European Scientific Journal January 2014 edition vol.10, No.3 ISSN: 1857 – 7881 (Print) e - ISSN 1857- 7431

A NOVEL METHODOLOGY TO DESIGN SECURITY PROTOCOLS BASED ON A NEW SET OF DESIGN PRINCIPLES

Rosa Elena Di Costanzo
Tecnológico de Monterrey Campus Toluca,
Associate Professor of Computer Science,
Dept. of Computer Science and Information Systems Mexico
Luciano Chirinos
Tecnológico de Monterrey Campus Toluca,
Full professor of Electronics and Mechatronics, Dept. of Mechatronics,
Head of the Mechatronics Undergraduate Program, Mexico

Abstract

This paper presents a novel design methodology based on a new set of design principles to develop step-by-step security protocols for up to three participants, guiding the designer on each step. It accompanies the designer through a succession of six abstraction levels proposed in this work: protocol objectives, protocol constraints, security mechanisms, message flow, protocol conformation and authentication tests. The methodology proposed is based on a new set of design principles extracted from different sources and combined using the systemic approach, which considers the designer and client's security and functional needs. The resulting model separates high-level tasks from implementation details, allowing the designer to specify the security requirements and functionality desired for each abstraction level. Consequently, the protocol design is linked with the best-fitting design principle. To corroborate the results of the methodology, the resulting protocol in the Alice and Bob notation in the fifth level is tested using the "Strand Spaces" Model. The Needham-Schroeder protocol with symmetric keys was successfully used as a test. The security goals achieved were: authentication, confidentiality, integrity, and non-repudiation.

Keywords: Security protocols, authentication, protocol design, methodology

1. Introduction

The design of security protocols is a problem constantly confronted by security experts. With increasing demands for functionality and reliability, protocol sizes and complexities have expanded. Currently,

designers face several challenges while creating a security protocol, namely the fundamental challenge of how to gather a series of rules for exchanging information that at the same time are complete, minimal and logically consistent, that may be efficiently implemented (Holzmann, 1991) and provide functional and/or security requirements. To face those challenges, there is a need for tools to help designers create protocols that respond to such increasing complexities. In this paper we present such a tool: a novel step-by-step methodology based upon a hierarchical structure, for the design of security protocols for two or three participants, where not only their security and functional needs are taken into account, but also the most appropriate design principles.

This methodology presents a new set of protocol design principles arranged in hierarchical levels that facilitate designers' creative process, showing a feasible, inexpensive and safe way to produce a protocol, in the Alice and Bob notation. In the lowest level we finish by converting the new protocol to the Strand Spaces Model, in order to corroborate results through authentication tests.

Although design principles have been in use since the 90s, in this paper we take a strictly systemic approach (Checkland, 1991; Van Gigch, 1974; Wilson, 1992), extracting their essence and integrating them in different abstraction levels. Many such principles are written to account for various aspects of security goals and mechanisms, plus the synergy obtained from their application. We based our search for design principles on those that are critically examined in the scientific literature and accepted by the scientific community, as presented in section 2.

With this methodology designers can formalize explicit assumptions about underlying cryptographic algorithms and about trust (Choi, 2006), clarify protocol objectives, define detailed specifications of internal actions of protocol agents and verify messages and their parts on receipt (Fidge, 2001).

Our first contribution is the proposal of a new set of design principles integrated at each abstraction level of the methodology, instead of continuously reviewing the principles at each step of a design process.

Our second contribution is the methodology itself, which not only involves the most relevant concepts of design principles on each abstraction level as presented in Section 3.1, but also includes, in Table 2, a path for the clear definition of protocol objectives that contains the main functionalities of security protocols appearing in the principal repositories (AVISPA, n.d.; SPORE, n.d.) and standards (ISTF, 2013), which typically include authentication of the participants and/or messages, and definition or use of secret keys for short or long term.

The third contribution, in Table 5, is a combination of mechanisms that helps to achieve authentication, one of the main goals of security protocols, and also helps with the other three: confidentiality, integrity and non-repudiation.

2. Theoretical framework

Typically there are three main techniques used to design a protocol: the first is the optimization of a current implementation without changing its functionality, but simply reducing its processing costs; the second is the development of new mechanisms, reengineering existing protocols, in order to reduce conflicts arising from a particular feature of the operating environment, for example to get fairness in a contract signing, using an online trusted third party. The third technique is the development of new protocols, tailored to exactly match the needs of a given application; its main drawback is the cost of developing a specialized protocol, being that the security of a protocol depends on information that is either given as informal contextual description or implicitly assumed. Current techniques for creating security protocols deal with the human interaction and knowledge; the designer states the security goals and finally creates a corresponding security protocol (Oceanasek & Sveda, 2006).

The idea of developing design principles for security protocols originated from the observation that successful attacks against these protocols were the result of malpractices in design, even more, it was observed that good design principles can lead us to build more robust protocols. Over time, design principles for security protocols have shown completeness and consistency and have been accepted by the scientific community.

This work is based on design principles proposed by the scientific community (Abadi & Needham, 1996; Anderson & Needham, 1995; Aura, 1997; Carlsen, 1994; Dong, Chen, Wen, & Zheng, 2007; Gritzalis, Spinellis, & Georgiadis, 1999; Khurana, Bobba, Yardley, Agarwal, & Heine, 2010; Malladi, 2008; Syverson, 1996; Xianxian, Hun, & Zhaohao, 2004; Zheng, 2000). Although these principles do not readily apply because they tend to be written independently and address multiple problems, here we present those considered most useful as a reference to compare designs in search of faults. We began with those provided by Abadi and Needham (1996) and enriched by other authors (Anderson & Needham, 1995; Aura, 1997; Carlsen, 1994; Dong, et al., 2007) that include principles appropriate to technological advances or meant to delve into cryptographic details.

2.1 Abadi and Needham Design Principles

The seminal work of Abadi and Needham (1996) is considered within the scientific community as a breakthrough in the development of security protocols; they noticed that several common characteristics difficult to analyze in published protocols could be avoided and guide to error prevention. Therefore, in order to prevent confusion and errors, they proposed a set of informal guidelines to complement formal design methods to a level of detail that indicates what and how.

2.2 Design principles from other authors

With the aim of updating the principles of Abadi and Needham (1996), based on an improvement in their application, in other viewpoints or on technology changes, we include additional principles, collected from accepted scientific literature.

Aura (1997) proposes five strategies against replay attacks whose systematic application leads to achieve desirable properties. Carlsen (1994) proposes principles directed toward understanding and classification of flaws in the protocols, measuring against different categories of faults that may be present explicitly in the specifications or implicitly when the specification has been incomplete; his proposal is closely related to the design level. Dong et al. (2007) introduce a new concept called "protocol engineering" into three groups: security (1-5), detailed design (6-9) and provable security (10). They visualize the design as Systems Engineering, pointing out the explicit relationship between security and design, including the environment in executions and emphasizing in obtaining strength in safety and accuracy of cryptographic services. Anderson and Needham (1995) extend to the public key scheme the principles of Abadi and Needham (1996), emphasizing the design level, explicit assumptions, and the extension to more complex protocols.

2.3 Authentication tests based upon the Strand Space Model

Authentication tests using the Strand Space Model (Guttman & Thayer-Fábrega, 2002) are evidences of authenticity that give the possibility to infer that participants in a protocol run are not intruders; in other words: are evidences of authenticity that allow inferring that some principal possessing relevant key material has received and transformed a message carrying a distinct value. The authors embody these ideas in three tests: outgoing, incoming and unsolicited; their results allow us to establish authentication without any consideration of the dynamic execution of protocols, involving the activity of several principals. Instead, it is sufficient to consider the possible behaviors of the principals independently.

The rules of the protocol determine which transformations are permitted, because security protocols are designed to control the ways that protocol participants transform messages (Doghmi, Guttman, & Thayer-Fábrega, 2007). The protocol determines when a critical value may be transmitted within new forms of messages, when such a critical value has occurred only within a particular set of cryptographic contexts, then a participant may be authenticated by the way it transforms into new contexts.

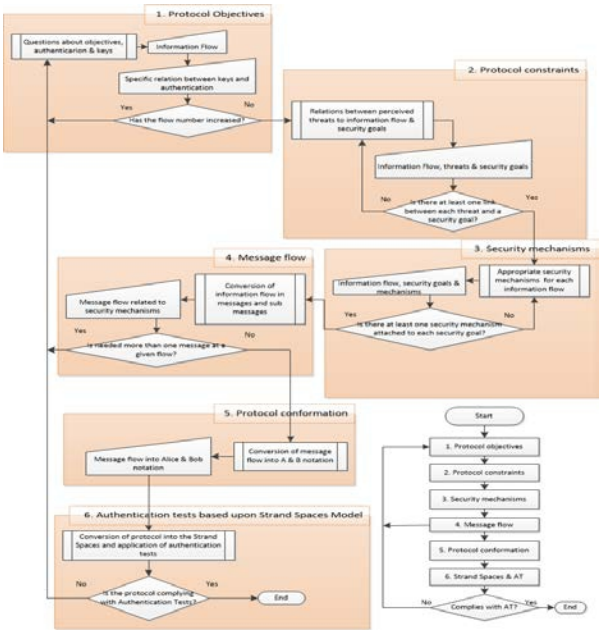
3. Methodology

From a systemic and systematic study of the design principles of Abadi and Needham (1996), Anderson and Needham (1995), Aura (1997), Carlsen (1994) and Dong et al. (2007), we decided to separate them based on security goals related to security mechanisms; because in some cases one principle affects different levels of abstraction in the design process. Once separated, we have classified them, placing each one in the level where it would provide more aid, according to the needs of functionality and security that we had to accomplish. We have identified six levels, depending on the problem being addressed; our methodology is presented in Figure 1, structured from the higher, first level to the fifth that has detailed implementation considerations, finishing in the sixth with authentication tests.

Figure 1: Methodology outline

3.1 Selection and classification of design principles

In order to combine the design process based on levels of abstraction,



with the set of selected design principles, we proposed the key terms of the principles, such as predictable numbers (counters, timestamps, nonces), encryption, digital signature, key, session key, long-term key, symmetric cipher and asymmetric cipher, through a systematic analysis process; relating each key term of each principle with the proposed security goals: authentication, confidentiality, integrity and non-repudiation. Then we assigned each principle with the abstraction level that best fitted it, taking into account primarily the problem faced in that level of the design process.

We obtained a group of five levels of questions and suggestions for designers for review considerations which in turn are constrained by decisions made at the highest levels, so that when a designer applies all the steps successively in the top-down methodology, all design principles are covered and applied.

3.1.1 Level 1

It is related upon the protocol´s operating environment, with explicit trust relationships on which it depends, allowing us to document the ways in which the security requirements are met within the domains that interact in the environment, the assumptions made in key agreements and the reasons for this necessary dependence.

3.1.2 Level 2

It analyzes the attack model and the skills of the adversary, the designer has to express their needs as a result of the threats perceived by each participant, expressing them first as security goals and then as requirements for the protocol; these data are directly related with the operating environment. At this point we must express the strength required for the protocol, checking operating conditions (existing circumstances) and restrictions (regulation or limitation) embodied in the previous level, detailing further assumptions of trust and secrecy for each participant, specifying the use of any certification authority.

3.1.3 Level 3

It is related to specific content, timing and performance conditions of the messages, as well as an understanding of the trust assumptions made in key agreements; it is also related to explicit mention of participants' names to show their identity and to authenticate its aliveness, as much as knowing the reasons to encrypt, digitally sign, generate nonces and time stamps, under a clear comprehension of the assumed properties. In the same way, it recommends to use available formalisms for specifying the messages, having the restrictions clear.

3.1.4 Level 4

It is related to the precise shape of the messages and their acting conditions; each message has to say what it means, so that their interpretation depends only on its content, without knowing the context. Each message has to be explicit in defining the parameters of the cryptographic primitives and any property that can be used to attack them.

3.1.5 Level 5

It is related to identification of details of the protocol and execution levels, and with details to be taken into account during the programming and implementation of the protocol. Among the recommendations are: to apply different words associated with each primitive type, placing an identifier in the protocol, in the step of the protocol, in the sub-elements of the message and in the run number; as the representation depends on the implementation, perform the bit count of all operating in the message.

3.1.6 Level 6

This level is concerned with authenticity tests; these tests give the possibility to infer, using the Strand Spaces model that participants in the execution of a protocol are not invaders. These tests are evidence that allow us to infer that a participant in possession of key has received and transformed a message, returning a different value.

4. Results

There are six levels in the model, in order to improve its usage we recommend to review the design using successive refinements in each.

4.1 Protocol objectives

Overview of what the designer wants to achieve with the protocol, indicating the operating environment, the participants, their roles, potential information flows, the flows order of presentation, the assumptions made in key agreements and the reasons for this necessary dependence.

The designer will describe the environment and through what means the information transmission will be carried out. The process will be of successive refinements, in order to achieve protocol´s objectives and its functional requirements.

The designer should be clear about:

1. What he wants to do with the protocol?
2. How many participants will the protocol have?
3. Who are the participants, the role of each and if it will need to include a trusted third party or authentication server, which can be

online or offline. The former is considered a tripartite protocol, the latter bipartite because we assume that keys have been previously established.

4. What information is going to be sent and received?
5. What is the order of the information that will be sent and received?
6. Development of an information flow table according to the number of participants, considering that the initiator is always (a) and may initiate the transmission with the second (b) or the third participant (s) also it must be taken into consideration that interactions are always alternated.

A simple example of these (partial) flows is presented in Table 1, where (a) initiates with a message to (b), who responds to (a) accepting the conversation; then (a) also asks a trusted third party to participate. Later we will delve further into these flows in more detail, using a thorough exemplification.

Table 1: Information Flow

Protocol step	a	Information flow	b		s
1		Intent to … with b →			
2		← Accept … with a trusted third party			
…		…			
n		Ask for … to the trusted third party →			

From Table 1 of Information Flow, the designer has to identify those data that each participant should uniquely know, because they could increase the flow number, if it's needed to spread a message into its parts. If so increased, assure that flows are alternated between participants; otherwise they must be reorganized.

Then the designer has to answer the following questions to validate that his choices meet the stated objectives previously defined at the beginning of this level; otherwise, he shall redefine the objectives and redo all following steps:

1. In what environment will the protocol operate? So that participants and infrastructure are directly related to objectives.
2. What authentication kind the designer wishes to achieve? So authentication is related to objectives.
3. The kind of keys to be used: short or long term? So keys are directly related to objectives.
4. If short term keys, they may be previously generated (key transport) or with the use of an online server (contributory keys), the establishment of keys is essentially a message authentication, where the key is the message

5. If long term keys, they could be: symmetric or asymmetric encryption (with the use of an online server); or previously generated keys.

By answering to the previous questions, the designer can locate more precisely on Table 2, the protocol type that he has to design.

4.2 Protocol constraints

Description of the environment and user-demanded conditions to be met by participants, based on perceived threats related to information flows and security goals to accomplish.

Once information flows are identified and design so far complies with objectives, the designer will then review the contents description, specifying in detail the information exchanged and threats perceived by each participant, applying the Dolev-Yao Model (Cervesato, 2001; Dolev & Yao, 1983), to link threats to security goals, as presented in Table 3.

There must exist at least one link between each threat and a security goal; otherwise the designer shall redefine the objectives going to the first level.

4.3 Security Mechanisms

Description of the security mechanisms appropriate for each information flow, necessary for the implementation of the constraints indicated.

The designer will link the security goals he wants to achieve with the security mechanisms listed in Section 3.1 and summarized in Table 5, analyzing each information flow and content separate into parts to determine what to encrypt, where to use a digital signature, or where to apply a hash, etc. At least one mechanism is required to deal with each detected threat related to each security goal, gradually scoring the resulting matrix of the previous level, taking into account the flow and the recommendations of the design principles. An exemplification of the process to link security goals with security mechanisms is shown in Table 4.

These eight mechanisms applied directly or in combination provide different strengths to protocols, knowing that higher strengths typically involve greater computational complexity, so we recommend analyzing the computational capabilities of participants to choose the appropriate mechanism. We also recommend not only to authenticate the entity, but also to ensure its aliveness and/or its freshness, so we must include appropriate mechanisms to do so as presented in Table 5.

There must be at least one security mechanism attached to each security goal; otherwise the designer shall redefine the objectives going to the first level.

To achieve the confidentiality goal, as a general rule encryption should be applied; it could also be for a given field or for a specified time. In order to ensure confidentiality and freshness of a message, appropriate protection for data integrity should be provided. To achieve integrity in general a hash function or in a particular field should be used. And to achieve non-repudiation, a digital signature or a confirmation of the message content (ensuring the integrity of the message) should be employed

Table 2: Relations between use or establishment of keys and authentication

Keys type	Use or setting keys	Keys usage	Participants	Authenti-cation
Session or short-term keys	Key transport	Previously generated keys	2 a, b	Sender
				Receiver
				Mutual
				Message
Session or short-term keys	Contributory agreement	With the use of an online server	3 a, b, s	Sender
				Receiver
				Mutual
				Server
				Message
Long-term keys	Symmetric cipher	Previously generated keys	2 a, b	Sender
				Receiver
				Mutual
				Message
		With the use of an online server	3 a, b, s	Sender
				Receiver
				Mutual
				Server
				Message
Long-term keys	Asymmetric Encryption System PKI	Previously generated keys	2 a, b	Sender
				Receiver
				Mutual
				Message
		With the use of an online server	3 a, b, s	Sender
				Receiver
				Mutual
				Server
				Message

Table 3: Information flow, threats and security goals

Links between information flow - threat - security goal							
Flow number	Information direction	Information exchanged	Threat	Authen-tication	Confiden-tiality	Inte-grity	Non repu-diation
1	From a to b	Intent to form a session key with b	Communi-cating with an attacker	X			X

Table 4: Information flow, security goals and mechanisms

Table linking information flow - security goal - mechanism						
Flow number	Infor-mation flow	Information exchanged	Authenti-cation	Confiden-tiality	Inte-grity	Non repudiation
1	From a to b	Intent to form a session key with b	X sender name			X Sender digital signature

Table 5: Summary of authentication

Table Summary of Authentication				
Authen-ticated entity	Directly authenticates the entity	Aliveness	Freshness	Authenticates entity indirectly
	Mechanism-1	Mechanism-2	Mechanism-3	Mechanism-4
Sender	name	Time stamp	Nonce or counter *Predictable quantity	Encryption
Receiver	name	Time stamp	Nonce or counter *Predictable quantity	--------
Message	Digital signature **Non repudiation	Hash function **Integrity and confidentiality	--------	--------

* A predictable quantity can be a nonce, a counter or time stamp.

** Using the selected mechanism, you can also achieve the indicated security goals.

4.4 Message Flow

Conversion of information flow in messages and sub messages for the application of the specific mechanisms needed to meet security goals. Information flow will be transformed into messages and sub-messages, by reviewing the detail achieved in the third level; in such case that more than one message is needed at a given flow, the process must return to the first level, to restate order and content. We will form Table 6 analyzing and relating each of the resulting messages with the appropriate mechanisms according to the characteristics and restrictions set out in the third level. In Table 6 is shown an example of the process and results, to detail security mechanisms for each message.

4.5 Message Contents

It is the conformation of each message with its specific contents and succession, in an orderly form. Message flow will be transformed into the Alice and Bob notation, by reviewing the detail achieved in the fourth level. We will form Table 7 analyzing and transforming each message with its recommended mechanism(s), to the appropriate Alice and Bob notation (Abadi & Rogaway, 2002; Doghmi, et al., 2007) described in Table 7.

The notation used in level 5 considers messages A as a set of terms freely generated from atomic message sets (t), consisting of nonces, time stamps, names of agents, participants or principals, label and a set of keys K, concatenation messages, using the semicolon notation: m_1; m_2 and encryption with $\{|m|\}_k$, where $k \epsilon K$ and $m_1, m_2 \epsilon A$, with *h(t)* as the result of applying the hash function to message t, N_a, N_b as nonces generated by the corresponding participant (Abadi & Rogaway, 2002; Guttman & Thayer-Fábrega, 2002).

Table 6: Messages and mechanisms

	Information flow data	Security mechanism
Flow number	1	
Message flow	From a to b	
Information exchanged	Intent to form a session key with b	
Message parts	1	
Part-1	Sender name	Digital signature
•••		
Part-i		

We also assume two functions, the first mapping between principals a, b... to their public keys k_a, k_b… and the other, a pair of principals' ⟨a, b⟩ toward their shared symmetric keys k_{ab}. The set of keys K, comes with an inverse operator, mapping each member of a pair of keys from an asymmetric cryptosystem to the other: $(k_a)^{-1} = k_a^-$, or each symmetric key to itself: $(k_{ab})^{-1} = k_{ab}$, private keys are denoted as kp_a y kp_b. Function $Parts_k$, maps a message $t \epsilon A$ to a set containing all sub-terms of t that are accessible by knowing the set of keys K, and $[[t]]_a$ the message t digitally signed with a's private key.

Table 7 is made from Table 1 made at the first abstraction level to check the orderly succession of participants who exchange messages, here we develop a similar table in Alice and Bob notation, with the contents of each message, and an arrow indicating the directions of sending and receiving according to their communication requirements, arranged so that they achieve the protocol's objective; all participants should appear in the table.

Table 7: Message flow and contents

Protocol step	Message flow	Message contents
1	$a \longrightarrow b$	$[[a]]_a$
2	$b \longrightarrow a$	$[[b; \{\|a; N_{b1}\|\} k_{bs}]]_b$
…	…	…
n	$a \longrightarrow s$	$[[a; b; N_a; \{\|a; N_{b1}\|\} k_{bs}]]_a$

4.6 Authentication tests based upon the Strand Spaces Model

Is the conversion of protocol into the Strand Spaces model (Guttman, 2002; Oceanasek & Sveda, 2006) and its verification by authentication tests

(Guttman & Thayer-Fábrega, 2002), which require protocol conversion into the Strand Spaces model, to be analyzed in order to confirm that they are complying with any of the three tests; otherwise we will return to level 1.

4.7 Application example

The aims of this protocol is to establish a shared session key between two parties with mutual authentication using symmetric cryptography supported with an online server, with contributory agreement. To show an example of this methodology, we used the well-known NSSK Protocol, also known as Symmetric Cryptography NSPK.

4.7.1 Protocol objectives

The designer will describe the environment and through what means the information transmission will be carried out, applying the questions proposed in 4.1.

1. To establish a session key between two parties within a public network, in order to protect subsequent communication.
2. Three participants.
3. Initiator - a, Receiver - b, Server - s (trusted third party).
4. Data necessary to build a session a key among the three participants, enabling secure communication between the first two a and b.
5. Initiator (a) will call the other participant (b). The receiver (b) responds accepting the transmission. Each participant (a) and (b) contributes with a portion of the session key.
6. The server (s) will bring together contributions, forms the session key and the server (s) sends to the initiator (a) the joint key. The initiator (a) informs the recipient (b) the resulting key.
7. The resulting information flow is presented in Table 8.

Table 8: Information flow for NSSK Protocol

Pro-to-col step	a	Information flow	b	Information flow	s
1		Intent to form a session key with b →			
2		← Acceptance of deal, using a unique data for trusted third party			
3		Intent to form a session key with b →			
4		← Accepts, forming and sending the key			
5		Sends the session key →			
6		← Sends a secret data encrypted with the session key			
7		Sends the prior secret data, modified and encrypted with the session key →			

From Table 8 the designer has to identify those data that uniquely each participant should know, because it could increase the number of flows, and if so increased, assure that they are alternated between participants; otherwise Table 8 must be reorganized to achieve the objective of the protocol.

Based on the operational environment and the client's restrictions, the designer states the answers to the questions posed at this Level 1, to select an option on Table 2 in order to define more clearly the protocol's objective:

1. Protocol operates in Internet.
2. Requires mutual authentication between participants and unauthenticated messages.
3. Short-term keys.
4. Contributory agreement, with the use of an online server.

4.7.2 Protocol constraints

Once information flows are identified the designer will review the contents description, to specify in detail the information exchanged and threats perceived by each participant, then applying the Dolev-Yao Model (1983) he will link them to the security goals to fulfill, as presented in Table 9.

Table 9: Constraints for NSSK Protocol

Flow number	Infor-mation flow	Information exchanged	Threat	A	C	I	NR
1	From a to b	Intent to form a session key with b	You are communicating with an attacker	X			X
2	From b to a	Acceptance of deal, using a unique data for trusted third party	You are communicating with an attacker	X		X	X
3	From a to s	Intent to form a session key with b	A spy is recording and compromising the data transmission	X		X	X
4	From s to a	Accepts, forming and sending the key	A spy is recording and compromising the data transmission	X	X	X	
5	From a to b	Sends the session key	You are communicating with an attacker	X			
6	From b to a	Sends a nonce encrypted with the session key	You are communicating with an attacker	X			
7	From a to b	Sends the previous nonce, modified and encrypted with the session key	You are communicating with an attacker	X			

A = Authentication, C = Confidentiality, I = Integrity, NR = Non repudiation.

4.7.3 Security Mechanisms

The designer will link the security goals he wants to achieve with the security mechanisms listed in Section 3.1 and summarized in Table 5, analyzing each information flow and content separate into parts to determine what to encrypt, where to use a digital signature, or where to apply a hash, etc. At least one mechanism is required to deal with each detected threat related to each security goal, gradually scoring the resulting matrix of the previous level, taking into account the flow and the recommendations of the design principles, as presented in Table 10.

Table 10: Mechanisms for NSSK Protocol

Flow number	Infor-mation flow	Information exchanged	A	C	I	NR
1	From a to b	Intent to form a session key with b	X Sender name			X Sender digital signature
2	From b to a	Acceptance of deal, using a unique data for trusted third party	X Sender name		X Applying a hash function	X Sender digital signature
3	From a to s	Intent to form a session key with b	X Sender name		X Applying a hash function	X Sender digital signature
4	From s to a	Accepts, forming and sending the key	X Sender name	X Encryp-tion of sensitive data	X Applying a hash function	
5	From a to b	Sends the session key	X Sender name			
6	From b to a	Sends a nonce encrypted with the session key	X Sender name			
7	From a to b	Sends the previous nonce, modified and encrypted with the session key	X Sender name			

A = Authentication, C = Confidentiality, I = Integrity, NR = Non repudiation.

4.7.4 Messages

We will form Table 11 analyzing and relating each of the resulting messages with the appropriate mechanism according to the characteristics and restrictions set out in the third abstraction level according to Table 5.

Table 11: Information flow and mechanisms for NSSK Protocol

	Information flow data	Security mechanism
Flow number	1	
Information flow	From a to b	
Exchanged information	Intent to form a session key with b	
Quantity of parts	1	
Part-1	a	(a's) Digital signature

We will continue the process until the last flow is reached.

4.7.5 Sequence and Message content

Resulting Table 12 is taken from the first and the previous levels to check the orderly succession of participants who exchange messages, with an arrow indicating the directions of sending and receiving according to their communication requirements, arranged so that they achieve the protocol's objective. All participants should appear in the table:

Table 12: Messages flow and contents for NSSK Protocol

Protocol step	Message flow	Message content
1	a ⟶ b	$[[a]]_{a}$
2	b ⟶ a	$[[b; \{\|a; N_{b1}\|\} k_{bs}]]_{b}$
3	a ⟶ s	$[[a; b; N_a; \{\|a; N_{b1}\|\} k_{bs}]]_{a}$
4	s ⟶ a	$\{\|k_{ab}; b; N_a; \{\| k_{ab}; a; N_{b1}\|\} k_{bs} \|\} k_{as}$
5	a ⟶ b	$\{\| k_{ab}; a; N_{b1}\|\} k_{bs}$
6	b ⟶ a	$\{\|N_{b2}\|\} k_{ab}$
7	a ⟶ b	$\{\|N_{b2+1}\|\} k_{ab}$

4.7.6 Authentication tests based upon the Strand Spaces Model

Authentication tests require protocol conversion into the Strand Spaces model, to be analyzed in order to confirm that they are complying with any of the three tests; otherwise we will return to Level 1, as presented in Figure 2.

5. Conclusion and future work

This document proposed a new methodology for security protocol design step-by step, showing a feasible, inexpensive and safe way to deliver a protocol in the Alice and Bob notation and has demonstrated its utility through one application validated with authentication tests. The main contribution of this work is a novel and comprehensive methodology, based on a new set of design principles, which also meets the functional requirements and security goals requested by the protocol users. This work faces the security protocol's design challenge of gathering a complete and minimum series of rules that are at the same time logically consistent and possible to implement efficiently.

Step		a		b		Authentication Tests
1	*+n1*	•	$[[a]]_a$	•	*-m1*	
			→	Unsolicited Test		
		⇓		⇓	*+m2*	
2	*-n2*	•	$[[b; \{\|a; N_{b1}\|\} k_{bs}]]_b$	•		
			←	Outgoing Test		
		⇓		s		
3	*+n3*	•	$[[a; b; N_a; \{\|a; N_{b1}\|\} k_{bs}]]_a$	•	*-p1*	
			→			
		⇓		⇓		
4	*-n4*	•	$\{\|k_{ab}; b; N_a; \{\| k_{ab}; a; N_{b1}\|\} k_{bs} \|\} k_{as}$	•	*+p2*	
			←	Outgoing Test		
		⇓				
5	*+n5*	•	$\{\| k_{ab}; a; N_{b1}\|\} k_{bs}$	•	*-m3*	
			→			
		⇓		⇓	*+m4*	
6	*-n6*	•	$\{\|N_{b2}\|\} k_{ab}$	•		
			←	Outgoing Test		
		⇓		⇓	*-m5*	
7	*+n7*	•	$\{\|N_{b2+1}\|\} k_{ab}$	•		
			→	Outgoing Test		

Figure 2: Strand Spaces and authentication tests for NSSK Protocol

The presented methodology was proven with Guttman's authentication tests (2002), which give the possibility to infer that participants in a protocol run are not intruders (Dolev & Yao, 1983), also fulfills four of the most important goals of security protocols, namely authentication, confidentiality, integrity and non-repudiation, accompanying its designers and their clients in a systematic and systemic design process that includes the client's needs and the constraints that must be imposed to achieve the desired security goals, through a simple, structured and easy to apply process, it allows building complex protocols by successive refinements, from the establishment of secure associations in simple cases, to specialized protocols.

This methodology leads us to modular implementation, avoiding complex reasoning on a full protocol, allowing designers to focus on relevant variables within a certain abstraction level, without taking into account the complexities of pre-and post-levels; following a Top-Down and modular approach to software design and development (Kendall & Kendall, 2008).

We focused in this work on the first layer (application) of the seven-layer ISO/OSI Model (Gollman, 2006); based on its provision of a useful abstraction to discuss security in data transmission networks, and because it allows us to design services tailored to a specific need, reducing complexity,

standardizing interfaces, facilitating modular engineering and allowing interoperability between layers, which are the main advantages of ISO/OSI model (Gollman, 2006; Holzmann, 1991; Stallings, 2006).

According to literature related to specifications we found that it has been difficult to specify the desired properties in existing formalisms, because formal methods can express operation, but not typical security goals such as confidentiality, authentication, non-repudiation and integrity; in this work we involve these goals on each abstraction level. The presented methodology complements the conformed protocol (sequence of messages) with clear specifications and explanations of other forms coming from the detailed organization of the design principles. Once protocol specifications are made, the analysis and design continues: these processes consist of recurring activities of design on each abstraction level, until we reach the final product.

One limitation of this work is the restriction on two participants, or three if a trusted third party is needed. Another originates from the requirement of successive refinements: the designer that uses it must have a close relationship with clients in order to extract all their relevant needs, goals and constraints. Design principles were selected from current literature, when new principles, applications or technological developments arise that affect protocol's internal or operational environment, it might be necessary to review this methodology.

Further research emerges from this work, in the automation of the methodology, so that designer's time and resources may be used more efficiently, while assuring that sufficient assumptions are provided to specify and construct a model in a more dynamic environment. As we consider only four security goals, fulfillment of others requires additional work.

References:

Abadi, M., & Needham, R. (1996). Prudent Engineering Practice for Cryptographic Protocols. *IEEE Transactions on Software Engineering, 22*(1), 6-15.

Abadi, M., & Rogaway, P. (2002). Reconciling the Two Views of Cryptography (The Computational Soundness of Formal Encryption). *Journal of Cryptology, 15*(2), 103-127.

Anderson, R., & Needham, R. (1995). *Robustness Principles for Public Key Protocols*. Paper presented at the Advances in Cryptology CRYPTO'95.

Aura, T. (1997). *Strategies Against Reply Attacks*. Paper presented at the 10th. IEEE Computer Security Foundations Workshop CSFW'97, Rockport, MA.

AVISPA. (n.d.). Automated Validation of Internet Security Protocols and Applications. Retrieved from http://www.avispa-project.org/library/avispa-library-index.html

Carlsen, U. (1994). *Cryptographic Protocols Flaws. Know Your Enemy*. Paper presented at the Computer Security Foundations Workshop CSFW'94, Los Alamitos, CA.

Cervesato, I. (2001). The Dolev-Yao Intruder is the Most Powerful Attacker: Advanced Engineering and Sciences Division, ITT Industries.

Checkland, P. B. (1991). *Systems Thinking, Systems Practice*. New York, NY: John Wiley.

Choi, H. J. (2006). Security Protocol Design by Composition *Technical Report 657 (UCAM-CL-TR-657)*. Cambridge, UK: University of Cambridge.

Doghmi, S. F., Guttman, J. D., & Thayer-Fábrega, F. J. (2007). *Completeness of the Authentication Tests.* Paper presented at the 12th. European Symposium Research Computer Security ESORICS-07.

Dolev, D., & Yao, A. C. (1983, March). On the Security of Public Key Protocols. *IEEE Transactions on Information Theory, 2*, 198-208.

Dong, L., Chen, K., Wen, M., & Zheng, Y. (2007). *Protocol Engineering Principles for Cryptographic Protocols Design.* Paper presented at the 8Th, ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing.

Fidge, C. J. (2001). *A Survey of Verification Techniques for Security Protocols*. Queensland, AU: The University of Queensland.

Gollman, D. (2006). *Computer Security* (2 ed.). New York, NY: John Wiley.

Gritzalis, S., Spinellis, D., & Georgiadis, P. (1999). Security Protocols Over Open Networks and Distributed Systems: Formal Methods for their Analysis, Design and Verification. *Computer Communications, 22*(8), 695-707.

Guttman, J. D. (2002). *Security Protocol Design Via Authentication Tests.* Paper presented at the 15th. IEEE Computer Security Foundations Workshop CSFW'02.

Guttman, J. D., & Thayer-Fábrega, F. J. (2002). Authentication Tests and the Structure of Bundles. *Theoretical Computer Science, 285*(2), 333-380.

Holzmann, G. J. (1991). *Design and Validation of Computer Protocols*. Englewood Cliffs, NJ: Prentice Hall Software Series.

ISTF. (2013). Official Internet Protocol Standards *Internet Standards Task Force* Retrieved from http://www.rfc-editor.org/rfcxx00.html

Kendall, K. E., & Kendall, J. E. (2008). Software Quality Assurance *Systems Analysis and Design*. New York, NY: Pearson Prentice Hall.

Khurana, H., Bobba, R., Yardley, T., Agarwal, P., & Heine, E. (2010). *Design Principles for Power Grid Cyber-Infrastructure Authentication*

Protocols. Paper presented at the 43rd. Annual Hawaii International Conference on System Sciences, Honolulu, HI.
Malladi, S. (2008). *Prudent Engineering Practice to Prevent Type-Flaw Attacks under Algebraic Properties*. Madison, SD: South Dakota State University.
Oceanasek, P., & Sveda, M. (2006). *An Approach to Automated Design of Security Protocols.* Paper presented at the International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies ICNICONSMCL '06, Brno, CZ.
SPORE. (n.d.). Security Protocols Open Repository. Retrieved from http://www.lsv.ens-cachan.fr/Software/spore/
Stallings, W. (2006). *Cryptography and Network Security. Principles and Practices* (4 ed.). New York, NY: Pearson Prentice Hall.
Syverson, P. (1996). *Limitations on Design Principles for Public Key Protocols.* Paper presented at the IEEE Symposium on Security and Privacy, Oakland, CA.
Van Gigch, J. P. (1974). *Applied General Systems Theory*. New York, NY: Harper & Row.
Wilson, B. (1992). *Systems: Concepts, Methodologies and Applications* (2 ed.). New York, NY: John Wiley.
Xianxian, L., Hun, J., & Zhaohao, S. (2004). Design Principles and Security of Authentication Protocols with Trusted Third Party. *AUUG The Organization for Unix, Linux and Open Source Professionals.* Retrieved from http://www.auug.org.au/resources/proceedings/auug2004/li.shtml
Zheng, H. (2000). *Engineering Principles for Designing Secure Protocols*. Helsinki, FI: University of Helsinki, Department of Computer Science.