European Network for Cyber Security

# Distribution Automation RTU Security Requirements

Version Jaunuary 2016

ENCS

ENCS

# Table of Contents

# 1 Introduction

This catalog describes security requirements for the procurement of distribution automation RTUs. The purpose is to support procuring secure distribution automation systems. The requirements focus on RTUs because they act as access points in the substation and control the systems directly interacting with the distribution automation processes. The RTU is the shield between the outside communication and the field systems.

Beyond device security, the catalog describes security requirements for all end-to-end secured communication between the RTU and the central systems, and for processes the vendor should implement to keep the RTU secure during its lifecycle.

These requirements have been developed by the European Network for Cyber Security (ENCS) for the grid operators Alliander (NL), EDP Distribuição (PT), Enexis (NL), E.ON Group (Europe wide), EVN (AT), and Stedin (NL). These grid operators intend to use the requirements as the basis for future tenders.

## 1.1 Scope

Unless stated otherwise the requirements hold for RTUs used in distribution automation. The requirements have been developed for RTUs that are placed in medium to low voltage transformer stations. They can be used for other devices with a similar functions, and similar security risks.

The requirements are device-specific and are set to be fulfilled by the vendor. The requirements do not address operational security at the grid operator. For selected requirements operational recommendations are given.

### 1.1.1 Interoperability Requirements

The requirements are formulated in a technology and protocol independent manner. In this way they can be applied to many different types of RTUs, and in different types of infrastructures. Grid operators will choose different communication technologies and protocols, and different software systems based on their particular situation. The requirements in this document provide security for the RTU for all possible choices.

Users of the requirements may want to complement these security requirements with interoperability requirements. Some specific technologies may be required to integrate the RTU into a larger infrastructure. Examples where interoperability requirements may be needed are:

- The communication security requirements in section 2.3. To implement these requirements, the RTU needs to use the same protocol as the software connecting to it. Often protocols such as IPsec, TLS, or OpenVPN are used. Interoperability requirements may be needed on the version and configuration of these protocols.
- The access control requirements in section 0. The requirements ask for support for a central authentication server. Different technologies are available for such servers, such as RADIUS, TACACS, LDAP, and Active Directory. If a grid operators

is already using some of these technologies, they may want to also require it for the RTU.

- The logging requirements in section 3.2. More and more RTU logs are being imported in Security Information and Event Management (SIEM) systems. The SIEM may put particular requirements on the protocol used to send the logs and the format of the logs.
- The requirements for time synchronization in the logging requirements in section 3.2. Different technologies are available to provide this feature, such as NTP and GPS. If a grid operator is already using one of these technologies, they may require the RTU to use the same technology.

## 1.2 How to Read the Requirements

Each requirement is labelled with an identifier (Req._ID) and consists of the following three items:

- **Minimum Requirement:** A *mandatory requirement* is a compulsory need that a system, device, component, or entity must perform. Statements in the requirements of this document are compulsory for the vendor.
- **Awarding Criteria:** A*warding Criteria* are weighed and scored in the evaluation, but do not lead to direct exclusion of the vendor from the tender process. The weights and scores are not defined in this document but will be set by the utility starting the tender process.
- **Recommended Assurance:** *Recommended assurance* provides guidance for quality control. The vendor can see how the implementation of the requirement will be tested in a standard testing facility. Appendix A provides brief remarks and references concerning the most common testing procedures.

Items may be left out for a particular requirement if they are not used.

After these three items, further clarification on the requirement is given. The clarification can define certain terms, give examples of what is and is not allowed by the requirements, or give a recommendation on implementing the requirement. A requirement does not have to be implemented as in the recommendation, as long as a the Vendor provides a good justification on why their implementation meets the requirement (see requirement SUR.01 in Section 5).

The requirements use standard terminology from security and distribution automation where possible. If there is a possibility for confusion about a term, it will be defined in the clarification of the first requirement where it is used, and printed in bold there. A glossary of terms is also provided at the end of the document.

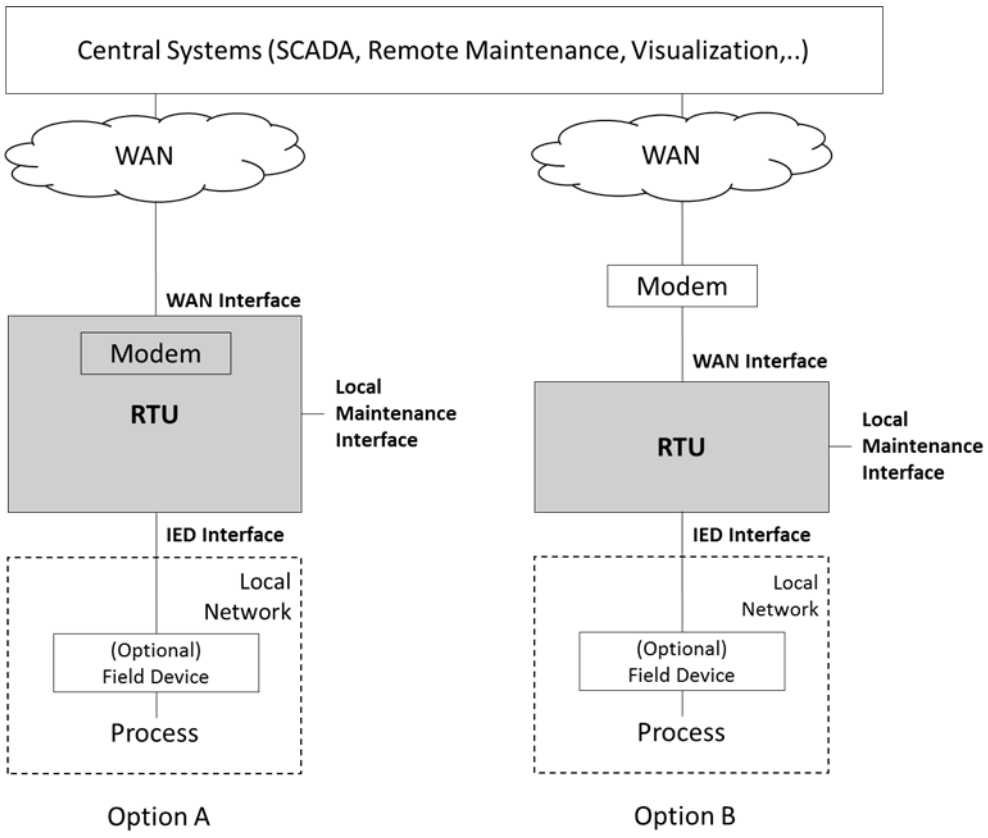
### 1.2.1 Wording

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [3]:

- *MUST This word, or the terms "REQUIRED" or "SHALL", mean that the definition is an absolute requirement of the specification.*
- *MUST NOT This phrase, or the phrase "SHALL NOT", mean that the definition is an absolute prohibition of the specification.*
- *SHOULD This word, or the adjective "RECOMMENDED", mean that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.*
- *SHOULD NOT This phrase, or the phrase "NOT RECOMMENDED" mean that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.*

### 1.2.2 Architecture

The security requirements assume the Distribution Automation reference architecture defined in [1]:




In particular, the requirements often reference the different interfaces in this architecture:

- The **WAN interface** connects the RTU to the central systems over the WAN. The WAN interface terminates any end-to-end secured communication over a WAN network between the central systems and the RTU. In Option A above, the WAN interface is directly connected to a wide-area network. In Option B, the WAN

interface connects the RTU to an external modem that facilitates the connection to the central systems over a wide-area network.
- The **Local Maintenance** interface allows service engineers to locally connect to the RTU to configure updates, make adjustments to settings, and conduct maintenance activities. Often this interface takes the form of an Ethernet of serial interface on the RTU.
- The **IED interface** connects the RTU to the field devices via the local network that is located inside the substation and connects the RTU to field devices and other devices in the substation. Typically, communication within a secondary substation uses the Modbus, IEC 61850, or IEC 60870-5-103 protocols.

Requirements also sometimes reference different functions on the RTU:

- The **firmware update** function refers to changing the firmware or any other software installed on the RTU.
- The **configuration** function refers to changing any setting on the RTU, including network settings, I/O settings, and the security configuration.
- The **sensor reading** function refers to accessing data from sensors connected to the RTU.
- The **control** function refers to sending commands to any actuators, such as switches or breakers, connected to the RTU.

The functions include access to data related to each function.

### 1.2.3 Stakeholders

The stakeholders concerned with the procurement and product lifecycle of the RTU are *Purchasers* and *Vendors*. This document uses the term *Purchaser* as replacement for utility, distribution system operator (DSO), grid operator or similar. The term *Vendor* stands for the party that sells the RTU. The document does not distinguish between a vendor and a manufacturer in case these are two separate entities. Ultimately, the Vendor is held responsible for the security features of the product, i.e., the RTU. In particular, the Vendor has to ensure that all components procured from a supplier satisfy the requirements in this document.

# 2 RTU Security Requirements

This section contains the technical requirements to keep the RTU itself secure. Care has been taken to align this requirements with common standards and best practices for security for devices used in the industrial control systems domain, such as the BDEW White Paper Requirements for Secure Control and Telecommunication Systems [4], the DHS Cyber Security Procurement Language for Control Systems [5], NERC CIP [6], the IEC 62351 series [7] and IEC 60870-5-7 [8], the IEC 62443 series (former ISA-99) [9], IEEE P1686 [10], and the WIB Process control Domain Security Requirements for Vendors [11]. A document [2] is available that shows the relation between the requirements in this document and these selected standards and best practice guides.

## 2.1 Future-Proof Design

The requirements in this section concern future-proof designs for the RTU. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SFR.**".

**SFR.01 Future-Proof Design**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL have sufficient reserves in memory and computing power to allow updates to security functions that security experts anticipate are necessary during the RTU's lifecycle. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Testing the performance of the RTU for algorithms and protocols anticipated for future use. |

In this document a **security function** refers to any function on the RTU that is needed for it to be operated securely. Security functions include access control, authentication, and encryption. All functions needed to implement the security requirements in this document shall be considered as security functions.

There are several sources of expert forecasts on what security functions are needed in the future. It is recommended that Vendors consult these sources when determining which algorithms and protocols are needed in the future.

The ENISA documents on algorithms, parameters, and key sizes [12] marks some algorithms as suitable for future use, while others are only suitable for legacy use. If legacy algorithms are used by the RTU, there should be sufficient resources to update it to an algorithm in the same category suitable for future use.

Recommendations on which key sizes provide sufficient security in the future are available from e.g. NIST [18], BIS [38], and ANSSI [38]. One way to show that sufficient computational resources are available, is to show that the RTU can support the key sizes required by these document at the end of the RTUs lifecycle.

The German Federal Office for Information Security (BSI) classifies IPsec and IKEv2 options in [15]; for each option BSI states a year until which the option is considered secure. The

label "2021+" means that the option is considered secure until the year 2021 and beyond. This gives a prediction of which IPsec options the RTU should be able to support during its lifecycle.

If for a protocol used by the RTU a newer version of the protocol specification is available or is being prepared, this version also gives information on security function the RTU may need to support in the future.

**SFR.02 Remote Firmware Updates**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL support updating all security functions through remote firmware updates. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor. |

This requirement does not forbid updates of security functions over the Local Maintenance interface. Such a requirement would be operational and is left to the grid operator to decide. SIR.03 details verification of the integrity of firmware updates.

## 2.2 Cryptographic Algorithms and Protocols

The requirements in this section concern how to choose cryptographic tools and key lengths. Requirements are grouped into different items. Each item has a unique identifier with prefix **"SPR."**.

**SPR.01 Cryptographic Algorithms and Key Lengths**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL use for security functions only cryptographic algorithms for which a description is publicly available, and which have been thoroughly reviewed by independent cryptographers.<br>2. The RTU SHALL not use for security functions a choice of cryptographic algorithms, protocols, and parameters if there are vulnerabilities known for them.<br>3. If for a security function algorithms are available in [12], the RTU SHALL use one of these algorithms.<br>4. The RTU SHALL use from [12] only those cryptographic algorithms, and parameters considered suitable for legacy or future use.<br>5. The RTU SHALL use the algorithms in [12] implemented exactly as they are described there without any modifications. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor can be used to establish that only allowed cryptographic algorithms, protocols, and parameters are used. |

- Functional security tests can be used to verify that the algorithms are implemented as described in [12]. Certifications are also available to test that the protocols have not been modified:
  - Cryptographic primitives can be certified with the NIST Cryptographic Algorithm Validation Program (CAVP) [23].
  - Compliance with the usage of TLS as in IEC 62351 can be in the form of a Protocol Implementation Conformance Statement as in IEC 60870-5-7 [8].

A **cryptographic protocol** is a protocol used for security functions, such as authentication protecting confidentiality or integrity. Cryptographic protocols are implemented using cryptographic algorithms, such as symmetric and asymmetric ciphers, and hash functions. The cryptographic algorithms again depend on certain cryptographic parameters. The most well-known example is the key size. If the key size for an algorithm is too small an algorithm becomes vulnerable to brute-force attacks. Correct choices for other cryptographic parameters, such as the initialization vector, are equally important for the secure functioning of a protocol.

Vulnerabilities are considered known if they are in a public vulnerability database, or if an advisory on them has been published. The ENISA report [12] provides a good overview of the state-of-the-art for cryptographic primitives such as block ciphers, cryptographic hash functions, stream ciphers, public-key primitives, and a key size analysis. The report is updated annually to be in accordance with technical and scientific progress. When an algorithm is marked as suitable for legacy use in this reports, it means that there are no known vulnerabilities and the algorithm is considered good for current use. When it is marked at suitable for future use, it is expected to remain secure for 10 to 50 years.

Some algorithms in [12] are not even allowed for legacy use, and are marked with an "X" in the legacy column. Such algorithms are broken and considered insecure. They must not be used on the RTU for security functions. Examples are:

- The MD5 hash algorithm: an attacker can construct two distinct files with the same MD5 hash value. In particular, it would be possible to produce a second firmware image with different content but matching hash value.
- The RC4 stream cipher: encryption can be broken due to biases in the key stream.

It is allowed to use algorithms for which vulnerabilities are known if they are not used for security functions. For instance, cyclic redundancy check (CRC) codes can be used by the RTU to detect accidental errors in the transmission of a message. They should however not be used to check against deliberate modifications by attackers (as required in SIR.02) as there are vulnerabilities known for them.

To interpret the requirement, it is important to distinguish between cryptographic protocols and communication protocols, such as TLS, IPsec or IEC 104. Communication protocols usually use several cryptographic protocols to implement their security features. Often they offer different options for each feature. For instance, the TLS protocol allows both RSA and (elliptic curve) Diffie-Hellman for key exchange, and allows for different key sizes for each protocol. If vulnerabilities are known for some of the cryptographic options allowed

by a communication protocol, it does not mean the communication protocol should not be used. Instead, only secure options should be used, and others disabled.

For several communication protocols commonly used in RTUs there are vulnerabilities known for all the cryptographic protocols used in older protocol versions. In that case the older protocol version should not be used. Examples are:

- All versions of SSL and TLS versions before 1.2 have known vulnerabilities. If the RTU uses TLS, it must user version 1.2 or greater.
- SNMP versions before version 3 have known vulnerabilities.

Communication protocols with known vulnerabilities can be used if they are encapsulated in other protocols that provide the security functions. The most common case is that vulnerable protocols are encapsulated in secure network or transport layer protocols, such as IPsec, OpenVPN, or TLS.

Many industrial protocols, such as IEC 60870-5-104 or Modbus, do not implement any security. Such protocols should therefore always be encapsulated in secure lower layer protocols. Security for the commonly used IEC 60870-5-101 [19] and IEC 60870-5-104 [20] protocols, is specified in IEC 60870-5-7 [8]. Implementations of this standard are however not always available. Also, the standard is updated at irregular intervals which can lead to conflicts with the state of the art in cryptographic security. In case such a conflict arises, the Vendor should follow the state of the art described in [12].

**SPR.02 Cryptographic Number Generation**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL use a dedicated cryptographic pseudo-random number generator, as defined in FIPS 186-2 [24], FIPS 140-2 (Annex C) [26], AIS 20 [26], or AIS 31 [27], to generate random numbers used for security functions. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Proof of the implementation could be the reports of a standardized test procedure such as the NIST Cryptographic Algorithm Validation Program (CAVP) [23].<br>• NIST SP 800-22 [39] provides a standardized test suite to look for biases found in non-cryptographic random number generator during a black-box test. |

Random values are used for security function for instance in the generation of digital signatures and cryptographic keys, or in authentication protocols.

The basic random number generators in many programming languages, such as the `rand()` function in the C programming language, do not satisfy the requirements in the mentioned standards. For Linux-based systems one can instead use `/dev/random`. The German BSI recommends in [28] to use kernel versions starting from 2.6.21.5, 3.2, 3.5, 3.6 and 3.7. It is recommended to monitor vulnerabilities in implementations and update kernels accordingly.

ENISA provides further requirements on pseudo-randomness generation in [12].

**SPR.03 Key Management**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU MUST support remote updates of all credentials and cryptographic keys.<br>2. The RTU MUST support limiting the duration of a session to a time length that is configurable by the purchaser. |
| *Awarding Criteria* | 3. The RTU SHOULD support establishing a fresh key for each communication session.<br>4. The RTU SHOULD support using different keys for different services and applications. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Functional tests can be used to establish the functionality is present on the RTU. |

Establishing a session key can only be done if the RTU and the hosts it communicates with use the same protocol. Hence, there may be interoperability requirements. For example if the RTU uses IPsec, it can be convenient if the RTU supports version 2 or newer of the IKE protocol to allow for easier negotiation. Such interoperability requirements are specific to the situation of the Purchaser, and are therefore not included in this document. The Purchaser should add them to their tender document if required.

Because the RTU supports key updates, it is possible to give each RTU individual keys. It is strongly recommended that this is done by Purchasers operating the RTU.

Pre-shared keys are considered less secure than session keys. Attacks on cryptographic algorithms often require a large amount of encrypted data. By using a session key the amount of data encrypted with one key is limited. Therefore, it is preferred that a fresh key is generated for each session. In this context, a key should be considered fresh if it was generated by a cryptographic random number generator (as defined in SPR.02) or a cryptographic key exchange algorithm (such as Diffie-Hellman key exchange), and was not used before.

Using TLS has the advantage that it allows different keys for different services and applications, so that awarding criterion 4 is met.

## 2.3 Communication Security

The requirements in this section concern communication security for the RTU. Each item has a unique identifier with prefix "**SCR**.".

**SCR.01 Confidentiality**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL protect the confidentiality of communication on the WAN interface by encrypting it using a protocol allowed by SPR.01.<br>2. The RTU SHALL store passwords together with a salt using a cryptographic hash function allowed by SPR.01. |
| *Recommended Assurance* | • This requirement is verified in a functional security test. The test should in particular ensure that the allowed cryptographic algorithms are supported and that disallowed algorithms are rejected. |

For SCADA traffic encryption could be implemented on the network layer, using for instance IPsec or OpenVPN, on the transport layer, using TLS, or on the application layer. All these solutions are allowed, as long as they meet requirement SCR.01.

The WAN interface covers both Ethernet and serial communication e.g., IEC 60870-5-101 and IEC 60870-5-104. Encryption on the Maintenance and IED interfaces is not required, as intercepting traffic on them is not possible without local access in the substation, and the value of the information that can be captured in one substation is low.

Special protection is required for passwords in point 2, because should be the only truly confidential information stored on the RTU. The requirements in this document are set up to allow for different keys for each RTU. If the Purchaser indeed uses different keys in operations, attackers will benefit little from getting the keys out of the RTU. They must already compromise the RTU to get the key, and they cannot use the keys on other RTUs.

It is still recommended to use different passwords for each RTU. Attackers that compromise the RTU may still acquire passwords by capturing them when they are sent to the RTU. Using different passwords does require support from the tools used for maintenance, and the central servers to remember the passwords. Engineers and operators cannot be expected to remember passwords for hundreds of RTUs.

**SCR.02 Message Integrity**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL verify the integrity of application layer messages received on the WAN and Local Maintenance interface using a message authentication algorithm allowed by SPR.01.<br>1. If the RTU detects that a message has been modified or if it cannot verify the integrity of the message, it SHALL reject or drop the message.<br>2. The RTU SHALL allow parties it communicates with on the WAN or Local Maintenance Interface to verify the integrity of application layer messages it sends by using a message authentication algorithm allowed by SPR.01. |

| | |
|---|---|
| *Awarding Criteria* | 3. The RTU SHOULD verify the cryptographic integrity of messages received on the IED interface.<br>4. The RTU SHALL allow parties it communicates with on the IED Interface to verify the integrity of application layer messages it sends by using a message authentication algorithm allowed by SPR.01. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Functional tests can be used to verify that the RTU supports the required functionality.<br>• Carrying out a penetration test can be used to determine if the RTU verifies message integrity under all conditions. |

Message integrity is usually verified using a message authentication code (MAC) or a block cipher in authenticated encryption mode, such as Galois Counter Mode (GCM). Algorithms for these are available in [12]. To be able to verify the integrity of an application layer message, the entire message should be given as input to the message authentication algorithm. No message fields should be left out.

The integrity of messages without application layer payload, such as acknowledgements, does not have to be protected. Headers from lower layer protocols also do not have to be protected. If these headers however include counters or information on the message's source, this information may still require integrity protection to meet requirements SCR.04 and SCR.05.

If IPsec is used to fulfil this requirement the Encapsulating Security Payload (ESP) should use one of the authenticated cipher modes (AES-GCM or AES-CCM). Alternatively, the Authentication Header (AH) should be configured using one of the allowed cryptographic algorithms (see SPR.01).

This requirement concerns cryptographic message integrity. CRC checksums do not fulfil the requirement. They are not allowed by requirement SPR.01.

A message is dropped if the RTU does not send a reply. A message is rejected if the RTU replies with an error message or NACK.

On the IED interface message integrity checks are not a minimum requirement. Many IEDs do not support it yet. To exploit the lack of integrity checks, attackers also first need to have access to the networks in the substation.

### SCR.03 Firmware Integrity

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL verify the integrity of firmware images before they are applied.<br>2. The RTU SHALL reject firmware updates if it detects the firmware has been modified, or it cannot verify the firmware's integrity. |

| | |
|---|---|
| *Recommended Assurance* | • The functional requirement can be verified by testing the implemented firmware-update functions. |

Firmware integrity is usually verified by calculating a hash value of the firmware. Hash functions are described in the ENISA document [12].

**SCR.04 Message Freshness**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL be able to detect replay attacks on the WAN and Local Maintenance interface.<br>2. If the RTU detects that a message is replayed, it MUST reject or drop the message. |
| *Awarding Criteria* | 3. The RTU SHOULD be able to detect replay attacks on the IED interfaces. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor on the mechanisms used to protect against replay attacks.<br>• Functional testing can be used to verify if the mechanisms are indeed implemented. |

To prevent replay attacks all messages should be secured by one of the following means:

- By adding a counter.
- By adding an authenticated nonce. It is essential that the nonce is authenticated using a MAC algorithm.

VPN technologies such as IPsec need to explicitly enable replay protection in combination with message authentication (SCR.02).

**SCR.05 Message Authentication**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL be able to determine that the sender of a configuration change or a firmware update has a certain role.<br>2. The RTU SHALL be able to determine that the source of a sensor reading request or control command is a specific host in the DA system. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor on the mechanisms used for message authentication.<br>• Functional testing can be used to verify if the mechanisms are indeed implemented.<br>• Penetration tests can be used to ascertain that attackers cannot bypass the authentication mechanisms. |

Authentication concerns being able to determine the source of a message. There are different levels of detail possible here. The source can be a host in the network, but also a user of the distribution automation system (in the sense of RBAC, see section 3.1). The first minimum requirement only refers to roles, because SAR.02 does not require the RTU to distinguish between different users, only between roles.

The reason that the requirement for sensor readings or control commands is less strict, is that user authentication is currently not supported by SCADA protocols, such as IEC 60870-5-104, because they do not support multiple users. This shortcoming can be mitigated if the SCADA traces commands to individual users. The RTU is required to support such a mechanism by allowing to trace the origin of a message to a specific host, such as a SCADA front-end. This can be done for instance by using a VPN. When multiple users would be allowed by SCADA protocols in the future, the requirement should be updated to include authentication (and possibly non-repudiation) for sensor reading requests, and especially control commands.

This requirement is usually met by using message authentication code (MAC) or a block cipher in authenticated encryption mode, as for requirement SCR.03. These algorithms allow the RTU to check that a message is sent by someone who has access to the key used for them. Requirement SCR.05 puts restrictions on who can have the key. If pre-shared keys are used, different keys must be used for different roles or hosts. If session keys are used, the protocol used to agree on the session key should check whether the user making the request has a certain role or is in on a certain host.

**SCR.06 Non-Repudiation**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL support non-repudiation for firmware: when it install firmware, it SHALL be able to prove that the firmware came from the Vendor. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor on the mechanisms used for non-repudiation.<br>• Functional testing can be used to verify if the mechanisms are indeed implemented.<br>• Penetration tests can be used to ascertain that attackers cannot bypass the non-repudiation mechanisms. |

Non-repudiation means that a sender of the firmware should not be able to deny that he sent it. It is normally implemented using digital signatures. A hash value of the firmware is calculated, and signed using public-key cryptography. The private key is kept by the Vendor (see SDR.03). The public key for the validation of the signature can be installed on the RTU during the manufacturing process. SDR.08 defines Production Security & Credential Provisioning. It is not needed to keep the public key secret. Measures should be taken to make sure the correct key is installed however.

It is not necessary that the Purchaser establishes a Public Key Infrastructure (PKI) at the Central System for this purpose. The Vendor has to store the private firmware signing key.

## 2.4 System Hardening

The requirements in this section concern hardening of the RTU. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SHR**.".

### SHR.01 Device Hardening

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL have all unneeded services and applications removed, or disabled if removal is not possible.<br>2. The RTU SHALL not use services or applications for security functions if there are vulnerabilities known for them.<br>3. The RTU SHALL use only communication protocols that are needed to meet the functional requirements. |
| *Recommended Assurance* | • Vulnerability scanners can automatically check devices for known vulnerabilities.<br>• Carrying out a penetration test can provide further assurance that this requirement is adequately implemented.<br>• If high-impact functions are disabled in the RTUs code, the Purchaser can request a code review from the Vendor. |

Examples of unused services and application that should be removed or disabled are:

- Testing and debugging applications used for initialization or testing during the production process.
- Webservers used as graphical user interfaces (GUIs) or for maintenance purposes if maintenance is normally done through a specialized application.
- FTP servers used during installation.
- Drivers for hardware that is not in the RTU.
- A telnet service when SSH is also available.
- NTP or DNS servers if these are not used by other devices in the substation.

Vulnerabilities are considered known if they are in a public vulnerability database, or if an advisory on them has been published.

Webservers/GUIs are often prone to code injection, buffer overflows and other vulnerabilities, they pose a high risk when directly accessible from a remote connection. The OWASP list [30] provides a good overview of known web vulnerabilities.

### SHR.02 Interface Minimization

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL have any unneeded interfaces and ports removed, or disabled if removal is not possible. In particular, all hardware interfaces that are used for debugging MUST be completely removed after production. |

| | |
|---|---|
| | 2. The RTU SHALL not allow direct remote access to modules with IED functions. |
| *Recommended Assurance* | • Carrying out a penetration test can provide assurance that this design requirement is adequately implemented. |

Redundant and unused ports could include

- USB ports
- Ethernet ports
- Serial ports

Microcontrollers and processors are often equipped with hardware interfaces, such as JTAG, and Serial Wire Debug. These interfaces allow programming or debugging of the respective components and are required for example in the course of production. They should be disabled in operational systems.

**SHR.03 Account Hardening**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU MUST NOT contain active default, guest and anonymous accounts.<br>2. The RTU MUST not allow remote access to root accounts on the RTU.<br>3. The RTU SHALL have Vendor-owned accounts removed where feasible. |
| *Awarding Criteria* | 4. The RTU SHOULD support enforcing a password policy that only allows passwords of sufficient complexity. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Carrying out a penetration test can provide further assurance that this design requirement is adequately implemented. |

**SHR.04 Security-enhancing features**

| | |
|---|---|
| *Awarding Criteria* | 1. The RTU SHOULD deploy security-enhancing features of the underlying platform, implementation language and tool chain when it enhances the RTUs security. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor on which security enhancing features are used.<br>• Functional tests can be used to verify that features are indeed used. |

Examples of security-enhancing features are:

- Compiler options that enhance security, such as adding checks to buffer overruns to the code.

- Secured boot process where the boot loader verifies the integrity of the firmware at startup.
- Use of a secure element such as a Hardware Security Module (HSM) and Trusted Platform Modules (TPM).
- Encryption of non-volatile memory.
- Activation of read-out protection enabling functions of a microcontroller.
- Whitelisting of programs and services to prevent that malware is executed on the system.
- The use of processor features that enhance security, such as ARM TrustZones

Using these features is not needed to meet the security requirements in this document. They can however add an extra layer of defense.

## 2.5 Resilience

The requirements in this section concern resilience of the RTU and the communication sent and received by the RTU. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SRR**.".

**SRR.01 Message Validity Verification**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL verify the validity of all messages it receives.<br>2. The RTU SHALL reject or drop messages that are invalid or for which the validity cannot be verified. |
| *Recommended Assurance* | • It is recommended to carry out fuzzing tests on all interfaces.<br>• The Vendor should provide a detailed documentation of all security tests. |

A message is considered **valid** if it meets all protocol specifications, it makes sense for the RTU's configuration, and it meets all requirements the RTU has on data sizes. Examples of validity checks include checks of syntax, data format, and value ranges. The RTU should also check if registers or data objects reference by a message exists, and if the data fits into internal buffers allocated for it.

The requirement is valid for all network protocol layers, including the wireless protocols, TCP/IP stack, and application layer protocols.

**SRR.02 Fail-Secure Operation**

| | |
|---|---|
| *Minimum Requirements* | 3. The RTU SHALL be fail-secure, i.e., it SHALL be designed to fail in a manner that limits any security compromise of its own operation and security compromise of other devices.<br>4. The RTU SHALL not leak confidential information, such as keys or credentials, on any interface during a failure. |

| | |
|---|---|
| | 5. The RTU SHALL protect the integrity of security critical data during failures.<br>6. The RTU SHALL not allow access controls to be bypassed remotely during failures.<br>7. The RTU SHALL restore availability after software failures as soon as possible. |
| *Recommended Assurance* | • Analysis of the design documentation provided by the Vendor.<br>• Carrying out a penetration test can provide further assurance of the design robustness. |

Point 7 can be addressed by implementing a watchdog functionality that allows the device to maintain a secured operational state in case of a failure.

Examples for relevant failures are:

- Integrity errors, e.g. of configurations or log files;
- Failures during execution of cryptographic functions;
- Failures during validation of login credentials;
- Failures when entering data (wrong data format, wrong data length, invalid commands etc.).

# 3 Support for Secure Operation

The requirements in this section concern access control and logging of security events, two services needed to securely operate the RTU.

## 3.1 Access Control

The requirements in this section concern access control for the RTU. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SAR**.".

**SAR.01 Role-Based Access Control (RBAC)**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL allow to set access privileges for configuration and firmware update functions per role.<br>2. The RTU SHALL only grant access to configuration and firmware update functions if a user's role has the right privileges.<br>3. The RTU SHALL allow new roles to be defined for future applications.<br>4. The RTU SHALL allow to assign to each role individual security credentials and keys.<br>5. The RTU SHALL allow to set access privileges to sensor reading and control functions per host.<br>6. The RTU SHALL only grant access requests to sensor reading and control functions if the host has the right privileges. |
| *Awarding Criteria* | 7. The RTU SHOULD support central user authentication and authorization through a centralized server. |
| *Recommended Assurance* | • This requirement is verified in a functional security test. The test should in particular ensure that each role has only the defined and necessary privileges.<br>• Penetration testing can be used to make sure that the access controls cannot be circumvented by for instance privilege escalation. |

The requirements assumes that RBAC is implemented in the distribution automation that the RTU is part of. Users are individual employees at the Purchaser. They are given certain roles, such as SCADA operator or maintenance engineer, which determine what they are allowed to do on the RTU. The requirement is that the RTU enforces the privileges of each roles.

The requirements is not that the RTU implements the full RBAC for the DA system. It is not required to have user accounts for individual employees. Many RTUs could technically support this, for instance through creating (UNIX) user accounts for each employee. The accounts become difficult to manage however with large numbers of RTUs.

Instead, the preferred solution is the use of a by centralized authentication server using technologies such as LDAP, Active Directory, RADIUS, or TACACS. The awarding criterion ask that the RTU supports integration into these systems. The minimum requirements are set up in such a way that, even if the centralized server is down, the RTU still can distinguish different roles and provide a good level of security.

Grid operators that do not use a centralized authentication server, can still have some form of RBAC control by having account on the RTU for each role, and giving employees the credentials for the roles they have. This does require more management, and can become labor intensive if users change roles. But in this way they do benefit from the security provided by this requirement.

It is not required that the RTU distinguishes between different roles for sensor reading and control functions. It would be preferable if it did. But the protocols used to provide access to it, such as IEC 60870-5-104, do not support RBAC yet. Instead, the requirement is set up in such a way that it is possible to restrict access to sensor reading and control functions to specific hosts, such as the SCADA server. The SCADA server can then implement RBAC on the server side.

**SAR.02 User Authentication**

| | |
|---|---|
| *Awarding Criteria* | 1. The RTU SHOULD authenticate the communication parties on the WAN interface using a challenge-response protocol based on either message authentication codes or public-key certificates.<br>2. The RTU SHOULD terminate the connection if the user authentication fails.<br>3. The RTU SHOULD authenticate the communication parties on the Local Maintenance interface.<br>4. The RTU SHOULD support blocking authentication requests, either temporarily or permanently, from an account after a number of failed login attempts. The number of failed login attempts and the time the account is blocked SHOULD be configurable. |
| *Recommended Assurance* | • The implementation of user identification can be verified in a functional security test.<br>• Carrying out a penetration test can provide further assurance that this design requirement is adequately implemented. |

A service application on the service laptop or central system is considered a communication party. Any challenge data needs to be using cryptographic randomness as in SPR.02.

## 3.2 Logging

The requirements in this section concern logging of events. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SLR**.".

**SLR.01 Logging Security Events**

| | |
|---|---|
| *Minimum Requirements* | 1. The RTU SHALL log security events in a locally stored log.<br>2. The RTU SHALL take measures to prevent that attackers can modify, delete or overwrite the security log to hide their traces.<br>3. The RTU SHALL support automatically sending log events to a central logging server or SIEM.<br>4. The RTU SHALL support synchronization with a centrally maintained time. |
| *Awarding Criteria* | 5. The RTU should allow remote monitoring of information about the device status such as processor and memory usage.<br>6. The RTU should store for each security event at least the interface, the event type, a time stamp, and the user, role, or process causing the event. |
| *Recommended Assurance* | • The implementation of logging mechanisms can be verified in a functional security test.<br>• Carrying out a penetration test can provide further assurance that attackers cannot bypass detection mechanisms or modify the security log. |

In the requirements below **security events** are any events relevant to the secure operation of the RTU. Security events include at least the following:

- User Activities:
  - Successful logins
  - Failed login attempts
  - Changes of security credentials
  - Unauthorized file access
- Possible signs of attacks:
  - Resource exhaustion (DoS)
  - Messages whose integrity could not be verified
  - Invalid messages
  - Attempted replay attacks
  - Alarms on physical manipulations
- Updates or changes:
  - Firmware Updates or patches
  - Configuration Changes

Common methods to export security events to a central logging server are syslog and SNMP. Syslog allows integration with many different SIEM solutions.

Time synchronization is required to allow logs events from different devices to be correlated. Different technologies are available for time synchronization, such as NTP and GPS.

# 4 Product Lifecycle and Governance

The requirements in this section concern the processes used for developing, manufacturing, and provisioning of the RTU in a secure way. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SDR**.".

There will be no recommendation regarding quality assurance for the requirements in this section. It is recommended that the Purchaser asks for documentation to verify the implementation of the requirements.

All requirements hold for the complete contractually agreed lifecycle of the RTU. All requirements apply to the Vendor and suppliers. This includes in particular Third-Party Suppliers.

### SDR.01 Information Security Management System

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL implement an information security management system (ISMS) the scope of which includes at least all systems used to develop, test, manufacture and provision the RTUs and any software and hardware tools needed for the maintenance of the RTU. |
| *Awarding Criteria* | 2. The Vendor SHOULD have regular audits of the ISMS performed by an accredited external auditor.<br>3. The Vendors SHOULD provide a proof of the audit to the Purchaser on request.<br>4. The Vendor SHOULD obtain an ISO 27001 certification for the ISMS.<br>5. The Vendor SHOULD make a proof of the certificate available on request.<br>6. The Vendors SHOULD share their security policies with the Purchaser. |

Quality assurance certification schemes such as the ISO 9001 are not sufficient to meet this requirement.

### SDR.02 Configuration Management System

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL employ a configuration management system for the administration of (changes of) hardware configurations and source code of devices.<br>2. The Vendor SHALL ensure that the configuration management system stores for each change an explanation, the author, the parts changed, and the time at which it was made. |

| | |
|---|---|
| *Awarding Criteria* | 3. The Vendor SHOULD allow the purchaser to audit the configuration management system. |

**SDR.03 Secured Versioning**

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL ensure that all released versions of hardware and firmware of the RTU are uniquely identifiable.<br>2. The Vendor SHALL provide to the Purchaser a cryptographic hash value for each firmware version.<br>3. The Vendor SHALL be able to reproduce released versions within the contractually agreed product lifecycle, with traceability provided by the hash value(s) as identifier(s).<br>4. The Vendor SHALL version exchangeable hardware modules separately.<br>5. The Vendor SHALL digitally sign each firmware update supplied to the Purchaser.<br>6. The Vendor SHALL protect the firmware signing keys as highly confidential data.<br>7. The Vendor SHALL report it to the Purchaser if a firmware signing key is compromised. |

SPR.01 gives references for allowed cryptographic hash functions, and digital signing algorithms.

The ISMS required by SDR.01 is normally used to determine the measures needed to protect the firmware signing key. Point 6 of this requirement means that a compromise of the confidentiality of the key should be treated as a high impact event in the ISMS.

**SDR.04 Vulnerability Handling Process**

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL have an established and documented process to handle vulnerabilities.<br>2. The Vendor SHALL monitor information sources on vulnerabilities to determine if it has been affected.<br>3. The Vendor SHALL address vulnerabilities found by the Vendor itself, the Purchaser or system integrator, or external security researchers.<br>4. The Vendor SHALL disclose to the Purchaser all known vulnerabilities on the RTU as soon as possible.<br>5. The Vendor SHALL communicate vulnerabilities to the Purchaser in a secure manner. |

| | |
|---|---|
| | 6. The Vendor SHALL issue a recommendation on how to mitigate a vulnerability as soon as possible.<br>7. The Vendor SHALL evaluate the criticality of a vulnerability using established standards (such as CVSS [36]).<br>8. The Vendor SHALL prioritize fixing vulnerabilities based on the potential impact to the Purchaser. |

Standards are available to objectively assess the impact of vulnerabilities, such as CVSS [36]. These can be used as an aid to prioritize fixing vulnerabilities. It is however recommended that the Vendor also takes into account the specific design of the RTU, and how it is used by the Purchaser, when assessing the potential impact.

**SDR.05 Security Updates and Patching**

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL provide security updates or patches for the RTU to fix high impact vulnerabilities found during the RTU's lifecycle.<br>2. The Vendor SHALL test all security updates and patches prior to deployment. |
| *Awarding Criteria* | 3. The Vendor SHOULD provide documentation that all security patches were tested and validated prior to deployment.<br>4. The Vendor SHOULD provide tools enabling batch updating of RTUs.<br>5. The Vendor SHOULD release a patch or firmware update for a vulnerability no more than three months after it was reported to the Vendor. |

The Vendor is allowed to leave vulnerabilities with a low impact unpatched. Of course it is not recommended to do so. Low impact vulnerabilities should always be disclosed to the Purchaser by requirement SDR.04.

**SDR.06 Security Training and Awareness**

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL provide security training for the personnel.<br>2. The Vendor SHALL be able to document that the necessary knowledge to securely develop and securely produce products is in place.<br>3. The Vendor SHALL name a technical expert responsible for security-related matters who acts as contact person for the Purchaser.<br>4. The Vendor SHALL conduct a risk analysis of the firmware design and the corresponding system architecture. |

| | |
|---|---|
| *Awarding Criteria* | 5. The Vendor SHOULD provide documented professional experience in the area of IT security or a security certification, e.g., CISSP or CISM. |

**SDR.07 Production Security and Credential Provisioning**

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL ensure secure provisioning of cryptographic keys, passwords and initial security credentials during the manufacturing process.<br>1. The Vendor SHALL ensure a secure production area to ensure the secure initial provisioning of credentials and cryptographic keys to the device.<br>2. The Vendor SHALL establish a secure hand-over process of the provisioned information to the central systems of the Purchaser. |

Initial security credentials include passwords.

# 5 Assurance

The requirements in this section concerns measures the Vendor should take to make sure the RTU will work securely. Requirements are grouped into different items. Each item has a unique identifier with prefix "**SUR**.".

### SUR.01 Design Evidence

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL document all interfaces of the RTU, including the protocols and services used on each interface.<br>2. The Vendor SHALL provide design evidence that sufficient reserves are available to update security functionality to meet requirement SFR.01.<br>3. The Vendor SHALL provide design evidence that only cryptographic algorithms, protocols, and parameters allowed by SPR.01 are used for security functions, including a description of which algorithms, protocols, and parameters are used for which functions.<br>4. The Vendor SHALL provide design evidence that cryptographic random number generation is implemented according to requirement SPR.02, including a description of which random number generator is used.<br>5. The Vendor SHALL provide design evidence that message integrity is protected as required in for SCR.02.<br>6. The Vendor SHALL provide design evidence that firmware integrity is protected as required in SCR.03, including a step-by-step description of the firmware update process.<br>7. The Vendor SHALL provide design evidence that the RTU has protection against replay attacks as required in SCR.04.<br>8. The Vendor SHALL provide design evidence that the RTU has implemented message authentication as required in SCR.05.<br>9. The Vendor SHALL provide design evidence that the RTU has implemented non-repudiation for firmware as required in SCR.06.<br>10. The Vendor SHALL provide design evidence that unused interfaces are disabled or removed to meet requirement SHR.02.<br>11. If interfaces or services or disabled and not removed, the Vendor SHALL provide information on how they have been disabled.<br>12. The Vendor SHALL provide design evidence that unused accounts have been removed to meet requirements SHR.03, including a list of all accounts enabled on the RTU on delivery. |

| | |
|---|---|
| | 13. If security-enhancing features as described in requirements SHR.04 are used, the SHALL provide design evidence on how they are used.<br>14. The Vendor SHALL provide design evidence on how the RTU has been made fail-secure to meet requirement SRR.02, including a list of all relevant failure types and their countermeasures.<br>15. The Vendor SHALL provide design evidence that RBAC is implemented as required in SAR.01.<br>16. The Vendor SHALL provide design evidence that security logging is implemented as required in SLR.01.<br>17. The Vendor SHALL provide design evidence at a level of detail that makes it easy to verify that the security requirements are implemented, and to test that they are implemented on the RTU as described.<br>18. The Vendor SHALL allow verification of the design evidence by an independent third party selected by the Purchaser. |

This requirement stresses that the Vendor provides the Purchaser design evidence. **Design evidence** consists of documents produced during the design and development processes that explain how the security requirements have been implemented on the RTU. The requirements in this document are formulated in a technology independent manner. The Vendor has different options to implement them. To allow the Purchaser to verify that the requirements are implemented correctly, it is important that they understand which option was chosen.

If design evidence is sensitive from a security or competitive viewpoint, the Vendor can supply it under an NDA, as long as the NDA allows for verification of the design evidence by the Purchaser or an independent third party.

### SUR.02 Security Testing

| | |
|---|---|
| *Minimum Requirements* | 1. The Vendor SHALL perform tests to verify that all the security requirements in this document have been implemented correctly.<br>2. These Vendor SHALL test the complete functional scope of the RTU, including the communication chain between the RTU and all connected field devices and the central systems.<br>3. The Vendor SHALL test both regularly used as well as rarely used functionalities of the RTU.<br>4. The Vendor SHALL document the concepts and details of the security tests in a comprehensible way.<br>5. The Vendor SHALL check the Implementation of Web services against the OWASP list of web vulnerabilities [30]. |

| | |
|---|---|
| | 6. The Vendor SHALL use vulnerability scanners to test each released firmware version on known vulnerabilities.<br>7. The Vendor SHALL allow the Purchaser to contract an independent test lab to perform a penetration test on the RTU. |
| *Awarding Criteria* | 8. The Vendor SHOULD conduct robustness tests, such as fuzzing or flooding, on all protocols used by the device both on the application layer and on lower protocol layers.<br>9. The Vendor SHOULD conduct design and code reviews and provide the results to the Purchaser. |

Examples of security tests to verify the requirements are given for each requirement under quality assurance.

**SUR.03 Secure Coding Practices**

| | |
|---|---|
| *Awarding Criteria* | 1. The Vendor SHOULD establish and enforce secure coding practices for the development of the RTU following best practices.<br>2. The Vendor SHOULD establish an internal code review process that takes security into account.<br>3. The Vendor SHOULD use automated code analysis tools to find security vulnerabilities. |

Examples of secure coding practices are the SEI CERT coding standards [40], available for different languages, and the MISRA C software development guidelines for embedded systems. [41]

# 6 Glossary

This glossary serves as inventory of technical terms and abbreviations used in the document. For detailed background information on cryptographic primitives or testing procedures we refer to the referenced literature.

| | |
|---|---|
| AES | Advanced Encryption Standard. Original name for this block cipher was Rijndael named after its designers Vincent Rijmen and Joan Daemen. |
| Application layer | OSI-Layer 5-7. |
| Authentication | When speaking about authentication one should distinguish between user authentication (e.g., sender/receiver) and message authentication. |
| Block cipher | Cryptographic primitive to encrypt/decrypt messages of fixed block length. Example: AES encrypts blocks of 128 bits (16 bytes) at a time. |
| Block cipher Mode of Operation | A mode of operation specifies how the message blocks are processed by the block cipher. Using a block cipher in CBC or CTR mode provides encryption only whereas using a block cipher in CCM or GCM mode encrypts the plaintext and produces a message authentication tag for the ciphertext. |
| Certificate | A digital certificate authenticates a public key or entity. See also Public-Key Infrastructure. |
| Confidentiality | Only authorized entities may access confidential data. To protect data from unauthorized access it can be encrypted. Then only entities with access to the secret keys can access the data after decrypting it. |
| Cryptographic hash function | Cryptographic hash functions should behave as one-way functions. They must be preimage resistant, 2nd preimage resistant, and collision-resistant. Changes in the input must produce explicitly different results in the output. Example: SHA-256. See also ENISA [12]. |
| Cryptographic protocol | A protocol used for security functions, such as authentication protecting confidentiality or integrity. |
| Cryptography | The ENISA Algorithms, Key Sizes and Parameters Report [12] provides an overview of the current state of the art. |
| DA | Distribution automation. |

| | |
|---|---|
| Data Integrity | See Integrity and Message authentication. |
| Design Evidence | Documents produced during the design and development processes that explain how the security requirements have been implemented on the RTU. |
| Digital Signature | Authenticates the sender. In practice digital signatures are implemented using elliptic curves (EC). See standards such as [14][18] and [25][31] for the implementation of the Elliptic Curve Digital Signature Algorithm (ECDSA). |
| EC | Elliptic Curve. See also ENISA [12]. |
| ECDSA | Elliptic Curve Digital Signature Algorithm. |
| Encryption | Using a cryptographic scheme the message is mapped to a random-looking undecipherable string (ciphertext). Decryption reverses the encryption process and can only be performed with the corresponding decryption key. This decryption key is either the same as the encryption key (symmetric cryptography) or the private key in a public-key cryptosystem. The confidentiality of the message can be guaranteed only while the keys are kept secret. |
| ENISA | European Union Agency for Network and Information Security. |
| EPRI | Electric Power Research Institute. |
| Fuzzing Test | A fuzzing test provides quality control of software used for secure network communication. A fuzzing test generates a high volume of mostly random data including malformated messages and observes the reaction of the device/system under test. More information on fuzzing is provided in [26][32]. |
| GPRS | General Packet Radio Service. |
| GPS | Global Positioning System. |
| Hash function | Function that maps a message to a bit string of fixed length (hash value). See also cryptographic hash function. |
| Hash value | Output of a (cryptographic) hash function. The length is fixed in the specs of the hash function. |
| ICS | Industrial Control System. |
| IED | Intelligent Electronic Device. |
| IED interface | See Deliverable D1.1 [1] on the reference architecture. |

| | |
|---|---|
| IETF | Internet Engineering Task Force. |
| Integrity | Data cannot be altered without authorization. See also message authentication. |
| ISO 27001 | ISO standard for information security. Current version at the time of writing: ISO27001:2013. |
| Key material | The term 'key material' includes all cryptographic keys. Examples: master key, symmetric session keys, private and public keys (public-key cryptography). |
| LAN | Local Area Network. |
| LDAP | Lightweight Directory Access Protocol. |
| Local Maintenance interface | See Deliverable D1.1 [1] on the reference architecture. |
| MAC | Message authentication code. Provides data integrity. Examples: CMAC, GMAC. See also ENISA [12]. |
| Message authentication | Messages should be protected against unauthorized modifications. The message should always be sent together with an authentication tag providing its authenticity. Such an authentication tag can be the second output of an authenticated cipher such as AES-CCM or AES-GCM or a message authentication code. |
| Message Validity | A message is considered valid if it meets all protocol specifications, it makes sense for the RTU's configuration, and it meets all requirements the RTU has on data sizes. |
| NESCOR | National Electric Sector Cybersecurity Organization Resource. Program issued by the US organization EPRI. See [27][33]. |
| NIST | National Institute of Standards and Technology. |
| Nonce | A nonce is a unique randomly generated string which can be used exactly once. Attachment of a nonce helps to prevent replay attacks. |
| NTP | Network Time Protocol. |
| OSI | Open Systems Interconnection. Reference model for network communications. |
| Password authentication | The user proves his/her identity using a password or PIN. |

| | |
|---|---|
| Penetration test | For a guideline refer to the EPRI program NESCOR, specifically the "AMI Penetration Test Plan". |
| PLC | Programmable Logic Controller. |
| Product lifecycle | The product lifecycle spans all stages of a product: starting from the design through the development and production to delivery and decommissioning.<br>The Purchaser and Vendor should agree on the length of the product lifecycle. |
| Public-key cryptography | Cryptographic scheme where a public key is published and henceforth can be used for encryption of messages or verification of digital signatures. Each public key has a counterpart, the corresponding private key. This key must be kept secret and is used for decryption or digital signing of messages. Public-key primitives have a high computational complexity for encryption and therefore are mostly used as part of a hybrid encryption scheme where the public key is used to communicate a common symmetric session key under which all further communication is encrypted.<br>Certificates administered by a public-key infrastructure are used to establish the authenticity of the public key. See also ENISA [12].<br>The most popular public-key encryption scheme is RSA. Digital signatures can be generated most efficiently with elliptic-curve based (EC) mechanisms. |
| Public-key infrastructure | System to generate, administer, and revoke certificates. |
| Replay attack | The attacker observes and captures data during a session with the intention of resending it later and thus impersonating one communication partner. |
| RFC | Requests for Comments. Published by the IETF. |
| Robustness test | A robustness test provides quality control by checking the design stability/robustness of the system. The tests check in particular the fault tolerance of the system. |
| RSA | Public-key cryptosystem named after its inventors Rivest, Shamir, and Adleman. |
| RTU | Remote Terminal Unit. |
| SCADA | Supervisory Control and Data Acquisition. |

| | |
|---|---|
| Security Event | Any event relevant to the secure operation of the RTU. |
| Security Function | Any function on the RTU that is needed for it to be operated securely, including access control, authentication, and encryption. |
| Session key | Symmetric key with a limited lifetime. |
| Symmetric cryptography | Sender and receiver hold the same key. Examples for symmetric primitives are block ciphers or MACs. |
| User Authentication | Verification of the identity of the communication partners (e.g., user on the RTU). Moreover, verification that the communication partners are still alive throughout a session. See also password authentication and user authentication. |
| WAN | Wide Area Network. |
| WAN Interface | Remote connection to Central System. See Deliverable D1.1 [1] for the Reference Architecture. |

# 7 References


[1] European Network for Cyber Security. Reference Architecture for Secure Distribution Automation. Deliverable D1.1 in the DA Member Project. Version 1.2, 2015.

[2] European Network for Cyber Security. Mapping of RTU Security Requirements. Deliverable D4.1 in the DA Member Project. Version 1.3, 2015.

[3] Internet Engineering Task Force. RFC 2119: Key words for use in RFCs to Indicate Requirement Levels, 1997. http://www.ietf.org/rfc/rfc2119.txt

[4] BDEW Bundesverband der Energie- und Wasserwirtschaft e.V., Anforderungen an Sichere Steuerungs und Telekommunikationssysteme (Requirements for Secure Control and Telecommunication Systems), v.01, 2008 (English and German).

[5] Department of Homeland Security (DHS). Cyber Security Procurement Language for Control Systems. September 2009.

[6] North American Electric Reliability Corporation (NERC) Critical Infrastructure Protection (CIP) Standards. http://www.nerc.com/pa/Stand/Pages/CIPStandards.aspx (last accessed on 17 January 2016)

[7] IEC 62351. Power systems management and associated information exchange – Data and communications security. Parts 1-8.

[8] IEC Telecontrol equipment and systems - Part 5-7: Transmission protocols - Security extensions to IEC 60870-5-101 and IEC 60870-5-104 protocols (applying IEC 62351).

[9] IEC 62443 and ISA99, Industrial Automation and Control Systems Security Standards.

[10] IEEE 1686 - Standard for Intelligent Electronic Devices (IEDs) Cyber Security Capabilities.

[11] Wurldtech Achilles Practices Certification. Based on International Instrument Users Association (WIB) "Process control Domain: security requirements for vendors." Version 2.0, October-2010.

[12] ENISA European Network and Information Security Agency, Algorithms, key size and parameters report 2014, 2014. (last accessed on 17 January 2016)

[13] Internet Engineering Task Force. RFC 4301: Security Architecture for the Internet Protocol. https://tools.ietf.org/rfc/rfc4301.txt (last accessed on 17 January 2016)

[14] Internet Engineering Task Force. RFC 7296: Internet Key Exchange Protocol Version 2 (IKEv2), 2014. https://tools.ietf.org/rfc/rfc7296.txt (last accessed on 17 January 2016)

[15] Internet Engineering Task Force. RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2, 2008. http://www.ietf.org/rfc/rfc5246.txt (last accessed on 17 January 2016)

[16] Bundesamt für Sicherheit in der Informationstechnik. TR-02102-3: Kryptographische Verfahren: Empfehlungen und Schlüssellängen Teil 3 – Verwendung von Internet Protocol Security (IPsec) und Internet Key Exchange (IKEv2). Bonn, Germany. Version 2015-01.

[17] Internet Engineering Task Force. RFC 5289: TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM), 2008. http://www.ietf.org/rfc/rfc5289.txt (last accessed on 17 January 2016)

[18] National Institute of Standards and Technology. Special Publication 800-57 Part 1 Rev. 3, Recommendation for Key Management, July 2012.

[19] IEC 60870-5-101. Telecontrol equipment and systems – Part 5-101: Transmission protocols – Companion standard for basic telecontrol tasks. Second edition. 2003-02.

[20] IEC 60870-5-104. Telecontrol equipment and systems – Part 5-104: Transmission protocols – Network access for IEC 60870-5-101 using standard transport profiles. Second edition. 2006-06.

[21] National Institute of Standards and Technology. Special Publication 800-38C: Recommendation for block cipher modes of operation. The CCM mode for authentication and confidentiality (including updates as of 07-20-2007). 2007.

[22] National Institute of Standards and Technology. Special Publication 800-38D. Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC. November 2007.

[23] National Institute of Standards and Technology. Cryptographic Algorithm Validation Program. http://csrc.nist.gov/groups/STM/cavp/ (last accessed on 17 January 2016)

[24] National Institute of Standards and Technology. Annex C: Approved Random Number Generators for FIPS PUB 140-2 [25], February 2012.

[25] National Institute of Standards and Technology. FIPS PUB 140-2, Security Requirements for Cryptographic Modules, May 2001.

[26] Bundesamt für Sicherheit in der Informationstechnik: Anwendungshinweise und Interpretationen zum Schema, AIS 20, Funktionalitätsklassen und Evaluationsmethodologie für deterministische Zufallszahlengeneratoren, Version 3.0, Bonn, Germany, May 2013. (in German)

[27] Bundesamt für Sicherheit in der Informationstechnik: Anwendungshinweise und Interpretationen zum Schema, AIS 31, Funktionalitätsklassen und Evaluationsmethodologie für physikalische Zufallszahlengeneratoren, Version 3.0, Bonn, Germany, May 2013. (in German)

[28] Bundesamt für Sicherheit in der Informationstechnik. TR-02102-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen. Bonn, Germany. Version 2015-01. (in German)

[29] Internet Engineering Task Force. PKCS #5: Password-Based Cryptography Specification Version 2.0, 2000. http://tools.ietf.org/rfc/rfc2898.txt (last accessed on 17 January 2016)

[30] Open Web Application Security Project. https://www.owasp.org/index.php/Data_Validation (last accessed on 17 January 2016)

[31] Bundesamt für Sicherheit in der Informationstechnik. TR-03116, Part 3, Kryptographische Vorgaben für Projekte der Bundesregierung – Intelligente Messsysteme. In German. Annually adapted. Bonn, Deutschland, Date: 2014.

[32] Ari Takanen, Jared DeMott, and Charlie Miller. Fuzzing for Software Security Testing and Quality Assurance (1 ed.). Artech House, Inc., Norwood, MA, USA, 2008.

[33] Electric Power Research Institute. National Electric Sector Cybersecurity Organization Resource. http://www.smartgrid.epri.com/nescor.aspx (last accessed on 17 January 2016)

[34] bcrypt. http://bcrypt.sourceforge.net/ (last accessed on 17 January 2016)

[35] scrypt. http://www.tarsnap.com/scrypt.html (last accessed on 17 January 2016)

[36] National Institute of Standards and Technology. NISTIR 7946. CVSS Implementation Guidance. April 2014.

[37] BSI, „TR-02102-1 v2015-1: Kryptographische Verfahren: Empfehlungen und Schlüssellängen," 2015

[38] ANSSI, „Mécanismes cryptographiques - Règles et recommandations, Rev 2.03,” 2014.

[39] National Institute of Standards and Technology. Special Publication 800-22 Rev. 1a, A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, April 2010.

[40] SEI CERT Coding Standards, https://www.securecoding.cert.org/confluence/display/seccode/SEI+CERT+Coding+Standards

[41] MISRA C software development guidelines for embedded systems, http://www.misra.org.uk/