INTERNATIONAL STANDARD

ISO/IEC 29128

First edition
2011-12-15

Information technology — Security techniques — Verification of cryptographic protocols

Technologies de l'information — Techniques de sécurité — Vérification des protocoles cryptographiques

Reference number
ISO/IEC 29128:2011(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword ... v

Introduction ... vi

1 Scope ... 1

2 Terms and definitions ... 1

3 Symbols and notation ... 2

4 General ... 3

5 Specifying cryptographic protocols ... 5
5.1 Objectives ... 5
5.2 The abstraction levels ... 5
5.3 The specification of security protocols ... 5
5.3.1 General ... 5
5.3.2 The symbolic messages ... 5
5.3.3 Observing messages ... 6
5.3.4 Algebraic properties ... 7
5.3.5 Protocol roles ... 7
5.4 The specification of adversarial model ... 8
5.4.1 Network specification ... 8
5.4.2 The attacker ... 8
5.4.3 The scenario ... 9
5.5 The specification of security properties ... 10
5.5.1 General ... 10
5.5.2 Trace properties ... 11

6 Cryptographic protocol assurance levels ... 12
6.1 General ... 12
6.2 Protocol Assurance Level 1 ... 13
6.3 Protocol Assurance Level 2 ... 13
6.4 Protocol Assurance Level 3 ... 14
6.5 Protocol Assurance Level 4 ... 14
6.6 Difference among Protocol Assurance Levels ... 14
6.7 Corresponding assurance levels in ISO/IEC 15408 ... 15

7 Security Assessment and Verification ... 16
7.1 Protocol specification ... 16
7.1.1 PPS_SEMIFORMAL ... 16
7.1.2 PPS_FORMAL ... 17
7.1.3 PPS_MECHANIZED ... 17
7.2 Adversarial model ... 18
7.2.1 PAM INFORMAL ... 18
7.2.2 PAM_FORMAL ... 18
7.2.3 PAM_MECHANIZED ... 19
7.3 Security properties ... 20
7.3.1 General ... 20
7.3.2 PSP_INFORMAL ... 21
7.3.3 PSP_FORMAL ... 21
7.3.4 PSP_MECHANIZED ... 22
7.4 Self-assessment evidence for verification ... 23
7.4.1 General ... 23
7.4.2 PEV_ARGUMENT ... 23
7.4.3 PEV_HANDPROVEN ... 23
7.4.4 PEV_BOUNDED ... 24
7.4.5 PEV_UNBOUNDED ... 24

8 Common Methodology for Cryptographic Protocols Security Evaluation 25
8.1 Introduction 25
8.2 Protocol specification evaluation 26
8.2.1 Evaluation of sub-activity (PPS_SEMIFORMAL) 26
8.2.2 Evaluation of sub-activity (PPS_FORMAL) 26
8.2.3 Evaluation of sub-activity (PPS_MECHANIZED) 26
8.3 Adversarial model evaluation 27
8.3.1 Evaluation of sub-activity (PAM INFORMAL) 27
8.3.2 Evaluation of sub-activity (PAM_FORMAL) 27
8.3.3 Evaluation of sub-activity (PAM_MECHANIZED) 28
8.4 Security properties evaluation 28
8.4.1 Evaluation of sub-activity (PSP_INFORMAL) 28
8.4.2 Evaluation of sub-activity (PSP_FORMAL) 28
8.4.3 Evaluation of sub-activity (PSP_MECHANIZED) 29
8.5 Self-assessment evidence evaluation 29
8.5.1 Evaluation of sub-activity (PEV_ARGUMENT) 29
8.5.2 Evaluation of sub-activity (PEV_HANDPROVEN) 30
8.5.3 Evaluation of sub-activity (PEV_BOUNDED) 30
8.5.4 Evaluation of sub-activity (PEV_UNBOUNDED) 30

Annex A (informative) Guidelines for Cryptographic Protocol Design 32

Annex B (informative) Example of formal specification 34
B.1 Symbolic specification of security protocols 34
B.1.1 Abstraction level 34
B.1.2 Protocol specifications 35
B.2 State transitions 37
B.2.1 Attacker model 37
B.2.2 Configuration state 37
B.2.3 Traces 38
B.3 Trace properties 38
B.3.1 Secrecy 38
B.3.2 Authentication 39

Annex C (informative) Verification examples 41
C.1 Sample protocol 41
C.2 Design artifacts 41
C.2.1 Input to protocol verification tool 42
C.2.2 Protocol Specification 43
C.2.3 Operating Environment 44
C.2.4 Security Properties 44
C.2.5 Evidence 44
C.3 Additional inputs for verification 47

Bibliography 49

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 29128 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

Introduction

The security of digital communications is dependent on a number of aspects, where cryptographic mechanisms play an increasingly important role. When such mechanisms are being used, there are a number of security concerns such as the strength of the cryptographic algorithms, the accuracy and correctness of the implementation, the correct operation and use of cryptographic systems, and the security of the deployed cryptographic protocols.

Standards already exist for the specification of cryptographic algorithms, and for the implementation and test of cryptographic devices and modules. However, there are no standards or generally accepted processes for the assessment of the specification of protocols used in such communication. The goal of this International Standard is to establish means for verification of cryptographic protocol specifications to provide defined levels of confidence concerning the security of the specification of cryptographic protocols.

Information technology — Security techniques — Verification of cryptographic protocols

1 Scope

This International Standard establishes a technical base for the security proof of the specification of cryptographic protocols. This International Standard specifies design evaluation criteria for these protocols, as well as methods to be applied in a verification process for such protocols. This International Standard also provides definitions of different protocol assurance levels consistent with evaluation assurance components in ISO/IEC 15408.

2 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

2.1

arity

number of arguments

2.2

cryptographic protocol

protocol which performs a security-related function using cryptography

2.3

formal methods

techniques based on well-established mathematical concepts for modelling, calculation, and predication used in the specification, design, analysis, construction, and assurance of hardware and software systems

2.4

formal description

description whose syntax and semantics are defined on the basis of well-established mathematical concepts

2.5

formal language

language for modelling, calculation, and predication in the specification, design, analysis, construction, and assurance of hardware and software systems whose syntax and semantics are defined on the basis of well-established mathematical concepts

2.6

adversarial model

description of the powers of adversaries who can try to defeat the protocol

NOTE It includes restriction on available resources, ability of adversaries, etc.

2.7

security property

formally or informally defined property which a cryptographic protocol is designed to assure such as secrecy, authenticity, or anonymity

2.8

self-assessment evidence

evidence that the developer uses to verify whether a specified protocol fulfils its designated security properties

NOTE It includes cryptographic protocol specification, adversarial model and output (transcripts) of formal verification tool.

2.9

protocol model

specification of a protocol and its behaviour with respect to an adversarial model

2.10

protocol specification

all formal and informal descriptions of a specified protocol

NOTE It includes all processes by each protocol participant, all communications between them and their order.

2.11

secrecy

security property for a cryptographic protocol stating that a message or data should not be learned by unauthorized entities

2.12

variadic

property of a function whose arity is variable

3 Symbols and notation

For the purposes of this document, the following symbols and notation apply.

ϕ security property of a protocol model

A,*B*	role names
m	message
r	random nonce
k	key
c	communication channel
enc	encryption function
dec	decryption function
<..,…>	paring operator
Send	sending process
Receive	receiving process

4 General

Verification of a cryptographic protocol involves checking the following artifacts:

a) specification of the cryptographic protocol;

b) specification of the adversarial model;

c) specification of the security objectives and properties;

d) self-assessment evidence that the specification of the cryptographic protocol in its adversarial model achieves and satisfies its objectives and properties.

The artifacts shall clearly state parameters or properties relevant for the verification. Examples include the bound used in bounded verification as later descibed in Clause 7.4.4.1 or assumed algebraic properties of cryptographic operators used in the protocol as described in Clause 7.1.2.3 and Clause 5.3.4.

The different Protocol Assurance Levels will lead to different requirements for these four artifacts. The stated requirements are only for design verification not implementation verification.

NOTE 1 For verifying an implementation, additional assurance requirements should be supplied and satisfied.

This International Standard does not specify precisely what proof methods or tools shall be used, but instead only specifes their properties. This encourages protocol designers to use the state-of-the-art for protocol verification in terms of models, methods, and tools.

Verification tools shall fulfil the following conditions.

a) The verification tools are sound.

The protocol designer or possibly an independent third party shall provide evidence of the correctness of the verification tool used. This may, for example, be in terms of a pencil-and-paper proof of the soundness of the calculus used or, in some cases, in terms of code inspection to see that the tool properly implements the calculus.

NOTE 2 This step is nontrivial, yet it is essential if machine checked proofs are to provide greater confidence than hand proofs. In theory, this can be done once and for all for a verification tool, although in practice, tools evolve over time.

b) The results of verification tools are documented in such a way that they are repeatable.

The protocol designer shall provide adequate documentation, including all inputs needed for the tool to construct a proof or (in the case of decision procedures) determine provability.

c) The verification tools are available for outside evaluation and use.

The protocol designer shall indicate all necessary tools to independently check the proofs.

NOTE 3 At least in theory, protocol verification canbe carried out by hand proofs, using paper and pencil. However, given the substantial amount of detail typically involved in security protocol verification, especially for the high Protocol Assurance Levels, confidence in the results is substantially increased by using mechanized tools such as model checkers and theorem provers. Thus, proofs only with paper-and-pencil are treated as lower assurance level (i.e. PAL2) than mechanized proof in this International Standard.

5 Specifying cryptographic protocols

5.1 Objectives

The goal of this part is to provide guidelines and minimal requirements for specifying cryptographic protocols.

5.2 The abstraction levels

The protocols can be specified at several levels of abstraction, each corresponding to a computation model. At the most abstract level, messages are terms constructed from symbols and the attacker is also modelled as a formal process. We will call this abstraction the *symbolic* level. In such a model, the resources (both time and space resources) are not considered.

Any other model can be defined as a refinement of a symbolic model. For instance we can interpret the symbols used in the symbolic model as functions on bitstrings, that can be computed in polynomial time.

Therefore, any cryptographic protocol consists in a symbolic specification and an interpretation in a given domain (e.g. bitstrings or structured data, or even material-dependent formats) of all the symbols, together with assumptions on their interpretation. Such hypotheses can ensure some correspondence between the properties at various abstraction levels.

NOTE In this International Standard, we only consider the symbolic specification of security protocols.

Further documents are required for the specification of other (lower) abstraction levels. Typically, it will be necessary to explain how to specify the interpretation domain and how to carry security guarantees across levels of abstraction.

5.3 The specification of security protocols

5.3.1 General

As explained, a symbolic specification is the necessary first part towards the full specification of a protocol. We list below the minimal mandatory parts in a symbolic protocol specification.

5.3.2 The symbolic messages

The first part consists in specifying what are the possible (valid) messages.

In this clause, the cryptographic primitives used in the protocol must be listed. Since we are talking about a symbolic specification, this part consists of providing with

1. a set of *function symbols* $\mathcal{F}$
   Each function symbol has either a fixed *arity*, that has to be specified, or it is variadic (in which case it has also to be specified).

2. a set of *name symbols* $\mathcal{N}$ that may be split into various syntactic categories that have to be specified.

3. a set of *variable symbols* $\mathcal{X}$.

4. a formal description of valid rules allowing to build messages using the function symbols.
   A (non exhaustive) list of possible ways to specify such a language is:

— Nothing: all terms that are built with the function symbols and following the arity restrictions are valid messages

— Some arguments are restricted to names: some of the arguments of function symbols are restricted to belong to some name categories

— Sorts: a type discipline is defined and only well-typed terms correspond to messages.

The set of valid terms (or messages) is written ($\mathcal{F}$, $\mathcal{N}$) (or ($\mathcal{F}$, $\mathcal{N}$, $\mathcal{X}$) when variables are involved)

EXAMPLE A typical example is encryption, that can be modeled by a symbol `enc`, whose arity is 2 or 3 (or 4), depending on whether the random seed is explicit or not (and whether the encryption algorithm is explicit or not). A specification has to make precise what is the arity of `enc` and what are the assumed types of its arguments. Typically, `enc` has an arity 3. As possible name categories there are the random seeds, whose symbols will start with r, the keys, whose symbols will start with k, the algorithms, whose symbols will start with α, and so on. If `enc` has been specified as being an arity 3 symbol, we can additionally restrict its arguments, specifying for instance that the first argument must be a key and the last one must be a random. In that case, $\texttt{enc}(k, k, r)$, $\texttt{enc}(k, \texttt{enc}(k, r_1, r_2), r_1)$, are valid terms while $\texttt{enc}(\texttt{enc}(k, k', r), k'', r')$ and $\texttt{enc}(r, k, r')$ are not valid terms. Examples of symbols that can be considered as variadicinclude the exclusive or $\oplus$, the arithmetic multiplication × or the concatenation ||.

5.3.3 Observing messages

This part consists of specifying some comparison predicates between messages.

Only the equality predicate is mandatory, since other predicates could be seen as Boolean functions and specified within the equality definition. It might however be useful to distinguish later between the computation abilities and the observation abilities. Moreover, in many current specification languages, properties of the function symbols are specified equationally (see clause 5.3.4), while it might be impossible to specify equationally the observation abilities.

This part consists in listing predicate symbols, together with their arity. Typical examples include typing predicates, equality, and `same_length` (that checks that its two arguments have the same length), `same_key` (that checks whether two ciphertexts are encrypted with the same key).

5.3.4 Algebraic properties

This part specifies when two valid terms represent the same message and, more generally, what are the interpretations of the predicate symbols listed in the previous clause.

For instance, when function symbols include both (symmetric) encryption and decryption, we might wish to state that $\mathtt{dec}(k, \mathtt{enc}(k, x, r)) = x$ where r is a symbol for a random seed to express probabilistic encryption: these are two term representations of the same message. We might also wish to state that, $x \oplus x = 0$ if we are using a symbol $\oplus$ meant to represent exclusive or.

As usual, we assume that any two terms that are not specified to be equal are different. The same rule applies to the predicates: everything that has not been specified to be true is, by default, false.

5.3.5 Protocol roles

A *role* is an interactive program that receives some input from the environment and sends messages to the environment. This is the atomic program component of a protocol: there is no communication that takes place inside a role.

Specifying a role requires to provide with:

1. A role name;

2. A finite list of formal parameters: these are the data, that can be used by the program without being generated or received from the environment;

3. A (usually finite, but this is not mandatory) set of control states;

4. A finite set of local variables and local names;

5. A specification of the sending and receiving abilities, as well as state transitions;

6. Formally, this amounts to specify two relations $q,v \xrightarrow{\text{Send}(c,m)} q',v'$ and $q,v \xrightarrow{\text{Receive}(c,m)} q',v'$, a communication channel c and a message m.

Such a specification does not commit to any particular programming language or any particular way to perform tests or moves. It only requires the specification of import/export data and communications with the environment.

EXAMPLE This is a possible specification of the responder role in the public key Needham-Schroeder protocol. We assume a single communication channel, which is omitted below.

1. role name: B;

2. parameters: the identity b of the agent running the instance of this role, the identity a, the private key of b, the public key of a;

3. local states: there are only three local states: q_0, q_1, q_f ;

4. local variables and names: n_B, r are local names and x, y are local variables

A specification of the transitions. Any other formally defined language can be substituted here:

$$q_0, n_B, r, x = 0, \mathrm{y} = 0 \xrightarrow{\mathbf{Receive}(m)} q_1, n_B, r, x = m, y = 0$$

$$q_1, n_B, r, x = m, y = 0 \xrightarrow{\mathbf{Send}\left(\mathbf{enc}\left(\mathbf{pub}(a),\langle \mathrm{m}', \mathrm{n}\rangle, \mathrm{r}\right)\right)} q_f, n_B, x = m, y = m'$$

$$\text{if } a = \pi_1\left(\mathbf{dec}(\mathbf{priv}(\mathrm{b}), x)\right), m' = \pi_2(\mathbf{dec}(\mathbf{priv}(b), x))$$

We use here a ternary encryption symbol `enc`, a decryption symbol `dec`, a pairing symbol ⟨_,_⟩ and projections symbols π_1, π_2.

NOTE In such an example, the transition is not specified when the test fails, meaning that there is no transition in this case: the program is stacked in state q_1. There are of course many other possible designs.

Sessions A *role instance* is a specific copy of a role, together with its actual parameters. This is sometimes called a *session*. As the same identity can run concurrently several copies of the same role, it might be convenient to include in the parameters an *identifier* also called sometimes *session number*, that will allow different role instances to be distinguished.

5.4 The specification of adversarial model

5.4.1 Network specification

This part specifies what are the (symbolic) communications devices and their reliability.

Typically, a list of channels (terms or symbols) is given, each of which with its own properties. For instance, one could specify a single public communication channel c which is under complete control of the attacker (who can intercept messages and send fake messages). But it could be refined further, distinguishing a more (or less!) secure proxy communication channel or a wireless channel that can only be eavesdropped, or even a private channel that is completely out of control of the attacker.

5.4.2 The attacker

This part specifies the computational abilities of the symbolic attacker. In other words, it specifies the messages m that can be computed from a set of messages S.

Typical specifications use inference systems such as the “Dolev-Yao inference system”. These rules might depend on whether there is an explicit decryption symbol or not, for instance. The simplest specification consists in having all function symbols explicit and public. Then the attacker, when given a set of names $\mathcal{N}$, is able to compute any term built on $\mathcal{F}$ and $\mathcal{N}$.

In addition, the attacker can use the predicate symbols of clause 5.3.3, even though such predicates are not used in the definition of the roles. Attacker's other capabilities are specified in clause 5.4.1 and depend on the reliability of the various channels.

From this clause and the previous one, it should be possible to define formally what are the possible execution traces.

NOTE 1 Typical attacks can be formally described as follows.

Eavesdropping attack is a typical security risk posed to networks. In some network environment, messages are broadcasted to everyone. This often can cause a problem that important messages such as passwords and credit card numbers might be delivered to unintended person. This attack can be formally described as a subset of the model in clause 5.4.1. That is, given a list of channels $\{c\}$, an attacker has no control on the channels $\{c\}$ but can listen to all messages S exchanged over the channels. Then, the whole knowledge of the attacker is any term which can be computed from a set of messages S.

Replay attack is another type of risk. In open networks like Internet, messages can be exchanged via routers which is under control of malicious person. In such an environment, messages such as passwords or credit card numbers exchanged over a network might be maliciously repeated or delayed by them. This often can cause a problem that unintended person impersonates a legitimate one by repeating the stored password as a proof of identity. This attack can be formally described as the model in clause 5.4.1 and a subset of the model in clause 5.4.1. That is, given a list of channels $\{c\}$ and a set of messages S exchanged over the channels $\{c\}$, an attacker has complete control on the channels $\{c\}$ and listen to all messages S. Thus, the whole knowledge of the attacker is any term which can be computed from a set of messages S. But, in replay attack, he uses only the elements of S and tries to impersonate some role.

NOTE 2 Extension of the model is required to describe a series of attacks such as denial-of-service attack and relay attack. Since these attacks are related to the physical properties of an actual system such as processing time of operations and communication speed via physical media, in order to describe such attacks, such physical properties should be somehow included in the extended model.

5.4.3 The scenario

The last part in the protocol specification consists in describing the execution environments that are considered.

This includes in particular the following important features: