How to bootstrap security for ad-hoc network: Revisited

Wook Shin, Carl A. Gunter, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka

Abstract There are various network-enabled and embedded computers deployed around us. Although we can get enormous conveniences by connecting them together, it is difficult to securely associate them in an ad-hoc manner. The difficulties originate from authentication and key distribution problems among devices that are strangers to each other. In this paper, we review the existing ways of initiating secure communication for ad-hoc network devices, and propose another solution. Exploiting Pairing-based cryptography and the notion of location-limited channel, the proposed solution bootstraps security conveniently and efficiently. Further, it supports ownership enforcement and key-escrow.

1 Introduction

The number of computer-embedded intelligent devices deployed around us keeps increasing as the technology evolves. The devices are sometimes network-enabled to give even more benefits. Although the advantages can be augmented when a user can connect the devices together on demand,

it is being obstructed by security and privacy threats. The communications over the intelligent and networked devices (called as "embedded devices" or just "devices", hereinafter) can be protected cryptographically, but bootstrapping security is not easy.

Security bootstrapping that includes key generation/distribution and authentication tends to impose configuration burdens upon users. For example, users need to follow a series of instruction steps for WPA2-PSK (WiFi Protected Access 2, Pre-shared key) configuration, even though the pre-shared key mode is the simplest option for using WPA. Establishing security among devices becomes more complicated in an ad-hoc network since there is no trusted entity always available online.

Wook Shin, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka
KDDI R&D Laboratories, Inc., 2-1-15 Ohara Fujimino-shi Saitama 356-8502, Japan, e-mail: {wookshin, kiyomoto, ka-fukushima, toshi}@kddilabs.jp

Carl A. Gunter
Dept. of Computer Science, University of Illinois at Urbana-Champaign, IL 61801, USA, http://seclab.uiuc.edu

In this paper, we look over the related existing technologies and propose a rather intuitive and useful way of bootstrapping security for networked devices. Taking advantage of Pairing-based cryptography and the notion of location-limited channel, the proposed method provides an easy, secure, and efficient way of creating private communication channels over devices. A user does not have to follow intricate commands, but just brings a special device close to other devices to create a secure channel. Besides, users can acquire privileged ownership and key escrow support on their own channels. Only a channel owner can manage membership of the owned channel and reveal any secret over the channel. Our method also can be applied to other forms of networks (e.g., home Wi-Fi networks), not only to ad-hoc networks.

2 Background

Stajano and Anderson[26] addressed security bootstrapping difficulties in an ad-hoc network, which are caused by the absence of an online trusted entity. To tackle the problem, the authors suggested using a side channel approach, instead of relying on public key infrastructures that require online servers to confirm the validity of signed certificates, or traditional symmetric key-based ticket solutions[21, 15, 23] that need a ticket granting server.

In their scheme, devices exchange authentication information via an out-of-band channel, and then authenticate each other online based on the exchanged information.

Balfanz et al.[6] extended the idea by Stajano and Anderson and clarified the notion of *preauthentication* information that is exchanged in *location-limited side channel*. They listed the characteristics of the side channel as *demonstrative identification*, *authenticity*, and *secrecy*. The communication media of the side channel need to have special physical characteristics (e.g., a very short communication range and directed propagation) so that users visually identify to whom they are talking. In another study[4], the authors demonstrated an alternative peer authentication using an IrDA implemented location limited channel. When a user brings a computer to a wireless access point (AP), the two devices exchange pre-authentication information via IrDA ports, and then contact each other over an 802.11 network to execute further handshake protocols. Mccune et al.[19] used two dimensional (2D) barcodes and camera phones. In this study, a camera phone is used to authenticate devices. A device displays 2D barcodes that contain authentication digests of the device, and then a camera phone reads the barcodes and authenticates the device online based on the digest. A variety of out-of-band communication mediums[12, 17, 18] have been proposed to deliver secret information as well, such as sound, gestures, and laser lights.

Although the above approaches have utilized diverse communication mediums, there is still room for improvement in the usability and security aspects. Some require a user to bring a device to the other device, but it is not very practical when devices are heavy and physically apart from each other. Some require a user to perform delicate tasks. In the laser light approach, a user must hold the light emitter stably to complete the information transmission, but it can be difficult for seniors, especially persons experiencing hand tremors. Some need special equipment. The 2D barcode scheme requires a device to have a display screen, which could increase costs for a small device like a finger oximeter. Instead of having a display, a device can have printed barcodes on its surface, but the printed information could be miss-

ing or replaced by something else. Audio and gesture signals can be observed by an attacker, so it is not useful in public places (e.g., an airport or a station).

Cryptographic techniques in previous efforts need to be reconsidered as well. Conventional public key cryptography (e.g., RSA) could impose a high computational load and power demand for small devices. Symmetric key schemes impose key management overhead as the number of devices increases, and threaten the security of others by exposing shared secrets if one of devices were compromised or if a malicious device were accidentally connected.

Additionally, users need administrative authority to give or deprive membership on the secure channel they create over multiple devices. Private devices would allow only authorized users to have the authority. Public devices may be open to anyone for channel creation, but each channel should be distinguished and managed by its creator. The enforcement of ownership is useful when devices are invited into an administrative domain (e.g., a home, hospital, or company network). The owner or administrator of the domain would allow invited devices to use and interact with other network resources for a limited period. Channel owners should also be able to investigate communication history and decrypt messages on owned channels. The key escrow is useful for auditing and tracing anomalies in institutions and enterprises. Such ownership representation and key escrow need to be supported cryptographically but have not been considered adequately in previous approaches.

Although it is not a necessary requirement of the security bootstrapping in general, we try to support protected broadcast. Sometimes, one-way and non-critical notifications need be broadcast to participants on a channel. For example, in a home automation environment, a sensor on the main door could wake up all devices in a room from sleep mode when the master entered.

3 Requirements and our approaches

We try to provide an easy method of bootstrapping security, so that anyone can securely create and manage private communication channels over embedded devices. Some embedded system applications are designed for even non-computer literate seniors[16]. After reviewing the existing technologies in ad-hoc security bootstrapping, we can list the requirements that our system has to meet: 1) *user-friendly way of establishing security*, 2) *ownership representation and key escrow support*, 3) *low overheadperformance, power, and key management*, and 4) *protected unicast and broadcast*.

In order to compose a solution addressing the purpose, we take the following approaches to exploit existing technologies.

How to intermediate: A USB flash drive is often used as a mediator of security establishment between Wi-Fi network devices, by delivering certificates or preshared keys. This kind of small mediator is rather handy for exchanging secret information between devices than direct contact of devices (cf. the IrDA approach[4]). Moreover, the intermediary could provide a user-friendly UI, store the configuration of created channels, and substantiate user ownership. It can also deliver security policies as the notion of “universal controller”[26], but policy enforcement is not the concern of this paper. For convenience, we call the intermediary AID (authentication intermediary device), hereinafter.

Communication media:Although a variety of communication mediums could be utilized to implement a location limited channel, a few wireless solutions seem to

be plausible considering usability. RF is one possible medium. Bluetooth, a popular RF technology, could be employed, but it has a long working range where we cannot identify hidden participants. Some RFID techniques can be empowered within a short range, like within a few centimeters. However, RFID tags only return stored information or static series of information. Some have achieved selective responses to unauthorized reading of RFID tags, but the tags can be duplicated. Moreover, the tags are not highly programmable and do not have sufficient computational power yet for our purpose.

A high-speed infrared solution like Giga-IR[2] could be useful. It uses a directed wave, so devices need to be aligned along the line-of-sight, although usability would be improved with accessories helping alignment (e.g., docks or clips). The virtue of this technology is it provides secure and high-speed transmission at low cost (the module costs about 20 cents).

Recent efforts in very short range wireless communications are also noticeable. Transfer Jet[3] is a promising technology. Although based on omni-directional electromagnetic waves, its working range is only 3 centimeters, which is fairly shorter than that of general Near Field Communication, so that a user can identify all participants to the established communication. This helps to fulfill the *authenticity* and *demonstrative identity*. We expect our scheme to be embodied in very short-range wireless communication, but it is not tied to a specific medium. Any wireless communication is applicable if it has a short working range sufficient for users to identify communication participants.

Cryptography: On the one hand, we want to take advantage of asymmetric key-based cryptography (e.g., key management and signing and non-repudiation functionalities). On the other hand, we cannot impose the burdens of computation and high demand of power upon the devices. Envisioning a small embedded device in a personal network, it is reasonable to have hardware constraints similar to the typical wireless sensor networks (WSN), that is, 8-bit microprocessors with several hundred kilobytes of RAM and ROM.

Additionally, we need to support privileged administration of the created communication channels, which means only the channel owner can administrate security parameters and channel membership. Moreover, *key escrow* needs to be supported so that users can decrypt all messages and investigate stored information over their own channels as needed. We also need to protect messages that are transferred between two devices and protect messages that are broadcast to all devices of a channel.

Pairing-based cryptography (or PBC) is very suitable for our purpose. Although it is not as light as Elliptic Curve Cryptography, PBC imposes very little performance and power overheads comparing to RSA[14, 27]. Moreover, it allow us to provide such useful functionalities with practical security as key escrow, ownership enforcement, and message unicast and broadcast.

4 System description

Although the functions of AID can be implemented on top of a variety of handheld devices, a cell phone might be the most plausible device for AID embodiment because cell phones are widely deployed and empowered to perform cryptographic computations. An overview of creating a secure channel in our system can be depicted with a simple example scenario; *a user found a public photo printer in a library and wants to print pictures stored in a digital camera. The user chooses a*

menu on her cell phone, and then brings the phone close to the photo printer and the digital camera one after the other. As a result, the two devices will share a paired secret, so that the user can send the pictures to the printer safely.

4.1 Identifiers

Based on the properties of PBC, two devices in our system will have mutually shared secrets derived from the IDs of the devices. There could be some consideration on how to generate and distribute IDs. Some identity-based encryption (IBE) applications use self-explanatory identifiers that can be uniquely inferred from some known properties of devices, such as the address or the network topology where the devices reside. Therefore, when a device wants to communicate, the former can easily acquire contact or the identifier of the latter if the former has one of either sets of information.

However, we do not use inferable identifiers since devices can have multiple IDs. The number of IDs for a device depends on how many channels associated with the device. IDs are generated and distributed by an AID, and there are two possible ways of ID generation: 1) The AID generates a set of IDs in advance for a given number of initial participants, and 2) generates an ID as the occasion demands. We use both methods. When a user generates a channel, the first is used, and when a new member is joining over the initial number of members, the second is used.

When a device wants to securely communicate with the other device, the former has to figure out how to make contact and what the ID is of the latter. On the other hand, the property of PBC builds a shared secret between pairs of IDs, even though the ID owners have not yet met. Authentication includes discovering the relationship between the contact and the ID of the peer, based on the shared secret. Contrary to the typical IBE applications, IDs are not inferable in our system. Therefore, devices acquire the information from an AID, otherwise they have to resolve it by themselves. These processes are discussed in Sect. 4.3.

4.2 Pairing based cryptography

The PBC bases on pairings that map a pair of elliptic curve points to an element of the multiplicative group of the finite field[1]. Bilinear pairings are special type of pairings defined as follows; G is an abelian group written in additive notation with identity element 0, and G_T is a cyclic group of order q written in multiplicative notation with identity element 1. We carefully select an elliptic curve $E(\mathbb{F}_q)$, and construct a Non-Interactive Key Distribution Scheme (NIKDS) as Boneh[7] and Sakai[24] proposed, by obtaining a map $\hat{e}$ that is derived from a Tate or Weil pairing on an elliptic curve, $\hat{e} : G \times G \to G_T$, which satisfies the following properties:

- Bilinearity: $\forall P, P', Q, Q' \in G$ we have $\hat{e}(P+P', Q) = \hat{e}(P, Q)\hat{e}(P', Q)$ and $\hat{e}(P, Q+Q') = \hat{e}(P, Q)\hat{e}(P, Q')$.
- Non-degeneracy: $\hat{e}(P, P) \neq 1$.
- Symmetry: $\forall P, Q \in G$, $\hat{e}(P, Q) = \hat{e}(Q, P)$.
- Computability: $\hat{e}$ can be efficiently computed.

[1] The descriptions of the pairings refer to articles in Galbraith and Pattersons' book[25] (Chapter IX and X, respectively), and an introduction of Menezes[20]. Details can be found in the references.

- Security: it is hard to compute the bilinear Diffie-Hellman problem and the decision-bilinear Diffie-Hellman problem.

The parameters $\langle G, G_T, \hat{e} \rangle$ are along with a cryptographic hash function, $\phi : \{0,1\}^* \rightarrow G$, that map arbitrary length binary strings onto elements of G.

When a channel c is created, an AID selects a channel secret $CS^c \in \mathbb{Z}_r{}^*$. The AID can get a public/private key pair of device i by computing $Pub_i{}^c = \phi(ID_i)$ and $Priv_i{}^c = [CS^c]Pub_i{}^c$, respectively. The AID should implement the following functions:

- PROC_INIT_CHN: when a user requests to create a secure channel c, the AID generates the following initial security parameters with a given number of initial participants n:
  1) CS^c, ϕ^c for the channel c
  2) $ID_i{}^c$ for the participating devices ($i = 1...n$), $ID_M{}^c$ for the AID, $ID_N{}^c$ for the channel network. $ID_M{}^c$ and $ID_N{}^c$ are used to derive the key pair of the AID and broadcast, respectively.
  3) private keys of devices, $Priv_i{}^c = [CS^c]\phi(ID_i{}^c)$. Note that, $\phi(ID_i{}^c)$ is the public key of i. The AID can optionally give the calculated public keys to devices to reduce the computational burden.
  4) a public/private key pair of the AID, ($Pub_M{}^c$, $Priv_M{}^c$).
  5) a public/private key pair of the channel network, ($Pub_N{}^c$, $Priv_N{}^c$).
- MSG_CHN_CNFRM: when the user brings the AID to device i, the AID asks if the channel (c, $Pub_M{}^c$) is already created on i.
- PROC_STORE_CHN: if the channel c has not been created yet on i, the AID sends $ID_i{}^c$ and other security parameters (cf. MSG_DVC_PRMTR). The AID may store the contact of the device (e.g., address) for an administrative purposes.
- MSG_DVC_PRMTR: the AID sends ($ID_i{}^c$, $Priv_i{}^c$, $\{ID_j{}^c\}$, ϕ^c, $ID_M{}^c$, $ID_N{}^c$, $Priv_N{}^c$) ($where\ i \neq j$) to i.

An embedded device needs to implement the following functions:

- PROC_CHN_CNFRM: receiving MSG_CHN_CNFRM, a device checks whether a channel c is created with the name of $Pub_M{}^c$, and returns yes or no confirmation.
- MSG_CHN_CNFRM_ACK: answers with the contact information.
- PROC_DVC_PRMTR: Receiving MSG_DVC_PRMTR, the device stores the security parameters.

After the AID distributes security parameters to devices, two devices will share a pairwise secret $K_{i,j}$ as $\hat{e}(Priv_i{}^c, Pub_j{}^c) = \hat{e}(Pub_i{}^c, Pub_j{}^c)^{CS^c} = \hat{e}(Priv_j{}^c, Pub_i{}^c)$ by the bilinearity and the symmetry.

4.3 Session key establishments

A device could be in the following status according to whether it knows the contact or the ID of its communication peer: *Stat1*: the device recognizes the other and knows how to initiate contact, but does not know the ID of the other yet, *Stat2*: the device knows the peer's ID, but does not know how to initiate contact, or *Stat3*: the device knows both of these pieces of information. The session key establishment can be processed differently depending on the states of the devices[2]:

[2] a) $H[x]$ is the hashed value of x, $\{x\}_K$ is an encrypted message x with a key K, and n_A is a nonce generated by A. b) session key expiration is not represented and addressed in handshake protocols.

- Handshake1: it is the session key establishment between a device A in *Stat3*, and the other device B in any other state. Since A knows the ID and the address of B, A can sends a session key establishment request to B using the shared secret between them. A session key is derived as the below handshake case 1 below shows. An intruder cannot impersonate A nor B without knowing one of their private keys.
- Handshake2: the session key establishment between a device A in *Stat2*, and the other device B in *Stat2* or *Stat1*. Although A knows B's ID, A has to resolve B's address. It is same as the Handshake1, except that the first message is broadcast to every device on the channel. Despite every device receiving the message, only B can acknowledge correctly.

Handshake1 (Handshake2):
1. $A \rightarrow B$ (ALL): c, $ID_A{}^c$, n_A, $H[c, \hat{e}(Priv_A{}^c, Pub_B{}^c), n_A, 0]$
2. $B \rightarrow A : n_B, H[c, \hat{e}(Priv_A{}^c, Pub_B{}^c), n_A+1, n_B, 1]$
(Key established as $H[c, \hat{e}(Priv_B{}^c, Pub_A{}^c), n_A+2, n_B+1]$)

- Handshake3: the session key establishment between two devices that are in *Stat1*. The devices need to exchange their ID and authenticate each other based on the shared information. Balfanz et al.[5] proposed a handshake protocol using pairing-based cryptography. We use a simplified version of the protocol. See the operation Mode3 below.

Handshake3:
1. $A \rightarrow B : c, ID_A{}^c, n_A$
2. $B \rightarrow A : ID_B{}^c, n_B, H[c, \hat{e}(Priv_B{}^c, Pub_A{}^c), n_A+1, n_B, 2]$
3. $A \rightarrow B : H[c, \hat{e}(Priv_A{}^c, Pub_B{}^c), n_A+1, n_B+1, 3]$
(Key established as $H[c, \hat{e}(Priv_B{}^c, Pub_A{}^c), n_A+2, n_B+2]$)

Devices can negotiate a session key with slightly fewer messages and steps in the case of Handshake1. Hence, in order to maximize the number of nodes in *Stat3*, we assume that an AID generates and distributes IDs in the following **accumulative way**:
1. Create a channel: a user creates a channel with k initial participants. The AID generates k number of IDs and private keysalong with other channel security parameters.
2. Distribute security parameters: when the AID touches a new device, an unassigned ID is associated with the contact (e.g., address) of the device. Therefore, on $n(\leq k)$th touch, the AID can pass $n-1$ associations to the new member. At this moment, everyone has k IDs, so that *Stat2* is always guaranteed with whomever they want to communicate. Also, nth one is in *Stat3* to $n-1$ old members.
3. Over the initial number: the number of members could grow over the expected k. Assume that l members joined and exceeded k, then the total number of members are $t = (k+l)$. When mth member joins ($k < m < t$), the member is in *Stat3* to $m-1$ old members by receiving the accumulated associations, whereas the old members are in *Stat1* to this since the member's ID is newly created. Also, the mth member is in *Stat1* to $t-m$ members who joined after.

Two other ways of ID distribution could be **Method1:** do not generate the initial set of IDs, but hand over the accumulated association to a new member, and

The negotiation of session key expiration can be done after the entities confirm mutual secret, according to security policy of each device.

Method2: do not generate the initial IDs nor pass the accumulated information. In the Method2, every device is in *Stat1* to everyone. In the Method1, every *m*th joined member is in *Stat1* to $(t-m)$ members who joined later (where t is the total number of members). If the proposed method is used in this paper, *m*th member is in *Stat1* to $(t-m)$ only when $m > k$, since k members are guaranteed to be in *Stat2* with each other. Considering the complete possible combinations of communication initialization, $t(t-1)$ connections might be initialized by devices in *Stat1* to the other with Method2, $\sum_{i=1}^{t} t-i = t(t-1)/2$ in *Stat1* with Method1, and $kl + \sum_{i=1}^{l-1} i = \{t(t-1) - k(k-1)\}/2$ (, where $l = t-k$) in *Stat1* with the accumulative way.

4.4 Broadcast in a channel

Although novel approaches of pairing-based group key agreement protocols have been proposed[7, 11, 10], they do not provide sufficient security under certain conditions and impose even more of a performance burden on senders than receivers[9]. The asymmetric overhead can be exploited by an inside attacker for denial of service attacks.

We just take a very simple approach of using the pair of channel network keys $Priv_N{}^c$ and $Pub_N{}^c$. A can send a message using $\hat{e}(Priv_A{}^c, Pub_N{}^c)$, then others decrypt the message using $\hat{e}(Pub_A{}^c, Priv_N{}^c)$. Since only channel participants have $Priv_N{}^c$, outsiders cannot send or receive a broadcast message correctly. In this way, the computational burdens of sender and receiver are not very different. Consequently, an inside attacker needs to pay as much power and computation costs as the victims.

5 Analysis

The security of Handshake1 and Handshake2 is based on the shared secret between A and B, which again depends on the security of PBC. In Handshake3, devices exchange the ID and other information to authenticate each other, and an insider intruder might be able to intervene in the communication. In this section, we try to check if a middleman can acquire a session key while two other devices execute the Handshake3 protocol. We model the protocol using Coloured Petri Nets (CP-Nets)[13] which has known to compactly model concurrent behaviors by allowing the net elements to have value, type, and supporting functional expressions.

The Fig.1(A) and (B) show the behaviors of A and B in Handshake3 protocol. The two entities communicate via three types of messages that draw from the first to third messages of the Handshake3. When A issues an initial message (MSG_JOIN in the graphs) to establish a session key using a channel number, own ID, and nonce, the message will be passed to B so that B can return a message (MSG_TEST_EC) based on their shared secret. A validates the digested message in the returned message, and confirm the message attaching a new digest(MSG_TEST_EC_ACK). Finally, they agree upon on a same session key. While the fourth message transmission of the protocol is omitted in the graphs to reduce the state space, we can confirm the agreement investigating tokens in the place, SharedPeer.

Additionally, we introduce an intruder to the model. The Fig.2(C) delineates the behavior of an intruder. As an insider, the intruder has the same security parameters

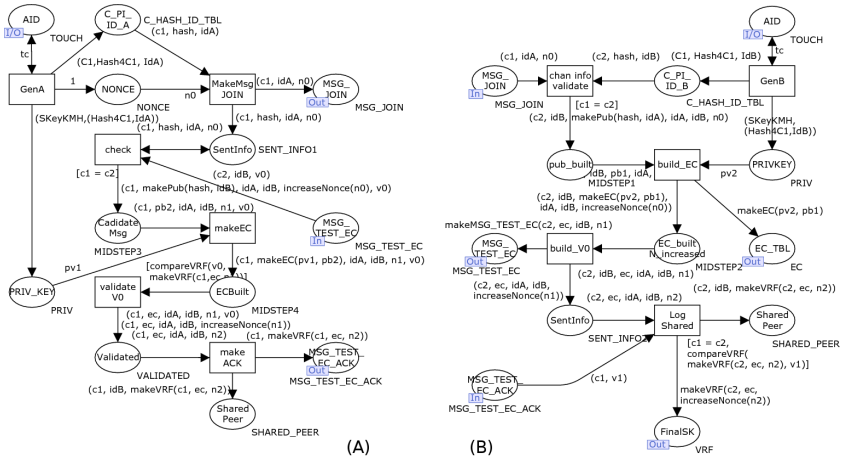

Fig. 1 The behavior of A (A) and B (B)

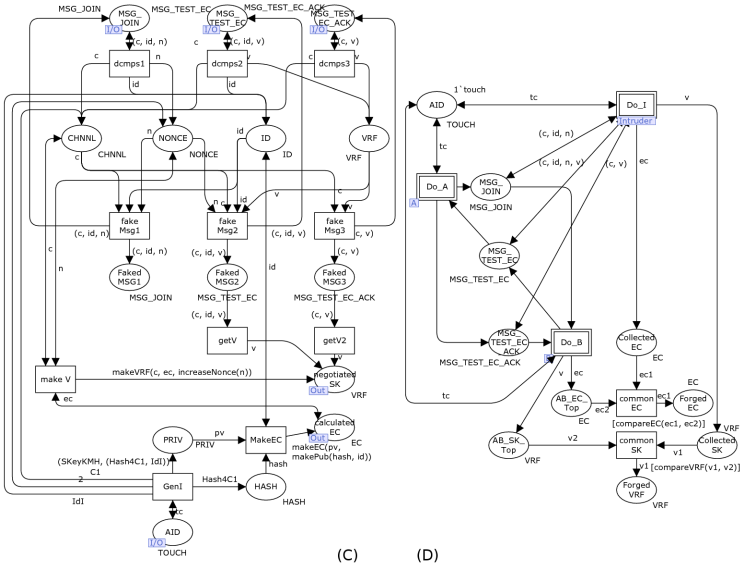

Fig. 2 The behavior of Intruder (C) and the top level diagram (D)

as A and B except the private keys, takes messages, decomposes the messages into parts, constructs new messages using the collections, and puts the synthesized messages into the communication. The Fig.2(D) shows a top level diagram where A, B, and Intruder interact. The intruder aims at agreed session keys between A and B. If the intruder succeeded a token is placed on the ForgedVRF. Also, the ForgedEC place stores shared secrets that the intruder have forged and collected.

CP-Nets provides an automatic analysis tool, CPNTools[1]. As a result of state space analysis and token game simulation in our model, we received the only one

dead transition commonSK and 0-bound with ForgedVRF (See Fig.2(D)), which means there is no session key exposed to the intruder. The intruder had tokens on 'ForgedEC', but only legal shared secrets between the intruder and B.

6 An application: Zigbee protocol

Zigbee[28] is a set of specification built upon IEEE 802.15.4 for wireless communications in a low-cost and low-power environment, which has similar target applications to this paper. The security service specification of Zigbee is based on Advanced Encryption Standard (AES). Communications among Zigbee devices are protected by link keys and network keys which are 128 bit keys for secure unicast and broadcast. The keys are obtained by pre-installation and key-transport. The link key is also established using a master key. The security between devices depends on how they initialize and install those keys. A special device that has a trust center role distributes keys and manages network and configuration.

There are some efforts to introduce public key cryptography to Zigbee security seeking for advantages in key management and additional functionalities such as signing and non-repudiation. Moreover, Nguyen and Rong[22] proposed using ID-based encryption for setting up the master key and the link key. In their system, a device provides its self-explanatory identifier to the trust center, then the trust center authenticates the device and gives the private key for the device.

Similarly, our system can be applied to Zigbee by establishing the three types of keys as follows:

1. The master key corresponds to the shared secret among devices. An AID can act as an offline trust center or domain controller, but the role can be delegated to an online entity by passing over security parameters if it is needed.
2. The link keys can be established via handshakes.
3. The network key corresponds to the broadcast key which the AID provides.

The expected advantages of using AID are support for multiple channels on top of Zigbee protocol, broadcast, and user-friendly interface, which are not included in the previous IBE scheme.

7 Discussions

The PBC has several advantages: it does not require an online trusted authority and imposes less overhead than the conventional public key cryptography. However, application of PBC is restricted because initial security parameters need to be generated by a trusted public key generator (PKG) and to be securely transplanted to devices. WSN applications overcome the restriction by having a base station perform the role of PKG and deliver the security parameters to sensor nodes before the nodes are deployed. Similarly in our system, the AID acts as the trusted entity and deliver security parameters using the out-of-band channel. Moreover, it enforces user ownership and support key escrow. Since security parameters of a channel are generated and distributed by an AID, only the user who has the AID can add or remove a channel member. After a channel is created, the channel participants also can authenticate the owner online relying on the public/private key pair of the AID.

We addressed ways to generate and distribute identifiers to save communication overhead. With an expected number of participating members k, we may expect

reduce communication overhead by eliminating the $O(k^2)$ possible handshake overhead. The benefit increases as k moves closer to the total number of actual participants t (See Sect 4.3).

When users connect their private devices together, they may need to decide which AID they will use. It depends on participants' security policy that who will create and own a secure channel. For example, if a user wants to use her own electronic reminder when she is in a hospital, she may need to ask a nurse to connect the reminder to the hospital network. It is different from the example scenario of Sect. 4. Security policy negotiation needs to be investigated further for connecting devices that have different security policy.

Since the AID functionalities are expected to be embodied with a handheld and user-friendly device like a cellphone or portable game player, other fancy techniques could be combined as well as a simple PIN-based protection. For example, such biometrics as gesture, voice, fingerprint, and finger vein recognition could be merged to attract users and expand usability. A user can cast "Abracadabra" and draw a spell mark in the air before creating a secure channel. We cannot guarantee that these technologies will strengthen security, but we presume a successful design.

The security of our system depends not only on the property of PBC, but also on the location limited channel. Communication media should be carefully chosen to avoid eavesdropping threats[8]. Key revocation and broadcast issues of PBC need to be considered further as well. In our scheme, an inside attacker can send a broadcast message impersonating another (e.g., use ${Pub_A}^c$ and ${Priv_N}^c$ to pose as A). We assume that broadcast messages are used to deliver non-critical notifications. Also, we assume the key revocation and membership changes are done by a user manually, expecting network size is manageable by the user.

8 Conclusions

We proposed a way of creating secure communication channels over ad-hoc network devices using an easy-to-use intermediary. We employed several concepts and technologies to mobile networks and wireless sensor networks, such as pairing-based cryptography, the notion of location limited channel, and the very short range wireless communication media.

We described and specified the steps of security bootstrapping. We also demonstrated the security of the proposed protocols using a model checking approach equipped with an automatic analysis tool.

Owing to the property of pairng-based cryptography, users can acquire security with low overhead and enforce their ownership over secure communication channels that are dynamically created over networked devices. Users can create multiple channels for their own purposes. Channel owners are identified using security parameters, and they can reveal any secret on their private channels as needed. Since the security parameters are generated and managed by a handheld device, users can create the security channels on-the-fly in an ad-hoc environment.

Our approach can be generally applied to any network where dynamic secure channel creation and ownership representation are required, such as home networks, medical sensor networks, and so on. As an example, we showed how our method could be applied to the Zigbee security service.

As our further study, we are going to implement the scheme using a cell phone and a high speed IrDA[2]. Key revocation and broadcast will be reinforced later.

References

1. Cpn tools. URL http://wiki.daimi.au.dk/cpntools
2. Giga-ir. URL http://techon.nikkeibp.co.jp/english/NEWS_EN/20080725/155461/
3. Transfer jet. URL http://www.transferjet.org/en/
4. Balfanz, D., et al.: Network-in-a-box: how to set up a secure wireless network in under a minute. In: Proc. of the 13th conference on USENIX Security Symposium, pp. 15–15 (2004)
5. Balfanz, D., et al.: Secret handshakes from pairing-based key agreements. In: IEEE Symposium on Security and Privacy, pp. 180–196 (2003)
6. Balfanz, D., Smetters, D.K., Stewart, P., Wong, H.C.: Talking to strangers: Authentication in ad-hoc wireless networks. In: Proc. 2002 Network and Distributed Systems Security Symposium (NDSS), pp. 23–35 (2002)
7. Boneh, D., Franklin, M.K.: Identity-based encryption from the weil pairing. In: Proc. of the 21st Annual International Cryptology Conference on Advances in Cryptology (CRYPTO), pp. 213–229. Springer-Verlag, London, UK (2001)
8. Cheung, H.: How to: Building a bluesniper rifle. Tom's guide (2005). URL http://www.tomsguide.com/us/how-to-bluesniper-pt1,review-408.html
9. Chien, H.Y.: Comments on an efficient id-based broadcast encryption scheme. IEEE Transactions on Broadcasting **53**(4), 809–810 (2007)
10. Choi, K.Y., Hwang, J.Y., Lee, D.H.: Id-based authenticated group key agreement secure against insider attacks. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences **E91-A**, 1828–1830 (2008)
11. Du, X., Wang, Y., Ge, J., Wang, Y.: An id-based broadcast encryption scheme for key distribution. IEEE Transactions on Broadcasting **51**, 264–266 (2005)
12. Goodrich, M.T., et al.: Loud and clear: Human-verifiable authentication based on audio. In: Proc. ICDCS 2006: 26th Conf. on Distributed Computing Systems, p. 10. IEEE (2006)
13. Jensen, K.: Coloured Petri nets: basic concepts, analysis methods and practical use, vol. 1, 2. Springer-Verlag, London, UK (1995)
14. Khalili, A., Katz, J., Arbaugh, W.A.: Toward secure key distribution in truly ad-hoc networks. In: Proc. of Symposium on Applications and the Internet Workshops, pp. 342–346 (2003)
15. Kohl, J.T., Neuman, B.C., Ts'o, T.Y.: The evolution of the Kerberos authentication service. IEEE Computer Society Press (1994)
16. May, M.J., Shin, W., Gunter, C.A., Lee, I.: Securing the drop-box architecture for assisted living. In: Formal Methods in Software Engineering (FMSE '06). ACM (2006)
17. Mayrhofer, R., Gellersen, H.: Shake well before use: Authentication based on accelerometer data. In: Proc. of the 5th International Conference on Pervasive Computing (Pervasive 2007), pp. 144–161 (2007)
18. Mayrhofer, R., Welch, M.: A human-verifiable authentication protocol using visible laser light. In: Proc. of the The Second International Conference on Availability, Reliability and Security, pp. 1143–1148 (2007)
19. Mccune, J.M., Perrig, A., Reiter, M.K.: Seeing-is-believing: Using camera phones for human-verifiable authentication. In: In IEEE Symposium on Security and Privacy, pp. 110–124 (2005)
20. Menezes, A.: An introduction to pairing-based cryptography. notes from lectures given in (2005). URL http://www.math.uwaterloo.ca/ ajmeneze/publications/pairings.pdf
21. Needham, R.M., Schroeder, M.D.: Using encryption for authentication in large networks of computers. Communnication of the ACM **21**(12), 993–999 (1978)
22. Nguyen, S.T., Rong, C.: Zigbee security using identity-based cryptography. In: Proc. of the 4th International Conference Autonomic and Trusted Computing (ATC 2007), *LNCS*, vol. 4610, pp. 3–12. Springer (2007)
23. Otway, D., Rees, O.: Efficient and timely mutual authentication. SIGOPS Oper. Syst. Rev. **21**(1), 8–10 (1987)
24. Sakai, R., Ohgishi, K., Kasahara, M.: Cryptosystems based on pairing. In: Proc. of Symposium on Cryptography and Information Security (2000)
25. Smart, N.P., et al.: Advances in Elliptic Curve Cryptography. No. 317 in London Mathematical Society Lecture Note Series. Cambridge University Press (2005)
26. Stajano, F., Anderson, R.: The resurrecting duckling: Security issues for ad-hoc wireless networks. In: 7th International Workshop on Security Protocols (LNCS 1976), pp. 172–194. Springer-Verlag (1999)
27. Szczechowiak, P., et al.: Testing the limits of elliptic curve cryptography in sensor networks. In: European conference on Wireless Sensor Networks (EWSN'08), pp. 305–320 (2008)
28. ZigBee Alliance: Zigbee specification (2008). ZigBee Document 053474r17