Contents

List of Figures

List of Tables

Abbreviations

ADVSIG	Advanced Signature
ASYM	Asymmetric
BS	Boosting Signal
CC	Channel Confidentiality
CO	Channel Occupancy
COA	Channel Origin Authentication
CS	Commitment Scheme
DC	Data Confidentiality
DE	Direct Logical Link Test Edge
DH	Diffie Hellman
DOA	Data Origin Authentication
DRP	Dropping
EAP-TLS	The Protected Extensible Authentication Protocol
ESP	Extended Strand Spaces
GDOI	Group Domain of Interpretation
GPS	Global Positioning System
GSAKMP	Group Secure Association Key Management Protocol
HF	Hash Function
HTTPS	Secure Hypertext Transfer Protocol
HIP-DEX	Host Identity Protocol Diet EXchange
MAC	Message Authentication Code
MIKEY	Multimedia Internet KEYing
ID	Identification
IoT	Internet of Thing
LR	Long Range
MANA	Manual Authentication
MITM	Man-In-The-Middle
N	Nonce
ND	Neighbour Discovery
NDP	Neighbour Discovery Protocol
NFC	Near Field Communication
NS	Needham Schroeder
OLSR	Optimised Link State Routing Protocol
OOB	Out-of-Band
OVH	Overhearing
PANA	Protocol for Carrying Authentication for Network Access
PKI	Public Keys Infrastructure

RADIUS	Remote Authentication Dial-In User Service
REL	Releasing
RPL	Replaying
RFID	Radio Frequency Identification
RF	Radio Frequency
XOR	Exclusive-OR operation
SAS	Short Authentication String
SMS	Short Message Service
SP	Original Strand Spaces
SR	Short Range
SRL	Single Relay
SYM	Symmetric
SUS	Suspending
TLS	Transport Layer Security
URN	Uniform Resource Name
WLAN	Wireless Local Network
WS	Wong-Stajano
6LoWPAN	IPv6 over Low power Wireless Personal Area Networks
L_HASH	Long Hash
S_HASH	Short Hash

List of Notations

$\mathcal{A}$	Set of Terms
$\mathcal{T}$	Set of predictable texts
$\mathcal{R}$	Set of unpredictable texts
$\mathcal{K}$	Set of keys
$\mathcal{D}$	Set of DH values
$\mathcal{K}_{Sym}$	Set of symmetric keys
$\mathcal{K}_{Ver}$	Set of verification keys
$\mathcal{B}$	Bundle
t	term
$\sqsubseteq$	Subterm relation
Σ	Strand Spaces
$\mathcal{P}$	Penetrator Strand
D_X	Set of DH values of entity X
st	Strand
$\preceq$	Partial order relation
o	channel

$--$	Secure Channel
$\#_o t$	Delaying term t on a channel o
l_n	Location of node n
R_n	Signal range of node n
τ_n	Timestamp of node n
$loc(st)$	Location of strand st
$dist(n, n')$	Distance between node n and n'
$dist(st, st')$	Distance between two strands st and st'
δ_{tp}	A processing delay time
$n_1 \rightarrow *n_2$	A path from node n_1 to n_2
v_c	Speed of signal in a specific environment
$\|a, b\|$	Distance between location a and location b
δ_{max}	Maximum interval
δ_{tp}	Processing delay
$plink(st, st', \rightharpoonup)$	Unidirectional physical link between two strands st and st'
$link(st, st', \rightharpoonup)$	Unidirectional logical link between two strands st and st'
$plink(st, st', \rightleftharpoons)$	Bidirectional physical link between two strands st and st'
$link(st, st', \rightleftharpoons)$	Bidirectional logical link between two strands st and st'
idf	Identification factor
$\mathcal{X}^{ESP}$	Penetrator strands in the extended Strand Spaces
$\mathcal{X}^{SP}$	Penetrator strands in the original Strand Spaces
$CS(r_a, r_b)$	A commitment scheme for a random pair (r_a, r_b)
$PR[f]$	Probability of the function f
$\mathcal{B}_o^{SP}(m, i)$	A transforming shape of OOB message $_o(m)$, i index the in an original protocol
$sk_o^r(m, i)$	a Initiator skeleton in $\mathcal{B}_o^{SP}(m, i)$
$sk_o^r(m)$	a Responder skeleton in $\mathcal{B}_o^{SP}(m, i)$
$_{lrp}(m)$	Message m on a long-range public channel
$_{srp}(m)$	Message m on a short-range public channel
$_{pro}(m)$	Message m on a protected public channel
$_{pri}(m)$	Message m on a private public channel
Aice(A) and Bob(B)	Communicating entities
ID_X	Device X identifier
r_x	Random number generated by entity X
k_X	Key generated by entity X

ks	Shared key
PK_X	Public key of entity X
PR_X	Private key of entity X
K	Key
m	Message/Data
m_x	Message/Data created by entity X
$MAC_k(m)$	Message authentication code of message m using shared key K
$h_k(m)$	Universal hash function of message m using the key K
$\{m\}_k$	Message m encrypted with key k
$h(m)$	One-way hash function of message m, or long-hash function
$S\_h(m)$	One-way short hash function of message m
,	Concatenation of different parts of a message
$\otimes$	Bitwise "XOR" operation
x	x in g^x is a secret value of Diffie-Hellman public key of entity X
g^x	Diffie-Hellman public key of entity X
g^{xy}	Diffie-Hellman shared key between X and Y
$(c, d) \leftarrow commit(m)$	Commitment algorithm takes m to produce commit value c and decommit d
$m \leftarrow open(c, d)$	Open commitment algorithm take c, d and produces a message m or error signal
c_X	Commitment value computed by entity X
d_X	Decommitment value computed by entity X
SAS_X	Short Authenticated String computed by entity X
Q_X	A number of instances of entity X in the network
MS_ID	32-bit unique manufactory identification.
Dev_ID	32-bit device identification.
MS_FR	manufactory fingerprinting used by third-party services to verify manufactories.
Dev_Status	the status of a device including sold, registered, unregistered, or brand-new
Dev_Desc	information describing device specification.
Dev_Key	the device DH key generated in secure pairing process
Dev_URN	the unique URN assigned by the manufactory to uniquely identify the device.
Dev_Addr	the network address of the device

Expires	the expiration date of device fingerprinting
PR_{SH}	the home gateway's private key
PB_{SH}	the home gateway's public key
Dev_FR	device fingerprinting formed as $\{Dev\_Key, Expires\}_{PR_{SH}}$
Dev_Info	the device information including MS_ID, Dev_ID, MS_FR, and Dev_URN
SH_Addr	the network address of the home gateway
Net_ID	the home network ID
Net_CONF	the network settings including Net_ID, SH_Addr provided by the home gateway
NetPass	the network group key including $\{Key, ID\}_{PR_{SH}}$ where ID is a sequence number of the Key
ks	the shared key conducted by two DH keys.

Chapter 1

Introduction

The term "Internet of Things" (IoT) is defined for such a huge picture as millions number of connected devices cooperating to accomplish some specific tasks required by users. Devices for instance are mobile phones, smart TVs, smart lights, fans, and etc that are normally constrained in resources. By the year 2020, it is expected to 16 billion interconnected devices [1]. Hence, applications of IoT are truly large and potential in both research and industrial areas.

According to the work [2], the properties of the Internet of Thing are usually defined as the uncontrolled environment, the heterogeneity, the need for scalability, and constrained resources:

- The *uncontrolled environment* is a place where many devices travel to untrustworthy surroundings.
- The *heterogeneity* is described that various devices from various manufactories can interoperate together.
- *Scalability* is demanded for scalable systems consisting of a vast amount of interconnected devices.
- *Constrained resources* in power capacity, computational capacity, memory and interfaces are normally found on devices in IoT.

These properties associates to major challenges in IoT, which could be named as low energy consumption requirement, limited radio frequency bandwidth requirement, and security requirement. To address these challenges, IoT fans are trying to propose their own new concepts of technologies including documents, schemes, and protocols. But, many aspects of IoT have not been standardised yet, especially regarding to secure mechanisms.

Secure mechanism aims to ensure things working properly in environments with presence of many adversaries. As one of main parts of secure mechanism, security protocols (also called cryptographic protocols) provide goals (or properties) such as secrecy (data is transferred such that only an intended receiver is able to understand) and authentication (providing the proof of origin of data to remote principals).

In this first chapter, we offer a brief introduction of secure physical communication for IoT, the need for lightweight secure mechanisms, and formal verification. We also give an overview of the main contribution of our work. We close this chapter with the outline of this thesis.

1.1 Motivation

For example, creating security domains from unassociated constrained devices is a key operation in the IoT network. Playing as a crucial role in IoT, device pairing protocols are responsible for two non-prior knowledge wireless devices to establish a secure connection. However, it is formally shown that pairing goals could not be offered by just cryptographic primitives [3]. To provide solution, a pre-authenticated auxiliary channel, human assisted or location limited, usually called out-of-band(OOB) channel is used. Thus, a great number of device pairing protocols with various OOB channels as documented in [4] have been introduced. Despite of that, many of them feature some flaws, e.g the Wong-Stajano protocol [5] as an instance. Additionally, they are currently not sufficiently effective on security requirement, and low bandwidth networks due to constructing on high secure channels, and large amount of exchange data.

Another important family of protocols in IoT is neighbour discovery protocols. Theoretically, they are designed to allow each participant to correctly identify other participants who are actual neighbours. Hence, discovery mechanisms fundamentally consider location information or even wireless signal range of each principal. However, wireless devices in current proposals are normally assumed to have the same physical wireless interfaces, this is not always true in practice. As a consequence, security flaws appear in some existing protocols such as ADVSIG [6] and Brands and Chaum protocol [7].

So far, we need both a more effective and provable security mechanisms and methods that allow us to avoid flaws as early as possible before our protocols are deployed. Formal methods are introduced as well-suited tools for our needs to reduce flaws at the protocol design step.

Reasoning about security properties for wireless protocols, a number of existing work have been proposed in literature. Interestingly, most of them are extended work of classic formal models such as BAN logic [8], inductive approach [9], authentication logic [10], deductive model checking [11], Petri Nets [12], simulation paradigm [13], Spi calculus [14], and Strand Spaces model [15].

Thank to these models, a wide range of protocols has been formally analysed. For instance, MANET routing problems have been studied in [13, 16–23] while neighbour discovery and distance bounding problems have been considered in [10, 24–28]

As we mentioned above, despite of helpfulness on analysing cryptographic protocols, classical formal methods were just designed for classical security properties such as data origin authentication and secrecy. For this reason, they are not suitable for reasoning about physical properties. In meanwhile, some existing extensions concerned on several aspects of physical properties in literature, but their attacker model is mainly based on classical and strong Dolev-Yao model [29]. Hence, in some cases, attackers can not be visually conducted, e.g, using a high power antenna to lift up the signal propagation distance, an attacker can persuade a victim to believe existence of connections between them.

1.2 Reseach Objective

The main objective of this PhD research is to study security mechanisms in context of Internet of Things, particularly in secure device pairing and secure neighbour discovery. This objective encompasses the following challenges which are to be specifically addressed:

- The design of security mechanisms should answer the constrained characteristics of devices in Internet of Things. For this purposes, the mechanisms would be effective in term of communication and computation, robustness.
- Security of proposed mechanisms must be against malicious physical attacks such as relaying, delaying, replaying, spoofing.
- The developed key agreements and link agreements between devices, and their accompanying security framework must be validated using formal methods to avoid undesired attacks.

In addition to the main goals, a straightforward, robust formal model analysing a wide range of secure wireless protocols facilitates reasoning about both cryptographic properties and physical properties. Physical attacks should be addressed in the model as well.

Worthy to note, the proposed formal model would fulfil our requirements:

1. the model is straightforward and robust;
2. the model has some facilities to enable reasoning about physical goals; or it is feasible to integrate extensions without heavy modification of core theories;
3. physical attacks must be considered in the core model;
4. the model is able to visually produce attack scenarios if such scenarios exist;
5. the model can be potentially deployed in an automatic verification tool.

1.3 Contribution

In this thesis, we focus on crucial aspects for security of wireless protocols: effectiveness, physical security properties, and formal verification. Our contributions are following:

1. We introduce a new device pairing protocol that is more secure and efficient than other competitors in term of communication cost, and remains the same attack probability. Then, as a proof of concept, we implement our protocol in an embedded system to show its usefulness.
2. We build our formalism based on the famous Strand Spaces model to capture the physical security characteristic of out-of-band channels. The adversary capabilities are also extended on these channels. Thank to our model, a flaw is discovered in Wong-Stajano, that has not introduced before. Additionally, we propose a procedure that transforms a model in our formalism of an initial protocol with out-of-band channels into a model in original Strand Spaces of a protocol that does not use any OOB channel while preserving security properties of initial protocol.
3. We make a comprehensive study on neighbour discovery protocol. Then, we find out a problem when signal ranges of two principals are different. As a consequence, we point out that time-based, or distance-based mechanisms cannot provide exact link agreement among principals. Apparently, neighbour discovery protocols using these techniques are proved as incorrect protocols.

4. Based on above mentioned enhanced security protocols and our formal method, we introduce a new concept of the secure bootstrapping scheme for Internet of Things that enables a resource constrained thing as a new member to securely join into a home network in circumstances where the home gateway is down, or the thing is second-handed. Furthermore, our scheme does not require pre-shared keys, or public keys, or even does not require a PKI infrastructure. Formal proofs of security properties are given as well.

1.4 Thesis Outline

This thesis is divided into 6 chapters with chapter 1 being this introduction. In chapter 2, we study a family of secure device pairing protocols using out-of-band channel. We give an detail of current device pairing approaches and discuss the different aspects of them. We also in this chapter propose our novel key agreement protocol using out-of-band channel. As proof of concept, an implementation of this protocol is deployed into two embedded systems. Chapter 3 is devoted to analyse formally security properties of secure device pairing protocols. We present our improved Strand Spaces theory, Wong-Stajano protocol flaw, and proof of our protocol. We also present a way to translate a protocol modelled in our extended Strand Spaces into a protocol modelled in original Strand Spaces without out-of-band channels. Chapter 4 studies neighbour discovery protocols and formal analysis of these protocols. We continue using Strand Space model as our tool in this chapter. In chapter 5, we propose a new secure bootstrapping scheme for constrained devices in Internet of Things. Chapter 6 concludes our thesis and presents future work.

Chapter 2

Secure Device Pairing Protocols

The need to secure communications between personal devices is increasing nowadays, especially in the context of Internet of Things. Authentication between devices which have no prior common knowledge is a challenging problem. One solution consists in using a pre-authenticated auxiliary channel, human assisted or location limited, usually called out-of-band channel. A large number of device pairing protocols using an out-of-band channel were proposed. However most of these proposals lacks of proofs, and therefore may be vulnerable to some attacks. Additionally, current approaches are not sufficiently convenient for IoT applications where the devices are strictly constrained, and network bandwidth is too expensive.

In this part, we study in depth current secure device pairing protocols. We found that current approaches are not effective in term of computation and communication for Internet of Things applications. Therefore, we introduce a new key agreement protocol between two wireless devices. This protocol, only using two wireless messages and one out-of-band message, offers better communication costs than currently existing solutions, yet still ensuring a reasonable security. Security of our proposal is validated by estimation of attack success probability in a computational model.

The chapter begins with a short introduction to out-of-band channels, and existing device pairing schemes. Then, our novel device pairing will come after that. Finally, we are willing to introduce flaws we found in some current approaches.

2.1 Out-of-Band Channels

Securing wireless communication is establishing an initial trust relation between non associated devices. Such a trust initialisation process is commonly called either *Secure*

Device Pairing, or *Secure Bootstrapping*, or *Secure First Connect*. Due to heterogeneity of devices and lack of official standards, no existing security infrastructure or schemes could provide a universal solution for this task. Additionally, unfamiliar devices with no common trust cannot take advantage from traditional cryptographic protocols (i.e. authenticated key exchange protocols) when there does not exist any pre-shared secret, or authenticated public keys.

Trying to solve this problem, a great body of work proposes some forms of human involvement in secure pairing process. This human involvement is achieved by using an auxiliary channel between the devices that is both observable and controllable by the owner of the devices. This auxiliary channel received various names such as *out-of-band channel*, or *human-assisted channel*, or *manual channel*. In this thesis, we adopt the general term out-of-band channel (OOB).

2.1.1 OOB Security Properties

One easily misunderstands concept of security properties of data or messages versus security properties related to a physical channel because they sometimes overlap. Data security properties are usually ensured using cryptographic primitives such as symmetric or asymmetric encryption algorithms, signatures or hash functions. A secure physical channel not only provides data security properties without help of cryptographic mechanisms, but also physical security properties such as stall-free, listener-ready, non-forwarding, time and distance constraint guarantees and so on. In this chapter, we consider the following security properties, where S and R denote principals, m a message, o a channel, T is an interval of time:

- *Data Origin Authentication (DOA)*: Data origin authentication is also called *message authentication*. Let m be a message originally created by S, then any receiver of m is able to authenticate S as an original source of the message. It means that in a particular penetrator cannot modify m; thus message authentication includes message integrity. However, a penetrator may block or replay m.

- *Data Confidentiality (DC)*: If a sender determines that only a specified R can observe content of a message m, then no one including penetrators, excepted R, is allowed to know the content of m.

- *Channel Origin Authentication (COA)*: Only a specified sender S can use a channel o to send messages and this fact is known to determined receivers. Thus, a penetrator cannot impersonate S on o. However, a penetrator can suspend message transmission to know messages before they reach their desired destination.

- *Channel Confidentiality (CC)*: Only a specified receiver R can receive messages on a channel o, and this fact is known to determined senders. Thus, a penetrator cannot impersonate R to receive message on o. A penetrator cannot overhear messages sent on o.
- *Channel Occupancy (CO)*: If a specific receiver R uses a channel o to communicate with someone during interval of time T, then there is indeed a sender using o with R during T. A penetrator cannot manipulate o during T if o is not exclusively used by the penetrator.

Note that, channel occupancy allows participants to ensure distance and presence of their protocol partners. Straightforwardly, penetrators cannot use suspending attack on messages over such channels.

Both the channel origin authentication and channel confidentiality properties were initially defined in[30]. The definitions above overlap: channel confidentiality implies data confidentiality and channel origin authentication implies data origin authentication. The definition of channel occupancy is an adaptation of locale occupancy property introduced in [31].

2.1.2 OOB Classification

In this subsection, we classify out-of-band channels according to their physical, but also the security properties they offer

2.1.2.1 Physical Types of Channels

Following the classification from [32], existing OOB channels can be grouped into categories depending on their physical characteristics: cable-based channels, audio-based channels, visual-based channels, tactile-based channels, motion-based channels, biometric-based channels, wireless-based channels, or channels based on a combination of previous types.

Cable-based

A cable-based connection is used in the *Resurrecting duckling policy* model proposed in [33] to map relationship between devices. A master device, so-called "Mother", imprints "duckling" slave devices. The slaves is either imprinted or imprint-able. Imprint-able state is the beginning state of a slave before it is chosen by a master. In meanwhile, the imprinted state is once a slave has got a secret from a master. The imprinted process actually bounds a slave to a master until the slave's death. As a consequence, the slave remains faithful to its master and obeys no one else. Because the secret key needs to be transferred from a master to a slave, the authors suggest that it could be sent in plain text over a physical connection (such as cable). Complex and heavy cryptographic key exchange like DH scheme is not recommended. This thing, hence, is not convenient in practice. Nevertheless, this approach takes advantages of minimal requirement of human interaction in authentication phase.

Audio-based

Audio channels could be used as secure channels. The work [34–39] proposes ideas to encode cryptographic materials into nonsensical audio sentences. Then after transmitted from a speaker to a microphone, the sentences are reconstructed into the cryptographic materials at the target device. User takes responsibility for comparing results and deciding the pairing.

Audio-based schemes normally require some kinds of physical interfaces such as speaker, microphones. But, these interfaces are often suffered from denial of services or noise environments. For instance, ambient noise in crowded environments (e.g. in subway, in airport, or in bars) makes the authentication either weaker or difficult in speaker-to-speaker, as well as in others. Moreover, handicap users are not suitable for these schemes.

Recently, researchers have made improvement on both security levels and usability, thank to advanced speech engines and audio codec technologies.

Visual-based

Using image comparison to set up a secure channel between devices has appeared early in literature. Precisely, cryptographic materials are encoded into images, and ask users to compare them on two devices. Approaches chose this method such as [40–44]. Despite of the requirement of a high resolution screen on each device, these approaches stated that screens are easily found in current laptops, PDAs, smartphones, and etc.

Visual-based schemes also share the limitations of audio-based ones on hardware requirements. Furthermore, cameras are sometimes strictly prohibited in high security areas such as military zones or bank offices, and barcodes do not work in low light conditions.

Tactile-based

BEDA [45], proposed by Soriente et. al, presents ways to transmit a secret among devices using very basic interfaces like buttons. There are four BEDA variants: Button-to-Button, Display to Button, Short Vibration to Button, and Long Vibration to Button. The only difference of these variants is the way a first device transfers a secret to others.

To transmit the secret code, two approaches are suggested. In the first approach, both devices get the same secret via the use of single button. In the Button-Button approach, the user simultaneously presses and then releases buttons on both devices until the secret is acquired. In Display to Button approach, a device equipped with an output interface signals the user to press a button on the other device. Then, idle time between two pushing actions are is to calculate the secret.

Wireless-based

In order to establish a secure channel, some types of wireless channels such as infra-red [46], ultrasound [47], RFID [48], Bluetooth, and NFC could be used. Talking to Stranger and other its variants [46–49] are examples. However, a drawback of those schemes is that they are strongly suffered by denial of service attacks or passive eaves-dropping attacks.

Pairing scheme using Bluetooth demands 4-digits pre-shared PIN code between two devices. Unfortunately, adversaries can guest and break the PIN code from long distance. As an alternative method, NFC, an extremely short communication, is concerned on solving limitations of Bluetooth and infra-red. In many scenarios, NFC combines with Bluetooth to offer quicker setup, and better security. Nevertheless, NFC does not provide any protection against eavesdropping attacks, as well as data corruption and data modification. In spite of that, it is hard to launch MITM attack in NFC authentication session. As this reason, NFC is still considered safety for current applications.

Biometric-based

A first work which tried to apply biometric data to establish a secure channel was found in Feeling-is-Believing [50] in which authors proposed a method to share secret key using authenticated biometric information.

In users' point of view, biometric-based schemes sound more secure and usable than others. But in practice, biometric processing is not sufficiently accurate and requires more calculation cost. To overcome this obstacle, thank to advanced technologies, the accuracy of recognition is significantly improved in many commercial devices such as modern smartphones and laptops. The only drawback of these such schemes is that biometric scanners must be equipped on both devices.

Accelerometer/Motion-based

Common uses of accelerometer include detecting and monitoring vibration, and detecting magnitude and direction. A first work using accelerometers for pairing devices was found in Smart-its-Friends scheme [51] in which two intended devices are held and shaken together simultaneously. By this way, the sensing information collected from accelerometers allows them to establish a common communication channel. Other variant approaches are [52–57].

A drawback of these approaches is that embedded accelerometers sometimes are not always easy to be deployed in big devices such as printers, projectors or laptops.

Combination

Claude Castelluccia and Pars Mutaf introduced *Shake Them Up* [58] to allow two constrained devices to share a secret in minimal requirements of both hardware and out-of-band channel. In this scheme, both involved devices are required to shake and twirl together in close proximity. During shaking process, both devices also exchange radio packets. When finishing the steps, they together get the same secret key. Attackers cannot interfere key exchange process because they cannot determine source of each radio packet due to the source indistinguishability achieved by CDMA-based system, and sharking devices in close proximity.

Varshavsky et al. introduced AMIGO [59] investigated that radio signal fluctuations is too hard to predict at a specific location and time. With this result, the authors pointed that any attacker who is not physically close to the signal source would see a different pattern of signal strength.

2.1.2.2 Channel Security Properties

The types of OOB channels can be also classified by security properties. We refer the classification in the work [4], and adapt each type with the definitions of channel security properties. In addition to, we divide the public channel into sub-public channels.

- a *private channel* ensures all channel security properties;
- a *protected channel* ensures channel origin authentication and channel confidentiality but not channel occupancy;
- a *public channel* ensures channel origin authentication but not channel confidentiality;

Additionally we classify public channels into two sub-types:

TABLE 2.1: OUT-OF-BAND CHANNEL CLASSIFICATION

OOB Channel Type	CO	COA	CC	Examples
Private	√	√	√	Cable
Protected	Ø	√	√	SMS, Encrypted email
Short-range Public	√	√	Ø	Button, Vibration, RFID, NFC...
Long-range Public	Ø	√	Ø	Screen to camera, Speaker to speaker, ...
Insecure	Ø	Ø	Ø	WIFI

TABLE 2.2: THREATS ON OUT-OF-BAND CHANNELS

Out of Band Channel	Attacker's Power			
Type	Overhear	Block	Suspend	Replay
Private	Ø	Ø	Ø	Ø
Protected	Ø	√	√	Ø
Short-range Public	√	√	Ø	Ø
Long-range Public	√	√	√	√

- a *short-range(SR) public channel* is a public channel offering channel occupancy,
- a *long-range(LR) public channel* is a public channel not offering channel occupancy.

A private channel could be established for instance by connecting a cable between two devices. A protected channel could be set by using a tactile based technique like authenticated server SMS, authenticated emails. A short-range public channel can be offered by motion-based techniques or NFC. A long-range public channel can use a visual based or sound-based technique. Table 2.1 summaries a comparison of OOB channel and give some examples for each type of OOB channels.

2.1.3 OOB Penetrator Model

As described in previous subsections, the penetrator is prevented from performing actions on private OOB channels, yet he can still launch some malicious actions on public or protected OOB channels. Table 2.2 presents the penetrator power for each kind of OOB channels.

In the table 2.2, overhearing means that attackers are capable knowing OOB messages when they are being transmitted. Suspending means that attackers are able to suspend sending events. Especially, suspending attacker can completely know message's content before messages are transmitted on a public OOB channel. Blocking means that attackers can drop any message over an OOB channel. Replaying means attackers are capable replaying OOB messages.

The difference between types of OOB channels and penetrator power for each kind is summarised in table 2.3.

Table 2.3: Out-of-Band Channel Summarization

Out of Band Chanel			Attacker's Power			
Pairing Method	Interface	OOB Type	Overhear	Block	Suspend	Relay
Resurrecting Duckling Model	Cable	Private	∅	∅	∅	∅
Motion-based Model	Accelerometer	SR Public	√	√	∅	∅
BEDA Methods						
☐ Button-Button	Button	SR Public	√	√	∅	∅
☐ Display-Button	Display, Button	SR Public	√	√	∅	∅
☐ Vibration-Button	Accelerometer, Button	SR Public	√	√	∅	∅
Audio-based Methods						
☐ Audio Context Recognition	Speaker, Microphone	LR Public	√	√	√	√
☐ Speaker-Speacker	Speaker	LR Public	√	√	√	√
☐ Speaker-Microphone	Speaker, Microphone	LR Public	√	√	√	√
Visual-based Methods						
☐ Barcode-Camera	Barcode, Camera	LR Public	√	√	√	√
☐ Visual Comparison	Screen	LR Public	√	√	√	√
☐ Blinking Light	LED light	LR Public	√	√	√	√
Wireless-based Methods	Wireless Interface					
☐ GPRS/3G/4G		LR Public	√	√	√	√
☐ Infrared		SR Public	√	√	∅	∅
☐ NFC		SR Public	√	√	∅	∅

2.2 Overview on Secure Device Pairing Schemes

We refer the survey [4] and sum up existing device pairing proposals. Then, we point out their limitations. To begin with, we describe some notations used to picturize the protocol as follows:

- An arrow shows direction of each message. A solid line represents for an unsecured channel, while a dotted line represents for a secured (authenticated) channel.
- When *Bob* or *Alice* receives a message over a insecure channel, this message could be altered by attackers. An apostrophe appears on a letter, e.g. A' or m', it means that the received message may be different from the sent one.

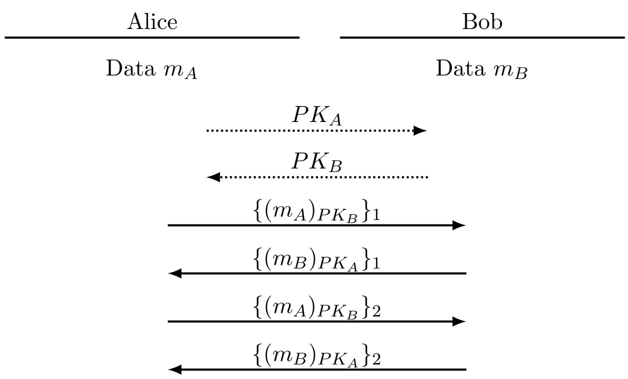
A protocol utilises commitment schemes which commit on an arbitrary non-hidden message m together with a hidden k-bit string r. These schemes are formalised by three algorithms.

- **commit(m)** takes a message m and produces two strings: a commit value c and decommit value d.
- **open(c,d)** takes m or an error signal.

2.2.0.1 Interlock Protocol

Proposed by Rivest and Shamir in 1984, Interlock protocol exploits user's knowledge of communication pattern or voice recognition as a long-range public out-of-band channel. In this scheme, two parties use their initial knowledge to mutually authenticate their public keys without any assistance of a third party. This protocol works as follows:

1. *Alice* and *Bob* exchange their public keys via a sound channel.
2. *Alice* produces $(m_A)_{PK_B}$ then sends the first half of the result $\{(m_A)_{PK_B}\}_1$ to *Bob*.
3. *Bob* produces $(m_B)_{PK_A}$ then sends the first half of the result $\{(m_B)_{PK_A}\}_1$ to *Alice*.
4. *Alice* sends to *Bob* the second half $\{(m_A)_{PK_B}\}_2$.
5. *Bob* sends to *Alice* the second half $\{(m_B)_{PK_A}\}_2$.
6. Both decrypt the combination of two halves with their own private key.

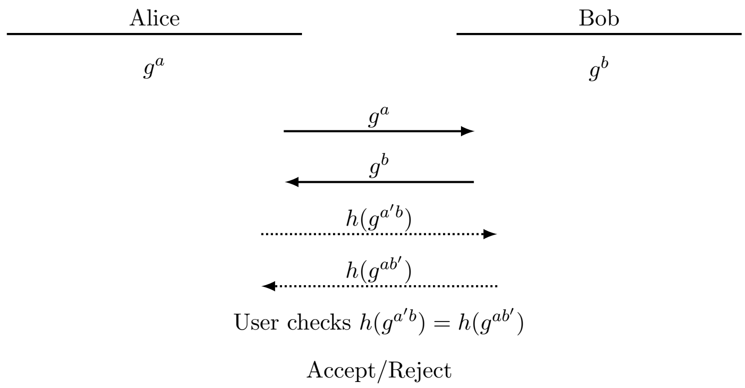


The strength of this protocol basically lies on encryption algorithms with which the attacker cannot decrypt the received halves of encrypted messages. In case of that the attacker intentionally send some new messages to destroy the communication, he will be revealed.

2.2.0.2 Maher Manual Authentication

In 1993, Maher got his patent for several pairing methods [60] allowing a user to share secret DH key manually. In one method, he applied a compression function(hash function) to interpret DH key g^{ab} into 4-digit number. The number than is showed on each device screen. A user gets easy to compare two numbers on both devices. The OOB channel in this work is regarded as a short-range public OOB channel. The protocol simply happens as follows:

1. *Alice* and *Bob* generate DH key g^a and g^b respectively.
2. *Alice* and *Bob* exchange their values.
3. Both calculate $h(g^{ab})$ to 4-digit hex number, and show the result on their device's screen.
4. The user decides accept or reject by pushing a button.

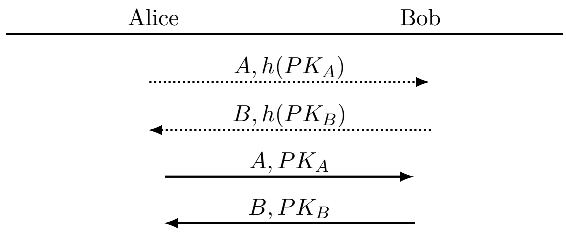


The protocol requires at least 80 bits over an out-of-band channel to get enough secure. At a result, this length of bit string is over-weighted on any out-of-band channel.

2.2.0.3 Talking to Strangers

Balfanz et al. proposed a new scheme in [61] using audio, or infrared channels to transmit a hash of public keys which is considered as a fingerprinting or digest value. The protocol works as follows:

1. *Alice* and *Bob* initially exchange their identifications and a hash of their pubic keys over an out-of-band channel.
2. Both sides exchange their ID and their public key on insecure channel.

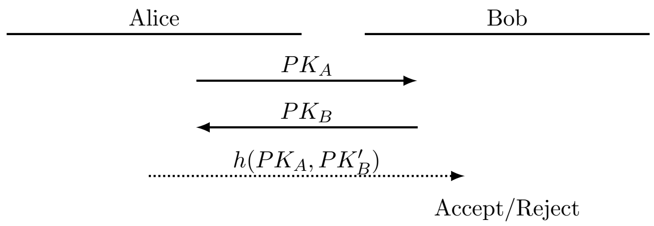


Security of this protocol mainly depends on security properties of out-of-band channels and the hash function. To prevent MITM attack, this protocol needs at least 80 bits information over each direction of OOB channel. Otherwise, the attacker can find out the pair of public keys PK'_A/PK'_B in which $h(PK_A) = h(PK'_A)$ and $h(PK_B) = h(PK'_B)$ to launches a MITM attack.

2.2.0.4 Visual authentication based on Integrity Checking

Saxena et al in [44] proposed a new pairing protocol, namely Visual authentication based on Integrity Checking(VIC) which works as follows:

1. *Alice* and *Bob* initially exchange their pubic keys.
2. *Alice* hashes both keys, and sends the hashing value to *Bob*.
3. After verification the *Alice*′s hashing value, *Bob* informs Accept or Reject back to *Alice*.



Security of VIC heavily depends on strength of the hash function and requires at least 80 bits on each OOB channel to prevent hash collision.

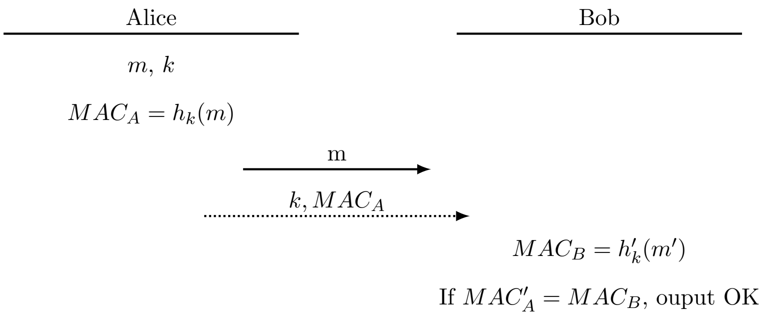
2.2.0.5 Manual Authentication(MANA) Protocols

Due to the long length of OOB messages in [44], and [61], some studies tried to truncate this length into 16 or 32 bits (4 or 8 hexadecimal digits, respectively), this could lead to security weakness. Adversaries probably recover the messages from the hash-codes. To overcome this problem, Christian Gehramann and Chirs J.Michell [62] proposed three type of manual authentication protocols: MANA I for Output-Input, MANA II for Output-Output, and MANA III for Input-Input. These protocols remarkably reduce bandwidth of OOB channel to k bits (16-20 bits) in each way while a MITM attack success probability is still hold at 2^{-k}.

MANA I Protocol

The authors presented a first example scheme [62], which is designed for situation in which one device has a keyboard, the other has a display. Although the scheme uses keyed check-function with short check-value (16 to 20 bits), it is proved to provide sufficient secure.

1. *Alice* sends a message m to *Bob* over an insecure channel.
2. *Alice* generates a random key k(16-20) bits. *Alice* also generates $h_k(m)$, then output to its display.
3. User enters $h_k(m)$ and k to *Bob*.
4. *Bob* uses k and recomputes $h_k(m)$, then compares the value that the user has entered.
5. The user copies success or failure.

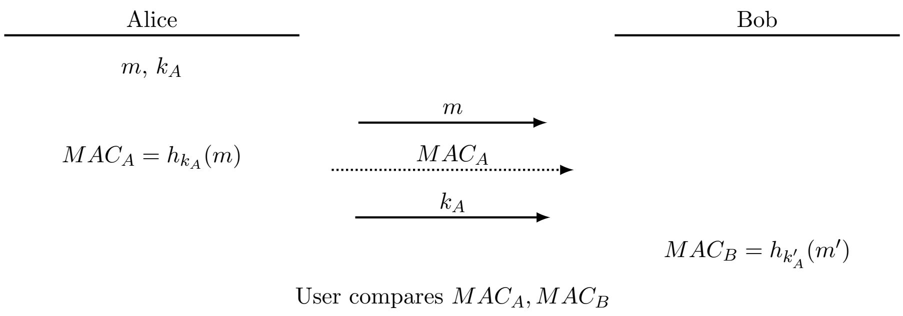


MANA II Protocol

MANA II [63] is a variant of MANA I. In this scheme, both devices are equipped with displays and keyboards. The protocol works as follows:

1. *Alice* sends a message m to *Bob* over an insecure channel.
2. *Alice* generates a random key k_A(16-20) bits. *Alice* also generates $h_{k_A}(m)$, then output it to the user over a secured channel.
3. *Alice* sends key k_A to *Bob* over the insecure channel.
4. *Bob* uses k_A and recomputes $h_{k_A}(m)$, then sends $h_{k_A}(m)$ to the user.

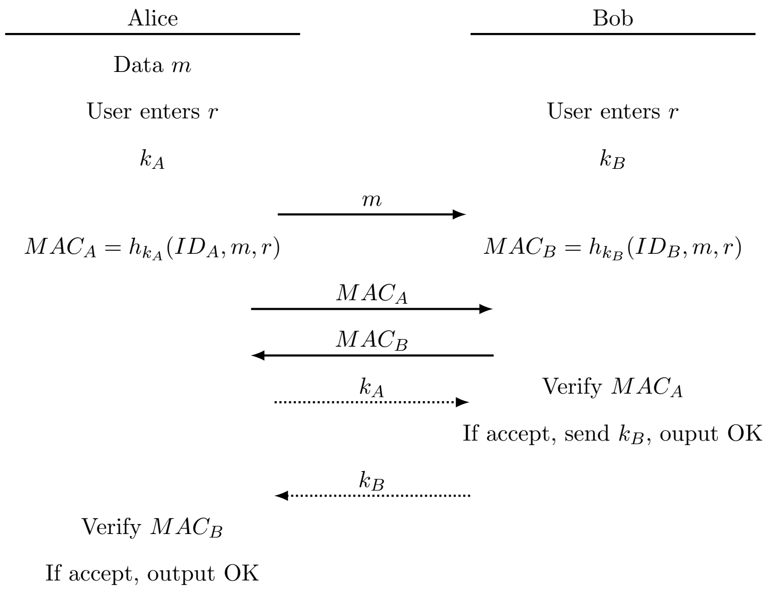
5. The user compares two values from both devices. Then if results are matched, the user indicates success to both devices over a secured channel.



MANA III Protocol

MANA III protocol [62] is designed for a situation in which both devices have keyboards. The keyboards serve as a long-range public OOB channel between two devices. Both devices are assumed on agreement on a public data m. Here, ID_A and ID_B are identifications of *Alice* and *Bob*, respectively. The scheme operates as follows.

1. *Alice* sends a message m to *Bob* over an insecure channel.
2. A user generates a short random bit-string (16-20) bits r and enters it to 2 devices.
3. *Alice* generates a key k_A, computes $MAC_A = h_{k_A}(ID_A, m, r)$, then sends MAC_A to *Bob* over a wireless link.
4. *Bob* generates a key k_B, computes $MAC_B = h_{k_B}(ID_B, m, r)$, then sends MAC_B to *Alice* over the wireless link.
5. When *Alice* receives MAC_B, then sends k_A to *Bob* over a secured channel.
6. When *Bob* receives MAC_A, then sends k_B to *Alice* over the secured channel.
7. *Alice* recomputes MAC_B and verifies its stored value of m, the expected identifier ID_B, and random value r.
8. *Bob* recomputes MAC_A and verifies its stored value of m, the expected identifier ID_A, and random r.
9. If(and only if) both devices indicate success, the user indicates success two both devices.



Analysis of MANA Protocols

An out-of-band channel used in MANA I and II could be a public or protected channel, while MANA III strongly requires a private channel as the key R must be confident. If leaking r, MANA III protocol is definitely a victim of MITM attack. Furthermore, success probability of attacks in MANA protocol is 2^{-k} where k is length of check-value messages in MANA I and MANA II, and is the length of r in MANA III. In term of optimisation, in MANA I and II, the user must compares check-values and the key k. However, in term of security, MANA I may provide a stronger security level than others.

2.2.0.6 Ephemeral Pairing Protocols

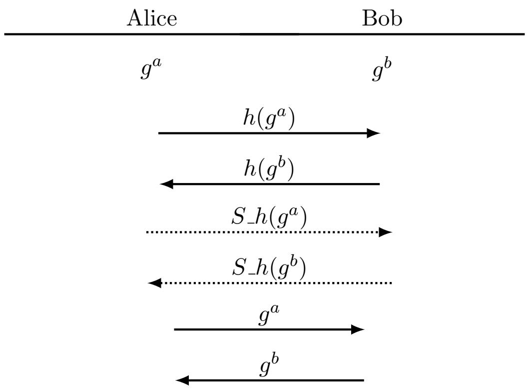
Hoepman AKA Protocol

Jaap-Henk Hoepman in [64] proposed his protocols by exploited long-range public out-of-band channel properties. In his protocol, parties securely share their DH public keys in two phases. First, long hashes of both sides' public keys are exchanged over insecure channels. Second, short hashes are sent over out-of-band channels. The detail protocol is illustrated as follows.

1. *Alice* and *Bob* generate DH key g^a and g^b respectively.
2. Both sides exchange a long hashing value of g^a and g^b over an insecure channel.

3. After the reception of the long hashing value, both sides calculate a short hashing value of g^a and g^b, exchange them over an out-of-band channel.

4. At the end, both sides reveal their value g^a and g^b to each other in the insecure channel.



Improved Ephemeral Pairing Protocol

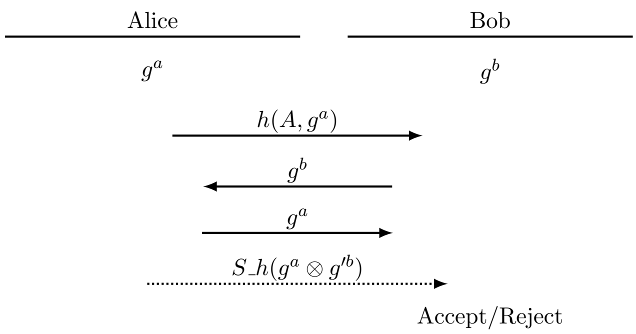
Nguyen and Roscoe [65] proposed an improved version of Hoepman protocol by cutting off one message on long-rang public OOB channel. The protocol works as follows:

1. *Alice* and *Bob* generate DH key g^a and g^b respectively.

2. *Alice* sends $h(A, g^a)$ to *Bob*.

3. Both sides exchange g^a and g^b to each other.

4. At the end, *Alice* calculates a short hash $S\_h(g^a \otimes g'^b)$, then sends it to *Bob* over the out-of-band channel.



Analysis of Ephemeral Protocols

Security of Hoepman protocol heavily depends on freshness of DH public keys in each protocol session. Otherwise, an attacker is able to discover a matching key with the same short hash output, then successfully launches a MITM attack. Hoepman also provided a proof of his protocol in the Bellare-Pointcheval- Rogaway model. The improvement scheme of Nguyen and Roscoe remains the same requirement.

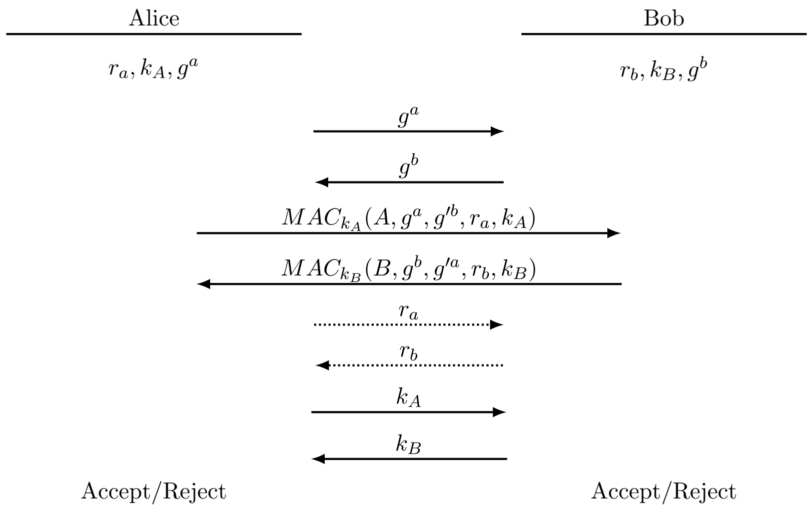
2.2.0.7 Wong-Stajano Multichanel Security Protocols

Wong and Stajano [5] proposed new mutual authentication and key agreement protocols over bidirectional and unidirectional public out-of-band channels. Their protocols exploit a short authenticated string over visual channels which provide data origin authenticity.

Protocol with Bidirectional Channel

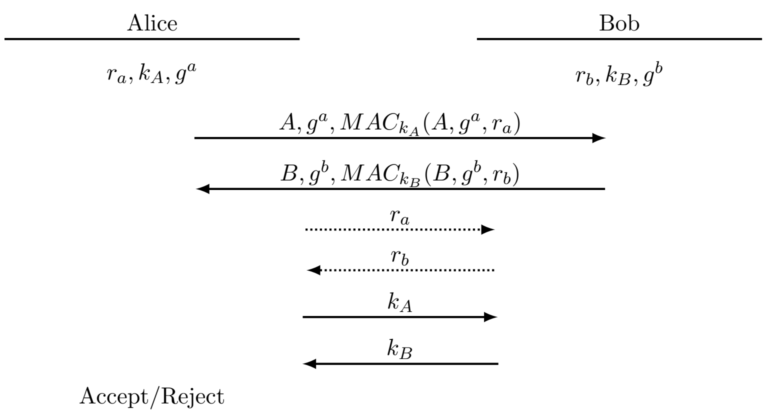
Wong and Stajano introduced a new variant of MANA III protocol which works as follows:

1. *Alice* generates a random number r_a, a key k_A, and DH key g^a.
2. *Bob* generates a random number r_b, a key k_B, and DH key g^b.
3. Both sides exchange g^a and g^b to each other.
4. *Alice* calculates $MAC_{k_A}(A, g^a, g'^b, r_a, k_A)$, then sends it to *Bob*.
5. *Bob* calculates $MAC_{k_B}(B, g^b, g'^a, r_b, k_B)$, then sends it to *Alice*.
6. Both sides exchange their r_a and r_b to each other over an out-of-band channel.
7. At the end, *Bob* sides exchange their k_A and k_B to each other.



In term of usability, the above protocol spends 6-move insecure communication, and long hash of DH public keys that put a heavy pressure on constrained devices. Realising this limitation, the authors improved their first work by an improved version over the bidirectional long-range public out-of-band channel. The new protocol works as follows:

1. *Alice* generates a random number r_a, a key k_A, and DH key g^a.
2. *Bob* generates a random number r_b, a key k_B, and DH key g^b.
3. *Alice* calculates $MAC_{k_A}(A, g^a, g'^b, r_a, k_A)$, then sends it with *Alice*'s identification A and g^a to *Bob*.
4. *Bob* calculates $MAC_{k_B}(B, g^b, g'^a, r_b, k_B)$, then sends it with *Bob*'s identification B and g^b to *Alice*.
5. Both sides exchange their r_a and r_b to each other over an out-of-band channel.
6. At the end, *Bob* sides exchange their k_A and k_B to each other.
7. *Alice* informs to *Bob* Accept or Reject the session.

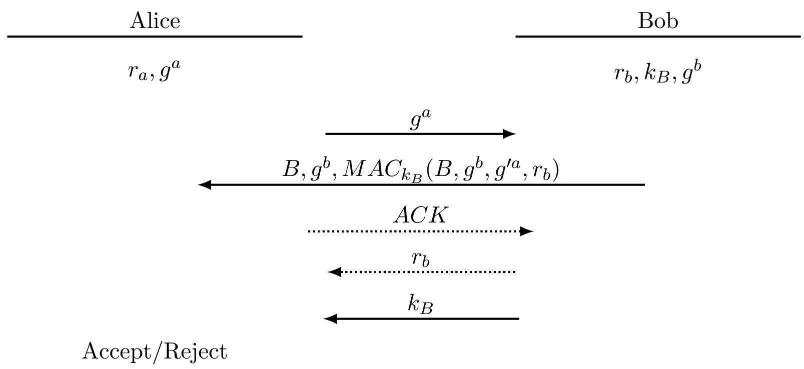


The improved protocol combines the first 4 messages of MANA III variant to 2 messages, and uses the MAC function, instead of a general hash function.

Protocol with Unidirectional Channel

In the same paper, Wong and Stajano also proposed a new protocol with unidirectional long-range public out-of-band channel. This version only spend 3 move on the wireless channel rather than 4-move in bidirectional version. The protocol works as follows:

1. *Alice* generates a random number r_a and DH key g^a.
2. *Bob* generates a random number r_b, a key k_B, and DH key g^b.
3. *Alice* sends g^a to *Bob*.
4. *Bob* calculates $MAC_{k_B}(B, g^b, g'^a, r_b, k_B)$, then sends it with *Bob*'s identification B and g^b to *Alice*.
5. *Alice* informs to *Bob* that she has already received his message over a short-range public out-of-band channel.
6. *Bob* sends r_b to *Alice* over the out-of-band channel.
7. *Bob* sends k_B to *Alice*.
8. *Alice* informs to *Bob* Accept or Reject the session.

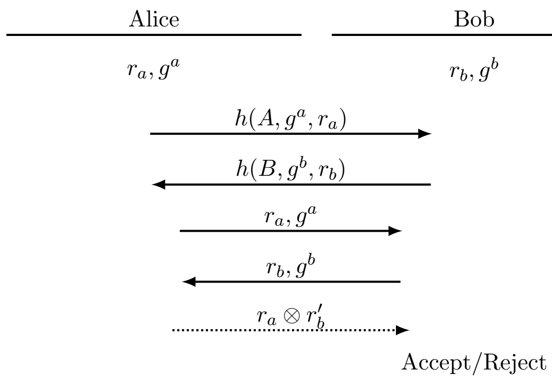


In term of efficiency, this unidirectional protocol only spends 5 messages including 2 out-of-band messages, that decreases both computation and communication cost compared to costs of two previous protocols.

Improved Wong-Stajano Key Agreement Protocol

Nguyen and Roscoe [65] proposed an variant of Wong-Stajano protocol. The new scheme cuts off long keys, and replaces two different authenticated string r_a and r_b by a single value $r_a \otimes r'_b$. The protocol works as follows:

1. *Alice* generates a random number r_a and DH key g^a.
2. *Bob* generates a random number r_b, and DH key g^b.
3. *Alice* calculates $h(A, g^a, r_a)$, and sends it to *Bob*.
4. *Bob* calculates $h(B, g^b, r_b)$, then sends it to *Alice*.
5. *Alice* sends r_a and g^a to *Bob*.
6. *Bob* sends r_b and g^b to *Alice*.
7. *Alice* calculates $r_a \otimes r'_b$, then pushes it on a long-range public out-of-band channel to *Bob*.
8. *Bob* informs to *Alice* Accept or Reject the session.



Analysis of Wong-Stajano Protocols

Wong-Stajano protocols were designed against either passive or active attacks. Passive attacks are resisted by DH components, while active attacks are resisted by short hash values over OOB channel. However, our work has successfully exploited the protocols. The counterexamples will be presented in the following section.

2.2.0.8 Short Authenticated String-Based Authentication Key Agreement Protocols

MANA-based protocol family usually requires a strong assumption on a channel on which adversaries are not able to suspend or replay any OOB message. To ease this strict condition, Serge Vaudenay introduced a protocol [66] based on Short Authentication String (SAS) in 2005. This scheme uses $k - bit$ on the long-range public OOB channel, and still preserves 2^{-k} attack success probability.

4-Move SAS-based Mutual-Authentication

Vaudenay presented his first authentication protocol based on a short authenticated string over an OOB channel [66]. The protocol is illustrated as follows.

1. *Alice* types a message m_A, then picks a random value r_a. *Bob* types a message m_B, then picks a random value r_b
2. *Alice* computes $(c_A, d_A) \leftarrow commit(0, m_A, r_a)$, and sends m_A, c_A to *Bob*.
3. *Bob* computes $(c_B, d_B) \leftarrow commit(1, m_B, r_b)$ sends m_B, c_B to *Alice*.
4. *Alice* sends d_A to *Bob*. Then *Bob* computes $(0, r'_a, m'_A) \leftarrow Open(c'_A, d'_A)$.
5. *Bob* sends d_B to *Alice*. Then *Alice* computes $(1, r'_b, m'_B) \leftarrow Open(c'_B, d'_B)$.
6. *Alice* computes $SAS_A \leftarrow r_a \otimes r'_b$, and sends SAS to *Bob* over an out-of-band channel.
7. *Bob* computes $SAS_B = r'_a \otimes r_b$, then sends SAS_B back to *Alice* over his out-of-band channel.
8. Both sides compare their calculated SAS value to received one from the other.

Alice		Bob
m_A, r_a		m_B, r_b
$(c_A, d_A) \rightarrow commit(0, m_A, r_a)$	$\xrightarrow{(m_A, c_A)}$	
	$\xleftarrow{(m_B, c_B)}$	$(c_B, d_B) \rightarrow commit(1, m_B, r_b)$
	$\xrightarrow{d_A}$	$(0, r'_a, m'_A) \leftarrow Open(c'_A, d'_A)$
$(1, m'_B, r'_b) \leftarrow Open(c'_B, d'_B)$	$\xleftarrow{d_B}$	
$SAS_A \leftarrow r_a \otimes r'_b$	$\xrightarrow{SAS_A}$	check $SAS'_A = r'_a \otimes r_b$
check $SAS'_B = r_a \otimes r'_b$	$\xleftarrow{SAS_B}$	$SAS_B \leftarrow r'_a \otimes r_b$
verify *Bob* and m'_B		verify *Alice* and m'_A

3-Move SAS-based Mutual-Authentication

Sylvain Pasini and Serge Vaudenay [67] attempted to decrease interaction cost of the protocol [66] down to 3 moves by advantage of a random oracle model. The protocol runs as follows.

1. *Alice* types a message m, then picks a random value r_a. *Bob* picks a random value r_b
2. *Alice* computes $(c_A, d_A) \leftarrow commit(m, r_a)$, and sends c to *Bob*.
3. *Bob* sends r_b to *Alice*.
4. *Alice* sends d_A to *Bob*. Then *Bob* computes $(r'_a, m') \leftarrow Open(c'_A, d'_A)$.
5. *Alice* computes $SAS_A \leftarrow r_a \otimes r'_b$, and sends SAS to *Bob* over an out-of-band channel.
6. *Bob* computes $SAS_B = r'_a \otimes r_b$, then sends SAS_B back to *Alice* over his out-of-band channel.
7. *Bob* verifies SAS and *Alice*. *Alice* compares both values of SAS and verifies *Bob*.

Alice		Bob
m, r_a		r_b
$(c_A, d_A) \rightarrow commit(m, r_a)$	$\xrightarrow{c_A}$	
	$\xleftarrow{r_b}$	
	$\xrightarrow{d_A}$	$(r'_a, m') \leftarrow Open(c'_A, d'_A)$
$SAS_A \leftarrow r_a \otimes r'_b$	$\xrightarrow{SAS_A}$	
	$\xleftarrow{SAS_B}$	$SAS_B \leftarrow r'_a \otimes r_b$
check $SAS'_B = SAS_A$		if $SAS'_A = SAS_B$
accept *Bob*		accept *Alice*

3-Move SAS-based Cross Authentication

New SAS-based Cross Authentication based on the previous 3-move mutual authentication protocol improves a number of exchanged messages and uses a strongly universal hash function family. The protocol is presented as follow.

1. *Alice* types message m, then picks a random value r_a. *Bob* picks a random value r_b

2. *Alice* computes $(c_A, d_A) \leftarrow commit(0, m, r_a)$, and sends (c_A) to *Bob*.

3. *Bob* sends (r_b) to *Alice*.

4. *Alice* sends d_A to *Bob*. *Bob* computes $(0, m', r'_a) \leftarrow Open(c'_A, d'_A)$.

5. *Alice* computes $SAS_A \leftarrow r'_b \otimes h_{r_a}(m)$, and send SAS_A to *Bob* through out-of-band channel.

6. *Bob* computes $SAS_B \leftarrow r_b \otimes h_{r'_a}(m')$ and send SAS_B to *Alice* through out-of-band channel.

7. *Alice* and *Bob* check received SAS with their own SAS. *Alice* verifies *Bob* and m. And, *Bob* verifies *Alice* and m.

Alice		Bob
m, r_a		r_b
$(c_A, d_A) \rightarrow commit(0, m, r_a)$	$\xrightarrow{(c_A)}$	
	$\xleftarrow{(r_b)}$	
	$\xrightarrow{d_A}$	$(0, m', r'_a) \leftarrow Open(c'_A, d'_A)$
$SAS_A \leftarrow r'_b \otimes h_{r_a}(m')$	$\xrightarrow{SAS_A}$	
	$\xleftarrow{SAS_B}$	$SAS_B \leftarrow r_b \otimes h_{r'_A}(m)$
check $SAS'_B = SAS_A$		if $SAS'_A = SAS_B$
accept *Bob* and m		accept *Alice* and m'

MANA IV Protocol

Presented a new method based on Vaudenay protocol [66], Sven Laur and Kaisa Nyberg [68] claimed that their proposal was more general, and had weaker security assumptions than one in [67]. Three round cross-authentication protocol MANA IV with l-bit OOB messages is presented as follows.

1. *Alice* computes $(c_A, d_A) \leftarrow commit(r_a)$ for random r_a, and sends (m_A, c_A) to *Bob*.

2. *Bob* chooses random r_b, and sends (m_B, r_b) to *Alice*.

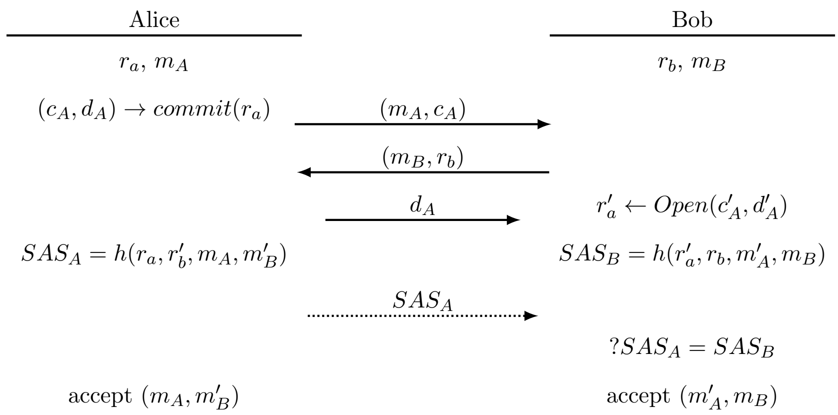
3. *Alice* sends d_A *Bob*.

4. *Bob* computes $r'_a \leftarrow Open(c_A, d_A)$ and halts if $r'_a = NULL$. Both parties compute a test value $SAS = h(r'_a, r_b, m_A, m_B)$ from received messages.

5. *Alice* sends SAS_A to *Bob* over a long-range public OOB channel.

6. Both parties accept (m_A, m_B) iff the local l-bit test value SAS_A and SAS_B coincide.

Specification: h is a keyed hash function with sub-keys k_A, k_B.



Manually authenticated MA-DH

When revising MANA IV protocol, Laur and Kyberg [68] indicated that the protocol was not computational efficiency. For this reason, they proposed a new modified protocol, namely Manually Authenticated Diffie-Hellman (MA-DH). The protocol runs as follows.

1. *Alice* computes $(c_A, d_A) \leftarrow commit(g^a)$, and sends (ID_A, c_A) to *Bob*.
2. *Bob* computes g^b for a random b, and sends (ID_B, g^b) to *Alice*.
3. *Alice* sends d_A to *Bob*.
4. *Bob* computes $k'_A \leftarrow Open(c'_A, d'_A)$ and halts if $g'^a = NULL$. Both parties compute $sid = (ID_A, ID_B)$ and $SAS = h(sid, g^a, g^b)$.
5. Both parties accept $key = (g^a)^b = (g^b)^a$ iff the l-bit test values SAS_A and SAS_B coincide.

Alice		Bob
g^a		g^b
$(c_A, d_A) \leftarrow commit(g^a)$	$\xrightarrow{(ID_A, c_A)}$	
	$\xleftarrow{(ID_B, g^b)}$	
	$\xrightarrow{d_A}$	$g'^a \leftarrow Open(c'_A, d'_A)$
$sid = (ID_A, ID'_B)$		$sid = (ID'_A, ID_B)$
$SAS_A = h(sid, g^a, g'^b)$		$SAS_B = h(sid, g'^a, g^b)$
	$\xrightarrow{SAS_A}$	
		$?SAS_A = SAS_B$
accepts $key = (g^a)^b = (g'^b)^a$		accepts $key = (g'^a)^b = (g^b)^a$

Analysis of SAS-Based Protocols

Vaudenay and Pasini used Bellare-Rogaway model to prove their protocols. They also claimed that attack success probability is 2^{-k} for k-bit SAS, and $Q_A Q_B 2^{-k}$ where Q_A and Q_B are a number of instance of *Alice* and *Bob* in network. For instance, k is 50-bits in multi-party network, and 15-bits in 2-party network.

However, Laur and Nyberg in [68] did not agree with Vaudenay results because their proofs are not sufficient, and Random Oracle(RO) model and Common Reference String (CRS) model are not suitable for ad hoc networks.

Laur and Nyberg showed in their model that attack success probability against MANA IV is 2^{-k} where k is length of SAS value.

2.2.0.9 Cagalj-Capkun-Hubaux Protocols

Cagalj, Capkun and Hubaux [69] proposed three various protocols which are based on original DH protocol, and are assisted by human operations. The first protocol is based on visual comparison of short strings (DH-SC), the second one on distance bounding(DH-DB), and the third one on integrity codes(DH-IC). Each of them is presented below.

Diffie-Hellman key agreement protocol with String Comparison

1. *Alice* and *Bob* selects secret exponents a and b, and choose random number r_a and r_b respectively.
2. *Alice* and *Bob* calculate DH public parameters g^a and g^b.

3. *Alice* computes $m_A \leftarrow 0, ID_A, g^a, r_a$. *Bob* computes $m_B \leftarrow 1, ID_B, g^b, r_b$.

4. *Alice* computes $(c_A, d_A) \leftarrow commit(m_A)$ and sends (c_A) to *Bob.*

5. *Bob* computes $(c_B, d_B) \leftarrow commit(m_B)$, and sends (c_B) to *Alice.*

6. *Alice* sends d_A to *Bob*. *Bob* computes $m'_A \leftarrow Open(c'_A, d'_A)$ and verifies that 0 appears at beginning of m'_A. If verification is successful, *Bob* sends d_B to *Alice.*

7. *Alice* computes $m'_B \leftarrow Open(c'_B, d'_B)$ and verifies that 1 appears at beginning of m'_B

8. If the verification of *Alice* is successful, both parties go to the next stage.

9. *Alice* computes $i_A \leftarrow r'_b \otimes r_a$, *Bob* computes $i_B \leftarrow r_b \otimes r'_a$.

10. *Bob* sides exchange their short values over an out-of-band channel. If the received value equals the computed value, each side informs a successful signal.

Note that, 0 and 1 are public values used to prevent reflection attack. The identification ID_A and ID_B are human readable, e.g. email addresses or names. The authors used screens or human-voice as an out-of-band channel to securely transmit authenticated values.

Alice		Bob
Give ID_A, g^a		Give ID_B, g^b
r_a		r_b
$m_A \leftarrow 0, ID_A, g^a, r_a$		$m_B \leftarrow 1, ID_B, g^b, r_b$
$(c_A, d_A) \leftarrow commit(m_A)$		$(c_B, d_B) \leftarrow commit(m_B)$
	$\xrightarrow{c_A}$	
	$\xleftarrow{c_B}$	
	$\xrightarrow{d_A}$	$m'_A \leftarrow Open(c'_A, d'_A)$
$m'_B \leftarrow Open(c'_B, d'_B)$	$\xleftarrow{d_B}$	Verify 0 in m'_A, $i_B \leftarrow r_b \otimes r'_a$
Verify 1 in m'_B, $i_A \leftarrow r'_b \otimes r_a$		
	$\xrightarrow{i_A}$	
	$\xleftarrow{i_B}$	
If $i_A = i'_B$, *Alice* outputs m'_B		If $i'_A = i_B$, *Bob* outputs m'_A

Diffie-Hellman key agreement protocol with Distance Bounding

Distance bounding protocol allows both devices to verify the distance between them. Hence, this protocol can act as a out-of-band channel.

Protocol DH-DB is mainly built on the DH-SC. Upon reception of commitment d_A, d_B, both devices execute distance bounding protocol to exchange r_a, r_b, i_A, i_B. According to time-flight estimation, each device can extract upper bound distance of the other devices.

Finally, both devices exchange y_A, and y_B to open the commitment x_A, x_B respectively. At the last step, each device check if i'_A, i'_B and i_A, i_B equal or not. Note that, this last step is done by device itself, whereas in DH-SC the comparison is performed by the users.

Alice		Bob
Give ID_A, g^a		Give ID_B, g^b
r_a, r_a		r_b, r_b
$m_A \leftarrow 0, ID_A, g^a, r_a$		$m_B \leftarrow 1, ID_B, g^b, r_b$
$(c_A, d_A) \leftarrow commit(m_A)$		$(c_B, d_B) \leftarrow commit(m_B)$
$(x_A, y_A) \leftarrow commit(0, r_a)$		$(x_B, y_B) \leftarrow commit(1, r_b)$
	$\xrightarrow{c_A, x_A}$	
	$\xleftarrow{c_B, x_B}$	
	$\xrightarrow{d_A}$	$m'_A \leftarrow Open_{(}c'_A, d'_A)$
$m'_B \leftarrow Open_{(}c'_B, d'_B)$	$\xleftarrow{d_B}$	Verify 0 in m'_A
		$i_B \leftarrow r_b \otimes r'_a$
Verify 1 in m'_B		
$i_A \leftarrow r'_b \otimes r_a$		
	-DB Phase-	
r_a bits $= r_{a1}, .., r_{ak}$		r_b bits $= r_{b1}, .., r_{bk}$
i_A bits $= i_{A1}, .., i_{Ak}$		i_B bits $= i_{B1}, .., i_{Bk}$
$\alpha_1 \leftarrow r_{a1} \otimes i_{A1}$	$\xrightarrow{\alpha_1}$	
	$\xleftarrow{\beta_1}$	$\beta_1 \leftarrow r_{b1} \otimes i_{B1} \otimes \alpha'_1$
	...	
$\alpha_i \leftarrow r_{ai} \otimes i_{Ai} \otimes \beta'_{i-1}$	$\xrightarrow{\alpha_i}$	
	$\xleftarrow{\beta_i}$	$\beta_i \leftarrow r_{bi} \otimes i_{Bi} \otimes \alpha'_i$
	...	
$\alpha_k \leftarrow r_{ak} \otimes i_{Ak} \otimes \beta'_{k-1}$	$\xrightarrow{\alpha_k}$	
	$\xleftarrow{\beta_k}$	$\beta_k \leftarrow r_{Bk} \otimes i_{Bk} \otimes \alpha'_k$
	-End of DB Phase-	
	$\xrightarrow{y_A}$	$0, r'_a \leftarrow Open(x'_A, y'_A)$
$1, r'_b \leftarrow Open_{(}x'_B, y'_B)$	$\xleftarrow{y_B}$	
$i'_{Bi} \leftarrow \alpha_i \otimes \beta'_i \otimes r'_{bi}$ (i=1,..,k)		$i'_{A1} \leftarrow \alpha_i \otimes r'_{a1}$
		$i'_{Ai} \leftarrow \alpha'_i \otimes \beta_{i-1} \otimes r'_{ai}$ (i=2,..,k)
If $i_A = i'_B$, *Alice* outputs m'_B		If $i'_A = i_B$, *Bob* outputs m'_A

Diffie-Hellman key agreement protocol with Integrity Codes

Integrity codes (I-codes) are used with communication channel such that that they are not possible to block signal without being detected. An integrity code is a seven-tuple $(\mathcal{S}, \mathcal{M}, \mathcal{E}, \mathcal{P}, l, t, e_c)$ where:

1. $\mathcal{S}$ is possible of source states.
2. $\mathcal{M}$ is a set of binary of sequence t 1's and $l - t$ 0's.
3. $\mathcal{E}$ is a set of source encoding rules $e_s : \mathcal{S} \rightarrow \mathcal{M}$, where $e_s \in \mathcal{E}$ is an injective function.
4. $\mathcal{P}$ is a set consisting of two power levels 0 and p with $p > 0$.
5. $e_c : \mathcal{M} \rightarrow \mathcal{P}^{\updownarrow}$ is a channel modulation function satisfying rules: (i) symbol ”1” is transmitted using power level p, symbol ”0” is transmitted using power level 0.

Based on a concept of I-codes, its application on DH-SC is straightforward. *Bob* and *Alice* complete four steps of DH-SC protocol at beginning. *Alice* encoded authentication value i_A into $I{-}code$, then transmits it in an out-of-band channel to *Bob*. After reception of $I - code$, *Bob* probably ensures that this $I - code$ is sent from Alice, and verifies if authenticated value i'_A is equal to i_B or not.

Alice		Bob
Give ID_A, g^a		Give ID_B, g^b
r_a		r_b
$m_A \leftarrow 0, ID_A, g^a, r_a$		$m_B \leftarrow 1, ID_B, g^b, r_b$
$(c_A, d_A) \leftarrow commit(m_A)$		$(c_B, d_B) \leftarrow commit(m_B)$
	$\xrightarrow{c_A}$	
	$\xleftarrow{c_B}$	
	$\xrightarrow{d_A}$	$m'_A \leftarrow Open(c'_A, d'_A)$
$m'_B \leftarrow Open(c'_B, d'_B)$	$\xleftarrow{d_B}$	Verify 0 in m'_A, $i_B \leftarrow r_b \otimes r'_a$
Verify 1 in m'_B, $i_B \leftarrow r'_b \otimes r_a$		
Push a button	$\xrightarrow{I - codes(i_A)}$	
Inform ”Message Sent”		Push a button
		Verify $i'_A = r'_b \otimes r_a$
If $i_A = i'_B$, *Alice* outputs m'_B		If $i'_A = i_B$, *Bob* outputs m'_A

2.2.0.10 Short Random String-Based Key Agreement Protocol

Authors in [38] claimed that SAS-based AKA methods strongly required assistance from users. Therefore, they proposed a scheme, namely Short Random String (SRS)-based key agreement protocol, automatically pairing two devices using an audio channel. The proposed scheme was claimed more advantage than SAS-based scheme in term of bandwidth utilisation.

The 3-move protocol between *Alice* and *Bob* is as follow.

1. A device A sends A's public key PK_A to device B.
2. A device B generates two random strings, r and SRS. Then B calculates a hashed value using PK_A, PK_B, SRS and r and sends this value to A.
3. B sends SRS to A over an out-of-band channel.
4. B sends r to A.
5. A verifies the hashed value received in step 2 using PK_A, PK_B, SRS and R. If it is successful, A generates agreed key using authenticated B's public key.

In the protocol, the verification process is only decided by A, then attackers easily impersonate A's public key. However, this attack can be prevented by human monitoring or confirming from $A$to B.

Alice		Bob
		r, SRS
	$\xrightarrow{PK_A}$	
	$\xleftarrow{c_B}$	$c_B =$
		$h(PK'_A, PK_B, r, SRS), PK_B$
	$\xleftarrow{SRS}$	
	$\xleftarrow{r}$	
$c_A =$		
$h(PK_A, PK'_B, r', SRS')$		
If $c_A = c'_B$		
accepts PK'_B		

Strength of the protocol completely lies on length of SRS string. If the length of SRS message is p, then success probability of attack is less than or equal 2^{-p}.

Summing up this part is presented at the table 2.4.

Table 2.4: Paring Method Comparision

Approach	**Number of Message**		**Computation**	**Communication**	**Required Crypto-graphic Primitives**	**OOB Type**
	Wireless Channel	**OOB Channel**	**Cost per Sid**	**Cost over OOB**		
Interlock	4	2	2 HASH	2* 80 bits	HF	Long-range public
Maher Manual Authentication	2	2	1 HF	2* 80 bits	HF	Long-range public
Talking to Strangers	2	2	1 HASH	2 * ID + 2 * HASH	HF	Long-range public
VIC	3	1	1 HASH	20 bits	HF	Long-range public
MANA I	0	1	1 HF	20 bits k + 20bits HF	MAC	Protected
MANA II	1	1	1 HF	20 bits k + 20bits HF	HF	Long-range
MANA III	4	1	2 HASH	16-20bits k	MAC	Long-range
Wong-Stajano MANA III	6	2	1 MAC	2 *(20 bits N)	HF	
Wong-Stajano Bidirectional Out-of-band Channel	4	2	1 MAC	2 *(20 bits N)	HF	Long-range
Wong-Stajano Unidirectional Out-of-band Channel	3	1	1 MAC	20 bits N	HF	Long-range
Improved Wong-Stajano	4	1	1 HASH + 1 XOR	1* (20 bits N)	HF + XOR	Long-range
Hoepman AKA	4	2	1 L_HASH + 1 S_HASH	2* (N bits S_HASH)	Short(S_) and Long(L_) HF	Long-range
Improved Ephemeral Pairing	3	1	1 L_HASH + 1 S_HASH	80 bits S_HASH	Short(S_) and Long(L_) HF	Long-range
4-move SAS Vaudenay	4	2	1 CS + 1 XOR	15bits SAS	CS + XOR	Long-range
3-move SAS Vaudenay	3	1	1 CS + 1 XOR	15bits SAS	CS + XOR	Long-range
SAS Cros AKA Pasini-Vaudenay	3	2	1 CS + 1 XOR + 1 HASH	2 * (15-20bits SAS)	CS + XOR + HF	Long-range
MANA IV	3	1	1 CS + 1 HASH	14bits SAS	CS + HF	Long-range
MA-DH	3	1	1 CS + 1 HASH	14bits SAS	CS + HF	Long-range
SRS-based AK	3	1	1 CS	15 bits SRS	CS	Long-range
DH-SC	4	2	2 CS + 1 XOR	15bits SAS	CS +XOR	Short-range
DH-DB	6	2	1 CS + 1 XOR	15bits SAS	CS +XOR	Short-range

2.3 2-Move Secure Device Pairing Protocol

In this section, we propose a novel pairing protocol. This proposal is relatively efficient in sense that it only requires 2 messages on a wireless channel plus one message on a unidirectional public OOB channel. In spite of this, we will see in section 2.3.1 that it still preserves an attack success probability of 2^{-k}, where k is length of random numbers. The picture 2.1 describes how the protocol works.

The protocol is depicted in Figure 2.1. It runs between the initiator *Alice* (A) and the responder *Bob* (B), who intend to securely exchange their public keys denoted as g^a and

g^b respectively. The protocol is presented as below.

1. *Alice* picks a random value r_a. *Bob* picks a random value r_b
2. *Alice* sends $(g^a, h(g^a, r_a))$ to *Bob*.
3. *Bob* sends (g^b, r_b) to *Alice*.
4. *Alice* sends $(r_a \otimes h_{r_b}(g^a, g^b))$ to *Bob* over a public channel.
5. *Bob* verifies received value and announces the result to *Alice*.
6. *Alice* confirms by pushing an Accept button.

Random values r_a, r_b and use of hash function in the protocol work as an instance of a commitment scheme whose design has strong results on minimising attack probability as we will see in section 2.3.1.

This protocol achieves strong security even with a low-bandwidth OOB channel. *Alice* and *Bob* are assumed to be honest or uncompromised. After receiving the third message *Bob* can perform verification and then securely notify *Alice* of an outcome with the human user's help.

Table 2.5 summarises a comparison of our protocol with main existing protocols exploiting a public out-of-band channel. We took into account a number of messages on wireless channel, a number of messages on OOB channel, computation cost, and existence of a formal proof of security (either in the Dolev-Yao model or computational model).

In term of computation complexity, our protocol can catch up with other competitors since one exclusive-or operation on a short string does not significantly impact the total cost. Concerning communication cost, our protocol uses the least number of messages. Furthermore, a security analysis has been performed both in Dolev-Yao and computational models.

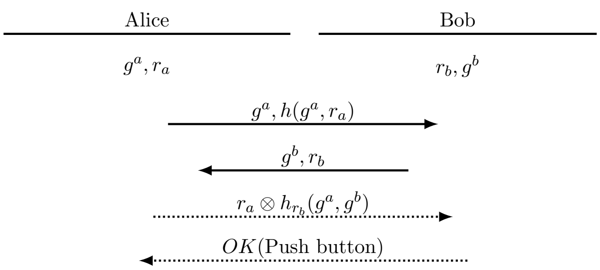


FIGURE 2.1: New 2-move Authenticated Key Agreement Protocol

TABLE 2.5: DEVICE PAIRING PROTOCOL COMPARISON

Protocol	Wireless Message	OOB Message	Computation Cost	Formal Proof
Bidirectional Wong-Stajano [5]	4	2	2*MAC	FAIL
Unidirectional Wong-Stajano [5]	3	2	1*MAC	FAIL
Improved Wong-Stajano [65]	4	1	2*HF + 1*XOR	∅
DH-SC [69]	4	1	2*CS + 1*XOR	√
4-Move SAS [66]	4	1	2*CS + 1*XOR	√
3-Move SAS [66]	3	1	1*CS +1HF + 1*XOR	√
MANA IV [68]	3	1	1*CS +1*HF	√
MA-DH [68]	3	1	1*CS +1*HF	√
SRS [38]	3	1	1*HF	FAIL
Our Proposal	2	1	1*HF +1*MAC + 1*XOR	√

2.3.1 Analysis In Computational Model

We give a sketch of proof of our protocol in a computational model. Our goal is to evaluate the successful attacking probability against the protocol. Similarly to [69] we conduct an analysis based on the model presented in [70].

We refer to the security definition in [69] which is presented as follows.

Definition 2.3.1. We say that a protocol is a secure protocol enabling authentication of DH public parameter between A and B if attacker cannot succeed in deceiving A and B into accepting DH public parameters different then g^{xa} and g^{xb}, except with a satisfactorily small probability $\mathcal{O}(2^{-k})$.

Lemma 2.3.2. For any interaction between strand st and st', and attacker X, attack success probability of X is lower or equal to $n.\gamma.2^{-k}$, where n is the number of participants on the network, γ is the maximum number of sessions for each participant, k is the length of short authenticated string.

Proof. For a normal run, the responder uses value of r_a extracted from messages received on OOB channel to open the commitment $h(g^a, r_a)$. If the responder opens successfully, an Accept statue is notified. So, to win the game, attacker X has to deliver $h(m)$ in the first message such that $h(m) = h(g^{xa}, (r_a \otimes h_{r_{xb}}(g^a, g^{xb})) \otimes (h_{r_b}(g^{xa}, g^b)))$ (1) for some m. Due to our assumption on MAC function equation (1) can be rewritten in $h(m) = h(g^{xa}, (r_a \otimes r_{xb} \otimes r_b \otimes h(g^a, g^{xb}) \otimes h(g^{xa}, g^b)))$ (2).

In the simplest case where attacker X is able to find a pair (g^{xa}, g^{xb}) such that $h(g^{xa}, g^b) = h(g^a, g^{xb})$, we can deduce from (2) to $r_{xa} = r_a \otimes r_{xb} \otimes r_b$ or $r_{xa} \otimes r_b = r_a \otimes r_{xb}$ (3).

Observe that X has to submit r_{xb} before actually knowing r_a. Similarly, X has to submit r_{xa} before actually seeing r_b. Thus irrespectively, the attacking strategy taken by X, r_a and r_b will be revealed after r_{xa} and r_{xb} have been generated and submitted. If it happens that both r_a and r_b are revealed in the same time, then we can pick an arbitrary one.

Assume that r_a is revealed after r_b, we have:

(i) r_a and r_b are independently and uniformly distributed random variables,

(ii) r_{xa} and r_{xb} must be generated and submitted before either r_a is revealed,

(iii) each principal can open at most γ sessions.

The same holds for a case where r_b is revealed after r_a. Therefore, $Pr[r_{xa} \otimes r_b = r_a \otimes r_{xb}] \leq n.\gamma.2^{-k}$.

In a normal case where X cannot find collision, let $t = h(g^a, g^{xb}) \otimes h(g^{xa}, g^b)$ since g^a and g^b are not changed over sessions. We have (i)$r_{xa} \otimes r_b = r_a \otimes r_{xb} \otimes t$, (ii) t could be constant, hence, $Pr[r_{xa} \otimes r_b = r_a \otimes r_{xb} \otimes t] \leq n.\gamma.2^{-k}$. □

2.3.2 An Implemetation on An Embedded System

We produce a prototype of our device pairing protocol using two Arduino boards (the source codes can be found at the appendix C). Because of lacking WIFI shields, two Ethernet shields are used instead to provide a network connect between two boards. The testing system includes:

- An Arduino Uno equips with an Ethernet shield, and a LED light.
- An Arduino Mega equips with an Ethernet shield, and a light sensor.
- Two boards connect via an Ethernet cable.

The testbed is presented at the figure 2.2. A visual light communication served as an out-of-band communication channel is intuitively established via a LED and a light sensor. Data transmitting in this channel are encoded by a sequence of blinking light. In particular, while the LED hitting the highest brightness represents for a bit "1", the LED dims to represent for a bit "0". Furthermore, distance between the LED and the light sensor is short enough so that the sensor can correctly recognise bits from the LED. The accuracy of OOB transmission strongly depends on this distance and noise of environment apparently.

All tests have been done in normal lab environment. We ran 10 times, and in each time 32-bits OOB message was transmitted between two boards. Under 3 centimetres, we get 100% successful rate. The rate reduces to 75% in 5 centimetres, and dramatically drops at further distance. However, in low noise environment, the results are remarkably improved. The result of executing time of some functions is presented in the table 2.6. With these results, we believe that our solution is practical.

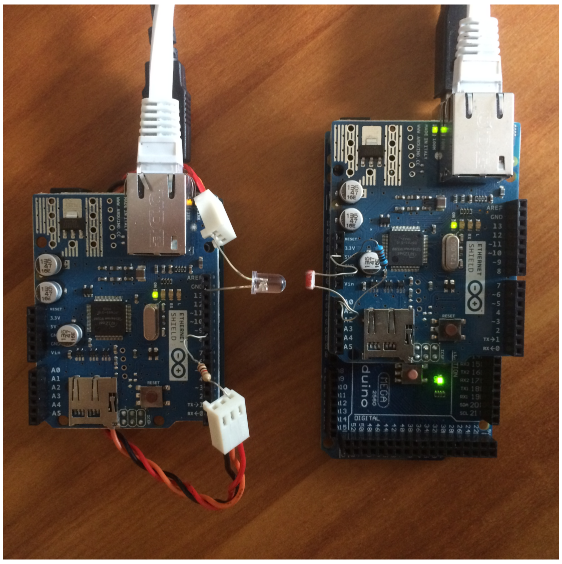


Figure 2.2: Prototypes

Table 2.6: Performance Evaluation of Device Pairing Protocol

Function	**Time(ms)**
Generating Keys	3898
commitment Calculation	15
commitment Validation	15
Transferring OOB message	3201
Complete Protocol	21760

2.4 Flaws Found in Some Pairing Protocols

In this section, we are going present flaws that we have found in some pairing protocols. These flaws have not revealed in any publication before.

We will explain clearly how come we can discover these attack in the next chapter. Briefly speaking, we constructed a formal model to analyse our protocol, and adapted it to do other existing device pairing ones introduced in this chapter. As a result of that, some flaws have been discovered.

To begin with, we would like to take some assumptions on channels, attacker capabilities, and user actions as below.

- Two participants don't play the protocol concurrently.
- The protocol replays when it accidentally gets an error.
- Attacker knows OOB message content before the message is delivered.
- Attacker can suspend user's actions.

Table 2.7: Attack Scenario Against Initiator's Guarantee in Wong-Stajano Protocol with Bidirectional Channel

Step 1.1	Attacker suspends the r_b sent by *Bob* on OOB channel
Step 1.2	Attacker drops the k_B sent by *Bob*.
Step 1.3	Attacker starts a new session with Alice
Step 2.1	*Alice* sends $(A, g^a, MAC_{k_{A2}}(A, g^a, r_{A2}))$ to the Attacker on wireless channel
Step 2.2	Attacker sends $(B, g^x, MAC_{k_X}(B, g^x, r_B))$ on wireless channel
Step 2.3	Attacker drops R_{A2} sent by *Alice* on OOB channel
Step 2.4	Attacker releases r_b at Step 1.1 on OOB channel
Step 2.5	Attacker sends k_X to *Alice* on wireless channel
Step 2.6	At the end of the execution, *Alice* believes she shares a fresh session key with Bob, known actually by the Attacker

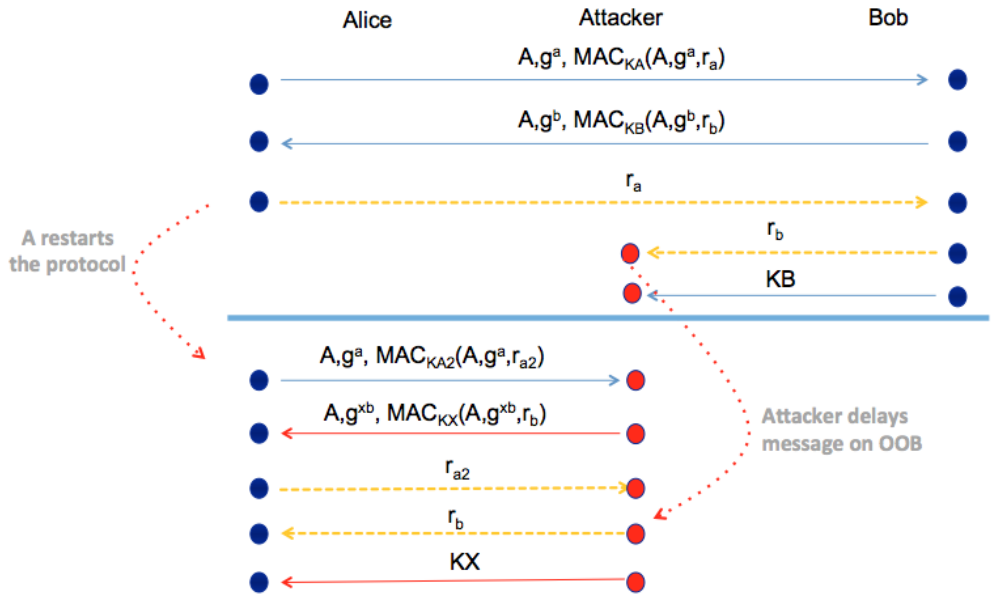


Figure 2.3: Attack on Wong-Stajano Protocol using Bidirectional Channel with Unidirectional Channel

Furthermore, protocols are considered in theoretical perspective where OOB channels are classified in our category. In particular, these following protocols are assumed to use long-range public out-of-band channels which only provide channel origin authentication.

2.4.1 Attack on Wong-Stajano Protocol using Bidirectional Channel

The Wong-Stajano protocol using bidirectional channel was presented at 2.2.0.7 in this chapter. The protocol aims to provide a key agreement between two participants. However, we found a counterexample in which the protocol goal does not hold for the initiator. The counterexample is illustrated in the figure 2.3, and is precisely detailed in table 2.7.

Table 2.8: Attack Scenario Against Initiator's Guarantee in Wong-Stajano Protocol with Unidirectional Channel

Step 1.1	Attacker intercepts g^a sent by *Alice* on wireless channel
Step 1.2	Attacker replies with $(B, g^x, h_{k_X}(B, g^x, g^a, r_x))$ to *Alice* on wireless channel
Step 1.3	Attacker suspends r_b sent by *Bob* on OOB channel, and starts a new session with Alice
Step 2.1	*Alice* sends $g^{a'}$ on wireless channel
Step 2.2	Attacker responds $(B, g^x, h_{k_X}(B, g^{a'}, g^{x'}, r_b))$ on wireless channel
Step 2.3	Attacker drops ACK sent by *Alice* on OOB channel
Step 2.4	Attacker release r_b sent by *Bob* on OOB channel at Step 1.3
Step 2.5	Attacker sends k_X to *Alice* on wireless channel
Step 2.6	At the end of the execution, *Alice* believes she shares a fresh session key with *Bob*, known actually by the Attacker

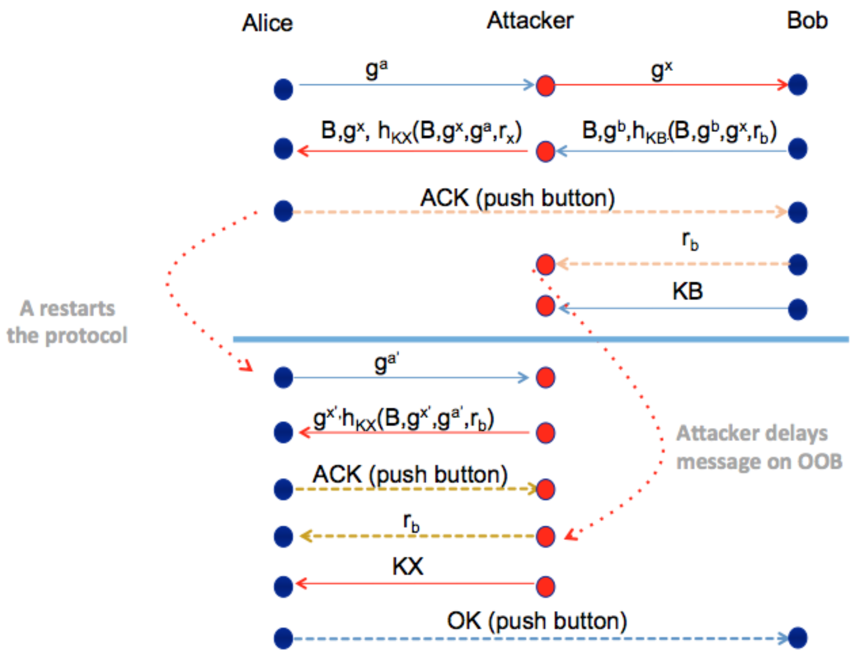


Figure 2.4: Attack on Wong-Stajano Protocol using Unidirectional Channel

Additionally, we found the protocol is being used in the Pico system of the authors [71].

2.4.2 Attack on Wong-Stajano Protocol using Unidirectional Channel

The counterexample of Wong-Stajano protocol using unidirectional channel is illustrated in the figure 2.4, and is precisely detailed in table 2.4.

2.4.3 Attack on SRS-AKA Protocol

Attack against SRS-AKA is closely identified with one in Wong-Stajano protocol using unidirectional channels. The attack scenario is illustrated in figure 2.5 and detailed at

Table 2.9: Attack Scenario Against Initiator's Guarantee in SRS-AKA Protocol

Step 1.1	Attacker intercepts PK_A sent by *Alice* on wireless channel
Step 1.2	Attacker replies with $h(PK_A, PK_X, r_x, SRS_X), PK_X$ to *Alice* on wireless channel
Step 1.3	Attacker suspends SRS sent by *Bob* on OOB channel, and starts a new session with *Alice*
Step 2.1	*Alice* sends PK_A on wireless channel
Step 2.2	Attacker responds $h(PK_A, PK_X, r_x, SRS), PK_X$ on wireless channel
Step 2.4	Attacker release SRS sent by *Bob* on OOB channel at Step 1.3
Step 2.5	Attacker sends r_x to *Alice* on Wireless channel
Step 2.6	At the end of the execution, *Alice* believes she shares a fresh session key with Bob, known actually by the Attacker

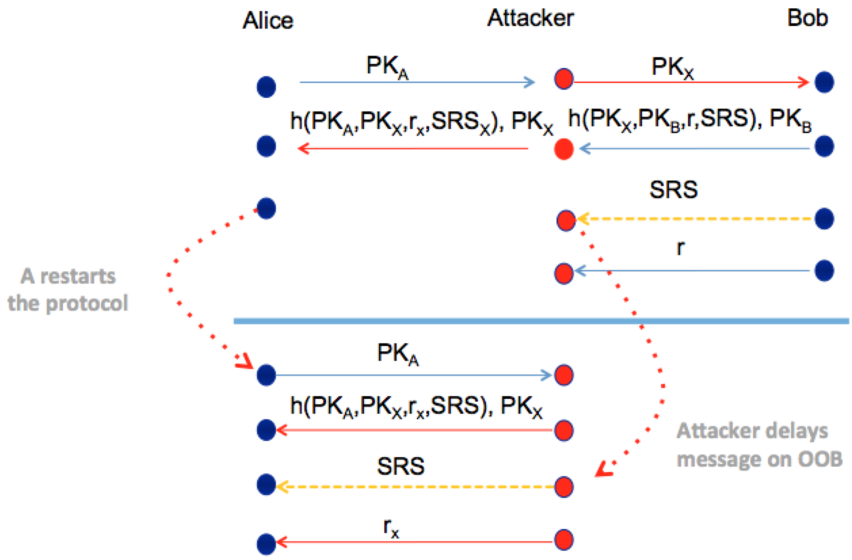


Figure 2.5: Attack Against SRS-AKA Protocol

table 2.9.

2.4.4 Attack on Hoepman AKA Protocol

Since the Hoepman protocol is quite similar to Wong-Stajano protocol with bidirectional channel, the attack found on Wong-Stajano protocol might be used against Hoepman one. But, the difference between two protocols is while two random numbers are exchanged over OOB channel in Wong-Stajano, two short short-hashing values are used in Hoepman version. This apparently lets the attacker a big challenge to break the Hoepman protocol. Nevertheless, due to the less strong hash function, if the attack can find a collision of the short hash function in polynomial time, it is definitely able to launch MITM attack. We take this assumption and present the attack scenario in figure 2.6, and table 2.10.

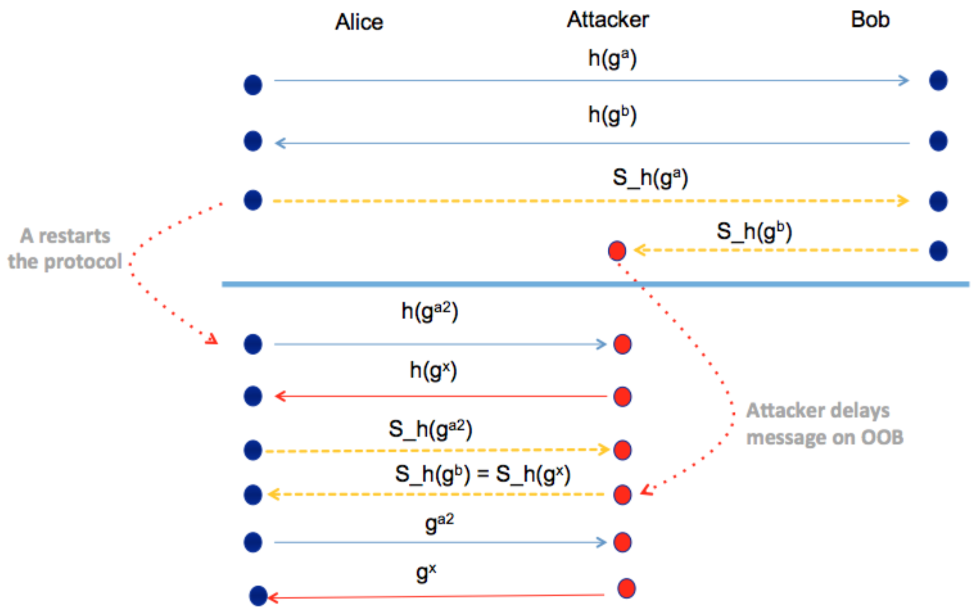


FIGURE 2.6: Attack Against Hoepman-AKA Protocol

TABLE 2.10: ATTACK SCENARIO AGAINST INITIATOR'S GUARANTEE IN HOEPMAN PROTOCOL

Step 1.1	Attacker suspends the $S\_h(g^b)$ sent by *Bob* on OOB channel
Step 1.2	Attacker finds g^x so that $S\_h(g^x) = S\_h(g^b)$. It does not necessary to find $h(g^x) = h(g^b)$
Step 1.3	Attacker starts a new session with Alice
Step 2.1	*Alice* sends $h(g^{a2})$ to the Attacker on wireless channel
Step 2.2	Attacker sends $h(g^x)$ to *Alice* on wireless channel
Step 2.3	Attacker drops $S\_h(g_{a2})$ sent by *Alice* on OOB channel
Step 2.4	Attacker releases $S\_h(g^b)$ at Step 1.1 on OOB channel
Step 2.5	*Alice* sends g^{a2} to Attacker on wireless channel
Step 2.6	Attacker sends g^x to *Alice* on wireless channel
Step 2.7	At the end of the execution, *Alice* believes she shares a fresh session key with Bob, known actually by the Attacker

2.5 Conclusion

In this chapter, we have conducted a deep survey on out-of-band channel types and secure device pairing protocols. Moreover, we have provided a novel solution to the fundamental issues of key agreement over a radio links. To our knowledge, our proposal requiring only 2 wireless radio messages and one out-of-band message is more economic than existing protocols in term of communication cost. Yet it still provides a security level equivalent to one of existing protocols. We also gave a sketch of proof in a computational model. Additionally, an implement of our protocol has been conducted and tested in an embedded system to show that it is practical via our benchmark. In mean while, we found some flaws of Wong-Stajano protocols, Hoepman protocol, and SRS-AKA protocol, this has not introduced before.

Chapter 3

Analysis of Secure Device Pairing Protocols

Our objective in this chapter is to propose a formalism which models device pairing protocols in a natural manner, and permits verification of security properties relevant to these protocols. We conceive such a formalism as an adaptation of Strand Spaces [15]. The model of Strand Spaces is a flexible formalism which represents protocols as a set of local views of participants in a run of a protocol. Taking advantage of this flexibility, our model extends Strand Spaces to deal with OOB channels. Moreover, the attacker model must be refined to take into account the different types of channels, i.e. unsecured channels and OOB channels.

Thank to our improved model, a device pairing protocol with unidirectional out-of-band channel proposed by Wong and Stajano [5] is discovered with a flaw which has not introduced before. More seriously, this protocol is using in current their products such as [71, 72]. Ultimately, we produce a procedure which transforms a model of an initial protocol in our extended Strand Spaces to an equivalent model without any out-of-band channel in original Strand Spaces.

The chapter 3 begins with some related work. Then we conduct our improved Strand Space to deal with out-of-band channels and device pairing problems. Analysis of Wong-Stajano protocol is presented later. Additionally, a proof of our proposed protocol in previous chapter is also offered. At the end of this chapter, our out-of-band translation is presented.

3.1 Related Work

Whereas a great deal of work tackles problems of formal verification of classical authentication protocols (see for instance [73] for an introduction to the topic), to our knowledge few address problems in cases of multichannel protocols.

Presented in [74], a question arises: are auxiliary channels necessary to provide authentication without pre-shared knowledge? Using BAN logic [3], they proved that device authentication using a single channel is not possible . From this analysis, they proposed an extension of BAN logic taking into account OOB channels, and using this extension the *Talking to Strangers* protocol from [61] and a simplified version of Wong-Stajano protocol [5] were shown to be correct. However, as we have seen in subsection 2.4.2, the Wong-Stajano protocol is vulnerable to an attack. In fact, the proposed formalism did not offer enough expressiveness to correctly model the Wong-Stajano protocol. Formal verification of specific versions of Bluetooth protocols has received a lot of attention in literature. Several proposals were introduced to take into account Bluetooth security weaknesses from a version 2.0 to a brand new version 4.0.

Some verification tools have been applied such as ProVerif in[75], and PRISM probabilistic model checker in [76]. These work are a first steps towards an automated analysis of formal model of human-assisted protocols.

3.2 Extended Strand Spaces with Out-of-Band Channels

Due to lack of place, we do not recall here the whole theory of Strand Spaces, but focus on the extensions necessary to examine secure pairing protocols based on Diffie-Hellman scheme [77]. For a complete background on Strand Spaces the reader can consult [15, 78, 79](we recap the Strand Spaces theory in the appendix A). The extensions mainly concern the algebra and the penetrator model.

Before presenting our extension of Strand Spaces, we formulate some supplementary assumptions concerning the execution of device pairing procedures, that we will have to take into account.

3.2.1 Model Assumptions

We now make several practical assumptions in our model as follows:

- The hash functions used in the secure device pairing protocol are perfect, that is the attacker cannot perform with success the following attacks: collision attack, pre-image attack, and second-image attack.
- There is no more than one instance of a particular role uses an OOB channel on each side at a given time.
- When one device sends the Accept/Reject information, the other confirms this decision.
- After a device pairing procedure, the communication session will start later. But in case of no evidence of exchanging procedure, the device pairing procedure replays again with a new session.

3.2.2 Extension to the Algebra

In the thesis, we will use the term *regular* as legitimate or trusted. Therefore, *regular strand* to refer to a run of some legitimate roles of a protocol. In the same way, we will use *regular node* to refer to a send or receive event occurring on a regular strand. Additionally, we use *regular keys* to refer to trusted keys.

Our definition of Strand Space algebra is based on the definitions from [79], which adds to model the possibility to deal with DH operation, hash functions, and signatures. To take into account device pairing protocols, we do not need to consider signatures (neither asymmetric encryption), but must add keyed hash function, or MAC function. We thus redefine the set of terms as follows:

Definition 3.2.1. The set of *terms* $\mathcal{A}$ is assumed to be freely generated from four disjoint sets: predictable texts $\mathcal{T}$, unpredictable texts $\mathcal{R}$, keys $\mathcal{K}$, and Diffie-Hellman values $\mathcal{D}$.

The set of keys $\mathcal{K}$ is divided into two disjoint sets: verification keys $\mathcal{K}_{Ver}$, and keys for symmetric encryption $\mathcal{K}_{Sym}$. The set $\mathcal{T}$ also includes a set of $\mathcal{T}_{Name}$ containing identifications of participants.

Compound terms are built by these operations:

- join: $\mathcal{A} \times \mathcal{A} \rightarrow \mathcal{A}$, which represents concatenation of terms.
- encr: $\mathcal{K}_{Sym} \times \mathcal{A} \rightarrow \mathcal{A}$, which represents encryption.
- DH: $\mathcal{D} \times \mathcal{D} \rightarrow \mathcal{D}$, which represents the Diffie-Hellman operation. We denote the range of DH by $\mathcal{D}_{DH}$.

- hash: $\mathcal{A} \rightarrow \mathcal{K}_{Sym}$, representing hashing into keys. We denote the range of hash by $\mathcal{K}_{hash}$.
- MAC: $\mathcal{K}_{Sym} \times \mathcal{A} \rightarrow \mathcal{K}_{Sym}$, representing MAC operation with a key into keys.

Terms will be denoted by t, t' possibly indexed by an integer. The elements from the set of unpredictable or random texts $\mathcal{R}$ are used to play the role of nonces in protocols and will be denoted by r possibly indexed with the identifier of an agent. The elements of $\mathcal{K}$ (resp. $\mathcal{D}$) will be denoted by k (resp. d) possibly indexed by an identifier (resp. integer). In the following, $encr(k,t)$, $hash(t)$ and $MAC(k,t)$ will be respectively noted $\{t\}_k$, $h(t)$ and $h_k(t)$. The term $join(t,t')$ will be noted t, t'.

In our extension, we will need to explicitly distinguish between different channels. We thus need to define what is a channel.

Definition 3.2.2 (Channel). A *channel* is a group of strands which can exchange messages in the same region.

One strand may use more than one channel. For example, given two channels ch_1 and ch_2 and 3 strands: st_1, st_2, st_3, the strand st_1 and st_2 may use ch_1, whereas st_2 and st_3 use ch_2.

If no supplementary assumption is declared, a channel is by default an unsecured public wireless channel. Any specific assumption on a channel, must be specified before formalising the protocol. Since a protocol may use several channels, when sending or receiving a term, the used channel must be specified. The definition of signed term is modified in consequence.

Definition 3.2.3 (Signed term). A *signed term* is a triplet $\langle \delta, t, ch \rangle$, noted $\delta_{ch}t$, where δ is $+$ (sending) or $-$ (reception), t a term, and ch the channel on which t is sent or received.

Actually, we will see in subsection 3.2.3 that the terms manipulated by a penetrator may receive another sign. By convention, we will specify the channel only when using an OOB channel: $-_{ch}t$ means that the term t is received on the OOB channel ch, and $-t$ will denote the reception of t on the public wireless channel.

Based on this new definition of signed terms, the definitions of *strand space*, *node*, *edge*, *originating term*, *uniquely originating term*, and *bundle*, *height of a strand* are the same in [15].

We refine the notions of *subterm* and *component* from previous works on Strand Spaces as follows.

Definition 3.2.4 (Subterm). We say that t is a *subterm* of t', written $t \sqsubseteq t'$ if:

- $t = t'$ or
- $t' = (t'_1, t'_2)$ then $t \sqsubseteq t'_1$ or $t \sqsubseteq t'_2$,
- if $t' = \{t''\}_k$, then $t \sqsubseteq t''$,
- if $t' = h(t'')$, then $t \sqsubseteq t''$,
- if $t' = h_k(t'')$, then $t \sqsubseteq t''$,
- if $t' = DH(d_1, d_2)$, then $t \sqsubseteq d_1$ or $t \sqsubseteq d_2$

Definition 3.2.5 (Component). We say that a term t is a *component* of term t', written $t \sqsubseteq_c t'$, if t' can be obtained by concatenating t with others terms.

For example, the term $(A, g^a, h(A, g^a))$, where g^a denotes a Diffie-Hellman key, contains three components: A, g^a, and $h(A, g^a)$.

At last, we introduce the notion of boxed term.

Definition 3.2.6 (Boxed term). For a given bundle, we say that a term t is *boxed* at node n, if there exists terms t' and t'' such that $t \sqsubseteq t'$, $t' \sqsubseteq term(n)$, and t' has one of the following forms: $\{t''\}_k$, $h(t'')$, $h_k(t'')$.

3.2.3 Extended Penetrator Model

The new penetrator model must take into account the different kind of channels used in the secure device pairing protocols. Concerning wireless channels, the original Dolev-Yao model [29] is broadened with DH, hash and MAC operations as following. We denote $\mathcal{D}_P$ as a set of attacker's DH key.

- **F**. Fresh DH key: $\langle +d \rangle$ where $d \in \mathcal{D}_P$ with $\mathcal{D}_P \subset \mathcal{D}_{DH}$.
- **H**. Hashing: $\langle -t, +h(t) \rangle$
- **MAC**. MAC: $\langle -t, -k, h_k(t) \rangle$ where k is a key generated by the attackers.

As described above, OOB channels intentionally limit penetrator's capacities in term of message manipulation. He is prevented from performing actions on private OOB channels, however, he is still able to realise some actions on public or protected OOB channels. We therefore need specific strands to model actions of penetrators on various types of OOB channels. To do so, we extend signed terms with a new event, $\#_o t$, meaning that penetrators suspend the message t on OOB channel o. This brand new event is only adopted for long-range public or protected OOB channels. Consequently, we extend the penetrator model with following two penetrator traces on OOB channels:

- **OVH**. Overhearing : $\langle -_o m, +m \rangle$ where o is an OOB channel of type public.
- **DRP**. Dropping : $\langle -_o m \rangle$.
- **SUS**. Suspending : $\langle -_o m, \#_o m \rangle$ where o is a OOB channel of type long-range public or protected.
- **REL**. Releasing : $\langle \#_o m, +_o m \rangle$ where o is a OOB channel of type long-range public or protected.
- **RPL**. Replaying : $\langle -_o m, +_o m, +_o m \rangle$ where o is a OOB channel of type long-range public.

The dropping attack can be modelled by SUS strand without the REL strand. Moreover, REL strand only works for a term t over a long-range public OOB channel o when there exists a SUS strand for t over o.

Having defined the penetrator model, we can now define the notion of revealed term.

Definition 3.2.7 (Revealed term). For a given bundle, a term t is called to be *revealed* at node n if:

- $t \sqsubseteq term(n)$, and t can be obtained by the penetrator using his knowledge at node n, and
- for any n' that precedes n, or $(n' \preceq n)$, such that $t \sqsubseteq term(n')$ the penetrator cannot obtain the t using his knowledge at node n'.

3.2.4 Pairing Agreement

Straightforwardly, a goal of secure pairing device protocols is to ensure that two devices with no prior shared knowledge and sharing a common OOB channel, receive the same agreement dataset after acceptance notification. To formalise corresponding security property, we adapt the definition of *agreement property* from [80] to our situation.

Definition 3.2.8 (Agreement Property). We say that a protocol ensures Initiator A *agreement* with Responder B on a set of data items ds, if whenever A (acting as Initiator) completes a protocol run, apparently with Responder B, then B has previously been running the protocol acting as Responder apparently with A, and each such run of A

corresponds to a unique run of B. Furthermore the two agents receive the same ds at the end of a run.

A penetrator can attack the protocol if at the end of its run, both devices reach to Accept state, yet having a different agreement dataset.

3.3 Analysis of Wong-Stajano Protocol

This section applies our model to analyse Wong-Stajano protocol with Unidirectional Channel. Wong and Stajano proposed in [5] new mutual authentication and key agreement protocol over bidirectional and unidirectional out-of-band channels. The out-of-band channels ensure data origin authenticity but does not provide confidentiality. Their protocols exploited a short authenticated string over visual channels which provides integrity and data origin authenticity. The Wong-Stajano (WS) protocol with unidirectional public out-of-band channel is presented in figure 3.1. Its model in our extension of Strand Spaces is defined below.

Definition 3.3.1. Given an infiltrated Strand Spaces Σ, $\mathcal{B}$ is a Wong-Stajano protocol space if Σ is a union of three kinds of strands:

- Penetrator strands $st_p \in \mathcal{B}$,
- "Initiator strand" with trace $Init[A, B, r_b, k_B, g^a, g^b]$ defined to be: $\langle +g^a, -(B, g^b, h_{k_B}(B, g^b, g^a, r_b), +_o ACK, -_{o1} r_b, -k_B\rangle$, where $B \in \mathcal{T}_{name}$, $ACK \in \mathcal{T}$, and $g^a, g^b \in \mathcal{D}\backslash\mathcal{D}_P$,
- "Responder strand" with traces $Resp[A, B, r_b, k_B, g^a, g^b]$ defined to be: $\langle -g^a, +(B, g^b, h_{k_B}(B, g^b, g^a, r_b)), -_o ACK, +_{o1} r_b, +k_B\rangle$, where $B \in \mathcal{T}_{name}$, $ACK \in \mathcal{T}$, and $g^a, g^b \in \mathcal{D}\backslash\mathcal{D}_P$,

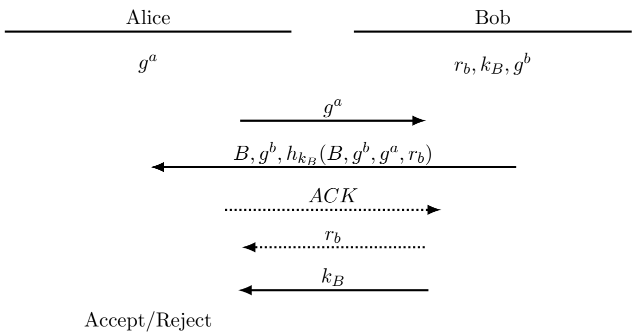


FIGURE 3.1: Wong-Stajano Protocol with Unidirectional Channel

with o a short-range public channel and $o1$ a long-range public OOB channel.

Unfortunately, agreement property does not hold for the Wong-Stajano protocol.

3.3.1 Responder's Guarantee for Wong-Stajano protocol

Responder's guarantee for Wong-Stajano protocol is stated as follows:
Let $\mathcal{B}$ be a bundle containing a strand st' in $Resp[A, B, r_b, k_B, g^a, g^b]$ of height 5. If st' uses the channel o in $\langle st', 3\rangle$ and $o1$ in $\langle st', 4\rangle$, and r_b, g^b, k_B uniquely originate on st', then $\mathcal{B}$ contains a unique strand st in $Init[A, B, r_b, k_B, g^a, g^b]$ of height 5 that also uses channel o and $o1$. Moreover, both strands agree on g^a, g^b.

Proving Responder's guarantee requires to prove following lemmas. Furthermore, these lemmas try to show existing regular(or trusted) nodes in Initiator strand.

Lemma 3.3.2. r_b uniquely originates on $\langle st', 2\rangle$

Proof. Providing that r_b uniquely originates on Σ, node $\langle st', 2\rangle$ is a positive node, and $r_b \notin K$, thus only possibility is that r_b uniquely originates at $\langle st', 2\rangle$. □

Lemma 3.3.3. There exists a regular node n_4 in Initiator strand such that $term(n_4) = -_{o1}r_b$.

Proof. Since Responder strand receives an acknowledgement from Initiator, Initiator apparently has a regular node with term $-_{o1}r_b$. Attacker cannot forge this acknowledgement because it is transmitted on an OOB channel. □

Lemma 3.3.4. There exists a regular node n_3 in Initiator strand such that $term(n_3) = -_oACK$.

Proof. Because the term of node $\langle st', 3\rangle$ is $-_oACK$, there is some Initiator strand which has a node n_3 which term is $+_oACK$. The node n_3 exploits the same OOB channel o than $\langle st', 3\rangle$. □

Then, to complete the proof of Responder's guarantee property, we should prove that:

There exists a regular node n_2 in Initiator strand such that

$$term(n_2) = -(B, g^b, h_{k_B}(B, g^b, g^a, r_b)).$$

To prove this, we should show that the term of node n_2 cannot be sent from a penetrator strand. It is easy to check for one of following strands: *M, C, R, S, K, E, D, F, H, MAC*. However we cannot conclude with the C strand.

Indeed, using the C strand, an attacker may send $(B, g^x, h_{k_B}(B, g^x, g^a, r_b))$ to Initiator strand. It supposes that he used before the MAC strand by that he knows k_B, and r_b which are normally sent after node $\langle st', 4\rangle$. The attacker may have learnt these values in a previous session. Let suppose that in a previous session, the attacker applies strand $SUS = \langle -_{o1}r_b, \#_{o1}r_b\rangle$ to suspend delivery of message to Initiator strand, and then receive k_B. If Initiator does not receive the value r_b before expiration time, he will restart a session according to the assumption. In current session, after sending $(B, g^x, h_{k_B}(B, g^x, g^a, r_b))$, the attacker can use $DRP = \langle -_{o}ACK\rangle$ to drop the ACK message on the OOB channel sent by Initiator. The attacker then executes $REL = \langle \#_{o1}r_b, +_{o1}r_b\rangle$ to deliver the message r_b to Initiator strand. Consequently, Responder's guarantee for Wong-Stajano protocol is not satisfied. Finally, after receiving the r_b message, new Initiator strand verifies MAC value in node n_2, then sends the Accept. The attack is successful. Finally, Responder cannot ensure for a regular Initiator strand.

3.3.2 Initiator's Guarantee for Wong-Stajano protocol

Initiator's guarantee for Wong-Stajano protocol is stated as follows:
Let $\mathcal{B}$ be a bundle containing a strand st in $Init[A, B, r_b, k_B, g^a, g^b]$ of height 5. If st uses a public the channel o in $\langle st, 3\rangle$ and o1 in $\langle st, 4\rangle$, and g^a is uniquely originated on st, then $\mathcal{B}$ contains a unique strand st′ in $Resp[A, B, r_b, k_B, g^a, g^b]$ of height 5 that also uses channel o and o1. Moreover, both strands agree on g^a and g^b.

As for Responder's guarantee, Initiator's guarantee does not hold for Wong-Stajano protocol. Trying to prove it leads to the attack scenario detailed in table 2.4.2 presented in the previous chapter.

3.4 Analysis of 2-Move Secure Device Pairing Protocol

We assume that:

- Participants reuse their public keys across protocol sessions.
- Hash function is perfect.
- Keyed hash function is simplified as a generic hash function exclusive-or with a key. And it is considered as a weaker version than the generic version used in the first

message. We light this assumption since length of output of keyed hashed functions is usually short over authentic channels. Therefore, attackers could exploit this weakness.

Our protocol is modelled as follows.

Definition 3.4.1. Given an infiltrated Strand Spaces Σ, $\mathcal{B}$ is the protocol space if Σ is a union of three kinds of strands:

1. Penetrator strand $st_p \in \mathcal{P}$,
2. "Initiator strand" with trace $Init[g^a, g^b, r_a, r_b]$ defined to be:
   $\langle +(g^a, h(g^a, r_a)), -(g^b, r_b), +_o(r_a \otimes h_{r_b}(g^a, g^b))\rangle$,
   where $g^a, g^b \in \mathcal{D} \backslash \mathcal{D}_P$,
3. "Responder strand" with trace
   $Resp[g^a, g^b, r_a, r_b]$ defined to be: $\langle -\{g^a, h(g^a, r_a)\}, +\{g^b, r_b\}, -_o(r_a \otimes h_{r_b}(g^a, g^b))\rangle$,
   where $g^a, g^b \in \mathcal{D} \backslash \mathcal{D}_P$,

with o a long-range public OOB channel.

We now proceed with showing correctness of our protocol by both proving Initiator and Responder guarantees.

3.4.0.1 Initiator's Guarantee

Proposition 3.4.2. Let Σ be a Strand Spaces of the protocol, and $\mathcal{B}$ a bundle containing Initiator's strand st with trace $Init[g^a, g^b, r_b, r_a]$ of height 3. If

- $g^a, g^b \notin \mathcal{D}_P$, and $g^a \neq g^b$, and
- r_a, r_b uniquely originate in Σ, and $r_a \neq r_b$,

then $\mathcal{B}$ contains Responder strand st' with trace $Resp[g^a, g^b, r_a, r_b]$. Both strands agree on g^a and g^b.

Proof. Basically, a proof proceeds according to following steps: at first we locate where r_a originates, and after that, we need to guarantee that all nodes in unique Responder's strand are regular. If any single node cannot be proved, the proof will fail. These steps are detailed in lemmas 3.4.3 to 3.4.7 below. $\square$

Lemma 3.4.3. r_a uniquely originates at $\langle st, 1\rangle$

Proof. Since r_a uniquely originates in Σ, and node $\langle st, 1\rangle$ is a positive node and the first node of strand st, then no strand other than st, can emit these terms. Therefore, r_a must originate at $\langle st, 1\rangle$. $\square$

Lemma 3.4.4. There is a regular node n_3 such that $term(n_3) = -_o(r_a \otimes h_{r_b}(g^a, g^b))$ on Responder strand.

Proof. Since $term(\langle st, 3\rangle) = +_o(r_a \otimes h_{r_b}(g^a, g^b))$, only a regular Responder strand can use the channel o to receive this message. Let call n_3 the node which receives $(r_a \otimes h_{r_b}(g^a, g^b))$, and n_3 belongs to some $Resp[*, g^b, *, r_b]$. $\square$

Note that, g^b and r_b could be sent from the attacker, hence let assume that B receives $(r_a \otimes h_{r_{xb}}(g^a, g^{xb}))$ in which r_{xb} and g^{xb} are sent from the attacker. Moreover, in case $n_3 \in SUS$, then the attacker would get r_a and r_b at this step. We will check them in following lemmas.

Lemma 3.4.5. There is a regular node n_1 such that $term(n_1) = -(g^a, h(g^a, r_a))$ on Responder strand.

Proof. Following the proof of lemma 3.4.4, Responder verifies the committed value at n_1 using r_a extracted in n_3. If the verification fails, Responder shows a Reject status immediately.

The Responder calculates $r'_a = (r_a \otimes h_{r_{xb}}(g^a, g^{xb})) \otimes (h_{r_b}(g^{xa}, g^b))$ where r_{xa}, r_{xb} and g^{xa}, g^{xb} are created by some attackers. From the assumption on keyed hash functions, $r'_a = (r_a \otimes r_b \otimes r_{xb} \otimes h(g^a, g^{xb}) \otimes h(g^{xa}, g^b))$ (1).

From following facts,

(i) r_a is revealed after seeing r_{xb},

(ii) r_{xb} is committed in $\langle st, 2\rangle$ just after $\langle st, 1\rangle$, and r_{xa} is committed before knowing r_b,

(iii) r_a and r_b uniquely originate in Σ,

(iv) hash functions are perfect,

we can deduce that attackers have not ability to generate such r'_a in (1) before n_1. Since n_1 maps to the first node of some Responder strand st' in which r_b belongs, n_1 must be a regular node. $\square$

Lemma 3.4.6. There is a regular node n_2 such that $term(n_2) = +(g^b, r_b))$ on Responder strand. Moreover, $n_1 \preceq n_2 \preceq n_3$.

Proof. Using results of lemmas 3.4.4 and 3.4.5, we have two regular nodes n_1 and n_3 in some Responder strands $Resp[*, g^b, *, r_b]$. In fact, there is a regular node n_2 with $term(n_2) = +(g^b, r_b))$ such that $n_1 \preceq n_2 \preceq n_3$. □

Lemma 3.4.7. Both participants agree on g^a and g^b.

Proof. For arbitrary *Alice*, *Bob* and r_a, if strand $st' \in Resp[g^a, g^b, r_a, r_b]$, then sign of $\langle st', 2\rangle$ is positive. Moreover, according to two previous lemmas, there is a relationship $n_1 \preceq n_2 \preceq n_3$. When $r_b \sqsubseteq term(\langle st', 2\rangle)$, there is at most one such st'.

Now, let check if g^a actually originates on st or not. Since g^a stays in the same box with r_a at $\langle st, 1\rangle$, the attacker cannot produce a term corresponding to $term(\langle st, 1\rangle)$ with a fake g^x. Therefore, g^a must originate on st.

Using the same argument, since g^b and r_b stay in the same box at n_3, and r_b uniquely originates in Σ, Initiator receives the correct g^b. □

So the protocol satisfies injective agreement for Initiator A.

3.4.1 Responder's Guarantee

Proposition 3.4.8. Let Σ be a Strand Spaces of the protocol, and $\mathcal{B}$ be a bundle containing Responder's strand st' with trace $Resp[g^a, g^b, r_a, r_b]$ of height 3. If

- $g^a, g^b \notin \mathcal{D}_P$, and $g^a \neq g^b$, and
- r_a, r_b uniquely originate in Σ, and $r_a \neq r_b$.

Then $\mathcal{B}$ contains Initiator strand st with trace $Init[g^a, g^b, r_a, r_b]$. Both strands agree on g^a and g^b.

Proof. The proof of Responder's guarantee is nearly identical to Initiator's guarantee proof. We need to verify that all nodes in a unique $Init[g^a, g^b, r_a, r_b]$ are regular. Firstly, we need to locate r_b. These steps are detailed in lemmas 3.4.9 to 3.4.13 below. □

Lemma 3.4.9. r_b originates at node $\langle st', 2\rangle$.

Proof. r_b is a subterm of the positive node $\langle st', 2\rangle$, thus it could lie on $\langle st', 1\rangle$. However, according to the assumption, r_b is neither g^a nor $h(r_a, g^a)$, and r_b uniquely originates in Σ, then r_b must originate at $\langle st', 2\rangle$. $\square$

Lemma 3.4.10. There is a regular node n_3 such that $term(n_3) = +_o(r_a \otimes h_{r_b}(g^a, g^b))$

Proof. Since Responder must receive an authenticated message over o before notifying an Accept/Reject status, it means that some Initiator has sent a message on a channel o. Therefore, let call n_3 with $term(n_3) = +_o(r_a \otimes h_{r_b}(g^a, g^b))$. Even if n_3 is on a SUS or REL strand, its value is not modified due to the reception on OOB channel. Consequently, there is a regular Initiator strand $st \in Init[g^a, *, r_a, *]$ such that $n_3 \in st$. $\square$

We note that r_b and g^b could be generated by some penetrator strands. Suppose that Responder receives r_{xb} instead of r_b, and g^{xb} instead of g^b. So the $term(n_3)$ could be $+_o(r_a \otimes h_{r_{xb}}(g^a, g^{xb}))$.

Lemma 3.4.11. There is a regular node n_1 such that $term(n_1) = +(g^a, h(g^a, r_a))$.

Proof. Assume that n_1 resides on some attackers, and $term(n_1) = +(g^{xa}, h(g^{xa}, r_{xa}))$ where g^{xa} and r_{xa} are generated by an attacker. According to last analysis in lemma 3.4.10, n_3 could lay on SUS and be reused in another session against Responder.

Now Responder calculates $r'_a = (r_a \otimes h_{r_{xb}}(g^a, g^{xb}) \otimes (h_{r_b}(g^{xa}, g^b))$, then checks if $h(g^{xa}, r'_a)$ equals $h(g^{xa}, r_{xa})$ or not.

We have,

(i) r_a, r_b uniquely originate on Σ,

(ii) r_{xa} is submitted before $\langle st', 2\rangle$,

(iii) r_a is revealed only in $\langle st', 3\rangle$,

(iv) hash functions are perfect,

hence the attacker has no way to create such r_{xa} before $\langle st', 2\rangle$ where r_b originates. Therefore, n_1 is regular node. $\square$

Lemma 3.4.12. There is a regular node n_2 such that $term(n_2) = -(g^b, r_b))$, and $n_1 \preceq n_2 \preceq n_3$.

Proof. According to the lemmas 3.4.10 and 3.4.11, we have some regular strands $st \in Init[g^a, *, r_a, *]$ such that $n_1, n_3 \in st$. Hence, st must contain $\langle st, 2\rangle$ labelled as n_2 with $term(n_2) = -(g^b, r_b))$. Finally, we have $n_1 \preceq n_2 \preceq n_3$. □

Lemma 3.4.13. Both participants agree on g^a and g^b.

Proof. For arbitrary Alice, Bob and r_b, if strand $st \in Init[g^a, g^b, r_a, r_b]$, then signs of $\langle st, 1\rangle$ and $\langle st, 3\rangle$ are positive. Moreover, according to previous lemmas, relationship $n_1 \preceq n_2 \preceq n_3$ holds. When $r_a \sqsubseteq term(\langle st, 1\rangle)$, there is at most one such st.

Now, let check if g^a actually originates on st or not. Since g^a stays in the same box with r_a at $\langle st, 1\rangle$, the attacker cannot produce a term corresponding to $term(\langle st, 1\rangle)$ with a fake g^x. Therefore, g^a must originate on st.

Using the same argument, since g^b and r_b stay in the same box at n_3, and r_b uniquely originates in Σ, Initiator receives the correct g^b. □

So the protocol satisfies injective agreement for Responder B.

3.5 Analysis of Commitment Schemes

As introduced chapter 2, modern secure device pairing protocols usually take advantage of commitment schemes to provide provable security. Hence, to process commitment-based device pairing protocols quickly, we generalise and model commitment schemes in our formalisation to offer a useful tool. To do that, we at first define a commitment scheme such that when it is recognised in a protocol, it straightforwardly results two regular strands.

3.5.1 Formalism of Commitment Schemes

To begin with, we define a commitment scheme as follows.

Definition 3.5.1 (Commitment Scheme). A commitment scheme $CS(r_a, r_b)$ for a random pair (r_a, r_b) in which $r_a \neq r_b$ contains one of these strands:

- *3-Move strand*: $+c(r_a) \Rightarrow -(r_b) \Rightarrow +d(r_a)$;
- *4-Move strand*: $+c(r_a) \Rightarrow -c(r_b) \Rightarrow +d(r_a) \Rightarrow -d(r_b)$.

A generic commitment scheme-based device pairing protocol is modelled as follows.

Definition 3.5.2. Given an infiltrated Strand Spaces Σ, $\mathcal{B}$ is the device protocol space if $\mathcal{B}$ is a union of three kinds of strands:

1. Penetrator strand $st_p \in \mathcal{B}$,
2. "Initiator strand" with trace $Init[r_a, r_b]$,
3. "Responder strand" with trace $Resp[r_a, r_b]$,

with a public OOB channel o, r_a and r_b are random numbers.

Proposition 3.5.3 (Provable Bundle). Let $\mathcal{B}$ be a bundle of a secure pairing protocol using an out-of-band channel o in which:

- a regular random pair (r_a, r_b) uniquely originates in $\mathcal{B}$, and $r_a \neq r_b$;
- a commitment scheme $CS(r_a, r_b)$ is found in $\mathcal{B}$;
- a function $f(r_a, r_b)$ is second pre-image resistant;

If the output of f is transferred over an out-of-band channel o between two regular principals, there exist two unique regular strands st and st' in $\mathcal{B}$ using f such that $r_a \in st$ and $r_b \in st'$. Moreover, both strands agree on (r_a, r_b).

Proof. Since o is used in $\mathcal{B}$, there exist at least two different strands sharing o. Let call two strands be st and st', and $term(\langle st, i\rangle) = +_o(f)$, and $term(\langle st', j\rangle) = -_o(f)$. Generally, we can assume that st is Initiator and $r_a \in st$, st' is Responder and $r_b \in st'$.

Initiator Guaranty: As described above, a regular node $\langle st', j\rangle$ associates to o such that $term(\langle st', j\rangle) = -_o(f)$. We assume that when the protocol finishes, st gets $f(r_a, r_{x2})$, st' gets $f(r_{x1}, r_b)$ where r_{x1} and r_{x2} could be generated by attackers. Then, $f(r_a, r_{x2})$ must equal $f(r_{x1}, r_b)$ since they are transferred via o.

Observing that X has to submit r_{x2} before actually knowing r_a. Similarly, X has to submit r_{x1} before actually seeing r_b. Thus irrespectively of the attacking strategy taken by X, r_a and r_b will be revealed after r_{x1} and r_{x2} have been generated and submitted. If it happens that both r_a and r_b are revealed in the same time, then we can pick an arbitrary one.

Assume that r_a is revealed after r_b, we have:

(i) r_a and r_b are independently and uniformly distributed random variables,

(ii) r_{x1} and r_{x2} must be generated and submitted before either r_b or r_a are revealed,

(iii) each principal can open at most γ sessions.

(iv) there are n participants in the network.

(v) m_{x1} and m_{x2} are possibly unchanged.

(vi) f is second pre-image resistant.

The same holds for the case where r_b is revealed after r_a. Therefore, $Pr[f(r_a, r_{x2}) = f(r_{x1}, r_b)] \leq n.\gamma.2^{-k}$, where k is the length of r_a or r_b. When k is sufficiently large, the attack is impossible.

As consequence, the protocol satisfies the agreement on (r_a, r_b) for Initiator.

Responder Guaranty is quite identical to Initiator guaranty.

Finally, $\mathcal{B}$ is provable. □

3.5.2 An Example

Let's take a simple example. The protocol presented at the figure 3.2 aims to provide a data agreement between two participants. The protocol happens as follows.

1. Alice picks a random value r_a. Bob picks a random value r_b
2. Alice sends $m, h(m, r_a)$ to Bob.
3. Bob sends r_b to Alice.
4. Alice sends r_a to Bob.
5. Alice sends $h(r_a, r_b, m)$ to Bob over a long-range public out-of-band channel.
6. Bob verifies the received value and announces the result to Alice.
7. Alice confirms by pushing an Accept button.

The protocol is modelled as follows.

Definition 3.5.4. Given an infiltrated Strand Spaces Σ, $\mathcal{B}$ is the protocol space if Σ is a union of three kinds of strands:

1. Penetrator strand $st_p \in \mathcal{P}$,
2. "Initiator strand" with trace $Init[r_a, r_b, m]$ defined to be:
   $\langle +(m, h(m, r_a)), -r_b, +r_a, +_o(h(r_a, r_b, m)) \rangle$

3. "Responder strand" with trace $Resp[r_a, r_b, m]$ defined to be:
   $\langle -(m, h(m, r_a)), +r_b, -r_a, -_o(h(r_a, r_b, m))\rangle$

with a long-range public OOB channel o and a second pre-image resistance function h.

Proposition 3.5.5. Let Σ be a Strand Spaces of the protocol, and $\mathcal{B}$ a bundle containing Initiator's strand st with trace $Init[r_a, r_b, m]$ of height 4. If r_a, r_b uniquely originate in Σ, and $r_a \neq r_b$, then $\mathcal{B}$ contains Responder strand st' with trace $Resp[r_a, r_b, m]$. Moreover, both strands agree on r_a, r_b and m.

Proof. Intuitively, $\mathcal{B}$ contains a $3-Move$ commitment scheme $CS(r_a, r_b)$ with a function $h(r_a, r_b, m)$. According to the proposition 3.5.3, $\mathcal{B}$ is a provable bundle in which two unique regular strand st and st' have $r_a \in st$ and $r_b \in st'$. Furthermore, due to partial-order relationship [1] in st', st' has a height 4.

Since $m \sqsubseteq term(\langle st, 4\rangle) = h(r_a, r_b, m)$, m is ensured for data origin authentication. As a result, st' receives a correct m after the protocol finishes. Finally, st and st' agree on m. $\square$

3.6 Out-of-band Channel Transformation

We aim in this section to propose a translation procedure that transforms a model in our previous formalism of an initial protocol with OOB channels into a model in original Strand Spaces of the protocol that does not use any OOB channel while preserves security properties of initial protocol: if there is no attack against a transformed model there is no attack against an initial model. As a result, a protocol using OOB channels can now be verified using a security protocol analyser such as [81] or [82].

[1]partial-order relationship is defined at [15]

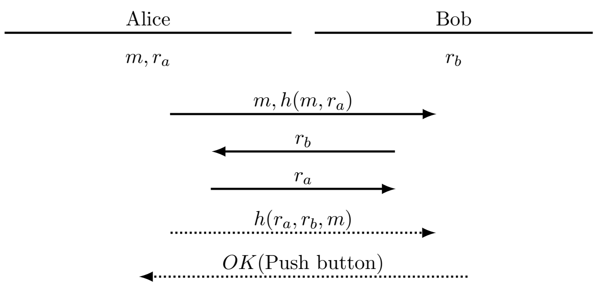


FIGURE 3.2: A Simple Data Agreement Protocol

3.6.1 Related Work

To our knowledge, out-of-band security property is only partially addressed in existing work. Most of current methods only deal with private or protected channels and simulate use of these out-of-band channels via a set of pre-shared or public keys, for example in [83–85].

Our approach is similar to Gavin Lowe and al. approach in [86] and [87]. In these studies, they specified specifications of out-of-band channels that are provided when a secure transport channel is used. Properties include confidentiality, no faking, no hijacking, and no redirecting. Then, the authors illustrated them via some cryptographic protocols using a set of private and public keys, but did not provide the correctness of these protocols. This work were implemented in Casper/FDR verification tool[81]. Basically, the differences between this work and ours are (i) they only consider transport layer while we can take into account both physical layer and transport layer, (ii) we provide a specific model of penetrator's abilities on OOB channel while they limit penetrator's abilities on out-of-band channels, and (iii) we ensure the correctness of protocols.

Another security protocol verification tool, Proverif [82], integrates out-of-band channels. There are two kinds of channels formalised in Proverif: *public* and *private*. Attacker can overhear on a public channel, whereas they cannot do anything on a private channel. Penetrator's capabilities are thus more limited than in our model.

3.6.2 Channel Property Transformation

The concept of the translation is simulating all security properties offered by each out-of-band channel by equivalent ones provided by cryptographic schemes. Firstly we state underlying assumptions:

(A1) All regular participants are honest.

(A2 All regular participants have pre-shared their identities, public keys, or pre-shared secret keys.

(A3) All regular keys are not known by any penetrator.

(A4) Each honest participant runs one instance of the protocol at a time.

To implement this idea, we introduce four specific cryptographic shapes [2], or sub-bundle $\mathcal{B}_o^{SP}(m, i)$ containing a sending skeleton $sk_o^e(m, i)$, and a receiving skeleton $sk_o^r(m, i)$,

[2] Definitions of shape and skeleton are defined at [88]

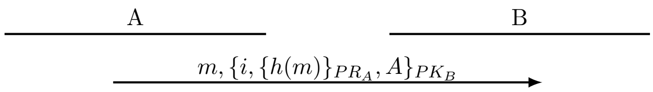


FIGURE 3.3: Shape 1

offering the same security quality as ${}_o(m)$ does. i is the index of ${}_o(m)$ in the original protocol.

In these shapes, Alice (A) and Bob (B) wish to agree on a message m. Attackers win if one of two participants gets different m.

To provide data origin authentication, a hash of m is encrypted by a public or private key. m is encrypted by public keys to provide data confidentiality. Keys are apparently assumed to not belong to any attacker.

Following definition allows characterising a fact that a principal receiving a message is able to access a given subterm of this message. In particular, if this subterm is encrypted, the principal owns keys to decrypt, and this subterm is not masked by a non-invertible function (hash or keyed-hash function).

Definition 3.6.1 (Extractable). t is called *extractable* from t' if $t \sqsubseteq t'$, and t is obtained by applying a limited number of operations of splitting and decryption with a set of keys k into t'.

Model of a long-range public channel in original Strand Spaces

Let $\mathcal{B}_{lrp}^{SP}(m, i)$ be a shape in the classical Strand Spaces pictured in figure 3.3 . $\mathcal{B}_{lrp}^{SP}(m, i)$ provides data origin authentication on m for Responder as a long-range public channel does. In this scheme, $\{h(m)\}_{PR_A}$ protected under PK_B will ensure integrity and origin of the message.

Proposition 3.6.2. Considering assumptions (A1) to (A4), the shape $\mathcal{B}_{lrp}^{SP}(m, i)$ containing two skeletons $sk_{lrp}^{e}(m, i)$ and $sk_{lrp}^{r}(m, i)$ holds data origin authentication for m.

Proof. Initiator's guarantee: Since its keys are not owned by any attacker, Initiator's messages cannot be forged. Additionally, m is signed by Initiator's private key, and is extractable. Therefore, $\mathcal{B}_{lrp}^{SP}(m, i)$ holds data origin authentication for $sk_{lrp}^{e}(m, i)$. Note that, attackers can drop messages, but Initiator goals are still satisfied.

Responder's guaranty: Using the unsolicited test [78] for uncompromised keys, the Responder skeleton is able to verify existence of the regular node on Initiator skeleton.

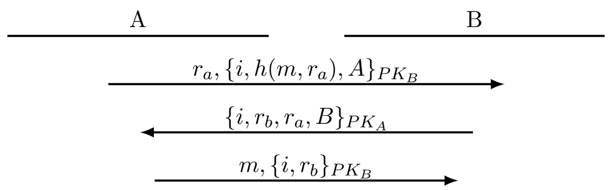


FIGURE 3.4: Shape 2

Data origin authentication of m is obtained in the hash value covered by Initiator's private key. m is clearly extractable. Finally, $\mathcal{B}^{SP}_{lrp}(m, i)$ holds data origin authentication for $sk^{r}_{lrp}(m, i)$. $\square$

Model of a short-range public channel in original Strand Spaces

We simulate a non-suspend channel by a unique instance of Initiator and a corresponding Responder in each protocol run. Precisely, in our proposed scheme, each side can ensure the unique execution of the other side. As the result of that, when a message is revealed, an attacker cannot reuse it in other sessions.

Let $\mathcal{B}^{SP}_{srp}(m, i)$ the shape displayed at figure 3.4 modelling in the original Strand Spaces. $\mathcal{B}^{SP}_{srp}(m, i)$ offers data origin authentication, non-replaying attack against Responder, and unique execution of each participant in each protocol session. As a consequence, $\mathcal{B}^{SP}_{srp}(m, i)$ provides all security properties as a short-range public out-of-band channel does. Additionally, we let r_a visible to attackers purposely because we allow dropping attack. By this way, attackers can produce the second message to obtain m, but definitely cannot reuse it.

We use a commitment scheme to keep away replaying attack. A firstly commits a value $h(m, r_a)$, then releases m after receiving a random challenge r_b from B. Although an attacker may produce a fake second message to obtain m, B is still able to verify the fresh of m by checking the value of r_b and r_a used in current session with A. r_b apparently is known only by A as encrypted by the A's public key. Meanwhile, the last message similar to one in $\mathcal{B}^{SP}_{lrp}(m, i)$ offers data origin authentication.

Proposition 3.6.3. Considering assumptions (A1) to (A4), $\mathcal{B}^{SP}_{srp}(m, i)$ containing two skeletons $sk^{e}_{srp}(m, i)$ and $sk^{r}_{srp}(m, i)$ holds data origin authentication, and unique execution of each sides.

Proof. Initiator's guarantee: Since its keys are not owned by any attacker, only Initiator can open the second message to obtain r_b. Therefore,the third message cannot be forged.

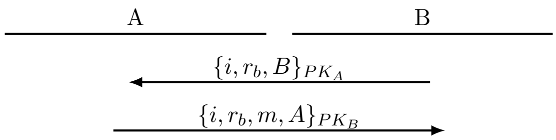


FIGURE 3.5: Shape 3

Hence, data origin authentication of m is ensured in the message received by Responder. Moreover, m is clearly extractable by Responder's private key.

We also have an outgoing test edge $\langle sk^e_{srp}(m,i), 1\rangle \Rightarrow \langle sk^e_{srp}(m,i), 2\rangle$ for a nonce r_a. Initiator ensures the existence of Responder, but it could be an attacker. Nevertheless, according to our assumption on uncompromised keys, and unique origination of r_b, Initiator knows attackers cannot forge or reuse the third message. Finally, there is a unique execution of Responder.

Responder's guarantee: Firstly, m is extractable in third message. Since the private key PR_A does not belong to attacker's keys, the attacker cannot generate the third message due to r_b encapsulated by Initiator's public key.

According to following reasons:

(i) Using the authentication test 2 [78] for r_b, Responder is able to verify existence of the two last regular nodes of Initiator skeleton.

(ii) Additionally, uniquely origination of r_a allows B to ensure for the unique Initiator.

(iii) Data origin authentication of m is provided by the hash value in the first message.

$\mathcal{B}^{SP}_{srp}(m,i)$ holds unique execution and data origin authentication for $sk^r_{srp}(m,i)$. □

Model of a protected channel in original Strand Spaces

Let be $\mathcal{B}^{SP}_{pro}(m,i)$ a shape described at figure 3.5 which offers data confidentiality of m and prevents Responder from replaying attack.

Clearly, due to the assumption on uncompromised keys , B ensures that m is confidential as encrypted by B's public key. Meanwhile, r_b plays as a challenge to offer non-replaying attack on m. Additionally, only A can open the first message, B ensures the origin of m.

Proposition 3.6.4. Considering assumptions (A1) to (A4), the bundle $\mathcal{B}^{SP}_{pro}(m,i)$ containing two skeleton $st^e_{pro}(m,i)$ and $st^r_{pro}(m,i)$ offers data confidentiality for m and prevents Responder from replaying attack.

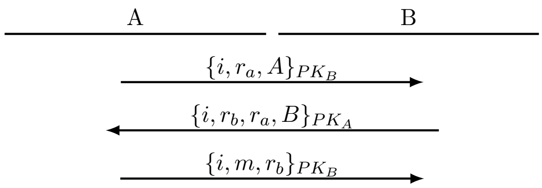


FIGURE 3.6: Protocol 4

Proof. Initiator's guarantee: Firstly, m is extractable and secure in second message since encrypted by Responder's public key. Assume that r_b uniquely originates in $\mathcal{B}^{SP}_{pro}(m, i)$, Initiator can ensure for the unique Responder skeleton in $\mathcal{B}^{SP}_{pro}(m, i)$. Finally, $\mathcal{B}^{SP}_{pro}(m, i)$ holds properties as data confidentiality, data origin authentication and non-replaying attack for $st^e_{pro}(m, i)$.

Responder's guarantee: Firstly, m is extractable and secure in second message by Responder's public key. When r_b uniquely originates in $\mathcal{B}^{SP}_{pro}(m, i)$, Responder ensures the existence of regular Initiator skeleton in $\mathcal{B}^{SP}_{pro}(m, i)$ according to the outgoing authentication test 2. Finally, $\mathcal{B}^{SP}_{pro}(m, i)$ holds properties as data confidentiality and non-replaying attack for $st^r_{pro}(m, i)$. □

Model of a private channel in original Strand Spaces

Let $\mathcal{B}^{SP}_{pri}(m, i)$ be the shape described at figure 3.6 which offers data confidentiality, data origin authentication, and prevents replaying, suspending and dropping attacks for participating strands.

We simulate non-dropping channel by that whenever attackers drop any message, the protocol will be stop at that point. Actually, $\mathcal{B}^{SP}_{pri}(m, i)$ is a variant of NS protocol [15] which holds injective agreement and secrecy on (r_a, r_b) between two participants. Injective agreement means that there are unique run of both participants in this protocol. As a result, replaying attacks are clearly avoided. Suspending attack apparently is useless as well. Meanwhile, m in encrypted by B's public key, so m is ensured data confidentiality.

Proposition 3.6.5. Considering assumptions (A1) to (A4), the shape $\mathcal{B}^{SP}_{pri}(m, i)$ containing two unique skeletons $sk^e_{pri}(m, i)$ and $sk^r_{pri}(m, i)$ offers data origin authentication, data confidentiality for m.

Proof. The protocol was proved in [15] to hold the injective agreement between Initiator and Responder on r_a and r_b. Since m is protected under PK_B, attackers cannot overhear m. Hence, both skeletons holds properties as data origin authentication and data confidentiality. □

Out-of-band Channel Translation

As can be seen above, four schemes could offers the same security properties as out-of-band channels do. Therefore, they can be used to simulate the use of out-of-band channels when a protocol is modelled in original Strand Spaces. Let's present how it works.

Let $st = \langle n_1, ..., n_{k-1}, n_k, n_{k+1}, ...\rangle$. We say that if a skeleton sk hooks into a strand st at the node n_k , then $st = \langle n_1, ..., n_{k-1}, sk, n_{k+1}, ...\rangle$

Definition 3.6.6 (OOB Replacement). The *oob replacement* replaces a $term(\langle st, i\rangle) =_o \{m\}$ in a strand st by corresponding sending or receiving skeleton that hooks into st at n. Precisely,

(i) $+_{lrp}(m)$ and $-_{lrp}(m)$ are replaced by $st^e_{lrp}(m, i)$ and $st^r_{lrp}(m, i)$ respectively;

(ii) $+_{srp}(m)$ and $-_{srp}(m)$ are replaced by $st^e_{srp}(m, i)$ and $st^r_{srp}(m, i)$ respectively;

(iii) $+_{pro}(m)$ and $-_{pro}(m)$ are replaced by $st^e_{pro}(m, i)$ and $st^r_{pro}(m, i)$ respectively;

(vi) $+_{pri}(m)$ and $-_{pri}(m, i)$ are replaced by $st^e_{pri}(m, i)$ and $st^r_{pri}(m, i)$ respectively;

Definition 3.6.7 (OOB Equivalent Strand). Let st^{ESP} be a strand of a protocol modelled in extended Strand Spaces. The strand st^{SP} is called be an *OOB equivalent strand* modelled in original Strand Spaces model for st^{ESP} if st^{SP} is the strand after applying the OOB replacement to all out-of-band messages in st^{ESP}.

By extension to the bundle, we have an OOB equivalent bundle as follows.

Definition 3.6.8 (OOB Equivalent Bundle). A bundle $\mathcal{B}^{SP}$ is called an *OOB equivalent bundle* for a bundle $\mathcal{B}^{ESP}$ modelled in extend Strand Spaces if $\mathcal{B}^{SP}$ consists of corresponding OOB equivalent strands of all ones in $\mathcal{B}^{ESP}$.

Note that, OOB equivalent bundle only applies for a normal execution of protocol in which no penetrator strand exists.

3.6.3 Attack Transformation

We simulate the suspending event by a storing penetrator strand which receives any message, and reuses it later. Precisely, the storing penetrator strand does not modify message content, instead of receiving the message from the other strand and sending it again.

Definition 3.6.9 (Storing Penetrator Strand). st_p^s is a *storing penetrator strand* if $\forall n' \in st_p^s, sign(n') = +$, then $\exists n \in st_p^s, sign(n) = -, term(n) = term(n')$.

Let t_1, t_2, t_3 respectively denote terms of first, second (if existing) and third (if existing) message in the four shapes defined in previous section. Let r_x denote a nonce generated by penetrator. Let st_p^s be a storing penetrator strand. $l, l1 \in st_p^s$ such that $term(h) = term(t1), term(l1) = term(l)$, and $term(l_2) = term(t_2)$. The table 3.1 depicts attacker's capabilities on each $\mathcal{B}_o^{SP}(m, i)$. The set of these strands is noted as $\mathcal{X}^{SP}$.

A natural question arising is: if there an attack against a transformed protocol in original Strand Spaces is found, is there a corresponding attack on the initial protocol in extended Strand Spaces? This problem is discussed in the subsection 3.6.5.

3.6.4 Proofs

The proof we will obtain in this paper follows a simple concept to establish the desired results: *A protocol bundle is secure in extended Strand Spaces model if its equivalent OOB bundle is secured.* Or, *whenever there is an attack on extended Strand Spaces model, there is an attack on the equivalent OOB bundle.* Formally presenting, we state this concept into a proposition as follows.

Proposition 3.6.10. Let $\mathcal{B}^{ESP}$ be a normal execution of a protocol $\mathcal{P}$ modelled in the extended Strand Spaces, and $\mathcal{B}^{SP}$ be an OOB equivalent bundle of $\mathcal{B}^{ESP}$. Whenever there is an attack against $\mathcal{B}^{ESP}$, then there is an attack against $\mathcal{B}^{SP}$.

This proposition is proved by lemmas from 3.6.11 to 3.6.14. The general idea of the proving is using its knowledge and capacities on terms in $\mathcal{B}^{ESP}$ to violate $\mathcal{B}^{ESP}$'s goals, an attacker can learn the same way to exploit $\mathcal{B}^{SP}$'s. At first we show that any term in $\mathcal{B}^{ESP}$ will appear in $\mathcal{B}^{SP}$.

Lemma 3.6.11. Let $\mathcal{A}^{ESP}$ be a set of all terms in $\mathcal{B}^{ESP}$, and $\mathcal{A}^{SP}$ be a set of all terms in $\mathcal{B}^{SP}$. For all $t \in \mathcal{A}^{ESP}$, then $t \in \mathcal{A}^{SP}$.

TABLE 3.1: ATTACK TRANSFORMATION FROM EXTENDED STRAND SPACES TO ORIGINAL STRAND SPACES

Attack	Type of Shapes			
	Long-range Public	Short-range Public	Protected	Private
OVH^{SP}	$\langle -t1, +t1, +m \rangle$	$\langle -t3, +t3, +m \rangle$	$\emptyset$	$\emptyset$
SUS^{SP}	$\langle -t1, +l \rangle$	$\emptyset$	$\langle -t2, +l \rangle$	$\emptyset$
REL^{SP}	$\langle -l, +l1 \rangle$	$\emptyset$	$\langle -l, +l2 \rangle$	$\emptyset$
DRP^{SP}	$\langle -t1 \rangle$	$\langle -t1,$ $+(\{i, r_x, r_a\}_{PK_A})$ $-t3 \rangle$	$\langle -t2 \rangle$	$\emptyset$
REP^{SP}	$\langle -t1,$ $+l1, +l1 \rangle$	$\emptyset$	$\emptyset$	$\emptyset$

Proof. According to the sub-sections 3.6.2, term m in all of four shapes $\mathcal{B}^{SP}_{o}(m,i)$ has the same structure and direction as one in $_o(m)$. For instance, message $_{lrp}(m,i)$ from A to B is presented as $(m, \{i, \{h(m)\}_{PR_A}, A\}_{PK_B})$ in $\mathcal{B}^{SP}_{lrp}(m,i)$. Moreover, defined in the definition 3.6.6, the replacement only happens on out-of-band terms. As a consequence, $\forall t \in \mathcal{A}^{ESP}, t \in \mathcal{A}^{SP}$. $\square$

We prove that the order of terms in $\mathcal{B}^{ESP}$ will be preserved in $\mathcal{B}^{SP}$. For an example in 3.5.2, strand $st = \langle +(m, h(m, r_a)), -r_b, +r_a, +_{lsp}(h(r_a, r_b, m))\rangle$ will be interpreted into $st^{SP} = \langle +(m, h(m, r_a)), -r_b, +r_a, +(h(r_a, r_b, m), \{4, \{h(r_a, r_b, m)\}_{PR_A}, A\}_{PK_B})\rangle$. Clearly, the term r_a in $\langle st, 3\rangle$ is placed before the term $h(r_a, r_b, m)$ in $\langle st, 4\rangle$ while the same term r_a in $\langle st^{SP}, 3\rangle$ is also placed before the term $h(r_a, r_b, m)$ in $\langle st^{SP}, 4\rangle$. The lemma below states this.

Lemma 3.6.12. $\mathcal{B}^{SP}$ preserves partial ordering $\preceq$ in $\mathcal{B}^{ESP}$.

Proof. Assume that $n \preceq n' \in \mathcal{B}^{ESP}$, and $t \sqsubseteq term(n) = +_o(m), t' \sqsubseteq term(n')$, there exists nodes $n_{sp}, n'_{sp} \in \mathcal{B}^{SP}$ so that $t \sqsubset term(n_{sp}), t' \sqsubseteq term(n'_{sp})$, and $n_{sp} \preceq n'_{sp}$. This is obtained by following facts.

- According to lemma 3.6.11, $\forall t, t' \in \mathcal{A}^{ESP}$, then $t, t' \in \mathcal{A}^{SP}$; and directions of t and t' in $\mathcal{B}^{ESP}$ are preserved $\mathcal{B}^{SP}$.
- After the replacement, $sk^{e}_{o}(m,i)$ is positioned before n'_{sp} in $\mathcal{B}^{SP}$ where term $term(n'_{sp})$ $= term(n')$;
- $\exists n_{sp} \in sk^{e}_{o}(m,i)$ so that $t \sqsubseteq term(n_{sp})$.

As the result of these, $n_{sp} \preceq n'_{sp}$. Likewise, we have the same proof if $term(n) = -_o(m)$, or $term(n') =_o (m)$. $\square$

We have proved that terms and their relationship in $\mathcal{B}^{ESP}$ are preserved in$\mathcal{B}^{ESP}$. Hence, attacker's capacities on any term in $\mathcal{B}^{ESP}$ are similar to ones on this term appearing in $\mathcal{B}^{SP}$. The below lemma shows this.

Lemma 3.6.13. Attacker's knowledge and capabilities in $\mathcal{X}^{ESP}$ on $_o(m)$ are similar to ones in $\mathcal{X}^{SP}$ on $\mathcal{B}^{SP}_{o}(m,i)$.

Proof. We are going to give a proof of a long-range public out-of-band channel. The proofs of other channels can be obtained in the same way. Assume that, the message $_{lrp}(m)$ is transmitted over a long-range public channels. Intuitively, attackers can overhear, suspend, release, drop, and replay the message.

According to the table 3.1, for the message $(m, \{i, \{h(m)\}_{PR_A}, A\}_{PK_B})$:

- m is extracted by OVH^{SP} and S;
- the message is received and held in a storing penetrator strand st_p^s by SUS^{SP};
- the message is sent from st_p^s by REL^{SP};
- the message is copied and replayed by REP^{SP};
- the message is dropped by DRP^{SP}

The attacker obviously cannot modify the message $(m, \{i, \{h(m)\}_{PR_A}, A\}_{PK_B})$ since the keys are pre-authenticated. Finally, what the attacker can do with $_{lrp}(m)$ is similar to what he does with $(m, \{i, \{h(m)\}_{PR_A}, A\}_{PK_B})$. □

As a consequence, when attacker knows exactly what he had and what he did in $\mathcal{B}^{ESP}$, he can produce the same process on $\mathcal{B}^{SP}$ to obtain his goal. The lemma below presents formally that when there is an execution $\mathcal{B'}^{ESP}$ of a protocol $\mathcal{P}$ which contains a sequence of attacker's events, a nearly similar sequence can be found in another execution of the OOB equivalent bundle of $\mathcal{P}$.

Lemma 3.6.14. If $\mathcal{B'}^{ESP}$ is an execution of a protocol $\mathcal{P}$ in which a attacker strand violates goals of $\mathcal{B}^{ESP}$, then $\mathcal{B'}^{SP}$, another shape of $\mathcal{B}^{SP}$, has an attacker strand violates $\mathcal{B}^{SP}$'s goals (similar to $\mathcal{B}^{ESP}$'s goals).

Proof. In general, we can assume that the goals of $\mathcal{B}^{ESP}$ are agreement and secrecy on a data set ds between participants A and B. Attackers win the game if A and B receive different ds. We assume that $\mathcal{B'}^{ESP}$ contains an attacker strand so that the attacker wins this game. Apparently, A has ds_A while B has ds_B, and $ds_A \neq ds_B$.

Mechanically, ds_A or ds_B are constructed by a sequence of events based on attacker's knowledge about terms, the protocol structure and his capabilities (precisely on Dolev-Yao strands, and $\mathcal{X}^{ESP}$). For example, to produce $\{m, A\}_k$, the attacker uses OVH on $+_o(m)$ from A to B somewhere in the protocol, SUS to suspend the message $+_o(m)$, REL to release the message, K to generate k, and E to produce $\{m, A\}_k$.

Provided in the lemma 3.6.13, attacker's capabilities in $\mathcal{X}^{ESP}$ on any term $t \in \mathcal{A}^{ESP}$ are similar ones in $\mathcal{X}^{SP}$ on t in $\mathcal{A}^{SP}$. For instance, an attacker's sequence in $\mathcal{B'}^{ESP}$ such as (OVH, SUS, REL, K, E) can be interpreted as (OVH^{SP},SUS^{SP}, REL^{SP}, K, E) to produce $\{m, A\}_k$ when $+_o(m)$ is translated into $sk_o^e(m, i)$.

Therefore, regarding to lemma 3.6.12, and in the same way the attack constructs ds_A or ds_B in $\mathcal{B'}^{ESP}$, the attacker can produce a sequence of events to construct ds_A or ds_B against goals of $\mathcal{B}^{SP}$ according to its knowledge and capacities on term $t \in \mathcal{A}^{ESP}$ and $t \in \mathcal{A}^{SP}$. Finally, $\mathcal{B'}^{SP}$ is another execution of $\mathcal{B}^{SP}$ which contains this sequence. □

3.6.5 Reversed Attack

The problem is when an attack against the transformed protocol is found in $\mathcal{B}^{SP}$, does a corresponding attack exist $\mathcal{B}^{ESP}$? If it is the case, is it possible to figure it out?

For the long-range public, we easily replace the $\langle -t1, +l \rangle$ by SUS strand, $\langle -t1, +t1, +m \rangle$ by OVH strand, and $\langle -l, +l1 \rangle$ by REL strand, and $\langle -l_1, +t1, +t1 \rangle$ by REP strand. Then we remove other related messages using the index number i, and remove related messages. The protected channels can proceed by the same way.

However, the case of a short-range public channel is more complicated. We have to search for patterns corresponding to short-range public channel in attack bundle. Whenever we find a negative node having a third message form, we must look for previous nodes using the index i. If they exist, then we remove them and replace the third one with DRP strand. Otherwise, if they do not exist, we replace the third message with OVH message, then remove related nodes in attack bundle using the i index, and finally remove $sk_o^r()$ and $sk_o^e()$.

Eventually, by this manual way, we can reverse an attack on original Strand Spaces into an attack on Strand Spaces. However, we note that if the protocol features messages sent on insecure channels similar to messages obtained when transforming bundles using OOB channels, we cannot conclude.

3.6.6 Example

In this part, we analyse Wong-Stajano protocol [5] using the out-of-band translation. The transformation of the protocol in original Strand Spaces, displayed in 3.7, is formally defined below.

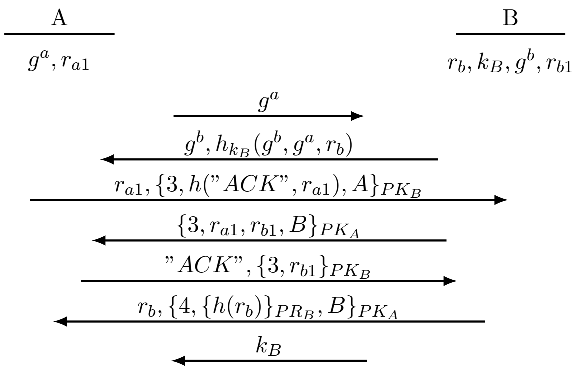


FIGURE 3.7: Transformed Wong-Stajano Protocol with Unidirectional Channel

Definition 3.6.15. Given an infiltrated Strand Spaces Σ, $\mathcal{B}$ is a model of Wong-Stajano protocol if Σ is the union of the three following kind of strands:

- *Penetrator strands* $st_p \in \mathcal{B}$,
- *Initiator strand* with strand: $Init[A, B, r_b, k_B, g^a, g^b, r_{a1}, r_{a1}, PK_A, PR_A, PK_B, PR_B]$ where $A, B \in \mathcal{T}_{name}$, and $g^a, g^b \in \mathcal{D} \backslash \mathcal{D}_P$,
- *Responder strand* with strand: $Resp[A, B, r_b, k_B, g^a, g^b, r_{a1}, r_{b1}, PK_A, PR_A, PK_B, PR_B]$ where $A, B \in \mathcal{T}_{name}$, and $g^a, g^b \in \mathcal{D} \backslash \mathcal{D}_P$,

Initiator's guarantee for transformed Wong-Stajano protocol is stated as follows:

Let $\mathcal{B}$ be a bundle containing a strand st' in
$Init[A, B, r_b, k_B, g^a, g^b, r_{a1}, r_{b1}, PK_A, PR_A, PK_B, PR_B]$ *of height 7.*
If r_{a1} uniquely originates on st, then $\mathcal{B}$ contains a unique strand st' in
$Resp[A, B, r_b, k_B, g^a, g^b, r_{a1}, r_{a1}, PK_A, PR_A, PK_B, PR_B]$ *of height 7.*
Moreover, both strands agree on g^a, g^b.

Proposition 3.6.16. Initiator's guarantee does not hold in transformed Wong-Stajano Protocol.

Proof. To prove Initiator's guarantee, we must ensure that all of nodes of Responder strand are regular. Let analyse them below. Since st receives a term

$$(r_b, \{4, \{h(r_b)\}_{PR_B}\}_{PK_A})$$

, we are sure that that a regular node, called $\langle n', 6\rangle$, owes this term. This node $\langle n', 6\rangle$ belongs to a regular strand st' in $Resp[A, B, r_b, k_B, *, g^b, *, r_{b1}, PK_A, PR_A, PK_B, PR_B]$.

Now, let analyse other positive nodes in st' including $\langle st', 2\rangle, \langle st', 4\rangle, \langle st', 7\rangle$. Observing that these nodes do not correspond to any authentication tests, they could be on some penetrator strands.

Looking at node $\langle st', 2\rangle$, despite of the origination of r_b in this node, r_b cannot be seen by current Initiator strand. Then, when r_b is received in $\langle st, 6\rangle$, it could be sent from other previous sections rather than the current section.

As a matter of fact, if an attacker X obtains r_b in a previous Initiator session with a regular Responder strand, attackers can reopen a session as a fake Responder, and later use r_b to reproduce $term(\langle st', 2\rangle)$. Formally saying, the penetrator strand is the following:

$$\langle -r_b, -k_X, -g^a, -g^x, +\{B, g^x, h_{k_X}(B, g^x, g^a, r_b)\}\rangle$$

where k_X and g^x are generated by the attacker. Intuitively, reusing $\{r_b, \{4, \{h(r_b)\}_{PR_B}\}_{PK_A}\}$ of previous section, and generating term($\langle st', 4\rangle$)and term($\langle st', 7\rangle$), attackers easily reproduce a fake Responder strand st' in order to finish a proper protocol run with current Initiator.

Finally, Initiator's guarantee does not hold.

□

From the counter-example found above for the transformed Wong-Stajano protocol, it is possible to rebuild the attack found in 2.4.2 against original Wong-Stajano protocol by proceeding a reverse analysis as presented in subsection 3.6.5.

Additionally, when we analyse the transformed Wong-Stajano protocol in AVISPA [89], we get the same result as we did in Strand Spaces model (the source code of AVISPA is attached in the appendix B). Through this result, we strongly believe that our implementation can work on other automatic security verification tools.

3.7 Conclusion

In this chapter, we extended the original Strand Spaces model to be able to analyse secure device pairing protocols. To achieve this, we modified the model so that it becomes possible to take into account protocols using several kind of channels, including OOB channels. The penetrator model has been adapted as consequence. This extension was used to formalise and analyse the Wong-Stajano mutual authentication protocol with unidirectional OOB channel. It successfully pointed us some flaws in the Wong-Stajano protocol that have never been noticed before to our knowledge.

Aforementioned works on this topic, mainly apply existing verification tools initially. We rather chosen to define a dedicated formalism able to model the specificities of device pairing protocols in a natural manner, and the results obtained so far seems promising.

In last contribution of this chapter, we gave an interpretation of these properties into cryptographic schemes by defining out-of-band equivalent strand and out-of-band equivalent bundle that describes a protocol modelling in our extended Strand Spaces into original theory. We also presented results that show how protocol's goals and attacks are translated.

In close future, we continue studying more out-of-band channel-based protocols using our mapping function. We will also analyse them using automatic security verification tools like AVISPA [89], Casper/FDR [81], and Proverif [83].

Chapter 4

Secure Neighbour Discovery Protocols

We have already presented a set of device pairing protocols and formally analysed them at the previous chapters. Basically, by adapting human assistance, wireless devices can establish secure connections among them. Before this stage, each device must know existence of its partners. The ability to determine the existence of participants within physical range in many systems from cellular infrastructure-based networks, wireless local area network to sensor networks, and short range wireless technologies is fundamental ranging problem. Thus, security mechanisms for ranging protocols should be seriously considered. Particularly, a secure ranging protocol must validate correctly both distance between participants and identification of participants.

One potential threat is that due to varieties of wireless interfaces with different signal power, false results from distance ranging and validating processes in many neighbour discovery mechanisms might appear. The threat is an old-school problem in traditional wireless networks, but it becomes serious in distributed wireless systems with heterogeneous devices. Attackers can take advantage to generate false connections, that significantly reduces stability and security of the systems. This has not mentioned before.

We put ourselves deeply between these problems, and find that time-based, or location-based neighbour discovery protocols cannot obtain correctly ranging goals. Additionally, some of them were exploited at the connection validating process. Meanwhile, current formal verification reasoning about security neighbour discovery protocols cannot resolve the problems. This motivates us to study more secure neighbour discovery mechanisms in context of Internet of Things where a huge amount of heterogeneous devices are inter-operating.

Contributions of this chapter are following:

1. We present existing neighbour discovery protocols, address their limitations, and present some incorrect existing protocols.
2. We adapt our formalisation on Strand Spaces with some extensions on physical characteristics and some helpful propositions to deal with communication statuese among principals.
3. Our model allows us obtain a notable result. We prove that time-based or distance-based neighbour discovery schemes cannot confidentially achieve their goals due to difference of physical signal power of principals.

Chapter 4 begins with a comprehensive survey of current proposals of secure neighbour discovery techniques, and vulnerabilities. We then present some incorrect protocols and conduct correct one. We introduce formalisation based on Strand Spaces in the next part. An analysis of ADVSIG [6] protocol concludes this chapter.

4.1 Overview on Neighbour Discovery Protocols

Neighbour discovery is the process by which a device in a network determines the number and identity of other in its vicinity. In wireless context, neighbours of a device are usually defined as ones that lie within its radio range. Devices considered as neighbours may cooperate in the performance of various tasks such as communications, sensing, and localisation. For instance, in perfect environment with no obstacle and noise, a device can determine its honest neighbours using flying time measurement. Particularly, a device, called A, broadcasts a greeting message to a potential device, called B, then B replies to A quickly after receiving this message. Afterwards, when completely observing the feed-back message, A measures the time-of-flight, and multiplies it by signal propagation speed in current medium to obtain the distance. If distance is lower than a pre-defined threshold, A concludes that B is its neighbour.

In this chapter, we need to explicitly distinguish between different types of links. A link is a generic term denoted a relationship between two participants. There are two kinds of links: *logical* and *physical*. While a logical link describes existence of path of a message from one strand to another, a physical one describes a directional and physical path without any relaying point from one strand to another one. Moreover, each type of link has two states: *unidirectional* and *bidirectional*. In one hand, a *unidirectional* link from A to B states that B can receive messages from A, but A cannot receive messages from B. In the other hand, a *bidirectional* link from A to B states that A and B can receive messages from each other.

Neighbour discovery protocols are apparently vital parts in current systems. Nevertheless, these protocols in wireless environment are easily abused by malicious activities of attackers. Tackling this problems, many approaches for securely discovering neighbours have been proposed.

In this session, we present some applications of neighbour discovery, their threats and vulnerabilities. Then we recap existing secure approaches in categories.

4.1.1 Neighbour Discovery Applications

Neighbour discovery protocol is discovered under a form of a principal part of many applications such as physical authentication, network authentication, routing built-up, and localisation. This following examples are referred from [90].

Physical Authentication

In some applications, value of distance between two devices is vital to assess authentication goals. For instance, in Passive Keyless Entry and Start [91], a RFID reader can estimate a travelling message flying time to imply distance to companion tags. Similarly to RFID, NFC technology allows smartphones to communicate with other devices such as speakers, headphones or even other smartphones in very short proximity. As the fact of that, neighbour discovery enables devices to find each other.

Network Authentication

Wireless network demands devices to be authenticated before they access and communicate with others. For instance, when a telephone is willing to access the Internet via a WLAN access point, it must discover the access point in the same physical range. For that purpose, neighbour discovery process is a primary part for wireless network authentication.

Localisation

When a device wishes to know its current location, it advertises HELLO messages to close GPS satellites, or close WLAN access points to obtain its location information. However, when GPS or WLAN signal is disabled by noise and obstacles, weather conditions, tree cover, or surrounding buildings, or walls, neighbours' location information could help. For instance, a car cannot locate itself in a long tunnel, then it derives its own location by asking surrounding cars. Hence, these processes apparently stand on a neighbour discovery protocol.

Routing Built-up

In ad-hoc network, when a device wishes to deliver a message from itself to a destination, it constructs a potential network topology by looking for many or all nodes in the network. According to the current network topology, an appropriate path will be picked up. In fact, exploring neighbours is always a primary step at the beginning.

4.1.2 Threat and Vulnerabilities

Classification of threats and vulnerabilities in neighbour discovery protocols is normally painful due to these following reasons. (i) Attackers could be either legitimate principals or outside attackers, or both. (ii) Some attacks happen across layers from physical layer to network one. Therefore, there are two ways to classify the threat and vulnerabilities. One is grouping attacks into internal or external attacks, and the other is grouping attacks into spoofing, relaying, or tunnelling attacks.

Attacks in the first way are:

- **Internal attacks** are types of attacks where attackers compromise several honest participants. Subsequently, they can imitate all honest behaviours, and intentionally generate fake information.
- **External attacks**, in the contrast, are types of attack where attackers are not capable of compromising honest participants and private information. However, they can overhear, reply, and jam messages.

This way sometime makes confusion in some kinds of attacks such relaying attack where attacker could be both insiders and outsiders. As an alternative way, the other classification considers three specific famous attacks on neighbour discovery protocols: *spoofing attack*, *relaying attack*, and *tunnelling attack*.

Spoofing Attack: In wireless context, spoofing attack is a situation in which an attacker (i) successfully pretends to be someone else to gain a connection to another participant, or (ii) pretends to own a connection to another participant, but actually does not have. The first type is called identification spoofing, while the second is link spoofing attack.

Relaying Attack: According to observation, relaying attack demonstrates a situation where messages are relayed between two honest participants by an attacker. Moreover, we consider two types of relay attack. One is store-and-forward relay where a message is completely received before relayed, while the other is fast relay where a message is relayed bit-by-bit.

Tunnelling Attack Tunnelling attack, a special kind of relaying attack, happens in a long distance. Two internal adversarial participants tunnel ND messages so that they appear as neighbours on routes constructed by routing protocols. As a result of that, traffic of some nodes on network probably is controlled by these adversarial nodes. In literature, wormhole attack is another name of tunnelling attack.

There are also two kinds of tunnelling attack in the routing context, one is in-band tunnelling, and the other is out-of-band tunnelling. In-band tunnelling describes that messages are encapsulated at one adversarial node, routed through the network as normal packets, and opened at an adversarial companion. In turn of out-of-band tunnelling, messages are routed in a fast private channel.

4.1.3 Secure Neighbour Discovery Techniques

Existing notable techniques of secure neighbour discovery can be classified into six primary categories: time-based, location-based, device fingerprinting-based, channel fingerprinting-based, directional antenna-based, connectivity-based. In this part, we presents some notable work in each category. This category is referred from [92].

Time-based Techniques

There are two main approaches using time-based techniques: single message-scheme and challenge-respond scheme.

In single message-schemes as [7, 93–95], each device equipped the same synchronized clock periodically broadcasts greeting authenticated messages including its current timestamp. Thank to precisely synchronized clock, any neighbour device receiving this message easily estimates the distance from itself to the source of the message.

To overcome clock synchronisation limitations, challenge-response scheme such as in [96] adopted the RTS/CTS mechanism of IEEE 802.11 into their proposals. In the mechanisms, a participant sends a challenge containing a timestamp, then receives a respond from the other. By that way, the distance between two participants can be calculated from the message-flight-time.

Location-based Techniques

The neighbour information in proposals [95, 97–99] provided at deployment stage allows participants to determine their neighbours' location. An alternative method [100] was using secure localisation schemes in which a device includes its location information in its packets.

However, the assumptions on trusted location information could be impractical. When an attacker compromises one or more honest devices, it can send counterfeit location information to its neighbours. Hence, combination of timestamps and location in [101] can offer better security mechanisms.

Device Fingerprinting Techniques

Complicated techniques using RF signal characteristics allow a device to identify other individual devices in its physical signal range. Particularly, techniques [102–104] are based on the fact that every device with a specific wireless adapter and driver differently generates a different signal pattern. By that way, a sensitive receiver probably identifies the source of signal. More complicated techniques [105–107] used other variables such as frequency error, SYNC correlation, I/Q offset error to enhance identification accuracy.

Channel Fingerprinting Techniques

Channel states, presented at [108–111], characterized by channel impulse response (CIR) in location-specific, or in time-specific can be used to increase accuracy of detecting source of signal. For instance, a device A will not observe correlated CIR from B when A stands outside the B's RF wavelength apart.

Directional Antennas-based Techniques

Multi-directional antennas used in [112], [113] were applied against worm hole attack. Under assumptions of a disk model, each antenna spans a specific zone and direction. By this characteristic, when a device sends a message in a specific zone, this message is received in opposite zone of another one.

Connectivity-based Techniques

Connectivity-based techniques basically can detect changes of muti-hops network when any wormhole is created. Moreover, some approaches tried to identify the wormhole and to remove its effects. Two main methods in the literature are centralised schemes and decentralised schemes.

In the centralised schemes [113], [114], visualisation of connectivity graph of the network constructed from coordinates of devices can be monitored either manually by human operator or automatically by software to detect and localise the wormhole.

In decentralised schemes, k-hops neighbour information used in [115] [116] were considered as local connectivity to detect and remove a false link in network. Some alternative methods [115] and [117] used features generalised edge-clustering coefficient to eliminate connectivity model.

4.2 Vulnerabilites of Exisiting Protocols

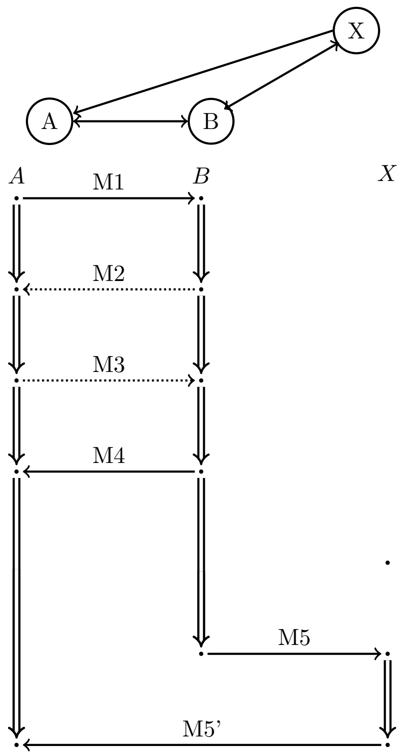
As introducing in the introduction section, we are concerning in this chapter problems associated with varieties of signal power of wireless interfaces. In particular, attackers facilitated with high power antennas can defeat validating goals in many secure neighbour discovery protocols. Actually, we found some work discussing this problem; they are [118] and [119]. This section highlights again these work.

4.2.1 Brands & Chaum Protocol Vulnerabilities

Brands & Chaum(BC) protocol [7], a famous distance bounding protocol - a kind of neighbour discovery protocol, enables a verifier to properly estimate the physical distance to its authenticated prover. The protocol works as follows.

1. Both sides generate their own random number, then rapidly exchanges bit-by-bit to together.
2. The verifier sends a message consisting of the two random numbers, and its signature to the verifier.
3. The verifier verifies the distance, random values, and the prover's identity in order to accept the connection

Figure 4.1: BC Protocol Attack



To simplify the protocol, we consider the rapid bit-exchange channel as a short-range public out-of-band channel for transmitting random values. The protocol is described below. Device A is the verifier while device B is a prover.

M1: $A \rightarrow B : \{A, B\}$
M2: $B \rightarrow A :_o \{r_b\}$
M3: $A \rightarrow B :_o \{r_a\}$
M4: $B \rightarrow A : \{r_a \otimes r_b\}$
M5: $B \rightarrow A : \{r_a, r_b\}_{PK_B}$

To violate the protocol's goal, attacker P must make a device A to accept him as the prover. Discovered in the paper [118], a flaw exploits the last message by using an advantage of a high power antenna. Actually, the attacker does not violate the ranging process, instead of the validation process. The attack scenario is presented at figure 4.1.

In this scenario, the attack cooperating with a close party to complete a protocol run. Attacker X overhears all messages exchanged between A and B during distance validation process (rapid bit-exchange), it drops message $M5$ from B to A, then it conducts and delivers $\{r_a, r_b\}_{PK_X}$ to A to complete the protocol run. As a result of that, A verifies values r_a and r_b and accepts X as a regular prover.

4.2.2 ADVSIG Vulnerability

ADVSIG proposed by INRIA group [6] is a kind of secure routing protocol fortifying for OLSR [120]. In this protocol, both devices wish to declare that they have a bidirectional physical link between them. Each message in the protocol consists of three parts: a link state declared by senders, a link proof(the link state of the previous message), and a timestamp. All information is signed by sender's private key PR. In this protocol, $\tau_0, \tau_1, \tau_2, \tau_3$ are timestamps at each sending event; and $ASYM\_LINK$ and SYM_LINK are link states. "$A : ASYM\_LINK$" states that B is obtaining a unidirectional link to A while "$A : SYM\_LINK$" states B is obtaining a bidirectional link to A. The protocol is presented as below.

M1: $A \to [B] : \{\emptyset, \emptyset, \tau_0\}_{PR_A}$
M2: $B \to [A] : \{\{"A : ASYM\_LINK", \tau_1\}_{PR_B}, \emptyset, \tau_1\}_{PR_B}$
M3: $A \to [B] : \{\{"B : ASYM\_LINK", \tau_2\}_{PR_A}, \emptyset, \tau_2\}_{PR_A}$
M4: $B \to [A] : \{\{"A : SYM\_LINK", \tau_3\}_{PR_B}, \{"B : ASYM\_LINK", \tau_2\}_{PR_A}, \tau_3\}_{PR_B}$

Moreover, every valid link state must satisfy a maximum interval δ_{max} so that $|\tau_s - \tau_r| < \delta_{max}$ where τ_s is the value clock of the sender, and τ_r is the value clock of receiver.

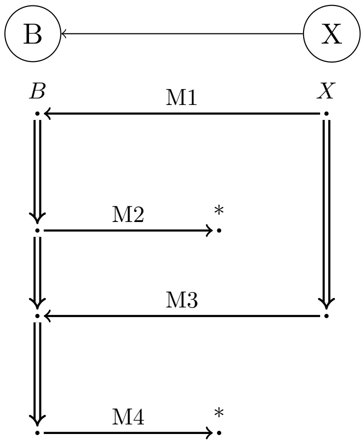
We realise that compared to OLSR specification, third message of ADVSIG changed SYM_LINK status into $ASYM\_LINK$ status. This change may affect to neighbours of A if no new message from A informs that A owns a bidirectional link A to B.

Provided that the third message does not include the proof from the second one, Responder cannot ensure if Initiator has received the second message or not. As the result, internal attackers can produce the first and third messages to valid a protocol run with Responder without receiving any message from Responder. Additionally, neighbours of Responder could be impacted by this attack due to two-hop sensing mechanism in the OLSR protocol. Figure 4.2 outlines an attacking scenario to ADVSIG where there only appears an unidirectional link X → B. Notation $- > *$ describes that the messages cannot be received by X.

In the attack scenario, we assume that X accidentally knows the existence of B out of X's physical signal coverage. By using a high power antenna, X can enlarge its signal range. Then X emits valid messages $M1$ and $M3$ to B, this makes B believe an existence of bidirectional physical link between B and X.

We found the same problem in the work [121]. Authors proposed a trusted-based security for OLSR protocols where all participants must trust together. However, internal

FIGURE 4.2: ADVSIG Attack



attackers can successfully create a fake bidirectional physical link to a target as they do in ADVSIG.

4.3 Correctness of ADVSIG

In this part, we would like to propose a simple correctness of ADVSIG scheme. We attach link proofs for the second and the third message. Thus, the first link proof in the second message is a hash of the first message encrypted by B's public key, while the second link proof in the third message is the link state of the previous one.

A's signature wrapped by hash function is impossible to be reproduced by any attacker. As the result of that, even the attacker incidentally receives the message of B, he cannot create a correct reply. The scheme is presented as follows.

M1: $A \rightarrow [B] : \{\emptyset, \emptyset, \tau_0\}_{PR_A}$
M2: $B \rightarrow [A] : \{\{"A : ASYM\_LINK", \tau_1\}_{PR_B},\ h(\{\{\emptyset, \emptyset, \tau_0\}_{PR_A}\}_{PR_B}), \tau_1\}_{PR_B}$
M3: $A \rightarrow [B] : \{\{"B : SYM\_LINK", \tau_2\}_{PR_A},\ \{"A : ASYM\_LINK", \tau_1\}_{PR_B}, \tau_2\}_{PR_A}$
M4: $B \rightarrow [A] : \{\{"A : SYM\_LINK", \tau_3\}_{PR_B},\ \{"B : SYM\_LINK", \tau_2\}_{PR_A}, \tau_3\}_{PR_B}$

4.4 Formal Analysis of Neighbour Discovery Protocol

As mentioned in the previous chapter, the problem existing in current secure neighbour discovery protocols associates to the incorrect ranging and validating processes which usually lies on some physical characteristics such as location, timestamp, and physical signal range. Meanwhile, existing formal models are partly facilitated with these characteristics to cope with the discussing problems. Therefore, in this section, we are going to produce a formalism to cover limitations of current formal model. To begin with, we will present related work and point out their limitations.

4.4.1 Related Work on Physical Characteristic Modelling

Most current approaches formalised time as timestamps to determine key-expiration [23] , and integrity of messages via round trip time-of-flight [24, 122]. Meanwhile, location information was considered in problems of correctness of routing protocols [27, 123], or evaluating distance between two neighbours in [122]. Barely found in literature, signal characteristic is an interesting feature mentioned in [122]. However, this feature is not truly helpful to reveal attacking location and direction.

Close to our approach, the models [21, 22] extended Strand Spaces to pick up vulnerabilities in ad-hoc routing protocols while other approaches [23, 28] expressed temporal phenomena, and notably key-expiration. In addition to, metric strand proposed in [31] was presented to deal with locale authentication. Nevertheless, no specified attack and proof was provided in this work.

4.4.2 Assumptions

Before presenting our model extensions of Strand Spaces, we formulate some supplementary assumptions concerning physical characteristics of wireless interfaces and environment, that we will have to take into account. We explicitly declare some reasonable assumptions as follows:

- Every participant is equipped with: (I) *various types of wireless devices with different signal powers*, (ii) *precise clock devices.* To make simplicity, radiation power is assumed to stay constantly during protocol execution.
- An *idealized communication environment* is regarded where wireless signal travels on non-obstacle path with speed of light v_c. Following this assumption, when a participant stands on signal propagation region of others, he can listen to their exchanged messages.
- A *secure key distribution function* is enabled on all participants.

Furthermore, our current work only focuses on *a static wireless network model* where devices do not change their position due to complexity of modelling dynamic networks. Dynamic network, hence, could be extended and considered in our future work.

4.4.3 Wireless Strand Spaces

Obviously, original Strand Spaces was designed for cryptographic protocols, so it apparently cannot analyse neighbour discovery protocols. To tackle this limitation, we facilitate Strand Spaces model with our extensions to account for wireless context, then we call the *wireless Strand Spaces.* In particular, a node in our model is equipped with *location, timestamp, signal range* information. Location shows where the node is standing, timestamp indicates when a node happens, and signal range refers to how far the wireless signal of the node can reach. As consequence, the definition of a wireless node is presented as below.

Definition 4.4.1 (Wireless Node). A *wireless node* n is a tuple of (t, l_n, τ_n, R_n) where t is a signed term, l_n is location, τ_n is a timestamp, and R_n is signal range.

Note that, l_n, *location of a node* n, is extensible to any Euclidean space. *Distance between nodes* n and n' is noted as $dist(n, n')$. *Timestamp*, τ_n, appears in a node to check freshness property, and it is a value of local clock when an event begins. *Signal range of a node* n, R_n, is a positive real number. According to assumption 4.4.2, signal power does not change during a protocol execution; hence, we denote R_{st} is physical signal propagation of strand st.

In realistic scenarios, there always exists a gap between two events, so a fixed value δ_{tp} is noted as a *processing delay* of edge $(+n) \Rightarrow (-n')$. We continue describing definitions of wireless strand, and wireless bundle.

Definition 4.4.2 (Wireless Strand). A *wireless strand* is a strand with wireless nodes.

Recall that static network is being discussed in this thesis; hence, a strand with fixed location is so-called *a fixed wireless strand.* Thus, all nodes in a fixed strand share the same location. We denote $loc(st)$ be the location of strand st, and $dist(st, st')$ be the distance between two strands st and st'. To avoid ambitious notions, notion of strand in this section is now refer to a wireless strand.

Existence of a physical link is usually hard to be precisely criticised due to environment complexity. Hence, in this work, a physical link is simply determined by a distance value among participants. Another speaking, by any mean, a bidirectional physical link exists between two participants if and only if the distance between them is lower than their own signal coverage. We formally define notations of links as follows. Noted that, the notation $\rightarrow^* n'$, referred from the original Strand Spaces, means that there is a path of a term from node n to node n'.

Definition 4.4.3 (Definition of Links).

- *Unidirectional physical link*:

  $\forall st, st' \in \mathcal{B}$, $plink(st, st', \rightharpoonup) \Leftrightarrow dist(st, st') \leq R_{st}$.

- *Bidirectional physical link*:
  $\forall st, st' \in \mathcal{B}$, $plink(st, st', \rightleftharpoons) \Leftrightarrow (dist(st, st') \leq R_{st}) \wedge (dist(st', st) \leq R_{st'})$.

- *Unidirectional logical link*:
  $\forall st, st' \in \mathcal{B}, n \in st, n' \in st', \exists(n_1 \rightarrow^* n_2) \Leftrightarrow \exists link(st, st', \rightharpoonup)$.

- *Bidirectional logical link*:
  $\forall st, st' \in \mathcal{B}$, $\exists link(st, st', \rightharpoonup) \wedge \exists link(st', st, \rightharpoonup) \Leftrightarrow \exists link(st, st', \rightleftharpoons)$.

4.4.4 Extended Penetrator Model

Along with Dolev-Yao penetrator model [29], this thesis considers two physical attacks: relaying attack and link spoofing attack. These attacks are apparently conducted from a sequence of atomic events such as sending events with high a power antenna and single relaying events respectively. Hence, we encode the atomic malicious actions into two new penetrator strands. Precisely, given penetrator strand st_p, and $n, n' \in st_p$.

SRL. *Single relay:*
$\langle -(t, l_n, \tau_n, R_n), +(t, l'_n, \tau_{n'}, R_{n'}) \rangle >$ where $\tau_{n'} - \tau_n = 0$.

BS. *Boosting signal:* $\langle +(t, l_n, \tau_n, R_M) \rangle$ where R_M could be an unlimited value.

At present, single relaying attack is possibly detected by advanced protection mechanisms presented in the previous section. Therefore, we consider the *weak penetrator model* which does not deal with relaying attack. In contrast, *strong penetrator model* has full attacker's capabilities.

4.4.5 Secure Neighbour Discovery Goal

We express formally secure neighbour discovery goals in Strand Spaces model as an authentication goals:

Secure Neighbour Discovery Goal: For all bundles $\mathcal{B}$, two roles $R, R' \in \mathcal{B}$, and strand st, there exists a strand st' such that if st $\in R$ has $\mathcal{B}_{height}$ i, and some protocol assumptions hold then $st' \in R'$ has $\mathcal{B}_{height}$. Moreover, there exists a physical link $plink(st, st', \rightleftharpoons)$.

Guttman stated that analysing authentication properties of a protocol means finding right choices for R and R' for i, and j, and necessary origination assumptions. However, this proving way could extremely cost time and effort when solving a complicated protocol. So, to ease this job, Guttman introduced authentication tests [124] as supporting tools. In this test, there indeed exists a bidirectional logical link between two regular strands so that they can communicate to each other.

Following to this idea, we construct our authentication link tests to guarantee whether a protocol satisfies the goal or not. At first, we introduce a link test edge, and an authentication logical link test. Then, we conduct authentication physical link tests based on time and location estimation.

4.4.6 Logical Link Tests

To get rid of link spoofing attack, a protocol should support a mechanism enabling a participant to verify whether his protocol-mate has already seen its messages or not. As a solution, challenge-respond mechanisms, described formally as authentication tests, could be used in which a participant delivers a random as a challenge and receives an answer only produced by an intended sender. The answer usually contains a cryptographic part as a proof that allows the participant verifies origin, integrity, or confidentiality of this message.

One possible particular proof, called *hidden proof*, is a nonce encrypted by pre-shared key or by public key of Initiator, or a hash of nonce and name of Responder. Another one, called *clear proof*, is a part of previous Initiator's message that contains Responder's ID and a timestamp, and is signed by private key of the Initiator. We define notations of *identification factor*, and a *provable component* to express these such proofs.

Definition 4.4.4. *An identification factor* in a component $\{c\}_k$ is:

- (clear-form) *Responder's ID_{Res} and a timestamp t* such that $(ID_{Res} \sqsubseteq \{c\}_k) \wedge (t \sqsubseteq \{c\}_k)$ in which k is a private key of Initiator. Or,
- (hidden-form) *a nonce N* such that N $\sqsubseteq \{c\}_k$ in which k is a pre-shared secret key or public key of Responder.

Definition 4.4.5. *A provable component* is a component which has one of forms:

- (clear-form):$\{m\}_{PR_{Res}}$ in which *idf* $\sqsubseteq \{c\}_{PK_{Init}} \sqsubseteq m$ where PR_{Res} is a private key of Responder, and PK_{Init} is the public key of Initiator;
- (hidden-from):$\{m\}_k$ or $\{h(m)\}_{PK_{Res}}$ in which *idf* $\sqsubseteq m$, and k is a pre-shared key, and PK_{Res} is a public key of Responder;

We conduct *a logical link test*, and *an authentication logical link test* which allows a participant to ensure that none of it's partners completes a protocol run without receiving its messages. We would like to revise the definition of *a test* defined in [124].

Definition 4.4.6 (A test). $t = \{m\}_k$ is *a test* for a in n if:

1. $a \sqsubseteq t$ and t is a component of n;

2. The term t is not a proper subterm of a component of any regular node $n' \in \Sigma$.

The edge $n \Rightarrow^+ n'$ is *a test* for a if a uniquely originates at n and $n \Rightarrow^+ n'$ is a transformed edge for a.

Definition 4.4.7 (Logical Link Test). The edge $n \Rightarrow^+ n'$ is a *logical link test* for an identification factor idf in $t = \{m\}_k \sqsubseteq term(n')$ if it is a test for idf in which $k \notin P$ and t is a provable component for idf in n'

The authentication logical link test below results a bidirectional logical link between two strands st and st' in a bundle.

Proposition 4.4.8 (Authentication Logical Link Test). Let $n \Rightarrow^+ n' \in st$ be a logical link test for an identification factor $idf \sqsubseteq t' \sqsubseteq term(n')$. There exist regular nodes $m, m' \in st'$ such that t' is a provable component of m', and $m \Rightarrow^+ m'$ is a transforming edge for idf.

Proof. Reusing the proof of proposition 20 in [78], we obtain regular edge $m \Rightarrow m'$ in a regular st'. On one hand, identification factor idf allows st' to ensure the existence of st in the protocol. On the other hand, a new provable component as well helps st ensure st' has already received idf. As a result, Initiator can ensure existence of a bidirectional logical link $link(st, st', \rightleftharpoons)$ with Responder. □

4.4.7 Authentication Physical Link Tests

In this sub-section, distance between two neighbours can be estimated through message time-of-flight in, or location between two nodes in a direct logical link test edge $n \Rightarrow n'$ ($\Rightarrow$ instead of $\Rightarrow^+$), or DE edge, for an identification factor idf. So, we introduce two authentication physical link tests: time-based and location-based authentication tests.

Time-based authentication test

Basically, distance between two participants can be determined by message travelling time. Formally speaking, the proposition below describes this idea.

Proposition 4.4.9. Consider a weak penetrator model, regular strands $st, st' \in \mathcal{B}$, nodes $n, n' \in st$, and $(+n) \Rightarrow (-n')$ is a DE edge for an identification factor idf. If $(\tau_{n'} - \tau_n) \div 2 \leq R_{st} \div v_c + \delta_{tp} \div 2$, then $\exists plink(st, st', \rightleftharpoons)$.

Proof. According to the proposition 4.4.8, there is a bidirectional logical link between st and st'. Furthermore, in absence of relaying attack, to receive message from st', st must stand within physical signal region of st'. As a result, we have $R_{st'} \wedge R_{st}$ (1). We then calculate the travelling time of idf between node n and n'.

$$\begin{aligned}
\tau_{n'} - \tau_n \geq \frac{1}{v_c}(dist(n,m) + dist(m',n')) + (\tau_{m'} - \tau_m) \\
\Leftrightarrow \tau_{n'} - \tau_n \geq 2 \times \frac{1}{v_c}(dist(n,m)) + \delta_{tp} \\
\Leftrightarrow \frac{1}{2}(\tau_{n'} - \tau_n) \geq \frac{1}{v_c} dist(st,st') + \frac{1}{2}\delta_{tp}
\end{aligned}$$

Additionally, according to proposition assumption, we have $\frac{1}{2}(\tau_{n'} - \tau_n) \leq \frac{1}{v_c}R_{st} + \frac{1}{2}\delta_{tp}$ (2). From (1) and (2), we conclude $dist(st,st') \leq R_{st} \leq R_{st'}$. Eventually, there exists a physical link between st and st'. □

In case of strong penetrator model, we need a strong assumption on similarity of signal ranges of all principals. The physical link, hence, could be obtained as below proposition.

Proposition 4.4.10. Given regular strands $st, st' \in \mathcal{B}$, assume that $R_{st} = R_{st'}$, $\exists link(st, st', \rightleftharpoons)$, nodes $n, n' \in st$, and $(+n) \Rightarrow (-n')$ is a DE edge for an identification factor idf. If $(\tau_{n'} - \tau_n) \div 2 \leq R_{st} \div v_c + \delta_{tp} \div 2$, then $\exists plink(st, st', \rightleftharpoons)$.

Proof. Using the proof of proposition 4.4.9, we have $(\tau_{n'} - \tau_n) \geq \frac{1}{v_c} dist(st,st') + \frac{1}{2}\delta_{tp}$. Additionally, since $R_{st} = R_{st'}$, we can conclude $dist(st,st') \leq R_{st} \cup dist(st,st') \leq R_{st'}$. Eventually, there exists a physical link between st and st'.

□

However, when removing the assumption $R_{st} = R_{st'}$ in proposition 4.4.10, we discover a flaw in the time-based authentication test. Let's analyse the problem into sub-cases:

Case 1: $R_{st} \geq dist(st,st')$ and $R_{st'} \geq dist(st,st')$. Obviously, both st and st' can receive messages of each other.

FIGURE 4.3: Link spoofing: Case 3

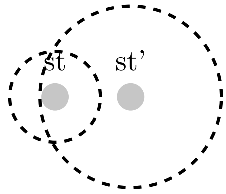
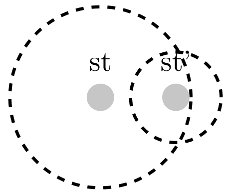


FIGURE 4.4: Link spoofing: Case 4



Case 2: $R_{st} \leq dist(st, st')$ and $R_{st'} \leq dist(st, st')$. Let's call $\Delta_d = dist(st, st') - R_{st}$, we calculate the message travelling time between n and n' as follows:

$$\begin{aligned}
\tau_{n'} - \tau_n \geq \frac{1}{v_c}(dist(n,m) + dist(m',n')) + (\tau_{m'} - \tau_m) \\
\Leftrightarrow \tau_{n'} - \tau_n \geq 2 \times \frac{1}{v_c} dist(n,m) + \delta_{tp} \\
\Leftrightarrow \tau_{n'} - \tau_n \geq 2 \times \frac{1}{v_c}(R_{st} + \Delta_d) + \delta_{tp} \\
\Leftrightarrow 2 \times \frac{1}{v_c} R_{st} + \delta_{tp} \geq 2 \times \frac{1}{v_c}(R_{st} + \Delta_d) + \delta_{tp} \\
\Leftrightarrow 0 \geq 2 \times \frac{1}{v_c}\Delta_d
\end{aligned}$$

The last equation is a contradiction when Δ_d is larger than zero, so this case will not happen.

Case 3: $R_{st} \leq dist(st, st')$ and $R_{st'} \geq dist(st, st')$. Let's call $\Delta_d = dist(st, st') - R_{st}$, we calculate the message travelling time between n and n' as follows:

$$\begin{aligned}
\tau_{n'} - \tau_n \geq \frac{1}{v_c}(dist(n,m) + dist(m',n')) + (\tau_{m'} - \tau_m) \\
\Leftrightarrow \tau_{n'} - \tau_n \geq 2 \times \frac{1}{v_c}(dist(n,m)) + \delta_{tp} \\
\Leftrightarrow \tau_{n'} - \tau_{n'} \geq 2 \times \frac{1}{v_c}(R_{st} + \Delta_d) + \delta_{tp} \\
\Leftrightarrow 2 \times \frac{1}{v_c} R_{st} + \delta_{tp} \geq 2 \times \frac{1}{v_c}(R_{st} + \Delta_d) + \delta_{tp} \\
\Leftrightarrow 0 \geq \delta_d
\end{aligned}$$

The only possible case is Δ_d equals zero, but it is too trivial. Otherwise, a fast relaying attack occurs at the side from st to st'. Case 3 is presented at figure 4.3.

Case 4: $R_{st} \geq dist(st, st')$ and $R_{st'} \leq dist(st, st')$. Presented at the figure 4.4, it similarly happens as case 3. If R_{st} is much larger than $dist(st, st')$, then a fast relaying attack occurs at the side from st' to st.

Location-based authentication test

Location information can help to determine the distance between two strands. Hence, we formally describe location-based authentication test using knowledge of participant locations.

Proposition 4.4.11. Consider a weak penetrator model, and location information is confidentially exchanged and stored. Given strands regular $st, st' \in \mathcal{B}$, and $\exists link(st, st', \rightleftharpoons)$, and let $d = |loc(st), loc(st')|$ be a distance between two location of st and st', if $d \leq R_{st}$ and $d \leq R_{st'}$, then $\exists plink(st, st', \rightleftharpoons)$.

Proof. Obviously, we have $|loc(st), loc(st')| = dist(st, st')$, then $dist(st, st') \leq R_{st}$ and $dist(st, st') \leq R_{st'}$. As a result, there exists a physical link between st and st'.

□

Note that, without assumption on trusted location information, the location-based protocols may contain flaws in some cases. For instance, st could not address correctly the physical location of st' , then an attacker uses his knowledge about location of st to make a fake link between them. Furthermore, location-based protocols share the same trouble with time-based protocols when analysed in our strong penetrator model.

4.5 Analysis of ADVSIG

In ADVSIG protocol, a node wishes to detect neighbours with bi-directional physical links. The protocol is presented as below.

M1: $A \rightarrow [B] : \{\emptyset, \emptyset, \tau_0\}_{PK_A}$
M2: $B \rightarrow [A] : \{\{"A : ASYM\_LINK", \tau_1\}_{PK_B}, \emptyset, \tau_1\}_{PK_B}$
M3: $A \rightarrow [B] : \{\{"B : ASYM\_LINK", \tau_2\}_{PK_A}, \emptyset, \tau_2\}_{PK_A})$
M4: $B \rightarrow [A] : \{\{"A : SYM\_LINK", \tau_3\}_{PK_B}, \{"B : ASYM\_LINK", \tau_2\}_{PK_A}, \tau_3\}_{PK_B})$

We analyse ADVSIG protocol to show usefulness of our model. There are three kinds of strands in ADVSIG protocol:

Definition 4.5.1. An infiltrated Strand Spaces $(\Sigma, \mathcal{B})$ is a model of ADVSIG protocol if Σ is the union of three following kind of strands:

- *Penetrator strands* $st_p \in \mathcal{B}$,
- *Initiator strand* with strand: $st = \{\langle st,1\rangle, \langle st,2\rangle, \langle st,3\rangle, \langle st,4\rangle\} \in Init[A, \tau_0, \tau_2]$, with trace $< +M_1, -M_2, +M_3, -M_4 >$.
- *Responder strand* with strand: $st' = \{\langle st',1\rangle, \langle st',2\rangle, \langle st',3\rangle, \langle st',4\rangle\} \in Resp[B, \tau_1, \tau_3]$, with trace $< -M_1, +M_2, -M_3, +M_4 >$.

Protocol assumption:

- ps1: $|\tau_s - \tau_r| \leq \delta_{max}$
- ps2: Time synchronisation mechanism is set on participants.

Initiator's Guarantee

Initiator states the guarantee as follows.

Suppose $\mathcal{B}$ is a wireless bundle. Under the conditions $ps1$ and $ps2$, if $\mathcal{B}$ contains a strand $st \in Init[A, B, \tau_0, \tau_1, \tau_2, \tau_3]$ with $\mathcal{B} - height$ 4, then $\mathcal{B}$ contains a strand $st' \in Resp[A, B, \tau_0, \tau_1, \tau_2, \tau_3]$ with $\mathcal{B} - height$ 4. Moreover, there exists a $plink(st, st', \rightleftharpoons)$.

Mechanically, to find out if the statement is correct or not, we start asserting the logical link guarantee between two participants. Let's call node n_1, n_2, n_3 and n_4 be $\langle st,1\rangle$, $\langle st,2\rangle$, $\langle st,3\rangle$, and $\langle st,4\rangle$ respectively.

Logical link guarantee: At first, we search for a DE edge with an identification factor:

- The edge $n_1 \Rightarrow n_2$ does not contain any identification factor in the challenge message.
- The edge $n_3 \Rightarrow n_4$: $term(n_3)$ contains a clear-form identification factor which is response's name B and a timestamp τ_2, all signed by A's private key, and $term(n_4)$ embodies the a clear-form provable component $\{\{"A : SYM\_LINK", \tau_3\}_{PK_B}, \{"B : ASYM\_LINK", \tau_2\}_{PK_A}, \tau_3\}_{PK_B}$.

The edge $n_3 \Rightarrow n_4$ satisfies the authentication logical link test. Therefore, A can verify a logical link between A and B, and B is a regular party. To find out if there exists $plink(st, st, \rightleftharpoons)$, we apply the time-based authentication test.

Physical link guarantee: To verify the distance between st and st', we apply the time-based authentication test for the edge $n_3 \Rightarrow n_4$. Let call τ_{n_4} be the timestamp of node n_4.

The test concludes that if $(\tau_{n_4} - \tau_2) \div 2 \leq R_{st} \div c + \delta_{tp} \div 2$ (1) then $\exists plink(st, st', \rightleftharpoons)$. Let's call node n'_3 and n'_4 be $\langle st', 3\rangle$, and $\langle st', 4\rangle$ respectively. And let $\tau_{n'_3}$ be the timestamp of node n'_3.

Basing on assumption ps1 and ps2, we calculate the message travelling time from node n_3 to n_4:

$$\begin{aligned} \tau_{n_4} - \tau_2 = \\ (\tau_{n'_3} - \tau_2) + (\tau_3 - \tau_{n'_3}) + (\tau_{n_4} - \tau_3) \\ \Rightarrow \tau_{n_4} - \tau_2 \leq 2 \times \delta_{max} + \delta_{tp}(2) \end{aligned}$$

Let $2 \times (1) - (2)$, we have

$$R_{st} \div v_c - \delta_{max} \geq 0(3)$$

If inequality (3) holds, then $\exists plink(st, st', \rightleftharpoons)$. $\square$

Responders Guarantee

Initiator states the guarantee as follows. *Suppose $\mathcal{B}$ is a wireless bundle. Under the conditions $ps1, ps2$, if $\mathcal{B}$ contains a strand $st' \in Resp[A, B, \tau_0, \tau_1, \tau_2, \tau_3]$ with $\mathcal{B}-height$ 4, then $\mathcal{B}$ contains a strand $st \in Init[A, B, \tau_0, \tau_1, \tau_2, \tau_3]$ with $\mathcal{B}-height$ 4. Moreover, there exists a $plink(st, st', \rightleftharpoons)$.*

The proof of this statement is quite identical to Initiator's guarantee. However, when looking for a bidirectional logical link, regrettably, we cannot find any logical link test edge in Responder strand. As a result, B can be a victim of spoofing attack. $\square$

Link spoofing attack on ADVSIG

Formally speaking, since n_1 and n_3 do not contain any proof of reception, they possibly lie on X strand. Actually, when X accidentally knows the existence of B but it cannot listen to B, X persuades B that B will own a bidirectional physical link with X by following strand.

$st_x = \{(M_1, loc(st_X), \tau_{x1}, R_M), (M_3, loc(st_X)\ , \tau_{x3}, R_M)\} \in Init[X, \tau_{x1}, \tau_{x3}]$ with trace $< +M_1, +M_3 >$.

4.6 Conclusion

This chapter has addressed a serious problem of current neighbour discovery protocols that have not introduced before. The problem comes when participants use wireless interfaces with different signal power, that leads the secure neighbour discovery protocols cannot correctly verify the distance between participants. Moreover, some of them were vulnerable to internal attackers due to incorrectly validating physical links.

We extended the original Strand Spaces model to be able to analyse secure neighbour protocols. To achieve this, we modified the model so that it becomes possible to take into account some physical properties including timestamp, location, and signal range. The penetrator model has been adapted in consequence. Thank our model, time-based, even location-based neighbour discovery techniques were formally proved that they would not guarantee the existence of physical bidirectional links.

Concerning future work, we will first try to extend the model in order to capture mobility and network topology. In a second step, we plan to study how we could automate the analysis procedure.

Chapter 5

A Secure Bootstrapping for Constrained Devices

High amount of heterogeneous devices interoperating together in Internet of Thing (IoT) associate to many challenges on scalability, effectiveness, constrained resources, and security for both research and industrial sectors. As one of the biggest challenges, secure bootstrapping for constrained devices are exceptionally paid attention by researchers. The term bootstrapping denotes process of configuring a device to properly participate in normal network operation. Bootstrapping is complete when a device receives all settings such as an identity, secret keys, a list of access control, protocol suits and etc. This includes any thing from physical link-layer information to application-layer one [125].

Securing bootstrapping process is essential at the first line of defence strategy. Indeed, the configuration information received in this stage is so sensitive that it should be secured to maintain security and stability of systems. Traditionally, security bootstrapping approaches mainly adopted device pairing protocols, key generation and distribution protocols, or secure configuration. However, in the context of Internet of Things, devices are usually various types and resource-constrained, establishing a secure channel among devices remarkably costs time and efforts. For example, devices don't always provide advanced user interfaces to input necessary information. The problem additionally enlarges when a controller and other devices do not have any prior knowledge about a new device. Furthermore, the scale of IoT networks might be sufficiently large, human assistance is expensive and low efficient in such networks.

To provide stability, secure bootstrapping schemes should concern on security for both physical access and network access. While secure physical access describes devices joining network must physically authenticate to a trusted center at the first place, secure network access describes the devices must authenticate to other authenticated and authorised ones in the network. However, problems may come from users when they accidentally add rouge devices into their network, that happens if they bought used or ingenious devices. Therefore, device validation must be regarded.

According to our review, existing approaches partly tackled the problems. Hence, this chapter offers a novel solution for secure bootstrapping schemes in Internet of Things that focuses on security, adaptability, and constrained resources. Precisely, our goals are providing a secure configuration between a new device and a controller, and a trust relationship among devices, even in some worst cases when controllers are inaccessible, when the devices are reused, and when the introducer could be lost.

The rest of this chapter is organised as follows. The first section introduces requirements for bootstrapping schemes for constrained devices. In section 5.1, we defined environment, problems, attacker model, and notations. Then, section 5.2 discusses about secure bootstrapping building blocks and related work. In section 5.3, we introduce our secure bootstrapping scheme for constrained devices in IoT. Section 5.4 presents formal proofs. Finally, section 5.5 concludes the chapter.

5.1 Environment, Problem Definition, Attacker Model

5.1.1 Environment

In this chapter, we examine problems of secure bootstrapping in home network. The environment includes these main factors:

- **A home gateway** playing as a *center controller* manages devices, checks origin of devices, and synchronises to the cloud service.
- **Cloud service** playing as a *remote controller* allows users to remotely manage their devices.
- **Things** have some functions on providing services to users, and also autonomously configure in the home network.
- **A smartphone** playing as an *introducer* introduces a new device to the home network, and manages remotely the home gateway or the cloud.

The things are constrained devices such as light sensors, temperature sensors, smart TVs, smart fans, and so on. Some such these devices definitely have limited computing power, limited amounts of memory; and they may have simple interfaces such as a button, a LED light, or a light sensor. Additionally, manufactures implant their information into devices in non-volatile memory. The information includes: a unique manufacture identification, a device identification, and a manufacture fingerprinting.

Smartphones and home gateway are assumed to have a trusted relationship through a secure TLS channel. Moreover, when the smartphone is accidentally lost, all its information stored the home gateway will be wiped out intermediately by users. Additionally, since attacker can exploit information stored in the smartphone, the smartphone is recommended to not contain any network group key, and devices' information.

5.1.2 Problem Definition

This chapter considers not only secure configuration between a new device and the home gateway, but also a trust relationship among devices. Furthermore, some worst cases are also regarded; particularly where home gateway is out-of-services, we cannot make a trust relation between device and the home gateway; where the device has been used before, its permanent embedded keys could be lost or compromised; where the smartphone is lost, all data stored inside the smartphone memory could be compromised; where there is no pre-share knowledge between a device and home gateway, pairing solution cannot be offered by any cryptographic protocol; and finally where devices have limited computational power, memory capacity, and communication interfaces, heavy cryptographic mechanisms such as public keys infrastructure (PKI) increase communication overhead and computational complexity.

Meanwhile, the problem of key revocation in distributed network is out of scope of this work. Additionally, this work does not consider the situation where attackers compromise trusted devices, or the smartphone, and the situation where the smartphone, home gateway, and the cloud service are out-of-services at the same time because these problems will reduce the bootstrapping problem.

The main contribution of this work is introducing an efficient and secure bootstrapping scheme using a simple out-of-band channel. Therefore, we state requirements for secure bootstrapping solutions for the Internet of Things as below.

(i) It must be secured against attacks on key agreements between a new device and other devices;

(ii) A solution should be intuitive for non-expert users, and should be difficult for installers to introduce any vulnerability into target systems;

(iii) The solution does not require a special additional setup hardware. The specialised hardware like ultrasound interfaces, or a large colour screen will significantly increase deployment cost.

(iv) The heavy cryptographic algorithms should not be applied on constrained devices, e.g. public key encryption.

(v) Interoperable security protocols between devices are efficient in term of complexity, communication cost, and size of protocols.

5.1.3 Attack Models

Attackers' goals are cracking down key agreements between the new device and home gateways, registration servers, or smartphones. Since our proposed solution adopts multiple communication channels, the penetrator model is refined to capture both Dolev-Yao attacks [29] on open channels such as wireless, or power line, and attacks on secure channels defined in 3.2.3. In particular, attackers can overhear, intercept, modify and inject arbitrary messages into public medium while limited capabilities, e.g. overhearing, dropping, delaying, and replaying messages on sorts of secure channels.

5.1.4 Notations

The following notation presented in the table 5.1 will be used in this chapter. Dev_Desc, or device description, describes in detail device information such as a device specification, a created date, a version, current firmware, and etc. Moreover, it is normally provided by manufactures.

Given to each device by a home gateway/a cloud service, a device fingerprinting, or Dev_FR includes a device DH key and its expiration date. Whenever any piece of a device fingerprinting is required to change, the fingerprinting owner must contact to a controller via bootstrapping process.

Net_CONF, created by a home gateway/a cloud service, is network settings containing network identification, and a network address of a home gateway. These information enable a device to maintain normal operations in the network.

A network group key, or $NetPass$, plays an important role to offer a cryptographic material for establishing secure communications among authorised devices. Thus, introducers should not store these information if it does not directly communicate with authorised devices in the home network. Otherwise, compromised or lost introducers are potential threats to the system. In particular, each $NetPass$, encrypted by the private

Table 5.1: Notations

MS_ID	32-bit unique manufacture identification.
Dev_ID	32-bit device identification.
MS_FR	manufacture fingerprinting used by third-party services to verify manufactures.
Dev_Status	the status of a device including sold, registered, unregistered, or brand-new
Dev_Desc	information describing device specification.
Dev_Key	the device DH key generated in secure pairing process
Dev_URN	the unique URN assigned by the manufacture to uniquely identify the device.
Dev_Addr	the network address of the device
Expires	the expiration date of device fingerprinting
PR_{SH}	the home gateway's private key
PK_{SH}	the home gateway's public key
Dev_FR	device fingerprinting formed as $\{Dev\_Key, Expires\}_{PR_{SH}}$
Dev_Info	the device information including MS_ID, Dev_ID, MS_FR, and Dev_URN
SH_Addr	the network address of the home gateway
Net_ID	the home network ID
Net_CONF	the network settings including Net_ID, SH_Addr provided by the home gateway
NetPass	the network group key including $\{Key, ID\}_{PR_{SH}}$ where ID is a sequence number of the Key
ks	the shared key conducted by two DH keys.

key of the home gateway, includes a sequence number, or ID, allowing devices to update keys when their $NetPass$ is obsoleted.

5.2 Related Work

As discussed in the previous section, a flexible solution should consider physical access, network access, and device validation as vital building blocks. So we call these blocks as secure device pairing block, a network association block, and a registration block. Each block means as below.

(i) The secure pairing block manages a device to securely pair with an introducer;

(ii) The registration block manages an introducer to provide registration information from a home gateway or registration services to a device,

(iii) The network association manages a device to join, or to recommission the network.

In practice, a bootstrapping scheme can be constructed by either one, two, or full three these building blocks, e.g device pairing protocols are the simplest ones. So, we classify current approaches depending on these tasks, and address their limitations as follows.

5.2.1 Pairing Block

Recent studies concerned on secure key sharing between a new device and an introducer. The easiest way is using a QR code (abbreviated from Quick Response Code) consisting of a one-time-password which either is implanted at the manufacture site, e.g in [126], or is generated at the bootstrapping process, e.g in [127]. Then, the introducer uses a camera to extract information in the QR code. However, this method meets a problem if QR codes are removed, or if devices do not have a screen. Some alternatives adopt kinds of out-of-band channels to securely transmit keys such as NFC, LED in[49], or human channels in [46].

Other main approaches [128–131] used pre-shared keys, or trusted public keys to establish secure channels between a device and a home gateway in 6LoWPAN [132]. However, because of exchanged through secure channels, pre-shared keys are not sufficiently convenient for users when they have to deploy keys in a farm of devices, and update keys for them. Alternative approaches use a PKI system to automatically deliver and to verify keys. The system indeed allows a home gateway to authenticate a new device, but it is not truly friendly to constrained devices.

Furthermore, it is clearly impossible for bootstrapping when the home gateway is disconnected to the network. Concerned as a solution, a trusted smartphone proposed in [127], [133] can act as an authenticator. Nevertheless, losing phones is definitely painful to users since their phone are storing a bundle of sensitive information such as network keys, or authentication keys.

5.2.2 Registration Block

Barely concerned in existing work, the device registration is partly built in device authentication schemes. However, in sensitive networks like home networks, new devices must be authorised before joining. The device registration is, hence, required at the early step to avoid rogue or fake devices.

Proposed in [126], a mothership acting as a registration point verifies device status, and generates a device fingerprinting whenever an introducer or a home gateway requests. In the same approach, a service registration for Zigbee devices is proposed in [134].

Standard protocols such as RADIUS [135], PANA [136, 137], EAP-TLS [138], HIP-DEX [139] offer strong authentication and registration services, but they are useful if a new device has obtained some common trusts , e.g. public keys, pre-shared keys, or trusted authentication servers.

5.2.3 Network Association Block

Generally, when successfully authenticated by a trusted center, a device obtains further configurations such as cluster head selection, routing protocols, or secure neighbour discovery protocols. Nonetheless, the way a new device joins in the network is presented unclear in current approaches. Hence, the device might receive incorrect information, or association protocols contain undesired flaws.

Furthermore, working in distributed environments, devices can loose their connections occasionally. Instead of re-authentication, the devices only need re-association using their authenticated materials such as session keys, or authenticated public keys. Some protocols such as GSAKMP [140], GDOI [141], PANA [136], EAP-TTLS [142], and MIKEY [143] offer association process, but they are too heavy on constrained devices.

5.3 A Proposal For Secure Bootstrapping in IoT

Combining all above facts in the previous section, current approaches are aggressively debated to be adopted in context of Internet of Things. In this part, as far as concerning on our requirements and tackling limitations of current approaches, we propose a novel secure bootstrapping scheme that offers lightness, adaptability, and security. This section shows the full version of our secure bootstrapping scheme which consists of three primary processes: pairing process, registration process and association process. We also attach re-association and re-bootstrapping processes.

5.3.1 Overview

Our general bootstrapping scheme is illustrated at the figure 5.1. The scheme contains a set of protocols enabling a new device to completely participate in an existing home network. A new device receives a set of information at manufacture via a secure channel. We assume that attackers cannot violate this process.

To begin a bootstrapping process, a device needs an introduction from a trusted smartphone. After establishing a secure channel via secure pairing process, the smartphone

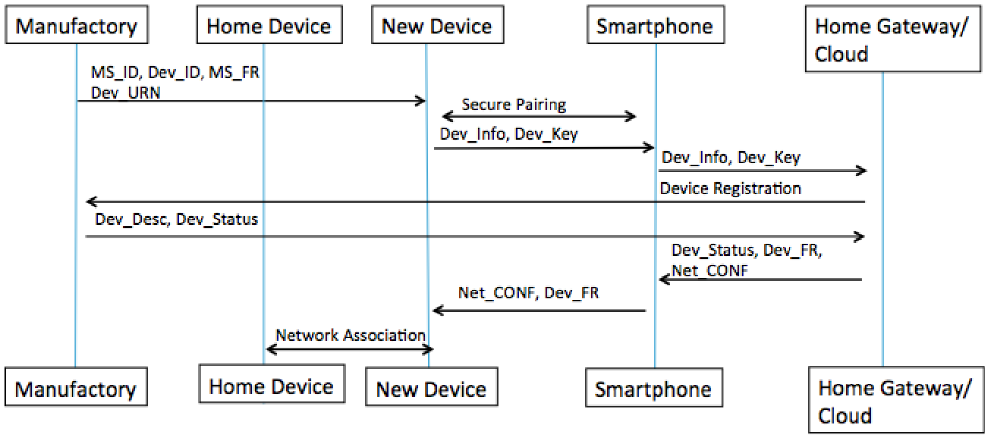


FIGURE 5.1: Secure Bootstrapping Scheme

receives all device's information including a device's DH key. The smartphone forwards this information to a home gateway/a cloud service via a secure TLS channel. The home gateway/ cloud service validates the device using its manufacture identification, and device URN (Uniform Resource Name). After successful validation, the home gateway/the cloud service produces a device fingerprinting and network configuration, and offers them to the device via the smartphone. Finally, the new device launches network association process with an authorised device to participate in the home network.

5.3.2 Pairing Process

Pairing process aims to create a secure communication between a new device (A) and a smartphone (B). It is mainly based on our proposed protocol in 3.4 which is the best secure and economic solution in its class. And according to our proof, this protocol could achieve strong security even with a low-bandwidth OOB channel.

The protocol is depicted in the figure 5.2 and presented as follows.

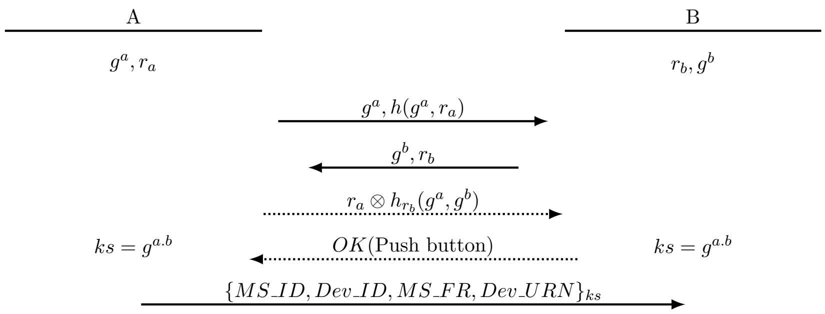


FIGURE 5.2: Secure Device Pairing Protocol

1. A and B generate its own device key g^a and g^b correspondingly.
2. A picks a random value r_a. B picks a random value r_b
3. A sends $(g^a, h(g^a, r_a))$ to B. B records g^a as Dev_Key of A.
4. B sends (g^b, r_b) to A.
5. A sends $(r_a \otimes h_{r_b}(g^a, g^b))$ to B over a public out-of-band channel.
6. B verifies received values and announces the result to A.
7. A confirms by pushing an Accept button.
8. Both side generate the shared key $ks = g^{a.b}$
9. A sends encrypted information including manufacture identification and its fingerprinting, device identification and its URN using ks to B.

5.3.3 Registration Process

The registration process happens between a smartphone and a home gateway/a cloud service. In this step, the new device wishes to be validated and to securely receives a device fingerprinting and network settings. The process starts after the smartphone receives a bunch of information from the new device including MS_ID, Dev_ID, MS_FR, and Dev_URN in previous step. The smartphone enrols the new device to the home gateway. The protocol is illustrated at the figure 5.3, and presented as below.

1. The smartphone opens a secure TLS connection to the home gateway.
2. The smartphone sends all device information, a device type, and the device's DH key to the home gateway.
3. The home gateway validates the device information to the manufacture over a secure HTTPS connection.
4. The manufacture sends the Dev_Desc and Dev_Status to the home gateway.
5. The home gateway possible requests a device registration to the manufacture site if the device has not been registered.
6. After successful validation and registration, the home gateway generates the Dev_FR, then sends Dev_FR, Net_CONF, Dev_Status, and PK_{SH} to the smartphone.
7. The smartphone offers Net_CONF, Dev_FR, and PK_{SH} to the new device.

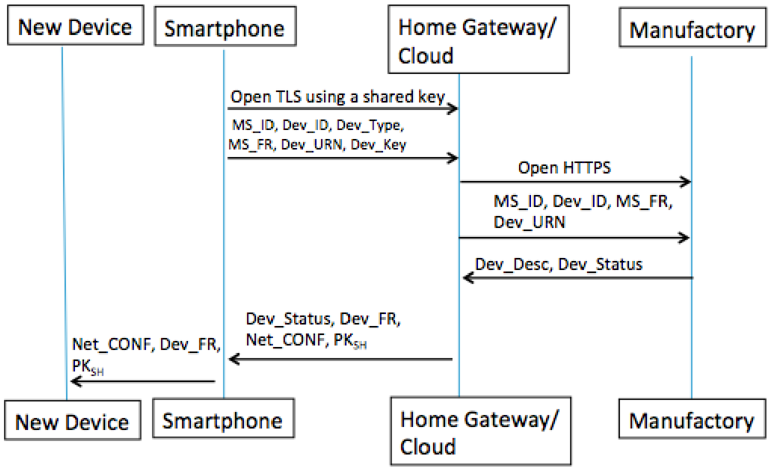


FIGURE 5.3: Registration Process

8. The smartphone cleans all received information when completing the process.

In case of inaccessible home gateway, the smartphone can forward the registration process to the cloud service as a remote controller. The cloud service then updates new information to the home gateway whenever it is available.

5.3.4 Association Process

Network association happens between an authenticated device (A) and an authorised device (B). The purpose of this process is exchanging the network group key or creating a per-neighbour key between two devices. In this work, we design a network group key exchange protocol.

A wishes to receive the network group key, or $NetPass$, from B. The protocol is illustrated in the figure 5.4, and happens as follows. In this figure, we denote Dev_A_FR as A's fingerprinting, and Dev_B_FR as B's one. g^a and g^b are DH keys generated at the pairing process. Apparently, g^a for g^b is a part of Dev_A_FR or Dev_B_FR correspondingly.

1. A sends r_a, Dev_A_FR to B.
2. B validates Dev_A_FR using PK_{SH}, and produces a shared key $ks = g^{a.b}$.
3. B sends $Dev_B\_FR, \{r_a \otimes r_b, NetPass\}_{ks}$ to A.
4. A validates Dev_B_FR using PK_{SH}, and produces a shared key $ks = g^{a.b}$.
5. A sends $\{r_b\}_{ks}$ to B.

Note that, it is possible for a home gateway playing in role of an authorised device.

5.3.5 Re-association Process

The re-association happens between two authorised devices A and B. They wishes to exchange their $NetPass$ securely. When one device receives a $NetPass$ from the other, it checks the received $NetPass$'s ID. If the new ID is bigger than its current $NetPass$'s ID, then the device will update its $NetPass$ key. The protocol is presented below and illustrated at the figure 5.5.

1. A sends r_a, Dev_A_FR to B.
2. B validates Dev_A_FR using PK_{SH}, and produces a shared key $ks = g^{a.b}$.
3. B sends $Dev_B\_FR, \{r_a \otimes r_b, NetPass_B\}_{ks}$ to A.
4. A validates Dev_B_FR using PK_{SH}, and produces a shared key $ks = g^{a.b}$.
5. A sends $\{r_b, NetPass_A\}_{ks}$ to B.

By the way of re-association, whenever a home gateway wishes to update the $NetPass$, it just disconnects and re-associates to neighbour devices to spread the new key.

5.3.6 Re-bootstrapping

Re-bootstrapping supports re-authentication and re-keying for devices, and allows them to be administratively switched to a different domain. Whenever re-bootstrapping process is demanded by a device, the secure pairing process will be established between the

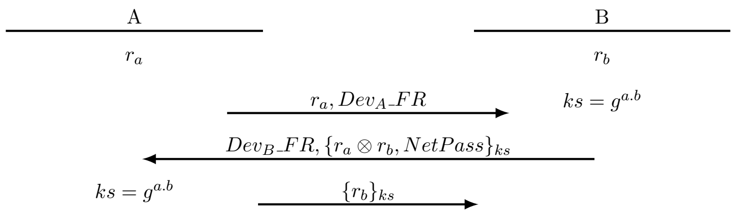


FIGURE 5.4: Network Association

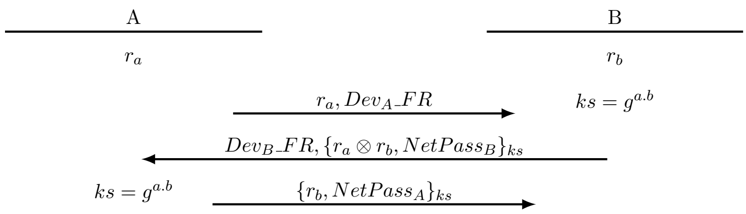


FIGURE 5.5: Network Re-association

device and the smartphone. After pairing properly, the smartphone chooses a device type then runs the registration process. The home gateway regenerates the device's fingerprinting, then sends Dev_Status, Dev_FR, Net_CONF, and PK_{SH} to the device via the smartphone.

5.4 Security Analysis

In this section, we present a sketch of proofs of our bootstrapping scheme since this scheme consists of some already-proved protocols. Additionally, these protocols in the bootstrapping scheme are proved in AVISPA [89] as correct protocols.

Our proposed bootstrapping scheme aims to (i) key agreement between a new device and a smartphone in device pairing process (ii) agreement on device fingerprinting between a new device and a home gateway in the registration process (iii) network group key agreement between an authenticated device and an authorised one in association process.

Proposition 5.4.1. The new device and the smartphone agree on the shared key $ks = g^{a.b}$ if the smartphone is not compromised.

Proof. According to the assumption on uncompromised smartphone, this secure pairing protocol is clearly proved in our last work [144]. We also showed that the successful attack probabilities is $Pr[attack] \leq n.\gamma.2^{-k}$ where n is the number of participants on the network, γ is the maximum number of sessions for each participant, k is the length of short authenticated string. □

Proposition 5.4.2. A new device and a home gateway agree on a device fingerprinting $Dev\_FR = \{Dev\_Key, Expires\}_{PR_{SH}}$.

Proof. According to the assumption on authenticated secure TLS channel between the smartphone and the home gateway, all messages transmitted on this channel are indeed secured. As a result of this, only regular smartphone can get the device fingerprinting generated by the home gateway/the cloud service.

Using the result of the proposition 5.4.1, whenever a new device gets a device fingerprinting from a regular smartphone, it can ensure this value produced by the home gateway/the cloud service. However, when the device might contacts to attackers, the home gateway does not authenticate this device. Hence,the fake one cannot be validated at the association process.

□

Proposition 5.4.3. An authenticated device(A) and an authorised device(B) agree on $NetPass$.

Proof. Intuitively, both sides agree on the shared key $ks = g^{a.b}$ by following facts.

(i) The proposition 5.4.2 says the agreement on Dev_A_FR between A and a home gateway.

(ii) In case of faked smartphones, A might obtain a fake device fingerprinting generated by attackers. However, by the assumption on uncompromised home gateway's private keys, B is able verify Dev_A_FR if it is generated by the home gateway or not.

(iii) According to the assumption on uncompromised B, A is able to verify Dev_B_FR.

Thus, after two first messages, two principals own the same shared key $ks = g^{a.b}$. Additionally, this protocol is a variant of *Needham Schroeder* protocol proved in [15] using a pre-shared key, A and B can ensure the agreement on r_a and r_b. Moreover, encrypted by ks in the last message, $NetPass$ cannot be obtained by any attacker. Therefore, A is able to ensure secrecy of $NetPass$. Finally, both sides agree on the secured $NetPass$. □

5.5 Conclusion

Recapping this chapter, not only does our scheme satisfies all of our mentioned requirements, it also provides unique features helpful to bootstrapping devices in some sensitive circumstances. Compared to our scheme, current proposed schemes do not provide the same fashion in term of security, light-weight and adaptability. Particularly, pairing methods such as [49] and [46] do not support authorisation and re-bootstrapping. In [126, 127], QR codes containing an initial key could introduce a weakness point for the system if they are lost. Alternative solutions using pre-shared keys in [128–131, 136] are not sufficiently convenient to users. Finally, lost smartphone in schemes [127, 133] leads to leak sensitive information such as pre-shared keys, re-bootstrapping information.

Eventually, this chapter offered a novel secure bootstrapping scheme for constrained devices in context of Internet of Things. Precisely, it takes advantage of unsupporting a PKI system, pre-shared keys, and pre-implanted keys. Furthermore, our scheme is not error prone to users since users only participate in the pairing process. In close future, we continue implementing remaining parts of our bootstrapping scheme, and evaluate its usability.

Chapter 6

Conclusion and Future Work

This chapter summaries the thesis results, discusses a few limitation, and introduces possible future work.

6.1 Summary

In this thesis, we have investigated three critical problems of current security protocol design in IoT, particularly in secure device pairing and neighbour discovery protocol, that received very little attention on physical security properties, physical attacks, and formal analysis. As our first contribution, we introduced our novel pairing protocol that is secure and efficient than other competitions in communication cost, but remains the attack probability. Precisely, it only uses two messages on wireless channels, and one on a public out-of-band channel. Additionally, an implementation on an embedded system was conducted to show the usefulness of our protocol. Meanwhile, we found attacks in some existing secure pairing protocols, some of which are using in commercial products.

We investigated that secure device pairing are strongly affected by many physical aspects that cannot be resolved by any classical formal method. Taking the obstacles into account, our model as the second contribution is constructed on famous Strand Spaces with supplement notations of channel. Our model also captures both physical attacks and specific attacks on secure channels. Along with the improved model, we proposed a procedure that transforms a model in our extended formalism of an initial protocol using OOB channels to a model in original Strand Spaces of a protocol that does not use any OOB channel. In addition, the translation exactly preserves security properties of initial protocol. As a result, our translation allows us to formalise out-of-band channels-based protocols in current automatic verification tools.

Our another main contribution is a study of current neighbour discovery protocols in wireless network, and formal models for such protocols. We spotted a problem where signal range of two principals is different. As a result, when analysed in our model, the time-based, or distance-based schemes do not exactly provide link agreement among principals. Finally, some protocols have been analysed in our model as our proof of concept.

Our final contribution was proposing a new secure and robust bootstrapping scheme for constrain devices. Our scheme takes more advantages than other competitors since it does not require pre-shared key, implanted public keys, and even PKI system. Furthermore, it still works when a home gateway is down, and a new thing is second-handed stuff.

6.2 Perspective

In Chapter 3, our model takes reasonable assumptions on which both principals in the protocol are honest, and out-of-band channels are at least authenticated. Hence, to take internal attacks into account, our model should express the attacks at a specific probability.

As introduced in Chapter 4, our formal model successfully reasons about physical properties-related protocols through specific examples. Currently, the model focuses on some particular physical aspects, yet does not include device mobility. For further work, we continue enlarging our model to cover dynamic networks, and topology aspects as well. After successfully reasoning about secure neighbour discovery protocols, we keep going to adapt our model to secure routing protocols, or secure protocols in vehicle networks. We strongly believe that our model achieves more promising results.

In chapter 5, our bootstrap framework is partly implemented in a prototype hardware. So, we are going to completely deploy the scheme using adaptation of current authentication systems such as RADIUS, PANA on 802.15.4 wireless technology. Obviously, this work is not too complicated to be complete. As a consequence, the framework will be comprehensively evaluated security, usability, and performance when fully constructed.

As a fact of that, manual proving task is prone to errors, we should looking for a way to implement our model into an automatic proving tool. In literature, Song [145] proposed an algorithm based on Strand Spaces model. And, the algorithm was also deployed in a tool named Athena. Unfortunately, we cannot access this tool since its authors do not publish their source code. Certainly, rebuilding the source code will take a plenty of time, so this work is planned in close future.

In long-term, we will bind all separated and unfinished pieces of current work. Moreover, an automatic or assistant proving tool is our first main goal. The tool, of course, will be fully adapted to study wider wireless protocol families, rather than two protocols discussed in this thesis. Visualising attacks could be interesting as well. Along with developing tools, a complete bootstrapping framework for IoT applications will be discussed more to become an industrial standard.

Appendix A

Strand Spaces Model

In 1997, Fabrega, Herzog and Gutman developed a new method to prove a protocol if it achieves authentication and secrecy properties or not. First published internally as a technical report, the work went out public in a conference paper [15] in a year later. Following this approach, the authors continued maturing their model. Basic Strand Spaces theory was extended with *honest idea* [146] which allows to learn general principles that limit penetrators' capabilities. After that, *mixed strands* [147] was proposed to study problems of mixed protocols. One huge milestone of Strand Spaces theory is *authentication tests* [78] proposed in 2000. In a study of cryptographic protocols, authors realized that after emitting a message containing a new value, a participant receives a cryptographic form of this value, this must exist a regular participant of the protocol sent it. This scheme, so called an authentication test, works as a powerful tool to minimise work of proving, and straightforwardly give results of authentication protocols. From 2002 to 2007, Strand Spaces theory was enriched with *shape of bundle, skeleton and homomorphism* [88]. These theories study about a sort of protocols that share the same forms. Current developments of Strand Spaces are focusing on various of types of protocols such as TLS [148], location-awareness protocols [31], and DH protocols [79].

Definitions used in thesis are referred from [15], [78] and [88]. Furthermore, to make readers conformable with the theory, the Needham-Schroeder(NS) protocol [149] presented below is used as an example through this part.

1. $A \rightarrow B$: $\{r_a, A\}_{PK_B}$
2. $B \rightarrow A$: $\{r_a, r_b, B\}_{PK_A}$
3. $A \rightarrow B$: $\{r_b\}_{PK_B}$

A.1 Fundamental Theory

A security protocol is an ordered sequence of messages that participants exchange in a protocol. Let $\mathcal{A}$ a set of possible messages intentionally transferred in a security protocol, and elements of this set are referred to as *terms*. Additionally, the set $\mathcal{A}$ is algebra freely generated from of a set of text terms $\mathcal{T}$ and a set of cryptographic keys $\mathcal{K}$ by means of concatenation and encryption. While $\mathcal{T}$ contains textual information such as nonces, $\mathcal{K}$ contains symmetric and/or asymmetric cryptographic keys. The two sets $\mathcal{T}$ and $\mathcal{K}$ are disjoint.

Definition A.1.1. *Compound terms* are generated by two operators

- encr: $\mathcal{K} \times \mathcal{A} \rightarrow \mathcal{A}$ representing encryption
- join: $\mathcal{A} \times \mathcal{A} \rightarrow \mathcal{A}$ representing concatenation

For convention, from now we express the concatenation of two term t and t' as *t,t'* and encryption of term t with key K as $\{t\}_K$. We continue defining a relation, *subterm relation*, on the set $\mathcal{A}$.

Definition A.1.2. The *subterm relation* $\sqsubseteq$ over $\mathcal{A}$ is defined inductively as:

- $a \sqsubseteq t$ for $t \in \mathcal{T}$ iff $a = t$
- $a \sqsubseteq key$ for $t \in \mathcal{K}$ iff $a = key$
- $a \sqsubseteq gh$ iff $a \sqsubseteq gh, a \sqsubseteq h$ or $a = h$
- $a \sqsubseteq \{g\}_K$ iff $a \sqsubseteq g$ or $a = \{g\}_K$

Remark that, $t_1 \sqsubseteq t$ means t_1 is a subterm of t. A *subterm* is just a term that can be easily extracted from a term with an appropriate key. In NS protocol, initiator A starts sending to intentional responder B a message (or a *term*) of the form $\{r_a, A\}_{PK_B}$ where PK_B is the public key of B. Subterms of term $\{r_a, A\}_{PK_B}$ could be r_a, A, or $\{r_a, A\}_{PK_B}$. Since PK_B is a key, $PK_B \not\sqsubseteq \{r_a, A\}_{PK_B}$, except a case when PK_B is in a value of this message.

Terms are extended to *signed term* including a positive represented to a transmission, and a negative one represented to a reception.

Definition A.1.3. A *signed term* is a pair $\langle \delta, a \rangle$ with $a \in A$ and δ one of the symbols $+, -$. We will write a signed term as $+t$ or $-t$. $(\pm A)^*$ is the set of finite sequences of signed terms. We denote a typical element of $(\pm A)^*$ by $\langle \delta_1, a_1 \rangle, ..., \langle \delta_n, a_n \rangle$.

Note that, the unsigned term is the term without direction. For example, a signed term $+t$ has direction $+$, and an unsigned term t.

A sequence of sending and receiving messages in a protocol is called *strand*, and a set of strands is called *Strand Spaces*.

When modeling a protocol, the strand of a principal is a sequence of events as seen by that principal in a particular protocol run and in a particular instance of role. For example, the strand of the initiator for the NS protocol is $\langle +\{r_a, A\}_{PK_B}, -\{r_a, r_b\}_{PK_A}, +\{r_b\}_{PK_B}\rangle$. The first event is emission of $\{r_a, A\}_{PK_B}$ followed by reception of $\{r_a, r_b\}_{PK_A}$ and so on.

Definition A.1.4. A *Strand Spaces* is a set Σ with a trace mapping tr: $\Sigma \rightarrow (\pm A)^*$.

For a Strand Spaces Σ:

- A node is a pair $\langle s, i\rangle$, with $s \in \Sigma$ and i an integer satisfying $1 \leq i \leq length(tr(s))$. The set of nodes is denoted by $\mathcal{N}$. We will say the node $n = \langle s, i\rangle$ belongs to the strand s. Clearly, every node belongs to a unique strand.
- If $n = \langle s, i\rangle \in \mathcal{N}$ then $index(n) = i$ and $strand(n) = s$. Define $term(n)$ to be $(tr(s))_i$, i.e. the i^{th} signed term in the trace of s. Similarly, $uns\_term(n)$ is $((tr(s))_i)_2$, i.e. the unsigned part of the i^{th} signed term in the trace of s
- If $n, n' \in \mathcal{N}, n \rightarrow n'$ mean $term(n) = +a$ and $term(n') = +a$ It means that node n sends the message a which is received by n', creating a causal link between their strands.
- If $n, n' \in \mathcal{N}, n \Rightarrow n'$ mean n and n' occur on the same strand with $index(n) = index(n') - 1$. It expresses that n is an immediate causal predecessor of n' on the strand.
- $n \Rightarrow^+ n'$ to mean that n precedes n' (not necessarily immediately) on the same strand.
- $m \Rightarrow^* n$ means $m \rightarrow n_1 \Rightarrow m_2 \rightarrow n_2 \Rightarrow ......m_k \rightarrow n$ where n_i, m_i, and n stand on the same or different strands. A *path* created by $m \Rightarrow^* n$ forms a connectivity subgraph in the bundle.
- An unsigned term t *occurs* in $n \in \mathcal{N}$ if $t \sqsubseteq term(n)$.
- An unsigned term t *originates* on $n \in \mathcal{N}$ iff:$term(n)$ is positive, $t \sqsubseteq term(n)$, and whenever n precedes n on the same strand, $t \not\sqsubseteq term(n)$.
- An unsigned term t is *uniquely originating* iff t originates on a unique $n \in \mathcal{N}$

Let the Initiator strand be $st = \langle +\{r_a, A\}_{PK_B}, -\{r_a, r_b\}_{PK_A}, +\{r_b\}_{PK_B}\rangle$, and the responder strand be $st' = \langle -\{r_a, A\}_{PK_B}, +\{r_a, r_b\}_{PK_A}, -\{r_b\}_{PK_B}\rangle$.

The first node of Initiator strand is $\langle st, 1\rangle = +\{r_a, A\}_{PK_B}$. Moreover, $\langle st, 1\rangle \Rightarrow \langle st, 2\rangle$ means node $\langle st, 1\rangle$ is a causal predecessor of node $\langle st, 2\rangle$. r_a occurs in $term(\langle st, 1\rangle)$. Additionally, since sign of $\langle st, 1\rangle$ is positive, and there does not exist any predecessor of $\langle st, 1\rangle$, r_a uniquely originates at $\langle st, 1\rangle$.

Let $\mathcal{N}$ be a set of nodes, and let $\mathcal{E}$ be the union of the sets of $\rightarrow$ and $\Rightarrow$ edges. A directed graph $\mathcal{G}$ has a structure $\mathcal{G} = \langle \mathcal{N}, \mathcal{E}\rangle$. A *bundle* is a finite subgraph of this graph in which the edges express casual dependencies of the nodes.

Definition A.1.5. Suppose $\mathcal{N}_\mathcal{B}$ be a subset of $\mathcal{N}$, and $\mathcal{E}_\mathcal{B}$ be a subset of $\mathcal{E}$. Let $\mathcal{B} = \langle \mathcal{N}_\mathcal{B}, \mathcal{E}_\mathcal{B}\rangle$ be a subgraph of $\mathcal{G}$. $\mathcal{B}$ is a *bundle* if :

1. $\mathcal{N}_\mathcal{B}$ and $\mathcal{E}_\mathcal{B}$ are finite
2. If $n \in \mathcal{N}_\mathcal{B}$ and $term(n)$ is negative, then there is a unique n' such that $n' \rightarrow n \in \mathcal{E}_\mathcal{B}$
3. If $n \in \mathcal{N}_\mathcal{B}$ and $n \Rightarrow n'$ then $n \Rightarrow n' \in \mathcal{E}_\mathcal{B}$
4. $\mathcal{B}$ is acyclic

The graph consisting of the Initiator strand st and responder strand st' is called a NS bundle. Remarking that Guttman implicitly expressed that when a node transmits a message, there is more than one (or none) receiving node of the message. However, we hardly see more than two receptions in their studies.

Definition A.1.6. A node n belongs to a bundle $\mathcal{B} = \langle \mathcal{N}_\mathcal{B}, \mathcal{E}_\mathcal{B}\rangle$, written $n \in \mathcal{B}$ if $n \in \mathcal{N}_\mathcal{B}$. The $\mathcal{B} - height$ of a strand $s \in \mathcal{B}$ is the largest i such that $\langle s, i\rangle \in \mathcal{B}$.

For instance, in NS protocol, $\mathcal{B} - height$ of initiator strand is 3, similarly $\mathcal{B} - height$ of responder strand is also 3.

A.2 Component, Authentication Tests

After introducing Strand Spaces fundamental theory, basing on the fact that security authentication protocols usually use specific challenge-response methods to obtain goals, Thayer et al[124] continued publishing a theory of *authentication tests* as a tool to prove security properties for protocols easily.

Definition A.2.1. A term t' is called a *component* of term t if t' cannot be split to another term t'', and t is built by concatenating t' with arbitrary terms.

Conveniently, we refer the example of Thayer[31]. If we have a term like $B\{r_a, K, \{r_b\}_{PK_B}\}_{PK_A}, r_a$, then it contains three components: $B, \{r_a, K, \{r_b\}_{PK_B}\}_{PK_A}$, and r_a.

Definition A.2.2. For a strand s, a term t is *new* at $n = \langle s, i\rangle$ if t is a component of term(n), but t is not a component of node $\langle s, j\rangle$ for every $j < i$.

Taking NS protocol as an example, the component $\{r_a, r_b, B\}_{PK_B}$ is new at node $\langle st, 2\rangle$ since it is clearly not a component of $\langle st, 1\rangle$.

Definition A.2.3. Suppose that $n \in \mathcal{B}$ is positive, $a \in \mathcal{A}$ is a subterm of $term(n)$. The edge $n \Rightarrow^+ n'$ is a *transformed edge* for a if there exits a negative node $n' \in \mathcal{B}$, and there is a new component t_2 of n' such that $a \sqsubseteq t_2$.

Respectively, *a transforming edge* is denoted.

Definition A.2.4. Suppose that $n \in \mathcal{B}$ is negative, $a \in \mathcal{A}$ is a subterm of $term(n)$. The edge $n \Rightarrow^+ n'$ is a *transforming edge* for a if there exits a positive node $n' \in \mathcal{B}$, and there is a new component t_2 of n' such that $a \sqsubseteq t_2$.

For example, the edge $\langle st, 1\rangle \Rightarrow \langle st, 2\rangle$ is a transformed edge for the term r_a, and $\langle st', 1\rangle \Rightarrow \langle st', 2\rangle$ is a transforming edge for r_a.

Definition A.2.5. The edge $n \Rightarrow^+ n'$ is *a test* for $a \in \mathcal{A}$ if a uniquely originates at n, and $n \Rightarrow^+ n'$ is a transformed edge for a.

The transformed edge $\langle st, 1\rangle \Rightarrow \langle st, 2\rangle$ is a test for r_a since r_a uniquely originates at $\langle st, 1\rangle$.

Definition A.2.6. Suppose that $n, n' \in \mathcal{B}$.

1. The edge $n \Rightarrow^+ n'$ is *a outgoing test* for $a \sqsubseteq t = \{c\}_K$ if it is a test for a in which $K^{-1} \notin P$, a does not occur in any component of n other than t. Moreover, t is a test component for a in n.

2. The edge $n \Rightarrow^+ n'$ is *a incoming test* for $a \sqsubseteq t_1 = \{c\}_K$ if it is a test for a in which $K \notin P$, and t_1 is a test component for a in n'.

The transformed edge $\langle st, 1\rangle \Rightarrow \langle st, 2\rangle$ could be considered as an incoming test for r_a.

Subsequently, authentication tests [124] are provided as powerful and simple tools to guarantee existence of regular strands in a bundle.

Authentication Test 1: Suppose that $n' \in \mathcal{B}$, and $n \Rightarrow^+ n'$ is outgoing test for $a \sqsubseteq t$ with $t = term(n)$. Then there exist regular nodes $m, m' \in \mathcal{B}$ such that t is a component of m, and $m \Rightarrow^+ m'$ is a transforming edge for a. In addition that a occurs only in component

$t_1 = \{c_1\}_{K_1}$ of m', that t_1 is not a proper subterm of any regular component, and that $K_1^{-1} \notin P$. There is a negative regular node with t_1 as a component.

Authentication Test 2: Suppose that $n \in \mathcal{B}$, and $n \Rightarrow^+ n'$ is incoming test for $a \sqsubseteq t'$ with $t' = term(n')$. Then there exist regular nodes $m, m' \in \mathcal{B}$ such that t' is a component of m', and $m \Rightarrow^+ m'$ is a transforming edge for a.

Definition A.2.7. A negative node is an *unsolicited test* for $t = \{c\}_K$ if t is a test component for any a in n and $K \notin P$.

Authentication Test 3: Suppose that a node n is in a bundle $\mathcal{B}$, and n be an unsolicited test for $t = \{c\}_K$, then there exists a positive regular node $m \in \mathcal{B}$ such that t is a component of m.

The proofs of these authentication tests are out of scope in this part. So if readers eager to deeply understand the proofs, please regard to the paper [78].

A.3 Shape and Skeleton

In design protocol, ones always desire that there is only one possible execution of their protocol in any scenario. Nevertheless, there might exist some executions relative to the protocol's assumptions. Actually, the executions of protocols normally have very few essentially different forms that called *shapes*. Then, authentication and secrecy properties can be determined by examining the shapes.

Precisely, a shape is a local execution by honest principals. Partial information about a principal's execution of a protocol is called *skeleton*. Skeletons are partial-ordered structures, or fragments of message sequence chart. Moreover, a skeleton is *realised* if it is not fragmented, i.e. it contains exactly the regular behaviour of some executions. A realised skeleton is a shape if it is minimal.

A preskeleton describes the regular parts of a set of bundles. Formally presented, preskeleton is defined as follows.

Definition A.3.1 (Skeleton). A four-tuple $\mathcal{A} = (node, \preceq, non, unique)$ is a preskeleton if:

1. $node$ is a finite set of regular nodes, $n_1 \in node$ and $n_0 \Rightarrow^+ n_1$ implies $n_0 \in nodes$;
2. $\preceq$ is a partial ordering on $node$ such that $n_0 \Rightarrow^+ n_1$ implies $n_0 \preceq n_1$;
3. non is a set of keys where if $K \in non$, then for all $n \in node, K \not\sqsubseteq term(n)$, and for some $n' \in node$, either K or K^{-1} is used in $term(n')$;

4. *unique* is a set of atoms where $a \in unique$, for some $n \in node, a \sqsubseteq term(n)$.

A preskeleton $\mathcal{A}$ is a *skeleton* if in addition:

4'. $a \in unique$ implies a originates at no more than one $n \in node$.

Definition A.3.2 (Shape). $\mathcal{A}'$ is a *shape* for $\mathcal{A}$ if (1) some $H : \mathcal{A} \to \mathcal{A}'$, (2) $\mathcal{A}'$ is realised, and (3) no proper skeleton of (A') satisfies (1) and (2).

A strand could be considered as a skeleton. And a bundle could be considered as a shape. Furthermore, a bundle containing a penetrator strand is also a possible shape of the protocol.

A.4 Penetrator Model

Original Strand Spaces theory uses Dolev-Yao [29] model as its penetrator model. Penetrator's power is built up from two ingredients: initially known keys available to the penetrator, and actions that allows the penetrator manipulates messages. The actions are summarized to discard message, to generate arbitrary messages, to concatenate messages together, and to apply cryptographic operation using available keys. The model is described as follows.

Definition A.4.1. A penetrator trace is *one of the following*:

M. Text message: $\langle +t \rangle$ where $t \in T$

F. Flushing: $\langle -g \rangle$

T. Tee: $\langle -g, +g, +g \rangle$

C. Concatenation $\langle -g, -h, +gh \rangle$

S. Separation into components: $\langle -gh, +g, +h \rangle$

K. Key: $\langle +K \rangle$ where $K \in \mathcal{K}_{\mathcal{P}}$

E. Encryption: $\langle -K, -h, +\{c\}_K \rangle$

D. Decryption: $\langle -K^{-1}, -\{c\}_K, +h \rangle$

This penetrator's trace set given here could be extended if desired without any modification on the whole model. However, the proofs should be adjusted to take into account the additional penetrator traces. This ability gives us an open space to add some physical penetrator traces without worry of proving way.

Appendix B

Analysis Wong-Stajano Protocol in AVISPA

B.1 Transformed Protocol

```
%Wong Stajano Protocol - OOB Transformation
%( F: hash function, Ks: pre-shared key,
%ACK: confirm message)
%( attacker knowledge: G, Hash, A, B)

A->B : Ga
B->A : Gb.h(Ga.Gb.Rb.Kb)

A->B : Rat.{F(ACK.Rat)}_Ks
B->A : Rbt.F(Rat.Rbt)
A->B : ACK.{F(ACK.Rat.Rbt)}_Ks

B->A : Rb.{4.F(Rb.Ks)}_Ks

B->A : Kb
A->B : {ACK.1.A.B}_Ks %confirm acceptance

%verify keys
SK = exp(Gb,Ea) or SK = exp(Ga,Ea)
A->B : {A.Rat3}_SK
B->A : {B.Rat.Rb3}_SK
A->B : {B.Rbt3}_SK
```

B.2 Source Code

```
%%Wong Stajano Protocol - OOB Transformation
% ( F: hash function, Ks: pre-shared key, ACK: confirm message)
% ( attacker knowledge: G, Hash, A, B)
%
% A->B : Ga
% B->A : Gb.h(Ga.Gb.Rb.Kb)
% A->B : Rat.{3.F(ACK.Rat)}_Ks
% B->A : Rbt.F(3.Rat.Rbt)
% A->B : ACK.{3.F(ACK.Rat.Rbt)}_Ks
% B->A : Rb.{4.F(Rb.Ks)}_Ks
% B->A : Kb
% A->B : {ACK.1.A.B}_Ks %confirm acceptance
%
% verify keys
% SK = exp(Gb,Ea) or SK = exp(Ga,Ea)
% A->B : {A.Rat}_SK
% B->A : {B.Rat.Rbt}_SK
% A->B : {B.Rbt}_SK
%

role alice(A, B : agent,
          G: text,
      ACK: message,
      F: hash_func,
          Ks: symmetric_key,
          SND,RCV: channel(dy))

played_by A def=
        local State   : nat,
        Ea: text,
        Ga, Gb: message,
        Kb,SK: symmetric_key,
        Rb, Rat, Rbt : text,
         Commit: hash(message. message.text.symmetric_key),
        Hb: hash(text.symmetric_key)

init State := 0

transition

1. State = 0 /\ RCV(start)
   =|>
```

```
   State' := 2 /\ Ea' := new()
               /\ Ga' := exp(G,Ea')
               /\ SND(Ga')

2. State = 2 /\ RCV(Gb'.Commit')
   =|>
   State' := 4 /\ Rat' := new()
         /\ SND(Rat'.{3.F(ACK.Rat')}_Ks)

3. State = 4 /\ RCV(Rbt'.Hb')
             /\ Hb' = F(3.Rat.Rbt')
   =|>
   State' := 6 /\ SND(ACK.{3.F(ACK.Rat.Rbt')}_Ks)



4. State = 6 /\ RCV(Rb'.{4.Hb'}_Ks)
             /\ Hb' = h(Rb'.Ks)
             /\ RCV(Kb')
         /\ Commit = F(Ga.Gb.Rb'.Kb')
   =|>
State' := 8 /\ SND({ACK.1.A.B}_Ks)
             /\ SK' := exp(Gb,Ea)
             /\ Rat':= new()
             /\ SND({A.Rat'}_SK)
             /\ secret(Rat',rat,{A,B})
             /\ witness(A,B,bob_alice_rat,Rat')

5. State = 8 /\ RCV({B.Rat.Rbt'}_SK)
   =|>
   State':= 10 /\ SND({B.Rbt'}_SK)
              /\ request(A,B,alice_bob_rbt,Rbt')

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role bob( B, A : agent,
          G: text,
          ACK: message,
          F: hash_func,
          Ks: symmetric_key,
          SND,RCV: channel(dy))

played_by B def=
```

```
        local State   : nat,
        Eb: text,
        Ga, Gb: message,
        Kb,SK: symmetric_key,
        Rb, Rat, Rbt : text,
         Commit: hash(message. message.text.symmetric_key),
        Ha: hash(message.text),
    Ha2: hash(message.text.symmetric_key)


init
        State := 1

transition

1. State = 1 /\  RCV(Ga')
   =|>
   State' := 3 /\ Rb' := new()
               /\ Kb' := new()
               /\ Eb' := new()
               /\ Gb' := exp(G,Eb')
           /\ SK' := exp(Ga',Eb')
           /\ Commit' := F(Ga'.Gb'.Rb'.Kb')
               /\ SND(Gb'.Commit')

2. State = 3 /\ RCV(Rat'.{3.Ha'}_Ks)
   =|>
   State' := 5 /\ Rbt' := new()
                   /\ SND(Rbt'.F(3.Rat'.Rbt'))

3. State = 5   /\ RCV(ACK.{3.Ha2'}_Ks)
               /\ Ha2' = F(ACK.Rat.Rbt)
               /\ Ha = F(ACK.Rat)
   =|>
   State' := 7 /\ SND(Rb.{4.F(Rb.Ks)}_Ks)
               /\ SND(Kb)

4. State = 7   /\ RCV({ACK.1.A.B}_Ks)
               /\ RCV({A.Rat'}_SK')
   =|>
   State' := 9 /\ Rbt' := new()
              /\ SND({B.Rat'.Rbt'}_SK)
              /\ secret(Rbt',rbt,{A,B})
              /\ witness(B,A,alice_bob_rbt,Rbt')
```

```
5. State = 9  /\ RCV({B.Rbt}_SK)
   =|>
   State' := 11 /\ request(B,A,bob_alice_rat,Rat)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role session(A,B : agent,
             G: text,
         ACK: message,
             F: hash_func,
             Kab: symmetric_key)

def=
  local SA, RA, SB, RB: channel (dy)

  composition
     alice(A,B,G,ACK,F,Kab,SA,RA)
  /\ bob  (B,A,G,ACK,F,Kab,SB,RB)

end role
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role environment() def=

    const a, b          : agent,
        g                   : text,
        ack                    : message,
        h           : hash_func,
    kab         : symmetric_key,
    rat, rbt,
        alice_bob_rbt,
        bob_alice_rat        : protocol_id

    intruder_knowledge={a,b,g,h}


   composition
      session(a,b,g,ack,h,kab)
   /\ session(a,i,g,ack,h,kab)
   /\ session(i,b,g,ack,h,kab)
```

```
end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

goal

  secrecy_of rat, rbt
  authentication_on alice_bob_rbt
  authentication_on bob_alice_rat

end goal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
environment()
```

B.3 Results

The attack on transformed WS protocol is found in AVISPA as follow.

```
SUMMARY
  UNSAFE

DETAILS
  ATTACK_FOUND
  UNTYPED_MODEL

PROTOCOL
  /Users/trungtran/span//testsuite/results/wong_avispa4.if

GOAL
  Secrecy attack on (n14(Rbt))

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 34303 states
  Reachable  : 13966 states
  Translation: 0.02 seconds
```

```
 Computation: 2.58 seconds

ATTACK TRACE
 i -> (a,6):  start
 (a,6) -> i:  exp(g,n21(Ea))

 i -> (a,3):  start
 (a,3) -> i:  exp(g,n1(Ea))

 i -> (b,4):  g
 (b,4) -> i:  exp(g,n11(Eb)).{g.exp(g,n11(Eb)).n11(Rb).n11(Kb)}_h

 i -> (a,6):  Gb(22).Commit(22)
 (a,6) -> i:  n22(Rat).{3.{ack.n22(Rat)}_h}_kab

 i -> (b,4):  n22(Rat).{3.{ack.n22(Rat)}_h}_kab
 (b,4) -> i:  n12(Rbt).{3.n22(Rat).n12(Rbt)}_h

 i -> (a,6):  n12(Rbt).{3.n22(Rat).n12(Rbt)}_h
 (a,6) -> i:  ack.{3.{ack.n22(Rat).n12(Rbt)}_h}_kab

 i -> (b,4):  ack.{3.{ack.n22(Rat).n12(Rbt)}_h}_kab
 (b,4) -> i:  n11(Kb).n11(Rb).{4.{n11(Rb).kab}_h}_kab

 i -> (a,3):  Gb(2).{exp(g,n1(Ea)).Gb(2).n11(Rb).Kb(4)}_h
 (a,3) -> i:  n2(Rat).{3.{ack.n2(Rat)}_h}_kab

 i -> (b,10):  Ga(31)
 (b,10) -> i:  exp(g,n31(Eb)).{Ga(31).exp(g,n31(Eb)).n31(Rb).n31(Kb)}_h

 i -> (b,10):  n2(Rat).{3.{ack.n22(Rat)}_h}_kab
 (b,10) -> i:  n32(Rbt).{3.n2(Rat).n32(Rbt)}_h

 i -> (a,3):  n32(Rbt).{3.n2(Rat).n32(Rbt)}_h
 (a,3) -> i:  ack.{3.{ack.n2(Rat).n32(Rbt)}_h}_kab

 i -> (a,3):  Kb(4).n11(Rb).{4.{n11(Rb).kab}_h}_kab
 (a,3) -> i:  {a.n4(Rat)}_dummy_sk.{ack.1.a.b}_kab
              & Secret(n4(Rat),set_92);  Witness(a,b,bob_alice_rat,n4(Rat));
              & Add a to set_92;  Add b to set_92;

 i -> (b,4):  {a.n4(Rat)}_dummy_sk.{ack.1.a.b}_kab
 (b,4) -> i:  {b.n4(Rat).n14(Rbt)}_(exp(g,n11(Eb)))
```

```
& Secret(n14(Rbt),set_110);  Witness(b,a,alice_bob_rbt,n14(Rbt));
& Add a to set_110;  Add b to set_110;
```

Appendix C

Implementation of 2-Move Secure Device Pairing on Arduino

C.1 UDP Client Source Code

```
#include <ecc.h>
#include <string.h>
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>
#include <sha1.h>

//client information
byte mac[] =  {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEB };
IPAddress client_ip(192,168,1,2);
unsigned int client_port = 9998;

//An EthernetUDP instance
EthernetUDP Udp;

//server information
IPAddress server_ip(192,168,1,1);
unsigned int server_port = 9999;

//client ECC key
EccPoint l_Q2;
uint8_t l_secret_server[NUM_ECC_DIGITS + 1];
uint8_t l_secret_client[NUM_ECC_DIGITS + 1];
uint8_t l_shared2[NUM_ECC_DIGITS +1];
```

```
uint8_t l_random2[NUM_ECC_DIGITS +1];
uint8_t l_shared1[NUM_ECC_DIGITS +1];// shared key from server, debugging purpose

//random of server
byte random_server[4];
//random of client
byte random_client[4];
//commitment from client
byte client_commit[20];
//decommitment from client
byte client_decommit[4];
//client say request
byte say = 0;

byte packetBuffer[UDP_TX_PACKET_MAX_SIZE +1]; //buffer to hold incoming packet,

//setup LED for visible light communication
const int ledPin = 9;      // the pin that the LED is attached to
byte byteOn = 150; //the brightness of 1
byte byteOff = 10; //the brightness of 0

unsigned long gtime1,gtime2;

//------------------begin setup--------------------------------//
void init_key()
{
  memset(l_secret_server,0,NUM_ECC_DIGITS  + 1);
  memset(l_secret_client,0,NUM_ECC_DIGITS  + 1);
  memset(l_shared2,0,NUM_ECC_DIGITS  + 1);
  memset(l_random2,0,NUM_ECC_DIGITS  + 1);

}
void setup()
{
    //start the Ethernet UDP
   Ethernet.begin(mac,client_ip);
   Udp.begin(client_port);

   Serial.begin(9600);

   unsigned long time1,time2;
   time1 = millis();
   //generate ECC key
   randomSeed(analogRead(0));
```

```
   getRandomBytes(l_secret_client, NUM_ECC_DIGITS * sizeof(uint8_t));
   getRandomBytes(l_random2, NUM_ECC_DIGITS * sizeof(uint8_t));
   ecc_make_key(&l_Q2, l_secret_client, l_secret_client);

    time2 = millis();
    Serial.print("Key generation ms:");
    Serial.println(time2-time1);

    //generate a client random
    getRandomBytes(random_client, 4 * sizeof(uint8_t));
    printDebug("random client",random_client,4);

    calculate_commit();

    //setup for VLC
    pinMode(ledPin, OUTPUT);
   delay(500);
}
//--------------------end setup--------------------------------//

int isRequestOne = 0;

//------------------begin loop----------------------//
void loop()
{
   //send hello 99 to server until receiving the request 1 from server
   if(isRequestOne == 0)
   {
      say = 99;
      sndMsg(&say,1);
      isRequestOne =1;
   }
   //when receive a message from client
   int packetSize = Udp.parsePacket();
   if(Udp.available())
   {
     Udp.read(packetBuffer,UDP_TX_PACKET_MAX_SIZE);
    // Serial.println("Contents:");
     //printHex(packetBuffer,packetSize);
   }

   //if it is a request
   if(packetSize == 1)
   {
```

```
      //process request from client
      byte request;
      request = packetBuffer[0];
      Serial.print("Received request:");
      Serial.println(request);
      processRequestedMsg(request);
    }
    //when receive a data from client
    if(packetSize > 1)
    {
      receiveData(packetBuffer,packetSize);
    }
    if(packetSize == 0) delay(200);
}
//------------------end loop----------------------//
//send a message
void sndMsg(byte *buf, int len)
{
    Udp.beginPacket(server_ip,server_port);
    Udp.write(buf,len);
    Udp.endPacket();
    delay(300);
}
//----------------------begin processing request--------------------------------//
//process request from server
void processRequestedMsg(int request)
{
    if(request == 1)
    {
      //send client key
      sndMsg(l_secret_client,NUM_ECC_DIGITS);
      //stop send hello to server
      isRequestOne = 1;
      //for debug
      printDebug("send client key",l_secret_client,NUM_ECC_DIGITS);
    }
    if(request == 2)
    {
      //send client commit
       sndMsg(client_commit,20);
      //for debug
      printDebug("send client commit",client_commit,20);
    }
    if(request == 21)
```

```
{
  //send request 3
  say  = 3;
  sndMsg(&say,1);
  //for debug
  Serial.println("Send request 3");
}
if(request == 5)
{
  unsigned long time1,time2;
   time1 = millis();
  //send decommit
   //padding random_server into 20byte hash key
   byte hashKey[20];
   memset(hashKey,0,20);
   memcpy(hashKey,random_server,4);
   //calculate SHA1 with key
   Sha1.initHmac(hashKey,20);
   byte hmacInput[49];
   memset(hmacInput,0,49);
   memcpy(hmacInput,l_secret_client,NUM_ECC_DIGITS);
   memcpy(hmacInput + NUM_ECC_DIGITS,l_secret_server,NUM_ECC_DIGITS);
   hmacInput[48]='\n';
   Sha1.print((char*)hmacInput);

   //take 4-frist byte of HMAC
   byte hashMac[4];
   memcpy(hashMac,Sha1.resultHmac(),4);

   //calculate decommit
   int i;
   for(i = 0;i<4;i++)
      client_decommit[i]= random_client[i]^hashMac[i];

   time2 = millis();
   Serial.print("Generating decommit value spends ms:");
   Serial.println(time2-time1);
   //send 32-bits decommit on VLC channel
   delay(1000);
   sndMsgtoLED(ledPin,client_decommit);

   printDebug("client decommit",client_decommit,4);
}
//request for debug key
```

```
    if(request == 51)
    {
      //send request 6
      say = 6;
      sndMsg(&say,1);
      Serial.println("Send request 6");
    }
}

//process received data from server
void receiveData(const byte *buf,int packetSize)
{

  if(say == 3)
  {
    //accept l_secret_server, check if the data is a key or not - based on size
     if(packetSize == NUM_ECC_DIGITS){
        memcpy(l_secret_server,buf, NUM_ECC_DIGITS);
        //for debug
        printDebug("Received server key",l_secret_server,NUM_ECC_DIGITS);

    // send say = 4
      say = 4;
      sndMsg(&say,1);
      Serial.println("Send request 4");
     }
     else
      Serial.println("Cannot parse client key");
  }
  else
  if(say == 4)
  {
    //accept server random
    if(packetSize == 4){
       memcpy(random_server,buf,4);
       //for debug
      printDebug("Received server random",random_server,4);
     }
     else
      Serial.println("Cannot parse server random value");

    //send ACK 41
    byte ACK = 41;
    sndMsg(&ACK,1);
```

```
    Serial.println("Send request 41");
  }
  //for debug only
  if(say == 6)
  {
    //generate shared key

     if(packetSize == 2)
     {
       if(buf[0] == 1 && buf[1] == 1)
         {
           Serial.println("Server accepts the connection");
            generate_sharedkey();
         }
       else if(buf[0] == 0 && buf[1] == 0)
         {
            Serial.println("Server rejects the connection");
         }
     }
  }
}

//----------------------tools------------------------//
void getRandomBytes(uint8_t *arr,int arrLen)
{
  int i;
  for(i =0;i<arrLen;i++)
    arr[i]=random(0,256);
}

void printHex(const byte* arr,int len)
{
  int i;
  char ptr1[10];
  char ptr2[10];
  String mystring;
  for (i=0; i<len; i++) {
      sprintf(ptr1,"%x",arr[i]>>4);
      sprintf(ptr2,"%x",arr[i]&0xf);
      mystring += ptr1;
      mystring += ptr2;
      Serial.print(mystring);
      mystring.remove(0,mystring.length());
  }
```

```
  Serial.println("");
}

//generate a shared key
void generate_sharedkey(){
       //generate a shared key
   unsigned long time1,time2;
   time1 = millis();
   if (!ecdh_shared_secret(l_shared2, &l_Q2, l_secret_server,l_random2))
    {
        Serial.println("shared_secret() failed (2)\n");
           return ;
    }
    time2 = millis();
    Serial.print("Generating shared key spends ms:");
    Serial.println(time2-time1);
    Serial.print("shared_secret:");
    printHex(l_shared2,NUM_ECC_DIGITS);
}

void printDebug(char *text,const byte *arr, int len)
{
   Serial.println(text);
   printHex(arr,len);
}

//----------------------send LED to server----------------------
void sndMsgtoLED(int LedPin,byte *rnd)
{
  Serial.println("Send message via LED");
   unsigned long time1,time2;
   time1 = millis();
  int i,j;
  byte temp[4];
  memset(temp,0,4);

  byte isOn =0;
  for(j =0;j<4;j++)
  {
    temp[j] =1;
    for(i=0;i<8;i++)
    {
        isOn = temp[j] & rnd[j];
        if(isOn > 0)
```

```
        {
          analogWrite(LedPin, byteOn);
          Serial.print("1");
          delay(100);
        }
        else
        {
          analogWrite(LedPin, byteOff);
          Serial.print("0");
          delay(100);
        }
        temp[j] = temp[j] << 1;
    }
    Serial.print(" ");
  }
  Serial.println("");
  analogWrite(LedPin, 0);
  time2 = millis();
  Serial.print("Sending 32-bits via VLC spends ms:");
  Serial.println(time2-time1);
}

void calculate_commit()
{
   unsigned long time1,time2;
   time1 = millis();
   unsigned char hashKey = 0;
   Sha1.init();
   Sha1.initHmac(&hashKey,1);
   char input[29];
   memset(input,0,29);
   memcpy(input,l_secret_client,24);
   input[24] = random_client[0];
   input[25] = random_client[1];
   input[26] = random_client[2];
   input[27] = random_client[3];
   Sha1.print(input);
   memcpy(client_commit,Sha1.resultHmac(),20);
   time2 = millis();
   Serial.print("Calculating commitment spends ms:");
   Serial.println(time2-time1);
}
```

C.2 UDP Server Source Code

```
#include <ecc.h>
#include <SPI.h>
#include <Ethernet.h>
#include <EthernetUdp.h>
#include <string.h>
#include <sha1.h>

//EEC Keys
EccPoint l_Q1;
uint8_t l_secret_server[NUM_ECC_DIGITS  + 1];
uint8_t l_secret_client[NUM_ECC_DIGITS + 1];
uint8_t l_shared1[NUM_ECC_DIGITS + 1];
uint8_t l_random1[NUM_ECC_DIGITS +1 ];

//server information
byte mac[] =  {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xEA };
IPAddress server_ip(192,168,1,1);
unsigned int server_port = 9999;

//An EthernetUDP instance
EthernetUDP Udp;

//random of server
byte random_server[4];
//random of client
byte random_client[4];
//commitment from client
byte client_commit[20];
//decommitment from client
byte client_decommit[4];

byte ACK = 0;

byte packetBuffer[UDP_TX_PACKET_MAX_SIZE + 1]; //buffer to hold incoming packet,

//visible light communication
int photocellPin = 0;     // the cell and 10K pulldown are connected to a0
int brightZero = 600;
int brightOne = 900;
int  photocellReading; //photocell value
byte vlcBuffer[4];
```

```
byte isAccept = 0;

//--------------------begin setup--------------------------------//
void init_key()
{
  memset(l_secret_server,0,NUM_ECC_DIGITS  + 1);
  memset(l_secret_client,0,NUM_ECC_DIGITS  + 1);
  memset(l_shared1,0,NUM_ECC_DIGITS  + 1);
  memset(l_random1,0,NUM_ECC_DIGITS  + 1);
}
void setup()
{
   //start the Ethernet UDP
   Ethernet.begin(mac,server_ip);
   Udp.begin(server_port);
   Serial.begin(9600);
   init_key();
   //generate ECC random and key
   randomSeed(analogRead(0));
   getRandomBytes(l_secret_server, NUM_ECC_DIGITS * sizeof(uint8_t));
   getRandomBytes(l_random1, NUM_ECC_DIGITS * sizeof(uint8_t));
   ecc_make_key(&l_Q1, l_secret_server, l_secret_server);

   //generate a random RA
    getRandomBytes(random_server, 4 * sizeof(uint8_t));
    //setup for VLC
    memset(vlcBuffer,0,4);
    delay(500);
}
//--------------------end setup--------------------------------//
//----------------------------------------------------------------//
byte say = 0;

//send a message
void sndMsg(byte *buf, int len)
{
    Udp.beginPacket(Udp.remoteIP(), Udp.remotePort());
    Udp.write(buf,len);
    Udp.endPacket();
    delay(300);
}

//process request from client
void processRequestedMsg(int request)
```

```
{
    if(request == 99)
    {
      //send say = 1
      if(say == 0)
      {
        say =1;
        Serial.println("Send ACK = 1");
        sndMsg(&say,1);
        isAccept = 0;
      }
    }
    if(request == 3)
    {
      //send l_secret_server
      if(say == 21)
      {
         Serial.println("Send server key");
         sndMsg(l_secret_server,NUM_ECC_DIGITS);
      }
    }
    if(request == 4)
    {
      //send ra
        Serial.println("Send server's random ");
        sndMsg(random_server,4);
    }
    if(request == 41)
    {
      //send say = 5
         say = 5;
         sndMsg(&say,1);
         photoReading();
         //---received client decommit then extract ra
         memcpy(client_decommit,vlcBuffer,4);
         printDebug("Received client decommit",client_decommit,4);
          extract_client_random();

          //send ACK 51
          ACK = 51;
          Serial.println("Send ACK = 51");
          sndMsg(&ACK,1);
          //generate random key when commitment is checked
          //commitment_check();
```

```
          if(commitment_check() == 0)
          {
             generate_sharedkey();
             isAccept = 1;
          }
          else
            isAccept = 0;
     }
    //request for debug shared key
    if(request == 6)
    {
       byte OK[2];
       if(isAccept == 1)
       {
         OK[0] =1;
         OK[1] = 1;
         sndMsg(OK,2);
       }
       else
       {
         OK[0] = 0;
         OK[1] = 0;
         sndMsg(OK,2);
       }
      say =0;
    }
}

//process received data from client
void receiveData(const byte *buf, int packetSize)
{
  if(say == 1)
  {
    //accept l_secret_client, check if the data is a key or not - based on size
     if(packetSize == NUM_ECC_DIGITS){

        memcpy(l_secret_client,buf,NUM_ECC_DIGITS);
        printDebug("Received client key",l_secret_client,NUM_ECC_DIGITS);
            // send say = 2
        say = 2;
        sndMsg(&say,1);
        Serial.println("Send request 2");
     }
     else
```

```
      Serial.println("Cannot parse client key");
  }
  else
  if(say == 2)
  {
    //accept commit h(l_client_secret,random_client)
    if(packetSize == 20){
       memcpy(client_commit,buf,20);
       printDebug("Received client commit",client_commit,20);
      //send ACK 21
       say = 21;
       sndMsg(&say,1);
       Serial.println("Send request 21");
     }
     else
      Serial.println("Cannot parse client commit value");
  }
}

//------------------begin loop----------------------//
void loop()
{
   int packetSize = Udp.parsePacket();

   if(Udp.available())
   {
     Udp.read(packetBuffer,UDP_TX_PACKET_MAX_SIZE);
     //Serial.println("Contents:");
     //printHex(packetBuffer,packetSize);
   }

   //when receive a request from client

   if(packetSize == 1)
   {
     //process request from client
     byte request;
     request = packetBuffer[0] ;
     Serial.print("Received request:");
     Serial.println(request);
     processRequestedMsg(request);
   }
   //when receive a data from client
   if(packetSize > 1)
```

```
   {
     receiveData(packetBuffer,packetSize);
   }
   if(packetSize == 0) delay(200);
}
//-----------------end loop----------------------//
//----------------------tools-------------------------//
void getRandomBytes(uint8_t *arr,int arrLen)
{
  int i;
  for(i =0;i<arrLen;i++)
    arr[i]=random(0,256);
}

void printHex(const byte* arr,int len)
{
  int i;
  char ptr1[10];
  char ptr2[10];
  String mystring;
  for (i=0; i<len; i++) {
      sprintf(ptr1,"%x",arr[i]>>4);
      sprintf(ptr2,"%x",arr[i]&0xf);
      mystring += ptr1;
      mystring += ptr2;
      Serial.print(mystring);
      mystring.remove(0,mystring.length());
  }
  Serial.println("");
}

//generate a shared key
void generate_sharedkey(){
       //generate a shared key
   if (!ecdh_shared_secret(l_shared1, &l_Q1, l_secret_client,l_random1))
    {
        Serial.println("shared_secret() failed (2)\n");
            return ;
    }
    Serial.print("shared_secret:");
    printHex(l_shared1,NUM_ECC_DIGITS);
}
void printDebug(char *text,const byte *arr, int len)
{
```

```
   Serial.println(text);
   printHex(arr,len);
}

//----------------receive decommit value on VLC channel
void photoReading()
{
   int loop_count =0;
   int index = 0;
   memset(vlcBuffer,0,4);
   while(loop_count <1000)
   {
       photocellReading = analogRead(photocellPin);
       if(photocellReading < brightZero)
       {
          //Serial.println(" - ");
          //index = 0;
          delay(10);
       }
      else{
          delay(45);
          for(index = 1;index <=32;index ++)
          {
             if(photocellReading < brightZero)
             {
                 delay(5);
                 index = index -1;
             }
             if(photocellReading <= brightOne && photocellReading > brightZero)
             {
                 //if(index <= 32)
                 //   VLCBuffer[index-1] = 0;
                 delay(100);
             }else
            if(photocellReading >= brightOne)
            {
                 //Serial.print("1");
                 if(index <= 32){
                  vlcBuffer[(index-1)/8] = vlcBuffer[(index-1)/8] | ((byte)1<<((index-1)%8));
                 delay(98);
                 }
             }
             photocellReading = analogRead(photocellPin);
          }
```

```
          //print_arr();
          break;
      }
      loop_count++;
   }
}

void extract_client_random()
{
       //calculate hmac
       //padding random_server into 20byte hash key
       byte hashKey[20];
       memset(hashKey,0,20);
       memcpy(hashKey,random_server,4);
       //calculate SHA1 with key
       Sha1.initHmac(hashKey,20);
       byte hmacInput[49];
       memset(hmacInput,0,49);
       memcpy(hmacInput,l_secret_client,NUM_ECC_DIGITS);
       memcpy(hmacInput + NUM_ECC_DIGITS,l_secret_server,NUM_ECC_DIGITS);
       hmacInput[48]='\n';
       Sha1.print((char*)hmacInput);

       //take 4-frist byte of HMAC
       byte hashMac[4];
       memcpy(hashMac,Sha1.resultHmac(),4);
       //calculate random_client by xor client_decommit with hashMac
       int i;
       for(i = 0;i<4;i++)
        random_client[i]= client_decommit[i]^hashMac[i];
       //print debug
       printDebug("extracting random_client:",random_client,4);
}

int commitment_check()
{
    unsigned char hashKey = 0;
    char hashResult[20];
    Sha1.init();
    Sha1.initHmac(&hashKey,1);

   char input[29];
   memset(input,0,29);
   memcpy(input,l_secret_client,24);
```

```
   input[24] = random_client[0];
   input[25] = random_client[1];
   input[26] = random_client[2];
   input[27] = random_client[3];
   Sha1.print(input);
   memcpy(hashResult,Sha1.resultHmac(),20);
   printDebug("hashResult:",(byte*)hashResult,20);
    if(memcmp(hashResult,client_commit,20) == 0)
    {
        Serial.println("commitment checked successfully");
        return 0;
    }
    else
      Serial.println("commitment checked fail");
      return 1;
}
```

Bibliography

[1] P Guillemin H Sundmaeker and P Friess. *Vision and challenges for realising the Internet of Things*. Number 978-92-79-15088-3. European Union, March 2010.

[2] Emmanouil Vasilomanolakis, Jörg Daubert, Manisha Luthra, Vangelis Gazis, Alexander Wiesmaier, and Panagiotis Kikiras. On the security and privacy of internet of things architectures and systems. In *International Workshop on Secure Internet of Things*. Springer, 2015. to appear.

[3] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36, February 1990. ISSN 0734-2071. doi: 10.1145/77648.77649.

[4] S. Mirzadeh, H. Cruickshank, and R. Tafazolli. Secure device pairing: A survey. *Communications Surveys Tutorials, IEEE*, 16(1):17–40, First 2014. ISSN 1553-877X.

[5] Ford Long Wong and Frank Stajano. Multichannel security protocols. *IEEE Pervasive Computing*, 6(4):31–39, 2007. ISSN 1536-1268.

[6] Daniele Raffo, Cédric Adjih, Thomas Clausen, and Paul Mühlethaler. An advanced signature system for olsr. In *Proceedings of the 2Nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN '04, pages 10–16, New York, NY, USA, 2004. ACM. ISBN 1-58113-972-1.

[7] Stefan Brands and David Chaum. Distance-bounding protocols. In Tor Helleseth, editor, *Advances in Cryptology EUROCRYPT 93*, volume 765 of *Lecture Notes in Computer Science*, pages 344–359. Springer Berlin Heidelberg, 1994. ISBN 978-3-540-57600-6.

[8] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36, February 1990. ISSN 0734-2071. doi: 10.1145/77648.77649.

[9] Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. *J. Comput. Secur.*, 6(1-2):85–128, January 1998. ISSN 0926-227X.

[10] Catherine Meadows, Radha Poovendran, Dusko Pavlovic, LiWu Chang, and Paul Syverson. Distance bounding protocols: Authentication logic analysis and collusion attacks. In Radha Poovendran, Sumit Roy, and Cliff Wang, editors, *Secure Localization and Time Synchronization for Wireless Sensor and Ad Hoc Networks*, volume 30 of *Advances in Information Security*, pages 279–298. Springer US, 2007. ISBN 978-0-387-32721-1.

[11] L. Buttyan and Ta Vinh Thong. Formal verification of secure ad-hoc network routing protocols using deductive model-checking. In *Wireless and Mobile Networking Conference (WMNC), 2010 Third Joint IFIP*, pages 1–6, Oct 2010.

[12] James L. Peterson. Petri nets. *ACM Comput. Surv.*, 9(3):223–252, September 1977. ISSN 0360-0300.

[13] Provable security of on-demand distance vector routing in wireless ad hoc networks. In Refik Molva, Gene Tsudik, and Dirk Westhoff, editors, *Security and Privacy in Ad-hoc and Sensor Networks*, volume 3813 of *Lecture Notes in Computer Science*, pages 113–127. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-30912-3.

[14] Martín Abadi and Andrew D. Gordon. A calculus for cryptographic protocols: The spi calculus. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, CCS '97, pages 36–47, New York, NY, USA, 1997. ACM. ISBN 0-89791-912-2.

[15] F.J. Thayer Fabrega, J.C. Herzog, and J.D. Guttman. Strand spaces: why is a security protocol correct? In *Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on*, pages 160–171, May 1998.

[16] Shu-Dong Shi. Formal verification and improvement of a secure protocol for ad hoc networks. In *Wireless Communications, Networking and Mobile Computing, 2008. WiCOM '08. 4th International Conference on*, pages 1–3, Oct 2008.

[17] Kurt Jensen. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use, Vol. 2.* Springer-Verlag, London, UK, UK, 1995. ISBN 3-540-58276-2.

[18] F. Erbas, K. Kyamakya, and K. Jobmann. Modelling and performance analysis of a novel position-based reliable unicast and multicast routing method using coloured petri nets. In *Vehicular Technology Conference, 2003. VTC 2003-Fall. 2003 IEEE 58th*, volume 5, pages 3099–3104 Vol.5, Oct 2003.

[19] G. Ács, L. Buttyan, and I. Vajda. The security proof of a link-state routing protocol for wireless sensor networks. In *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE International Conference on*, pages 1–6, Oct 2007.

[20] Gergely Ács, Levente Buttyán, and István Vajda. Modelling adversaries and security objectives for routing protocols in wireless sensor networks. In *Proceedings of the Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN '06, pages 49–58, New York, NY, USA, 2006. ACM. ISBN 1-59593-554-1.

[21] Shahan Yang. Modeling vulnerabilities of ad hoc routing protocols. In *In Proceedings of the ACM Workshop on Security of Ad Hoc and Sensor Networks*, pages 12–20, 2003.

[22] F. Nait-Abdesselam, B. Bensaou, and T. Taleb. Detecting and avoiding wormhole attacks in wireless ad hoc networks. *Communications Magazine, IEEE*, 46(4):127–133, April 2008. ISSN 0163-6804. doi: 10.1109/MCOM.2008.4481351.

[23] Yongjian Li and Jun Pang. Extending the strand space method to verify kerberos v. In *Proceedings of the Eighth International Conference on Parallel and Distributed Computing, Applications and Technologies*, PDCAT '07, pages 437–444, Washington, DC, USA, 2007. IEEE Computer Society. ISBN 0-7695-3049-4.

[24] Raphaël Jamet and Pascal Lafourcade. Formal model for (k)-neighborhood discovery protocols. In Evangelos Kranakis, editor, *Advances in Network Analysis and its Applications*, volume 18 of *Mathematics in Industry*, pages 181–207. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-30903-8. doi: 10.1007/978-3-642-30904-5_8.

[25] S. Capkun and J.-P. Hubaux. Secure positioning in wireless networks. *Selected Areas in Communications, IEEE Journal on*, 24(2):221–232, Feb 2006. ISSN 0733-8716. doi: 10.1109/JSAC.2005.861380.

[26] Federico Crazzolara and Glynn Winskel. Petri nets in cryptographic protocols. In *Proceedings of the 15th International Parallel &Amp; Distributed Processing Symposium*, IPDPS '01, pages 149–, Washington, DC, USA, 2001. IEEE Computer Society. ISBN 0-7695-0990-8.

[27] David Basin, Srdjan Capkun, Patrick Schaller, and Benedikt Schmidt. Let's get physical: Models and methods for real-world security protocols. In *Proceedings of the 22Nd International Conference on Theorem Proving in Higher Order Logics*, TPHOLs '09, pages 1–22, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-03358-2.

[28] Robin Sharp and Michael R. Hansen. Timed traces and strand spaces. In *Proceedings of the Second International Conference on Computer Science: Theory and Applications*, CSR'07, pages 373–386, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-74509-2, 978-3-540-74509-9.

[29] D. Dolev and Andrew C. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, Mar 1983. ISSN 0018-9448.

[30] UeliM. Maurer and PierreE. Schmid. A calculus for secure channel establishment in open networks. In Dieter Gollmann, editor, *Computer Security — ESORICS 94*, volume 875 of *Lecture Notes in Computer Science*, pages 173–192. Springer Berlin Heidelberg, 1994. ISBN 978-3-540-58618-0.

[31] F.Javier Thayer, Vipin Swarup, and JoshuaD. Guttman. Metric strand spaces for locale authentication protocols. In Masakatsu Nishigaki, Audun JAzsang, Yuko Murayama, and Stephen Marsh, editors, *Trust Management IV*, volume 321 of *IFIP Advances in Information and Communication Technology*, pages 79–94. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-13445-6.

[32] Khan Pathan Al-Sakib. *Security of Self-Organizing Networks: MANET, WSN, WMN, VANET*. ISBN: 978-1-4398-1920-3. Auerbach Publications, October 14, 2010.

[33] Frank Stajano and Ross Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In Bruce Christianson, Bruno Crispo, JamesA. Malcolm, and Michael

Roe, editors, *Security Protocols*, volume 1796 of *Lecture Notes in Computer Science*, pages 172–182. Springer Berlin Heidelberg, 2000. ISBN 978-3-540-67381-1.

[34] M.T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and clear: Human-verifiable authentication based on audio. In *Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on*, pages 10–10, 2006.

[35] Claudio Soriente, Gene Tsudik, and Ersin Uzun. Hapadep: Human-assisted pure audio device pairing. In *Proceedings of the 11th international conference on Information Security*, ISC '08, pages 385–400, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-85884-3.

[36] Felix Xiaozhu Lin, Daniel Ashbrook, and Sean White. Rhythmlink: securely pairing i/o-constrained devices by tapping. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 263–272, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1.

[37] Nitesh Saxena, Md. Borhan Uddin, and Jonathan Voris. Universal device pairing using an auxiliary device. In *Proceedings of the 4th symposium on Usable privacy and security*, SOUPS '08, pages 56–67, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-276-4.

[38] Young Sam Kim, Seung-Hyun Kim, and Seung-Hun Jin. Srs-based automatic secure device pairing on audio channels. In *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pages 1–6, 2010.

[39] Stephan Sigg, Dominik Schuermann, and Yusheng Ji. Pintext: A framework for secure communication based on context. In Alessandro Puiatti and Tao Gu, editors, *Mobile and Ubiquitous Systems: Computing, Networking, and Services*, volume 104 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 314–325. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-30972-4.

[40] Michael T. Goodrich, Michael Sirivianos, John Solis, Claudio Soriente, Gene Tsudik, and Ersin Uzun. Using audio in secure device pairing. *Int. J. Secur. Netw.*, 4(1/2):57–68, February 2009. ISSN 1747-8405.

[41] J.M. McCune, A. Perrig, and M.K. Reiter. Seeing-is-believing: using camera phones for human-verifiable authentication. In *Security and Privacy, 2005 IEEE Symposium on*, pages 110–124, 2005.

[42] Adrian Perrig and Dawn Song. Hash visualization: a new technique to improve real-world security. In *In International Workshop on Cryptographic Techniques and E-Commerce*, pages 131–138, 1999.

[43] Carl Ellison and Steve Dohrmann. Public-key support for group collaboration. *ACM Trans. Inf. Syst. Secur.*, 6(4):547–565, November 2003. ISSN 1094-9224.

[44] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel. In *Security and Privacy, 2006 IEEE Symposium on*, pages 6 pp.–313, 2006.

[45] Claudio Soriente, Gene Tsudik, and Ersin Uzun. Beda: Button-enabled device association. Inter- national Workshop on Security for Spontaneous Interaction (IWSSI), UbiComp Workshop Proceedings (2007), 2007.

[46] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel: Design and usability study. *Information Forensics and Security, IEEE Transactions on*, 6(1):28–38, 2011. ISSN 1556-6013.

[47] Rene Mayrhofer, Mike Hazas, and Hans Gellersen. An authentication protocol using ultrasonic ranging. Technical report, 2006.

[48] GeorgeT. Amariucai, Clifford Bergman, and Yong Guan. An automatic, time-based, secure pairing protocol for passive rfid. In Ari Juels and Christof Paar, editors, *RFID. Security and Privacy*, volume 7055 of *Lecture Notes in Computer Science*, pages 108–126. Springer Berlin Heidelberg, 2012. ISBN 978-3-642-25285-3.

[49] R. Mayrhofer and M. Welch. A human-verifiable authentication protocol using visible laser light. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, pages 1143–1148, 2007.

[50] Ileana Buhan, Jeroen Doumen, Pieter Hartel, and Raymond Veldhuis. Feeling is believing: a location limited channel based on grip pattern biometrics and cryptanalysis, 2006.

[51] LarsErik Holmquist, Friedemann Mattern, Bernt Schiele, Petteri Alahuhta, Michael Beigl5, and Hans-W. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In GregoryD. Abowd, Barry Brumitt, and Steven Shafer, editors, *Ubicomp 2001: Ubiquitous Computing*, volume 2201 of *Lecture Notes in Computer Science*, pages 116–122. Springer Berlin Heidelberg, 2001. ISBN 978-3-540-42614-1.

[52] Jonathan Lester, Blake Hannaford, and Gaetano Borriello. Are you with me?" – using accelerometers to determine if two devices are carried by the same person. In *In Proceedings of Second International Conference on Pervasive Computing (Pervasive 2004*, pages 33–50, 2004.

[53] Rene Mayrhofer and Hans Gellersen. Shake well before use: Authentication based on accelerometer data. In Anthony LaMarca, Marc Langheinrich, and KhaiN. Truong, editors, *Pervasive Computing*, volume 4480 of *Lecture Notes in Computer Science*, pages 144–161. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-72036-2.

[54] Ahren Studer, Timothy Passaro, and Lujo Bauer. Don't bump, shake on it: the exploitation of a popular accelerometer-based smart phone exchange and its secure replacement. In *Proceedings of the 27th Annual Computer Security Applications Conference*, ACSAC '11, pages 333–342, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0672-0.

[55] Bogdan Groza and Rene Mayrhofer. Saphe: simple accelerometer based wireless pairing with heuristic trees. In *Proceedings of the 10th International Conference on Advances in Mobile Computing ; Multimedia*, MoMM '12, pages 161–168, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1307-0.

[56] Ming Ki Chong, Gary Marsden, and Hans Gellersen. Gesturepin: using discrete gestures for associating mobile devices. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, MobileHCI '10, pages 261–264, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-835-3.

[57] Oyuntungalag Chagnaadorj and Jiro Tanaka. Mimicgesture: Secure device pairing with accelerometer-based gesture input. In Youn-Hee Han, Doo-Soon Park, Weijia Jia, and Sang-Soo Yeo, editors, *Ubiquitous Information Technologies and Applications*, volume 214 of *Lecture Notes in Electrical Engineering*, pages 59–67. Springer Netherlands, 2013. ISBN 978-94-007-5856-8.

[58] Claude Castelluccia and Pars Mutaf. Shake them up!: a movement-based pairing protocol for cpu-constrained devices. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, MobiSys '05, pages 51–64, New York, NY, USA, 2005. ACM. ISBN 1-931971-31-5.

[59] Adin Scannell, Alexander Varshavsky, and Anthony Lamarca. Amigo: Proximity-based authentication of mobile devices. In *In Ubicomp*, pages 253–270, 2007.

[60] Secure communication method and apparatus. U.S. Patent Number 5,450,493, sep 1995.

[61] Dirk Balfanz Smetters, Dirk Balfanz, D. K. Smetters, Paul Stewart, and H. Chi Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In Symposium on Network and Distributed Systems Security (NDSS '02), San Diego, CA,2002., 2002.

[62] C. Mitchell, C. Gehrmann, and K. Nyberg. Manual authentication for wireless devices. *Cryptobytes*, January 2004.

[63] K. Nyberg. C. Gehrmann. Security in personal area networks. *Security for Mobility*, pages 191–230, 2004.

[64] Jaap-Henk Hoepman. The ephemeral pairing problem. In Ari Juels, editor, *Financial Cryptography*, volume 3110 of *Lecture Notes in Computer Science*, pages 212–226. Springer Berlin Heidelberg, 2004. ISBN 978-3-540-22420-4.

[65] L. H. Nguyen and A. W. Roscoe. Authentication protocols based on low-bandwidth unspoofable channels: A comparative survey. *J. Comput. Secur.*, 19(1):139–201, January 2011. ISSN 0926-227X.

[66] Serge Vaudenay. Secure communications over insecure channels based on short authenticated strings. In Victor Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 309–326. Springer Berlin Heidelberg, 2005. ISBN 978-3-540-28114-6.

[67] Sylvain Pasini and Serge Vaudenay. Sas-based authenticated key agreement. In Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors, *Public Key Cryptography - PKC 2006*, volume 3958 of *Lecture Notes in Computer Science*, pages 395–409. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-33851-2.

[68] Sven Laur and Kaisa Nyberg. Efficient mutual data authentication using manually authenticated strings. In David Pointcheval, Yi Mu, and Kefei Chen, editors, *Cryptology and Network Security*, volume 4301 of *Lecture Notes in Computer Science*, pages 90–107. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-49462-1.

[69] M. Cagalj, S. Capkun, and J-P Hubaux. Key agreement in peer-to-peer wireless networks. *Proceedings of the IEEE*, 94(2):467–478, 2006. ISSN 0018-9219.

[70] Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In DouglasR. Stinson, editor, *Advances in Cryptology — CRYPTO' 93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer Berlin Heidelberg, 1994. ISBN 978-3-540-57766-9.

[71] Frank Stajano, Graeme Jenkinson, Jeunese Payne, Max Spencer, Quentin Stafford-Fraser, and Chris Warrington. Bootstrapping adoption of the pico password replacement system. In Bruce Christianson, James Malcolm, Vashek MatyÃ!'Å!', Petr Å venda, Frank Stajano, and Jonathan Anderson, editors, *Security Protocols XXII*, volume 8809 of *Lecture Notes in Computer Science*, pages 172–186. Springer International Publishing, 2014. ISBN 978-3-319-12399-8.

[72] Frank Stajano. Pico: No more passwords! In Bruce Christianson, Bruno Crispo, James Malcolm, and Frank Stajano, editors, *Security Protocols XIX*, volume 7114 of *Lecture Notes in Computer Science*, pages 49–81. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-25866-4.

[73] P. Ryan and S. Schneider. *The Modelling and Analysis of Security Protocols: The Csp Approach*. Addison-Wesley Professional, first edition, 2000. ISBN 0-201-67471-8.

[74] William Claycomb and Dongwan Shin. Extending formal analysis of mobile device authentication. *Journal of Internet Services and Information Security (JISIS)*, 1(1):86–102, 5 2011.

[75] Richard Chang and Vitaly Shmatikov. Formal analysis of authentication in bluetooth device pairing. In *Proc. of LICS/ICALP Workshop on Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis (FCS-ARSPA)*, Wroclaw, Poland, July 2007.

[76] Marie Duflot, Marta Kwiatkowska, Gethin Norman, and David Parker. A formal analysis of bluetooth device discovery. *International Journal on Software Tools for Technology Transfer*, 8(6):621–632, 2006. ISSN 1433-2779.

[77] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Trans. Inf. Theor.*, 29(2):198–208, September 2006. ISSN 0018-9448.

[78] Joshua D. Guttman and F. Javier Thayer. Authentication tests and the structure of bundles. *Theor. Comput. Sci.*, 283(2):333–380, June 2002. ISSN 0304-3975.

[79] J.C. Herzog. The diffie-hellman key-agreement scheme in the strand-space model. In *Computer Security Foundations Workshop, 2003. Proceedings. 16th IEEE*, pages 234–247, June 2003.

[80] Gavin Lowe. A hierarchy of authentication specifications. In *Computer Security Foundations Workshop, 1997. Proceedings., 10th*, pages 31–43, Jun 1997.

[81] Gavin Lowe. Casper: a compiler for the analysis of security protocols. In *Computer Security Foundations Workshop, 1997. Proceedings., 10th*, pages 18–30, Jun 1997.

[82] Bruno Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *14th IEEE Computer Security Foundations Workshop (CSFW-14)*, pages 82–96, Cape Breton, Nova Scotia, Canada, June 2001. IEEE Computer Society.

[83] Jesus Diaz, David Arroyo, and Francisco B. Rodriguez. On securing online registration protocols: Formal verification of a new proposal. *Knowledge-Based Systems*, 59(0):149 – 158, 2014. ISSN 0950-7051.

[84] Jun Han, Yue-Hsun Lin, Adrian Perrig, and Fan Bai. Mvsec: Secure and easy-to-use pairing of mobile devices with vehicles. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks*, WiSec '14, pages 51–56, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2972-9.

[85] Giampaolo Bella, Cristiano Longo, and LawrenceC Paulson. Verifying second-level security protocols. In David Basin and Burkhart Wolff, editors, *Theorem Proving in Higher Order Logics*, volume 2758 of *Lecture Notes in Computer Science*, pages 352–366. Springer Berlin Heidelberg, 2003. ISBN 978-3-540-40664-8.

[86] Christopher Dilloway and Gavin Lowe. On the specification of secure channels. In *Proceedings of the Workshop on Issues in the Theory of Security (WITS '07)*, 2007.

[87] Allaa Kamil and Gavin Lowe. Understanding abstractions of secure channels. In Pierpaolo Degano, Sandro Etalle, and Joshua Guttman, editors, *Formal Aspects of Security and Trust*, volume 6561 of *Lecture Notes in Computer Science*, pages 50–64. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-19750-5.

[88] Shaddin F. Doghmi, Joshua D. Guttman, and F. Javier Thayer. Skeletons, homomorphisms, and shapes: Characterizing protocol executions. *Electron. Notes Theor. Comput. Sci.*, 173:85–102, April 2007. ISSN 1571-0661.

[89] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P. C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron. The avispa tool for the automated validation of internet security protocols and applications. In *Proceedings of the 17th International Conference on Computer Aided Verification*, CAV'05, pages 281–285, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-27231-3, 978-3-540-27231-1.

[90] Marcin POTURALSkI. *Secure Neighbor Discovery and Ranging in Wireless Networks.* PhD thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, 2011.

[91] T.J. Waraksa, K.D. Fraley, R.E. Kiefer, D.G. Douglas, and L.H. Gilbert. Passive keyless entry system, July 17 1990. US Patent 4,942,393.

[92] Patrick Schaller Panos Papadimitratos, Marcin Poturalski. Secure neighborhood discovery: A fundamental element for mobile ad hoc networking.

[93] Gerhard P. Hancke and Markus G. Kuhn. An rfid distance bounding protocol. In *Proceedings of the First International Conference on Security and Privacy for Emerging Areas in Communications Networks*, SECURECOMM '05, pages 67–73, Washington, DC, USA, 2005. IEEE Computer Society. ISBN 0-7695-2369-2.

[94] Srdjan Čapkun, Levente Buttyán, and Jean-Pierre Hubaux. Sector: Secure tracking of node encounters in multi-hop wireless networks. In *Proceedings of the 1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, SASN '03, pages 21–32, New York, NY, USA, 2003. ACM. ISBN 1-58113-783-4.

[95] David B. Johnson Yih-Chun Hu, Adrian Perrig. Packet leashes: A defense against wormhole attacks in wireless ad hoc networks. San Francisco, CA, USA. INFOCOM 2003.

[96] J. Eriksson, S.V. Krishnamurthy, and Michalis Faloutsos. Truelink: A practical countermeasure to the wormhole attack in wireless networks. In *Network Protocols, 2006. ICNP '06. Proceedings of the 2006 14th IEEE International Conference on*, pages 75–84, Nov 2006.

[97] Loukas Lazos and Radha Poovendran. Serloc: Robust localization for wireless sensor networks. *ACM Trans. Sen. Netw.*, 1(1):73–100, August 2005. ISSN 1550-4859. doi: 10.1145/1077391.1077395.

[98] L. Lazos, R. Poovendran, and S. Capkun. Rope: robust position estimation in wireless sensor networks. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 324–331, April 2005. doi: 10.1109/IPSN.2005.1440942.

[99] S. Capkun, M. Cagalj, and M. Srivastava. Secure localization with hidden and mobile base stations. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–10, April 2006.

[100] Yanchao Zhang, Wei Liu, Wenjing Lou, and Yuguang Fang. Location-based compromise-tolerant security mechanisms for wireless sensor networks. *Selected Areas in Communications, IEEE Journal on*, 24(2):247–260, Feb 2006. ISSN 0733-8716.

[101] Reza Shokri, Marcin Poturalski, Gael Ravot, Panos Papadimitratos, and Jean-Pierre Hubaux. A practical secure neighbor verification protocol for wireless sensor networks. In *Proceedings of the Second ACM Conference on Wireless Network Security*, WiSec '09, pages 193–200, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-460-7.

[102] Kasper Bonne Rasmussen and S. Capkun. Implications of radio fingerprinting on the security of sensor networks. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 331–340, Sept 2007. doi: 10.1109/SECCOM.2007.4550352.

[103] Nur Serinken Oktay Ureten. Wireless security through rf fingerprinting.

[104] Vladimir Brik, Suman Banerjee, Marco Gruteser, and Sangho Oh. Wireless device identification with radiometric signatures. In *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, pages 116–127, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-096-8. doi: 10.1145/1409944.1409959.

[105] S. Jana and S.K. Kasera. On fast and accurate detection of unauthorized wireless access points using clock skews. *Mobile Computing, IEEE Transactions on*, 9(3):449–462, March 2010. ISSN 1536-1233. doi: 10.1109/TMC.2009.145.

[106] B. Danev and S. Capkun. Transient-based identification of wireless sensor nodes. In *Information Processing in Sensor Networks, 2009. IPSN 2009. International Conference on*, pages 25–36, April 2009.

[107] O. Ureten and N. Serinken. Detection of radio transmitter turn-on transients. *Electronics Letters*, 35(23):1996–1997, Nov 1999. ISSN 0013-5194.

[108] Liang Xiao, L. Greenstein, N. Mandayam, and W. Trappe. Fingerprints in the ether: Using the physical layer for wireless authentication. In *Communications, 2007. ICC '07. IEEE International Conference on*, pages 4646–4651, June 2007.

[109] Liang Xiao, L. Greenstein, N. Mandayam, and W. Trappe. A physical-layer technique to enhance authentication for mobile terminals. In *Communications, 2008. ICC '08. IEEE International Conference on*, pages 1520–1524, May 2008. doi: 10.1109/ICC.2008.294.

[110] Liang Xiao, L.J. Greenstein, Narayan B. Mandayam, and W. Trappe. Using the physical layer for wireless authentication in time-variant channels. *Wireless Communications, IEEE Transactions on*, 7(7):2571–2579, July 2008. ISSN 1536-1276. doi: 10.1109/TWC.2008.070194.

[111] Ruoheng Liu and Wade Trappe. *Securing Wireless Communications at the Physical Layer*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 144191384X, 9781441913845.

[112] Lingxuan Hu and David Evans. Using directional antennas to prevent wormhole attacks. San Diego, California, USA, February 2004. In Network and Distributed System Security Symposium (NDSS 2004).

[113] Yanchao Zhang and Yanchao Zhang. Wormhole-resilient secure neighbor discovery in underwater acoustic networks. In *INFOCOM, 2010 Proceedings IEEE*, pages 1–9, March 2010. doi: 10.1109/INFCOM.2010.5462093.

[114] Weichao Wang and Aidong Lu. Interactive wormhole detection and evaluation. *Information Visualization*, 6(1):3–17, March 2007. ISSN 1473-8716. doi: 10.1057/palgrave.ivs.9500144.

[115] R. Maheshwari, Jie Gao, and S.R. Das. Detecting wormhole attacks in wireless networks using connectivity information. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 107–115, May 2007. doi: 10.1109/INFCOM.2007.21.

[116] W. Znaidi, M. Minier, and J.-P. Babau. Detecting wormhole attacks in wireless networks using local neighborhood information. In *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, pages 1–5, Sept 2008.

[117] Dezun Dong, Mo Li, Yunhao Liu, Xiang-Yang Li, and Xiangke Liao. Topological detection on wormholes in wireless ad hoc and sensor networks. *Networking, IEEE/ACM Transactions on*, 19(6):1787–1796, Dec 2011. ISSN 1063-6692.

[118] C. Cremers, K.B. Rasmussen, B. Schmidt, and S. Capkun. Distance hijacking attacks on distance bounding protocols. In *Security and Privacy (SP), 2012 IEEE Symposium on*, pages 113–127, May 2012.

[119] Jean Leneutre Lin Chen, Xiaoyun Xue. A lightweight mechanism to secure olsr. pages 887–895. IMECS, 2006.

[120] Optimized link state routing protocol (olsr), 2003.

[121] Asma Adnane, Christophe Bidan, and Rafael Timóteo de Sousa Júnior. Trust-based security for the {OLSR} routing protocol. *Computer Communications*, 36(10–11):1159 – 1171, 2013. ISSN 0140-3664.

[122] Marcin Poturalski, Panos Papadimitratos, and Jean-Pierre Hubaux. Towards provable secure neighbor discovery in wireless networks. In *Proceedings of the 6th ACM Workshop on Formal Methods in Security Engineering*, FMSE '08, pages 31–42, New York, NY, USA, 2008. ACM. ISBN 978-1-60558-288-7.

[123] P. Schaller, B. Schmidt, D. Basin, and S. Capkun. Modeling and verifying physical properties of security protocols for wireless networks. In *Computer Security Foundations Symposium, 2009. CSF '09. 22nd IEEE*, pages 109–123, July 2009.

[124] Joshua D. Guttman and F. Javier Thayer. Authentication tests. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, SP '00, pages 96–, Washington, DC, USA, 2000. IEEE Computer Society. ISBN 0-7695-0665-8.

[125] R. Moskowitz Z. Cao R. Cragie B. Sarikaya, Y. Ohba. Security bootstrapping solution for resource-constrained devices. Internet-Draft, July 10 2012.

[126] C. Jennings. Transitive trust enrollment for constrained devices. Network Working Group Internet-Draft, 4 2013.

[127] Seung Wook Jung and Souhwan Jung. Secure bootstrapping and rebootstrapping for resource-constrained thing in internet of things. *International Journal of Distributed Sensor Networks*, 2015.

[128] Hong Yu and Jingsha He. Trust-based mutual authentication for bootstrapping in 6lowpan. *Journal of Communications*, 7(8), 2012.

[129] HyunSoo Cha, Ki-Hyung Kim, and SeungHwa Yoo. Lbp: A secure and efficient network bootstrapping protocol for 6lowpan. In *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, ICUIMC '11, pages 54:1–54:8, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0571-6.

[130] Muhammad Ikram, Aminul Haque Chowdhury, Bilal Zafar, Hyon-Soo Cha, Ki-Hyung Kim, Seung-Wha Yoo, and Dong-Kyoo Kim. A simple lightweight authentic bootstrapping protocol for ipv6-based low rate wireless personal area networks (6lowpans). In *Proceedings of the 2009 International Conference on Wireless Communications and Mobile Computing: Connecting the World Wirelessly*, IWCMC '09, pages 937–941, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-569-7.

[131] R. Bonetto, N. Bui, V. Lakkundi, A. Olivereau, A. Serbanati, and M. Rossi. Secure communication for smart iot objects: Protocol stacks, use cases and practical examples. In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2012 IEEE International Symposium on a*, pages 1–7, June 2012.

[132] J. Hui D. Culler G. Montenegro, N. Kushalnagar. Transmission of ipv6 packets over ieee 802.15.4 networks. RFC 4944, 2007.

[133] J. Suomalainen. Smartphone assisted security pairings for the internet of things. In *Wireless Communications, Vehicular Technology, Information Theory and Aerospace Electronic Systems (VITAE), 2014 4th International Conference on*, pages 1–5, May 2014.

[134] Young-Guk Ha. Dynamic integration of zigbee home networks into home gateways using osgi service registry. *Consumer Electronics, IEEE Transactions on*, 55(2):470–476, May 2009. ISSN 0098-3063.

[135] A. Rubens Merit W. Simpson C. Rigney, S. Willen. Remote authentication dial in user service. RFC 2865, 2000.

[136] B. Patil H. Tschofenig D. Forsberg, Y. Ohba and A. Yegin. Rfc5191: Protocol for carrying authentication for network access (pana). IETF Request For Comments,, 2008.

[137] B. Sarikaya. Security bootstrapping of ieee 802.15.4 based internet of things. Network Working Group, 5 2015.

[138] R. Hurst D. Simon, D. Simon. The eap-tls authentication protocol. RFC 5216, March 2008.

[139] R. Hummen R. Moskowitz, Ed. Hip diet exchange (dex). Internet-Draft, July 2015.

[140] A. Colegrove G. Gross H. Harney, U. Meth. Group secure association key management protocol. RFC 4535, 2006.

[141] T. Hardjono B. Weis, S. Rowles. The group domain of interpretation. RFC 6407, Oct 2011.

[142] S. Blake-Wilson P. Funk. Extensible authentication protocol tunneled transport layer securuty authenticated protocol version. RFC 5281, 2008.

[143] F. Lindholm M. Naslund K. Norrman J. Arkko, E. Carrara. Mikey: Multimedia internet keying. RFC 3830, 2004.

[144] Trung Nguyen and Jean Leneutre. A secure and effective device pairing protocol. In *Consumer Communications and Networking Conference (CCNC), 2015 12th Annual IEEE*, pages 507–512, Jan 2015.

[145] Dawn Xiaodong Song. Athena: A new efficient automatic checker for security protocol analysis. In *Proceedings of the 12th IEEE Workshop on Computer Security Foundations*, CSFW '99, pages 192–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0201-6.

[146] J. Thayer, J. Herzog, and J. Guttman. Honest ideals on strand spaces. In *Proceedings of the 11th IEEE Workshop on Computer Security Foundations*, CSFW '98, pages 66–, Washington, DC, USA, 1998. IEEE Computer Society. ISBN 0-8186-8488-7.

[147] F. Javier Thayer, Jonathan C. Herzog, and Joshua D. Guttman. Mixed strand spaces. In *Proceedings of the 12th IEEE Workshop on Computer Security Foundations*, CSFW '99, pages 72–, Washington, DC, USA, 1999. IEEE Computer Society. ISBN 0-7695-0201-6.

[148] Allaa Kamil and Gavin Lowe. Analysing tls in the strand spaces model. *J. Comput. Secur.*, 19(5):975–1025, September 2011. ISSN 0926-227X.

[149] Roger M. Needham and Michael D. Schroeder. Using encryption for authentication in large networks of computers. *Commun. ACM*, 21(12):993–999, December 1978. ISSN 0001-0782.