proposer node is selected to broadcast an unconfirmed block in the round. The algorithm works as follows. A validators decides to broadcast a pre-vote for the proposed block which is followed by pre-commit step. In the pre-commit step if the proposer node has received more than 2/3 positive responses during the pre-vote step, the proposer broadcasts a pre-commit message. If the proposer node receives greater than 2/3 positive responses in the pre-commit stage, it enters the commit stage. In the commit stage, the proposer node validates the block and broadcasts a commit for that block. If the proposer node receives 2/3 positive responses in the commit stage, the block is accepted as valid and added to the blockchain.

There are strengths and weaknesses associated with each consensus algorithm. In the case of Tendermint the identity of each validator needs to be known to select a proposer node where in PoS, PoW, Ripple and DPOS, nodes are able to join the blockchain network ad hoc. PoW requires that miners hash the block header nonce numerous times until the target value is reached which can consume vast amounts of computer resources. Some computer resource savings are conserved when using DPoS and PoS because the number of miner nodes computing the block nonce hash is greatly reduced. In Tendermint and Ripple, computing resources are conserved because iterative nonce hashing in the consensus process is not performed. Each of the consensus protocols will tolerate a distinct level of adversity. Theoretically, using PoW an adversary will need to generate 51% of the total hashing power of a blockchain network in order to gain control. However using a selfish mining strategy where pooled miners are intent on breaking the PoW protocol, only 25% of the total hashing power is required to gain significant control[50]. In terms of faulty node tolerance, Tendermint is designed to handle up to 1/3 faulty nodes while Ripple can operate correctly when the number of faulty nodes in the UNL is less than 20%.

5.6.4. Blockchain Limitations and Vulnerabilities

Blockchain being highly transactional and iterative, demands a large amount of storage and computing power on large blockchain networks. Every node in the network is required to store each transaction and validate each transaction's associated chain. This will be a major challenge for DER equipment and embedded devices with limited computational power and storage. Additionally, blockchain is able to shield a party's private identity information using public key encryption, but transactional privacy cannot be fully preserved as portions of all transactions of a party must be public for validation purposes. Certain block chain services do provide partial obfuscation of transactions through the use of ring signatures. A ring signature[51] uses an account holder's keys and selected public keys from the blockchain to form a ring of possible signers. Ring members are able to validate encrypted transactions, but will not themselves be able to match a specific account holder with a transaction. Using ring signatures makes it impossible to detect the specific group member that created the signature. Generally speaking, most signers will encrypt a

[49] J. Kwon, "Tendermint: Consensus without mining," *URL http://tendermint. com/docs/tendermint { } v04. pdf*, 2014.

[50] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proceedings of International Conference on Financial Cryptography and Data Security*, Berlin, Heidelberg, 2014, pp. 436–454

[51] R. Rivest, A. Shamir, Y. Tauman, ASIACRYPT 2001. Volume 2248 of Lecture Notes in Computer Science, pages 552–565.

portion of the transaction such that a third party is still able to publicly confirm that a valid transaction took place.

It has been shown that selfish miners are able to control and exploit a blockchain network using relatively low network hashing powers[52]. Selfish mining generally entails hording mined blocks without broadcasting, and the corresponding private branch is made public only when required by the network. Prior to publishing the horded private chain, honest miners are wasting time and computing resources on an outdated public branch while selfish miners are mining their private chain without competitors.

5.6.5. Blockchain Used as a PKI

In summary, blockchain's security works because the ledger entries are signed by private keys, validated by public keys and have a standardized format. Additionally, distributing the maintenance of the blockchain to numerous parties and achieving a distributed consensus, works effectively when the majority of the parties have a mutual interest in ensuring a correct and consistent set of blockchain transactions.

A blockchain PKI consists of a set of entries that are associated with the identity of an individual entity. In the case of an entity such as a smart inverter, the entity's public key is placed in blockchain entry, where it is made accessible to relying parties. The entity's public key for example is signed by a certifying authority (CA) to assert that the public key belongs to the entity. The improvement over standard centralized X.509 v3 based PKI is that blockchain enables block replication by a decentralized set of entities which makes PKI evolutions such as certificate revocation, renewals and credential usage patterns visible to the other PKI members. This transparency increases overall PKI system reliability and accessibility. Blockchain also provides an immutable history by maintaining a timestamped blockchain where transaction accountability and validity is strictly maintained throughout the life span of the blockchain. In standard PKI, when an entity is no longer trusted, it is common practice to revoke the entity's certificate via OCSP or revocation lists. In many cases the revocation process using OCSP and revocation lists takes an extensive amount of time due to the fact that each authenticating party in the PKI must first gain access to such revocation resources. When an entity is no longer trusted in blockchain PKI, the CA simply broadcasts a revocation block to all nodes in the blockchain network.

Openchain[53] is an open source blockchain platform upon which a blockchain PKI can be constructed. In Openchain the consensus algorithm is called partitioned consensus where each blockchain network has a single transaction validator. As explained above, blockchain consensus processing and validation can be very resource intensive operations. Having a single transaction validator reduces the number of required resources substantially. Additionally, each Openchain blockchain network controls their own ledger, but the networks are able to communicate with one another. Distinct Openchain network transaction validators are able to selectively authenticate transactions depending on their specific realm of authority. This allows organizations to maintain

[52] I. Eyal and E. G. Sirer, "Majority is not enough: Bitcoin mining is vulnerable," in *Proceedings of International Conference on Financial Cryptography and Data Security*, Berlin, Heidelberg, 2014, pp. 436–454.
[53] "Openchain Project," 2018 .URL: https://www.openchain.org/

multiple blockchain instances under different authority types. Having an enterprise of distributed validators and authorities is very similar to a standard hierarchical PKI system, but with the added potential benefits of blockchain PKI (faster more reliable revocation, greater accountability/auditability).

Figure 10 shows at a high level a two network Openchain based blockchain system. Transactions and blocks are validated by validators N_1 and N_2 according to the specific security policies of the respective networks. The certifying authority is able to generate key-pairs and sign public keys for each of the respective blockchain nodes with the respective networks. The validators are able to check the signatures of signed blocks and transactions transiting between the networks or within the networks. The validator can also participate in or lead a respective consensus process with the other nodes. If a validator finds that a node is behaving maliciously, it can generate a revocation block/transaction that can be added to the shared blockchain.

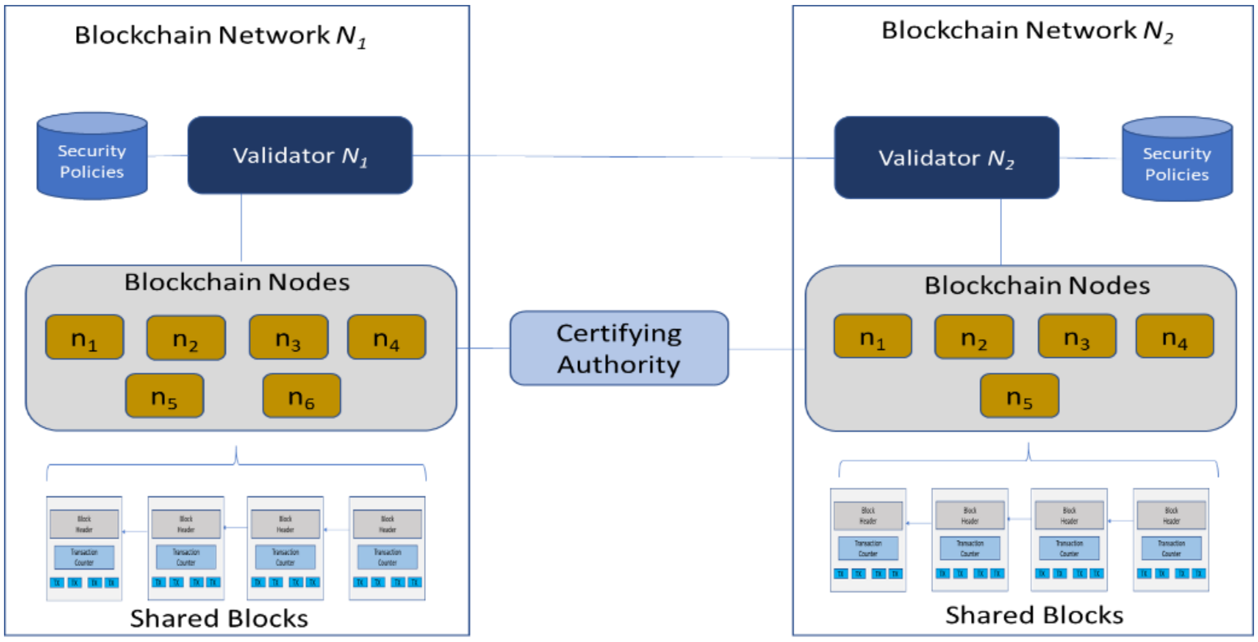


Figure 10: Openchain Blockchain PKI.

6. RECOMMENDATIONS FOR DER TRUST AND ENCRYPTION

6.1. Secure Application Enabled Devices

As quantum computing matures, the ability of an attacker to launch successful brute force attacks against encrypted communications channels will become a pervasive and relevant threat. The signature and encryption mechanisms and frameworks that will be adopted in the future may vary to some extent to accommodate longer key sizes (McEliece requiring a key size of 520,047). Current cryptographic processor architectures processing these substantially larger key sizes will undoubtedly experience high degrees of latency. In compute limited devices such as smart inverters, it will be necessary to incorporate miniaturized, low-power processors free of the von Neumann bottleneck effects that current processor architectures possess. Because of their low-power, nanoscale and inline computing characteristics, the incorporation of dedicated neuromorphic cryptographic co-processors within smart inverters is a viable solution. Such processors do this by departing from the theory of a Turing machine and the von Neumann architecture, and do not use sequential instruction sets or are composed of physically separated CPU, memory and permanent storage. Instead, neuromorphic processing systems are silicon based neural network counterparts of the human brain that perform calculations using high throughput weighted inter-synaptic learning. Because neuromorphic processors mirror the brain in terms of neuron and synaptic structure, they are particularly suited for efficient execution of artificial neural network encryption/decryption function approximators. It has already been illustrated that through the use of a constant time parallelized synaptic integrated neuromorphic sieve, it is possible to efficiently detect smooth numbers in discovering numerical encryption factors.[54] The neuromorphic sieve was implemented on an IBM Neurosynaptic System (NS1e)[55] which is a single chip neuromorphic coprocessor containing 4096 cores with 256 leaky integrate-and-fire (LIF)[56] neurons per core, and able to simulate a million spiking neurons while only using 100 milliwatts of power at the normal operating frequency of 1 KHz.

One experimental implementation currently available is BrainScale's neuromorphic hardware. BrainScale[57] is based on very large scale integration (VLSI) 20-cm-diameter silicon wafer containing 384 chips. Each chip contains 128,000 synapses and a maximum of 512 spiking neurons. In total on each wafer there are approximately 200,000 neurons and 49 million synapses. A BrainScale based function approximation neural network could potentially be trained to rapidly encrypt/decrypt data thousands of times faster than current processors[58].

6.2. Legacy Device Support

Utilities are slated to communicate securely via net-metering with smart inverters; however, in the current IEEE 2030.5 and CSIP, there are no specifications or requirements defined for legacy inverters. Historical trends within the power grid have traditionally shown that security needs alone

[54] J.V. Monaco, et al., "Integer factorization with a neuromorphic sieve," *2017 IEEE International Symposium on Circuits and Systems (ISCAS),* 2017.

[55] P.A Merolla, et al., "A million spiking-neuron integrated circuit with a scalable communication network and interface. Science," vol. 345, no. 6197, pp. 668–673, 2014.

[56] E. Hunsberger, C. Eliasmith, "Spiking Deep Networks with LIF Neurons", *arXiv, 2015.*

[57] "BrainScale", 2018, URL: https://www.hpcwire.com/2016/03/21/lacking-breakthrough-neuromorphic-computing-steadily-advance/.

[58] "Neural Networks and Deep Learning", 2018, URL: http://neuralnetworksanddeeplearning.com/chap4.html.

will shorten the lifecycle of critical operational components such as inverters and aggregators. For years to come, a number of inverters will continue to communicate with aggregators—and possibly utilities—using nonstandard proprietary protocols. If such legacy inverters are accessible over TCP/IP connections for example, they potentially will serve as launch points for an attacker. Such an attacker could, for example, easily perform reconnaissance on networks, identify attack vectors and disrupt grid services via DoS and cyber-physical attacks. In legacy inverters that support firmware updates, it may be possible to reduce the overall attack surface significantly by adding cryptographic and trust management capabilities via firmware updates.

6.3. Device Secure Key Storage

As per IEEE 2030.5 and the CSIP, the steps required for enabling PKI include:

- Generation of a key pair for each device from which a CSR will be created and submitted to a root CA.
- The root CA will sign a certificate and send it back to the manufacturer. The manufacturer will then place the keypair and certificate on the inverter.

Maintaining secure trust anchors in a secure communications network is extremely important, but very difficult to ensure when devices are minimally protected and widely distributed. It is important to securely store device private keys and provide a means for each device to securely attest to its identity. Additionally, it is important that distributed device private keys cannot be extracted from these devices physically or electronically. To physically protect the private keys, key storage components should be sealed in masked, tamper and extraction proof casings. To support proper trust establishment, communicating devices must provide a secure signing capability for the creation of attestation tokens. ARM TrustZone and MTM implementations are core lightweight solutions suited for small form-factor, compute limited inverters. Each of these solutions provide an attestation capability for trust establishment and options for secure key storage. Because both TrustZone and MTM are commercially available today, these are recommended solutions for inclusion in near-term smart inverter designs.

IEEE 2030.5 or CSIP do not explicitly mention is how the keypairs and certificate should be stored on the device. In higher security situations the private key should be stored in an obfuscated and secure physically tamper resistant section of the inverter. When security is of highest importance, the DER should also contain physical tracking capabilities enabling asset tracking through a GPS/RFID aware system. In this way if a device is physically removed to extract its private key, a monitoring system will be aware of the event. Tamper resistant mechanisms should be employed, include the use of hardened packaging where the private key itself is destroyed when an adversary attempts to extract it. Additionally, obfuscation of the private key[59] hides the byte sequence of the private key from an adversary. Un-obfuscated byte sequences can be viewed using various non-destructive electronic external interfaces or sometimes using advanced penetrating imaging[60].

Arguably, the loss of a single DER private key would not present a significant risk to the power system. However, if this communication channel is then used to gain a foothold at the aggregator

[59] S. Koteshwara, C. H. Kim and K. K. Parhi, "Functional Encryption of Integrated Circuits by Key-based Hybrid Obfuscation", March 2017.
[60] M. Holler, M. Guizar-Sicairos, E. Tsai, "High Resolution non-destructive Three-dimensional Imaging of Integrated Circuits", Nature, March 2017.

or utility server, the risk would be far more significant. If a private key is successfully extracted or recreated, an undetected adversary could potentially spoof downstream operational components. The amount of damage that a device can have in the infrastructure depends entirely on its role and the number of nodes (i.e., aggregator, microinverter, etc.) communicating with it.

6.4. Certificate Maintenance

The obvious maintenance issues in a PKI system is keeping certificates updated. (1 year certificate life is the usual recommended renewal interval). This requires that a key pair be generated for each device prior to expiration, a Certificate Signing Request (CSR) issued to the CA, and the CA then returning the signed certificate to the device for secure placement. In addition, if for example, it has been detected that a private key has been compromised, there needs to be a Certificate Revocation List (CRL) or other mechanism which warns the CA and all communicating nodes of the compromise. The CA then needs to correspond with the device administrator at which time a new key pair is generated, CSR submitted to the CA, and new certificate and key pair placed securely on the device. Additionally, all parties communicating securely with the device need to be warned through a running Online Certificate Status Protocol Service (OCSP)[61] that a new certificate must be obtained prior to communicating with that compromised device. A CRL is an offline way to notify the nodes under a PKI that a new certificate needs to be obtained prior to communicating with a compromised device. While OCSP is a running service, CRLs are offline distributed lists that require periodic updating. Because OCSP draws from a centralized database of revoked certificates, it is the preferred means for certificate revocation information. In the case that network connectivity cannot be guaranteed, it is often necessary to resort to the use of CRLs.

The statements addressing the management of certificates in IEEE 2030.5 are as follows: "'IEEE 2030.5 Cert – Indef' is meant to convey that Manufacturing PKI certificates are indefinitely valid and the check is limited solely to a check of the signatures on the certificate chain. The phrase 'Optional OCSP' means that the server device (and optionally, the client device) may utilize Online Certificate Status Protocol as an additional mechanism to determine if a certificate has been revoked. OCSP may only be used to verify non-IEEE 2030.5 certificates."

Inverter manufacturers strictly following these guidelines are not required to limit the life of a device certificate or to validate IEEE 2030.5 certificates using OCSP or CRLs. If a manufacturer decides not to implement certificate revocation and replacement, for a critical period an adversary could potentially extract a private key from a device and masquerade as the legitimate device without other nodes being aware of the device compromise. Correct use of OCSP and CRLs provides an automated systematic notification of a certificate revocation throughout the PKI. Manufacturers would naturally be a logical party to manage certificate revocation because they know device statuses, if the devices have the latest secure firmware, and whether the equipment is capable of further upgrades. However, in cases where vendors go out of business, this model would not be successful.

6.5. Legacy Device Trust

The current IEEE 2030.5 standard does not suggest support for legacy inverters, but it is possible to communicate to legacy inverters. These devices that have limited computing capabilities and do

[61] M. Myers, "RFC 2560 Internet Public Key Infrastructure Online Certificate Status Protocol – OCSP," June 1999.

not have pre-placed keys or key management hardware modules (i.e., MTM and TrustZone), so alternative authentication methods would need to be incorporated. These alternative methods essentially involve assembling a fingerprint for a device using operational or physical characteristics. A legacy inverter itself may not have the capability to encrypt/decrypt, etc., but an aggregator could positively identify an inverter's fingerprint. Such a fingerprinting scheme could be used to mitigate device IP or MAC address spoofing[62].

To be closely compliant with IEEE 2030.5's TLS secure communications scheme, legacy inverters or their utility/aggregator gateways would require firmware upgrades. Ideally, future truck rolls could be avoided if the legacy device upgrades would include the ability to calculate device fingerprints via Physical Unclonable Functions (PUF)[63,64] and the necessary TLS libraries. The steps involved in configuring a legacy device for secure IEEE 2030.5 TLS compliant communications is as follows:

1. A key pair is generated in the device using a PUF calculated seed.
2. The keypair is then used to form a CSR which is submitted to a CA.
3. The CA then returns a signed certificate to the device.
4. TLS secured communications can then be conducted.

One advantage to using a fingerprint over a randomly generated key as the root of trust is that if the algorithm used to generate the fingerprint is kept centrally secret, the fingerprint cannot be easily extracted and used in masquerading. Additionally, the device's physical identity can be positively verified via signed content.

6.6. Physical Security

While this document focused on data-in-flight trust and encryption of the communication protocols, physical security should also be studied. For instance, it is possible to mask the microprocessor chip type and manufacturer with an opaque conformal coating or some other obfuscation method so that the architecture (and associated vulnerabilities) are not known to the adversary. Anti-tamper protections like those used with AMI meters should be used by PV inverter manufacturers and additional physical security options should be investigated and recommended to the solar industry. These considerations will need to be included in a comprehensive DER cybersecurity approach and should be included in future work.

[62] E. Cardenas, "MAC Spoofing--An Introduction," GIAC Security Essentials Certification. SANS Institute, February 2013.
[63] T. Bauer, J. Hamlet, "Physical Unclonable Functions: A Primer," IEEE Security and Privacy, December 2014.
[64] C. Jin, C. Herder, "FPGA Implementation of a Cryptographically-Secure PUF Based on Learning Parity with Noise," Cryptography, 2017.

7. CONCLUSIONS

In this paper, the state-of-the-art encryption and trust features for DER communication protocols were investigated and alternative methods were presented for possible inclusion in future standards. The California implementation of IEEE 2030.5 was studied in depth to determine what gaps exist in modern DER security standards. The majority of the gaps revolve around the permanence of the implementation or details where the standard is ambiguous or silent. Since most DER devices are likely to operate for decades, there is concern that the keys never expire, there is no formal revocation process, and there is no way to change the cryptographic algorithm. As the industry transitions from theory to implementation, it is anticipated that additional weaknesses will be exposed in the standards. To generate strong, standardized, trusted DER communications, the following improvements are recommended:

- Create a thorough Certificate Policy defining security procedures, policies, and practices.
- Utilize OCSP and CRLs for certificate revocation when compromise of keys is known or suspected.
- Define the security requirements for aggregator IEEE 2030.5 servers and TLS interception methods at utilities and aggregators.
- Employ Physical Unclonable Functions (PUFs) to create device fingerprints to ensure DER and server identities.
- Adopt post-quantum cryptography solutions as they are developed.
- Create a set of physical security requirements and include methods to prevent key extraction, like those provided by the Trusted Platform Module (TPM) and Mobile Trusted Module (MTM).
- Implement Trusted Execution Environments like TrustZone to prevent unauthorized tampering of executable code over the network.
- Continue to investigate the use of per-application VPNs and blockchain PKI which could potentially provide faster more reliable revocation with greater accountability and auditability. Additional research, prototyping, and pilot programs are needed to better understand how to effectively apply new technologies to DER security.

APPENDIX A

A.1 Encryption Methods

Classical encryption methods required a sender and receiver to agree on an invertible process to encipher and decipher messages. Mathematical encryption schemes that encipher data with a key value and can be mathematically reversed to restore the data with the same key are known as symmetric cryptography. In contrast, late in the twentieth century, mathematicians succeeded in devising asymmetric schemes using one-way functions with mathematically related public and private keys to overcome the need of separate channels for securely delivering shared secret keys.

Whether symmetric or asymmetric, encryption algorithms are categorized as stream ciphers or block ciphers. In the former, a message is encrypted one sequential character or digit at a time, while in the latter, a message is split into sequential blocks and encrypted one block at a time. Many crypto algorithms exist, featuring greater or lesser degrees of security, speed, compactness, or optimizations for hardware versus software implementations. We mention only a few cryptographic primitives here.

A.1.1. Symmetric Cryptography

Symmetric cryptographic algorithms, which must be mathematically reversible using the same key for both encryption and decryption, are frequently based on reversible substitution and permutation functions. As shown in Figure 11 the sender, depicted as 'Alice,' must transmit the cryptographic key, K, by means of a trusted conduit to the receiver, 'Bob.' Alice encrypts her message, m, with the key by means of some reversible mathematical function, E, composed of substitutions and permutations. The unintelligible ciphertext can now be transmitted over any unprotected open channel. Bob, having securely received the key from Alice, decrypts the message by reversing the function, E, with the shared key.

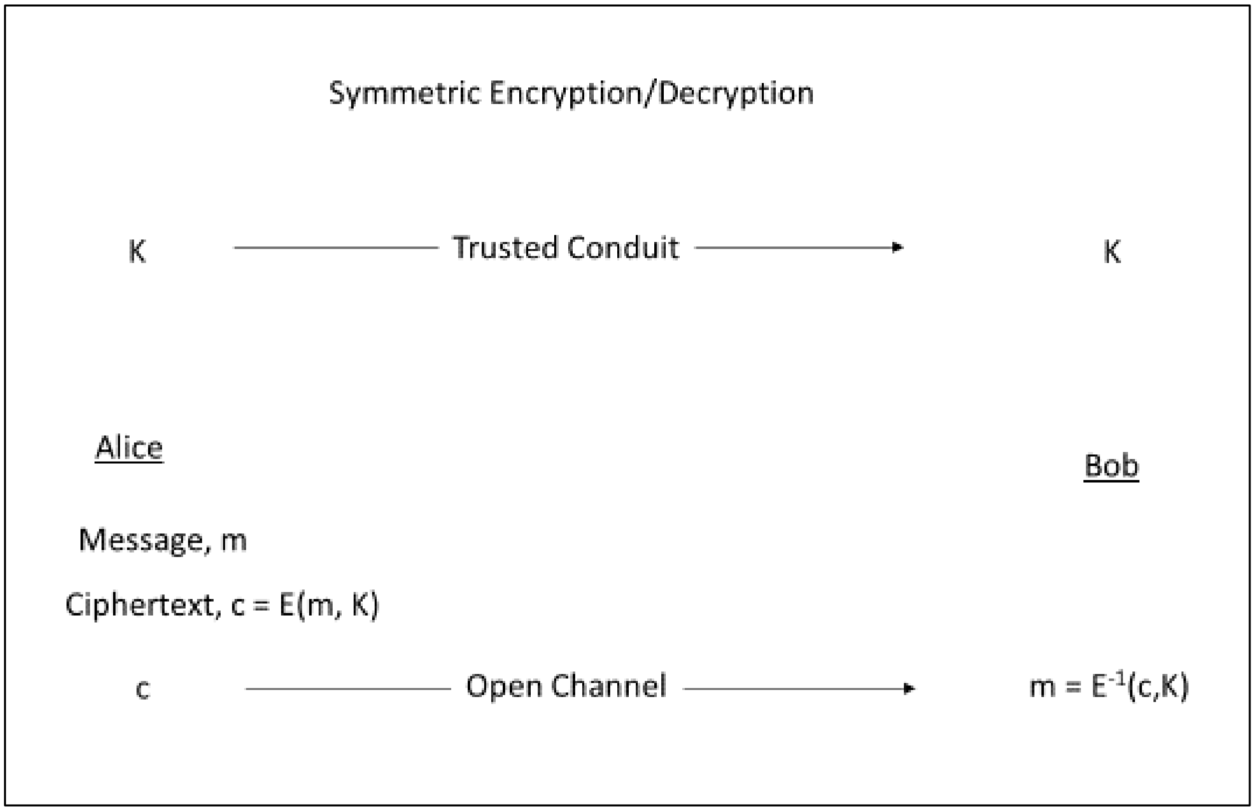


Figure 11: Symmetric Encryption/Decryption.

Advanced Encryption Standard (AES), standardized in NIST FIPS 197[65], is the symmetric algorithm most accepted for encryption uses. AES is a block cipher running each block of data through repeated cycles of substitution, shift, mix and addition operations.

Prior to AES, the accepted algorithm was the Data Encryption Standard (DES), another block cipher based on substitution and permutation operations. DES is deprecated due to brute force attacks against its relatively short key length, though it is considered secure when used in triplicate with three keys as in 3-DES.

A.1.2 Asymmetric Cryptography

Asymmetric cryptographic algorithms use pairs of mathematically related keys, with one part of the pair kept secret while the other is distributed to the public by a trusted party. As shown in Figure 12, the public key is used by the sender for encryption and the private one for decryption such that only the intended receiver holding the secret key is able to decipher the message. The two keys are related via a one-way mathematical function such that the private key cannot be determined from knowledge of the public key. Such one-way functions are for example based on large prime factorization or the discrete log problem, which are mathematically hard problems and computationally infeasible on conventional computers.

[65] NIST, FIPS 197, Advanced Encryption Standard (AES), URL: https://nvlpubs.nist.gov/nistpubs/fips/nist.fips.197.pdf

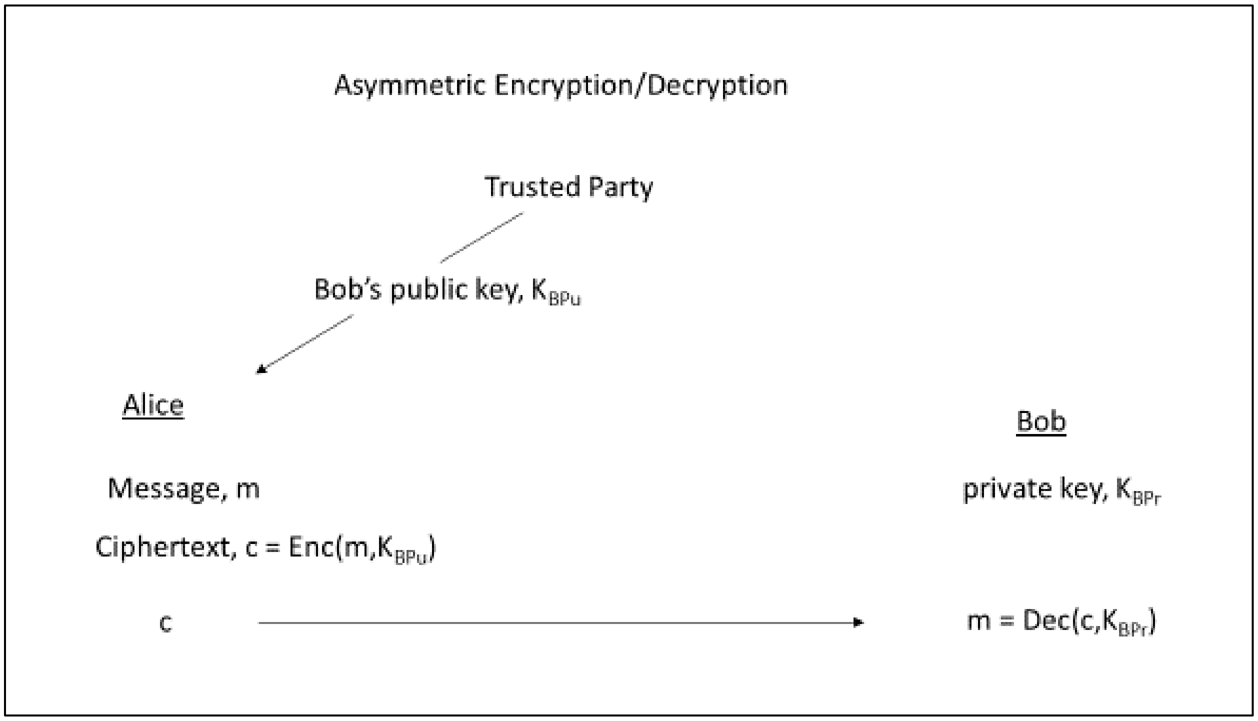

Asymmetric Encryption/Decryption
Trusted Party
Bob's public key, K_{BPu}
Alice
Bob
Message, m
private key, K_{BPr}
Ciphertext, c = Enc(m,K_{BPu})
c
m = Dec(c,K_{BPr})

Figure 12: Asymmetric Encryption/Decryption.

The earliest well-known asymmetric encryption algorithm is attributed to Rivest, Shamir and Adleman, the RSA algorithm from 1978, depicted in Figure 13, and is still used today. The algorithm is a block cipher consisting of a single modular exponentiation for both encryption and decryption, using the public key in one direction, and the private key in the other. For details on the mathematical relationship of public and private RSA keys, see The Handbook of Applied Cryptography[66].

[66] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, 1997.

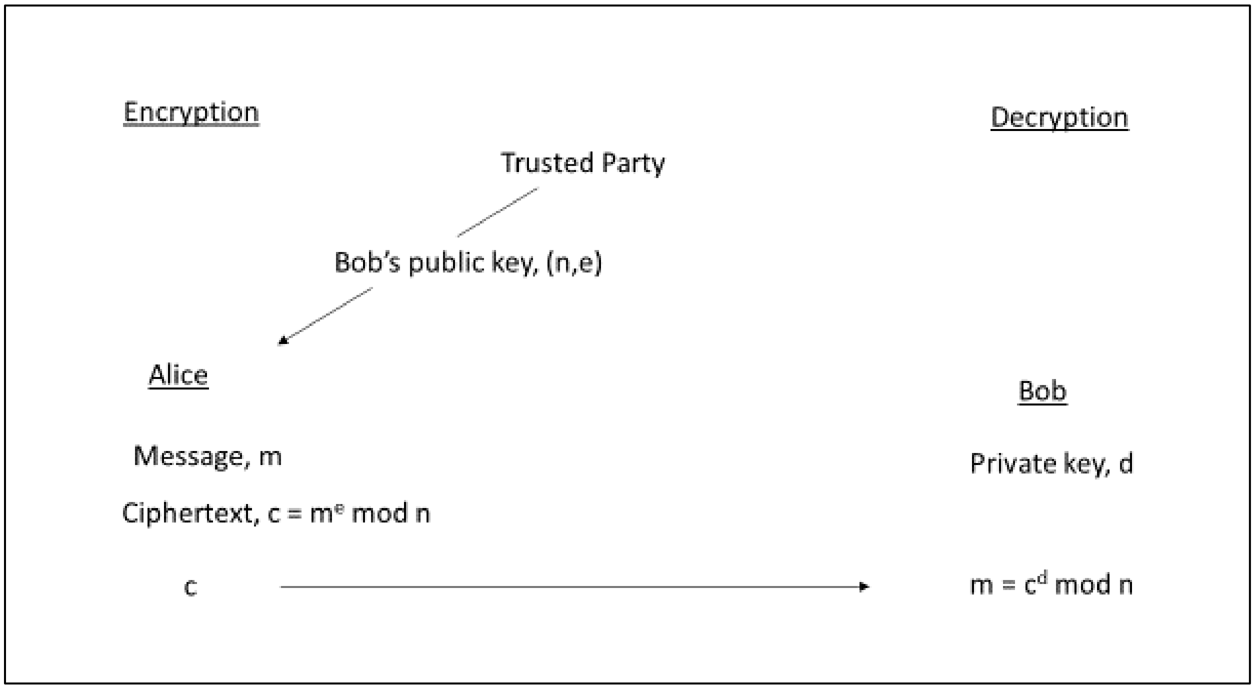


Figure 13: RSA Encryption/Decryption.

Elliptic-Curve Cryptography (ECC), NIST FIPS 186-3[67], has replaced RSA in some asymmetric implementations, offering greater security and faster processing speeds with smaller key sizes than RSA. In the case of ECC, the one-way function is the discrete-log problem derived from multiplication of a point on an elliptic curve[68]. ECC key establishment, encryption and decryption all rely on such point multiplication. As shown in Figure 14, addition of points P and Q on the elliptic curve E is the resulting intersection, R, of a straight line through the points with the curve itself, such that P + Q = R. Point doubling is the reflection of the intersection, R, of the tangent to the point P with the curve E, or P + P = 2P.

Any multiple of the elliptic curve point P can be found via repeated doubling and addition on the curve. However, the assumption of difficulty of inverting this operation provides sufficient computational intractability to consider the elliptic curve discrete log problem as a one-way function for asymmetric cryptography when used properly.

[67] NIST Computer Security Resource Center, FIPS 186-3, Digital Signature Standard (DSS), URL: https://csrc.nist.gov/csrc/media/publications/fips/186/3/archive/2009-06-25/documents/fips_186-3.pdf

[68] M. Musson, "Attacking the Elliptic Curve Discrete Logarithm Problem," Master Thesis of Science (Mathematics and Statistics), Acadia University, 2006

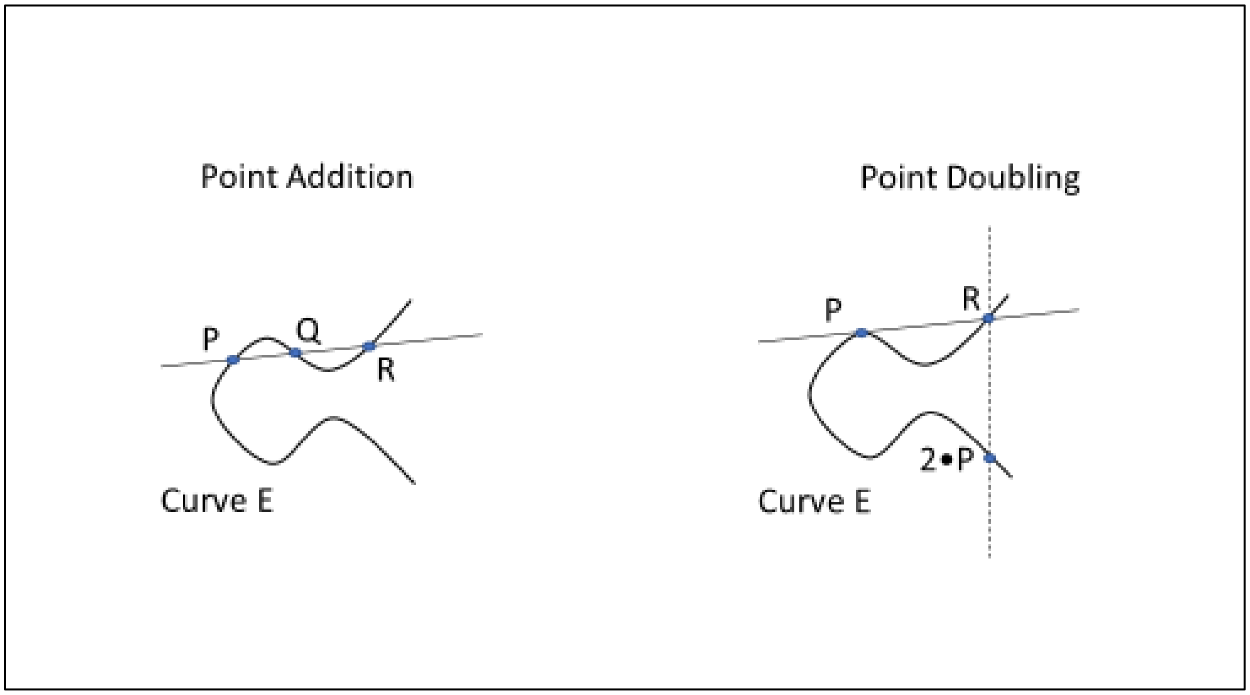


Figure 14: Elliptic Curve Operations.

An example elliptic curve encryption scheme uses the ElGamal cryptosystem[69] relying on simple elliptic curve math. As shown in Figure 15, Alice selects a secret integer, k, multiplies the curve base point, α, with her secret, k, resulting in the elliptic curve point y1=kα, and sends y1 to Bob over the open channel. At the same time, Bob selects a secret integer, a, multiplies the base point, α, with his secret, a, resulting in the point β=aα, and sends β to Alice over the open channel. Note that despite the fact that kα and aα are transmitted over an open channel, the secrets, k and a, are secure and cannot be recovered from kα and aα due to the difficulty of the elliptic curve discrete log problem. Alice next encrypts the message, x, by multiplying her secret, k, with the point β received from Bob, resulting in kβ, and adding this result to the message, x. The value y2 = x + kβ is the encrypted message and Alice sends y2 to Bob over the open channel. The point kβ is a secret shared by Alice and Bob because the point kβ, known by Alice, is kaα, and the point a•y1, known by Bob, is akα, equal to kβ. Therefore, Bob can recover the message, x, by calculating x = y2 - a•y1.

The elegantly simple math in this implementation, as well as other similarly described cryptographic primitives, however, must not be used without additional measures to prevent simple attacks. For an example of insecure implementation of elliptic curve, refer to *Breaking Plain ElGamal and Plain RSA Encryption* by Dan Boneh *et al*[70].

[69] W. Trappe, L. Washington, Introduction to cryptography: with coding theory (2nd ed.). Upper Saddle River, N.J.: Pearson Prentice Hall, 2006.
[70] D. Boneh, A. Joux, P. Nguyen, "Breaking Plain ElGamal and Plain RSA Encryption," URL: https://www-almasty.lip6.fr/~joux/pages/papers/PlainRSA.pdf

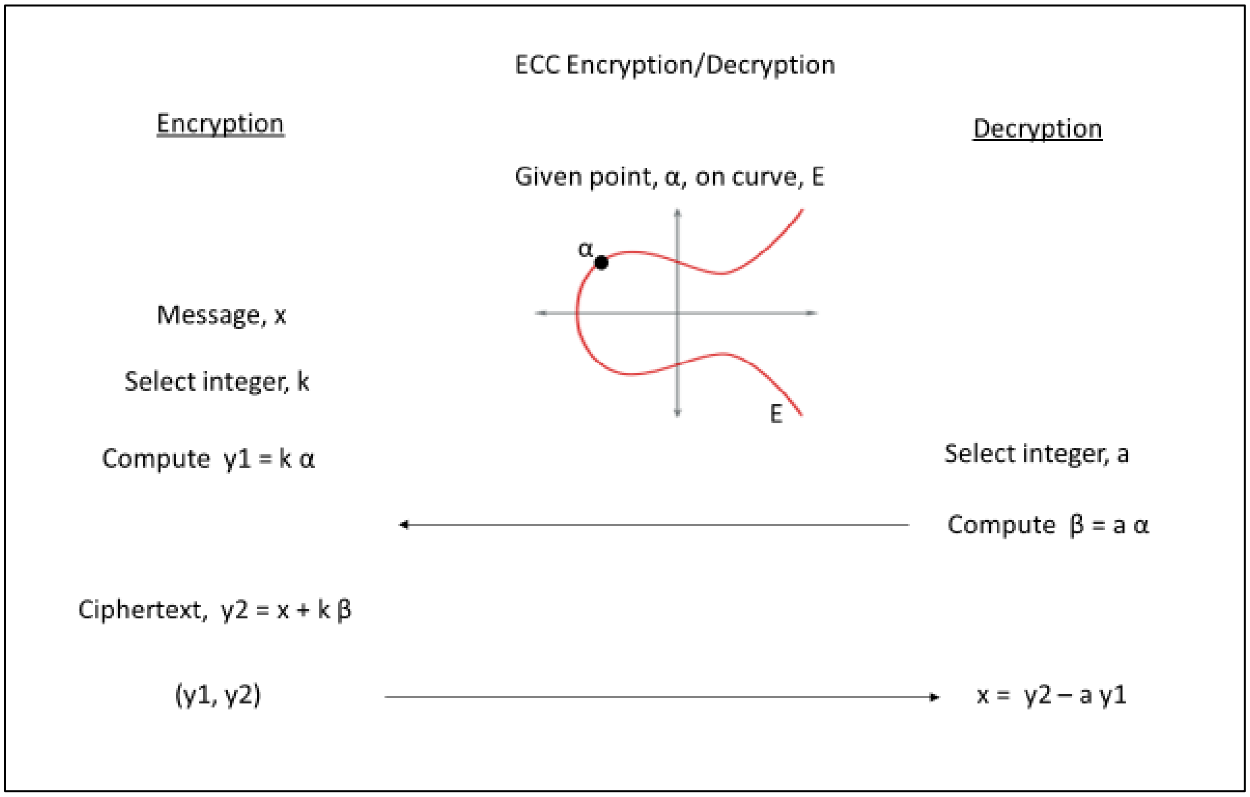


Figure 15: ECC Encryption/Decryption.

For more information on elliptic curve math, see Hans Knutson's article, *What is the math behind elliptic curve cryptography*[71].

A.1.3 Enabling Symmetric Cryptography via Asymmetric Cryptography

Symmetric cryptography schemes have the advantage of small key sizes and efficient computations when compared with typical asymmetric crypto schemes. Symmetric schemes, however, provide no method of securely sharing the required symmetric key. The common solution, shown in Figure 16, is to utilize the secure, though less efficient asymmetric algorithms with their asymmetric public/private key pairs to securely establish a shared symmetric key. That is, the symmetric key derivation material is the message to be sent asymmetrically, before proceeding with the more efficient symmetric cryptography for the bulk traffic encryption.

[71] H. Knutson, "What is the math behind elliptic curve cryptography," URL: https://hackernoon.com/what-is-the-math-behind-elliptic-curve-cryptography-f61b25253da3

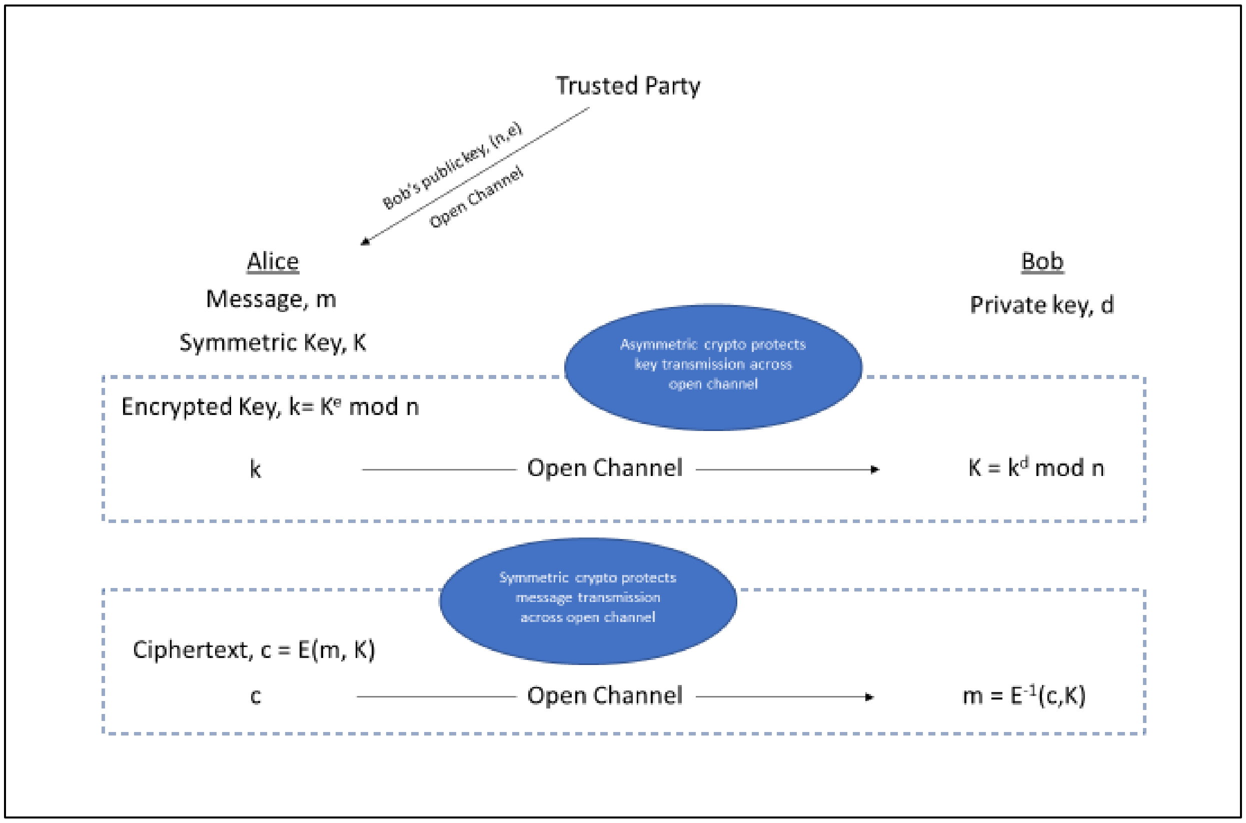


Figure 16: Asymmetric Crypto Enables Symmetric.

A.2 Digital Signing and Verification

Asymmetric cryptographic methods also enable digital signing and verification. The signature process is similar to asymmetric encryption and decryption, except that a message is signed with the signer's private (secret) key and verified with the signer's publicly known key. Theoretically, the unique digital signature of the message, once verified, proves that the message was sent by the indicated signer, and proves that the message was not modified after it was signed.

A.2.1 RSA Signatures

In a simple representation of RSA signature verification shown in Figure 17, Alice signs a message by encrypting a short hash of the message with her private key, *d*. Bob then hashes the received message, decrypts the signature with Alice's publicly known key, *e*, and compares his hash of the message with the hash recovered from the signature. As Alice is the only one with her private key, no one else can successfully create a signature that verifies against her publicly known key.

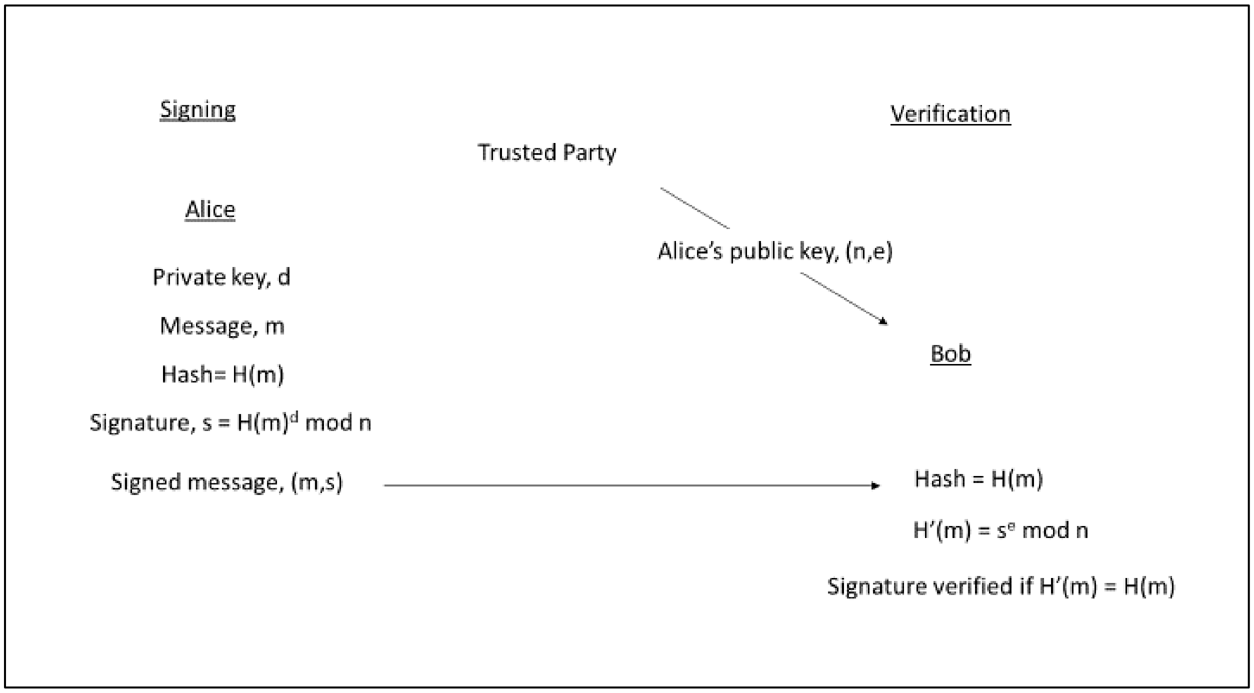


Figure 17: RSA Signature/Verification.

A.2.2 Elliptic Curve Signatures

Elliptic Curve Digital Signature Algorithm (ECDSA) is a more recent, commonly used signature/verification scheme.[72] Though the steps shown in Figure 18 appear more complex than the previously described for RSA signing and verification, the essentials are the same. The signer users her private key, d_A, to generate a public key, $Q_A = d_AG$, on the agreed upon elliptic curve, E, and a signature, *s*, equal to the encrypted digest or hash of her message. The recipient also calculates the message hash, *z*, and uses the sender's public key, Q_A, with the signature (*r, s*), to determine whether the point multiplications on the curve agreed.

[72] J. López, R. Dahab, "An Overview of Elliptic Curve Cryptography," Technical Report IC-00-10, State University of Campinas, 2000.

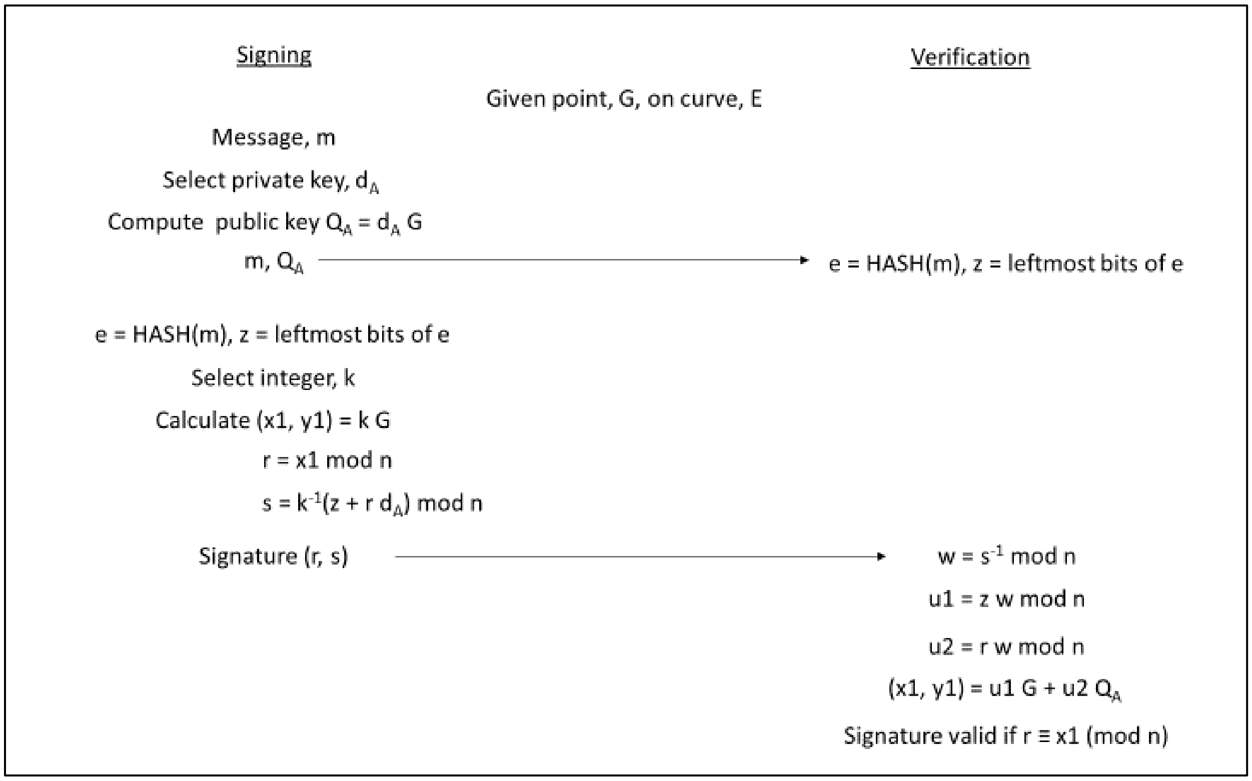

Signing
Verification
Given point, G, on curve, E
Message, m
Select private key, d_A
Compute public key $Q_A = d_A G$
m, Q_A
e = HASH(m), z = leftmost bits of e
e = HASH(m), z = leftmost bits of e
Select integer, k
Calculate (x1, y1) = k G
r = x1 mod n
$s = k^{-1}(z + r\, d_A) \bmod n$
Signature (r, s)
$w = s^{-1} \bmod n$
u1 = z w mod n
u2 = r w mod n
(x1, y1) = u1 G + u2 Q_A
Signature valid if $r \equiv x1 \pmod n$

Figure 18: Elliptic Curve Digital Signature Algorithm.

Again, note that the key usage for signing and verification is opposite that of encryption and decryption. Recall that for asymmetric encryption and decryption the sender and receiver use the public and private pieces respectively of the *receiver's* key. For signing and verification, the signer and verifier use the applicable pieces of the signer's, i.e. the *sender's,* key.

A.3 Public Key Distribution

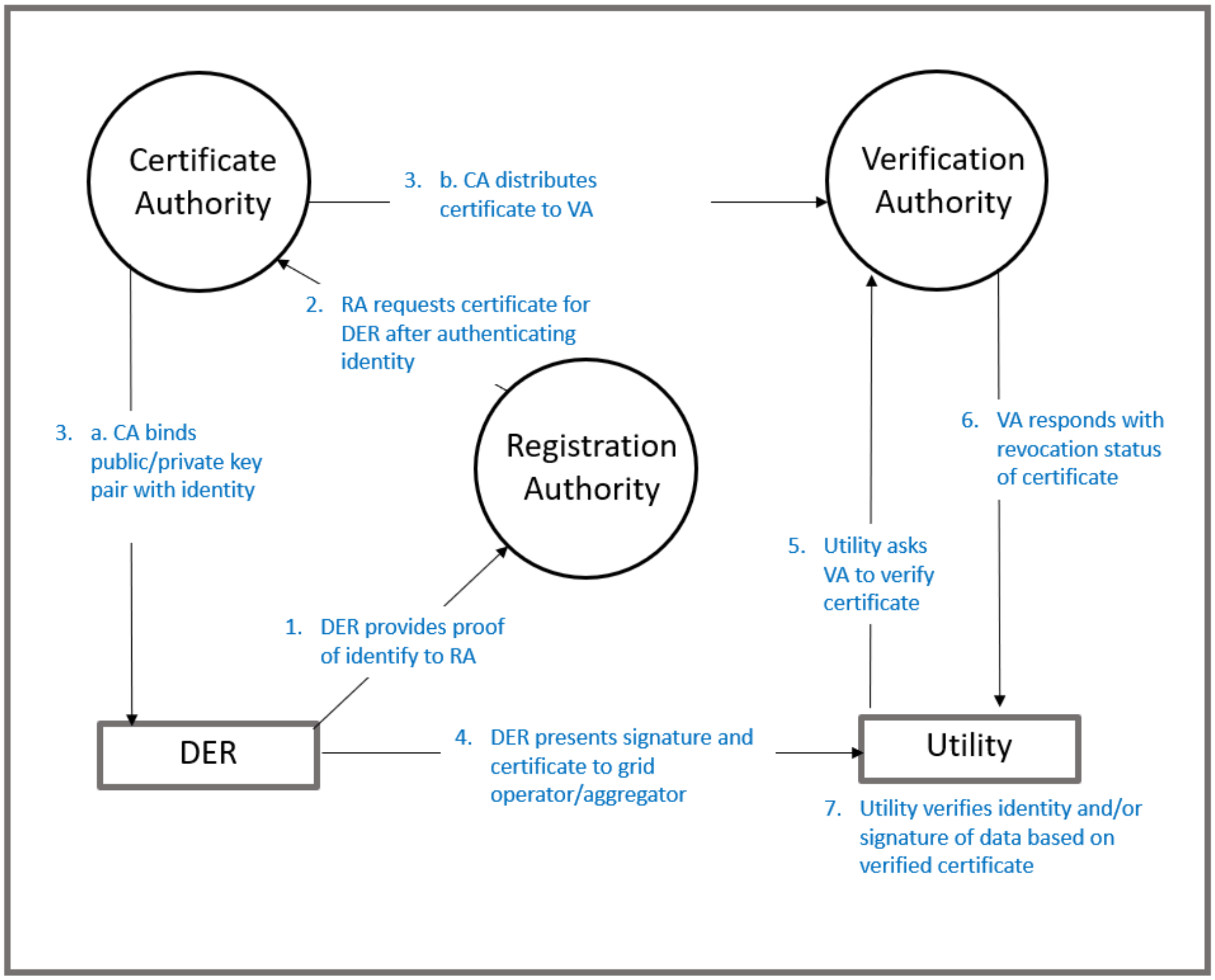
In order to have confidence that a public key belongs to a given entity prior to using that key for establishing a shared secret or verifying a digital signature, theoretically, the following Public Key Infrastructure (PKI) process is used to register, produce and verify a certificate carrying the entity's public key. The steps shown in Figure 19 for a DER certificate are as follows:

1. The DER provides proof of identity to the Registration Authority (RA)
2. The RA requests certificate for entity after authenticating identity

3a. The Certificate Authority (CA) binds public/private key pair with identity

3b. The CA distributes public portion of entity's data, i.e. certificate, to Verification Authority (VA)

4. Entity presents asymmetric-generated signature and public portion of certificate to other party (e.g. the utility)
5. The utility asks the VA to verify certificate
6. The VA responds with revocation status of certificate
7. The other party verifies entity's asymmetric-generated signature based on verified (non-revoked) certificate



Certificate Authority
3. b. CA distributes certificate to VA
Verification Authority
2. RA requests certificate for DER after authenticating identity
3. a. CA binds public/private key pair with identity
Registration Authority
6. VA responds with revocation status of certificate
5. Utility asks VA to verify certificate
1. DER provides proof of identify to RA
DER
4. DER presents signature and certificate to grid operator/aggregator
Utility
7. Utility verifies identity and/or signature of data based on verified certificate

Figure 19: Public Key Infrastructure.

Mutual authentication requires that the other party take the exact same steps 1-7 to prove its identity to the first entity. After mutual authentication, the communicating parties may establish a shared secret key via methods stated in their certificates and proceed with symmetric key encryption and decryption of their transactions.

DISTRIBUTION

1 Guohui Yuan
U.S. Department of Energy
1000 Independence Avenue SW
Washington, DC 20585

1 Kemal Celik
U.S. Department of Energy
1000 Independence Avenue SW
Washington, DC 20585

1 Dan Ton
U.S. Department of Energy
1000 Independence Avenue SW
Washington, DC 20585

1	MS0671	Jennifer Depoy	05628
1	MS0671	William Waugaman	05628
1	MS0671	Jason Stamp	05623
1	MS1033	Abraham Ellis	08812
1	MS1033	Jimmy Quiroz	08812
1	MS1033	Jay Johnson	08812
1	MS0161	Legal Technology Transfer Center	11500
1	MS0899	Technical Library (electronic copy)	

Sandia National Laboratories