Journal of Information Assurance and Security.
ISSN 1554-1010 Volume 7 (2012) pp. 259-274
© MIR Labs, www.mirlabs.net/jias/index.html

Key Initialization and Validation of Security Protocol Assumptions in a Home Health Care System

Kalvinder Singh[1], **Elankayer Sithirasenan**[2] **and Vallipuram Muthukkumarasamy**[3]

[1]Australia Development Lab, IBM, and Griffith University,
St Leonards, NSW 2065, Australia
kalsingh@au.ibm.com

[2]School of Information and Communication Technology, Griffith University,
Gold Coast Campus, Queensland 4222, Australia
e.sithirasenan@griffith.edu.au

[3]School of Information and Communication Technology, Griffith University,
Gold Coast Campus, Queensland 4222, Australia
v.muthu@griffith.edu.au

Abstract: **A home health care system can be used to monitor the elderly people or patients with chronic diseases. Information assurance, privacy, reliability and other requirements of home health care systems are complex. We show how keys can be securely established between the different components within the home health care system. We demonstrate that Genetic Design Methodology (GDM) can efficiently model the requirements of the complex home health care system and the key establishment protocols. We show that when new requirements are added to the system, the proposed model can successfully track if the existing key establishment protocol assumptions are still valid in the modified system. An implementation of the protocols is executed on mica2 motes and examined in detail. The time elapsed, complexity of the code and memory requirements are analysed. We show that a key establishment protocol based on RSA has advantages over a key establishment protocol based on ECC for this application.**

Keywords: security, sensors, health, networks, protocols

I. Introduction

The aging population and the increase of chronic diseases have placed an immense financial burden on health services in developed countries. Body sensors can be used to help reduce their costs significantly. Sensors can be used to remotely monitor elderly or patients suffering from chronic diseases and allow them to have relatively independent lives. Proposed healthcare systems contain many different components and hence are inherently complex [24].The complexity hinders security proofs and detailed analysis, so much so that theoretically proving that a protocol is secure within an entire system is rarely performed. Instead researchers only try to show that a sub–system is secure, for instance, communication between two body sensor nodes. Another problem is that complex systems rarely stay static, with new devices and algorithms a system may have different functionality from one day to the next. A proof that a protocol is secure for one system does not guarantee that it is secure on the other system.

Figure 1 demonstrates the way we modelled a typical home health care system proposed by [25, 37]. For our data flow analysis the body sensors, home sensors and mobile phone are used to gather information from the patient. The sensed data are then passed onto the home health controller. The home health controller analyser module will then analyse the data and send the analysis results to the decision module and hospital. The decision may be performed either at the home health controller decision module or at the hospital, depending on the results of the analysis. Once the decision is made it needs to be acted upon, this feedback may be a message sent to the patient that they should come in for an extra check–up or the sensors need to start measuring physiological data at higher sampling rates or that nothing needs to change.

A patient at home can have a number of body sensors that can communicate with home sensors, the health controller and a mobile phone. Home sensors, such as cameras, may only start recording if the body sensors detect that there may be a medical emergency, such as the patient lying horizontal in the kitchen. Surveillance software, such as S3 [19], can be used to detect if the patient is cleaning the kitchen or getting something from the ground or there is actually an emergency. If the software does detect an emergency, the hospital staff are notified, they examine the information and decide on the best course of action. The mobile phone is used to give feedback to the patient about the condition of their body, as well as the status of the sensors. The mobile phone can notify the patient of any detected emergency, allowing the patient to report back a false alarm if one has occurred. The mobile phone can be replaced with a PDA or any other hand–held communication device.

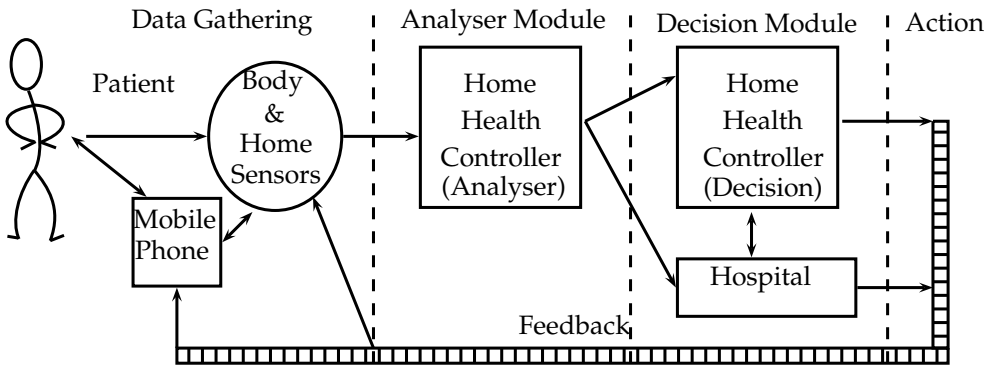

Figure. 1: Home Health Care System Data Flow Diagram

Health information collected from sensors needs to be secured and in some countries (for example the USA) security is mandated.Securing a home health care system becomes more challenging mainly because of the different requirements placed on various components in the system. For instance, the sensors have severe resource constraints than the constraints found in mobile phones, cameras or desktop computers. With differences in computing power, as well as in communication costs, a range of security protocols may be required to be deployed in the entire system. For instance, an efficient key establishment mechanism specifically suited for body sensors was proposed using physiological data [39]. However, the home health care system may send physiological data to medical staff or to an analytic engine [17]. The physiological data may also be sent to an actuator to release medicine into the body [17]. These may jeopardize the ability to use the physiological data in the key establishment protocol.

When the same physiological data is used for multiple purposes and/or the environment is heterogeneous and complex, it becomes important from a security or information assurance point of view to have a formal methodology to validate the system. A formal methodology is also important to insure that the information sent to medical staff and actuators to dispense medicine is accurate (secure), and the correct actions are taken. The formal methodology has a requirement that it can model both the security and privacy aspects as well as the assumptions and the application correctness.

In Section II we give a background wireless sensor security and formal verification mechanisms for security protocols. In Section III we will describe how Genetic Design Methodology (GDM) is currently used to model complex systems. Section IV supplies a description of a home health care system, and some of the components that belong in it. In Section V we will describe key establishment protocols for a complex sensor system. We will discuss the security of the key establishment protocol and how it is dependent on some assumptions about the environment. In Section VI we will show that GDM can be used as a formal analysis tool to verify both the system and in particular the assumptions made by the security protocols are correct. In Section VII we will describe the implementation of the key establishment protocol and some performance analysis. Section VIII concludes the paper.

II. Background

A Wireless Sensor Network (WSN) can consist of many different computing devices. Some have more computation power and/or memory than others. An example of a WSN is a Body Sensor Network (BSN). A BSN is a network of wearable heterogeneous sensors [1]. The sensors may spread over the entire body, and monitor and communicate a wide range of health related data. BSNs are used to monitor a patient's physical and biochemical parameters continuously in almost any environment and locations that the patient needs to go. BSNs can also be used by athletes to measure their performance characteristics. Another use for BSNs is user input into video games [1].

A challenge for BSNs is finding and deploying secure methods that allow the user to setup the BSN. One suggested solution is to have a special device that emits low powered messages in situations where the scope is a single person [17]. However, an intruder with sensitive enough equipment will be able to capture these messages. Initial work on cryptographically strong physiological data to initialize the network shows encouraging results [3, 44]. However, this has limitations since there is only a small number of cryptographically strong physiological data that an implementer could choose from.

A. Wireless Sensor Security

Security in sensor environments differ in many ways from that of other systems. Sensor nodes have little computational power, thus even efficient cryptographic ciphers must be used with care. Security protocols should use a minimal amount of RAM. Communication is extremely expensive; any increase in message size caused by security mechanisms comes at a significant cost. Energy is an important resource, as each additional instruction or bit transmitted means the sensor node is a step closer to becoming non–functional. Nearly every aspect of sensor networks is designed with extreme power conservation.

There are many aspects to WSN security [13]; ranging from data fusion security, location aware security, to the lower level security primitives such as cryptography, authentication and secure key establishment protocols. We shall not cover all aspects of WSN security in this paper, instead we will focus on some of the lower level primitives: authentication and key establishment protocols. However, concepts such as data

fusion security and location aware security also rely upon the lower level security primitives [27].
Several cryptography libraries using symmetric keys have been proposed [35, 23]. Much of the work on sensor protocols has used a symmetric key cryptography library. Recent work has shown that even asymmetric keys may be used in WSNs [46, 31]. Singh et al. [39] has proposed an efficient key establishment protocol using elliptic curves. However, they still consume considerably more resources than the symmetric counterparts. The limitations when verifying symmetric key authentication and key establishment protocols for WSNs are discussed in the next section.
Key establishment protocols are used to set up shared secrets between sensor nodes, especially between neighbouring nodes. When using symmetric keys, we can classify the key establishment protocols in WSNs into three main categories: Pair–wise schemes; Random key predistribution schemes; Key Distribution Center (KDC). The Pair–wise schemes and Random key predistribution schemes are designed for open environments, where there are many individual sensors [27]. The main difficulty with the above schemes is updating the keys between the nodes. Another drawback is that, when using the random key predistribution schemes, the shared keys cannot be used for entity authentication, since the same keys can be shared by more than a single pair of nodes [18]. The KDC mechanisms by themselves are not suitable for large scale WSN environments, although combinations of a KDC mechanism and the previously mentioned schemes have created hybrid protocols [11]. Some of the limitations in a KDC mechanism are:

- The KDC scheme relies upon other schemes to create the trusted intermediary.
- The key sizes in sensor nodes are not large enough, so over a period the key between the sensor and the trusted intermediary may become compromised. That is, if the KDC protocol messages were captured and saved by an adversary, then the adversary may calculate the new keys.
- Some sensor networks may not employ an encryption algorithm at all, however KDC protocols require an encryption algorithm to encrypt the new key.

The use of a password has been proposed as a way to initiate key establishment in a WSN [37]. However, the use of a PIN code or a password is not a pragmatic approach to BSNs since many of the sensors do not have a user-interface. Sensors also may be placed in hard–to–reach places, with some of the sensors implanted into the body. To complicate matters, the sensors may even harvest energy directly from the body [22], thus allowing the sensors to exist for very long periods of time. Updating keys with appropriate mechanisms is, therefore, an important requirement.
This paper uses the generic name *Secure Environmental Value* (SEV) referring to sensed data that can only be obtained by sensors in an environment. The SEV is usually hard to obtain through other means. Examples of an environment where SEVs may be found include:

- Human body, where it is difficult for an adversary to attach a device on the body without the knowledge of the person.
- A secured location, for instance a military base or unmanned vehicle, or a secure home environment.
- Hard to reach places, for instance a satellite in orbit.

The example environment used in this paper is the human body, where BSNs have been developed to measure the physiological values found in individuals [1]. Health sensors can use Inter–Pulse–Interval (IPI) or Heart Rate Variance (HRV) [2] as good sources for cryptographically random numbers and the physiological values can be used as a one–time pad. Recently, the EKE password protocol [5] was used in BSNs to increase the number of physiological values that can be used [39]. The physiological data replaced the password, in the EKE password protocol. A major limitation to the adoption of the above methodology is the lack of formal verification that the protocol, which makes a number of assumptions, is suitable in a complex sensor system.

B. Formal Verification

Formal methods to verify that a protocol is correct is an important area in the research community. Verifying a protocol provides the validity for the protocol and hence it is a significant step in analysing the protocol. The complexity of security protocols makes their verification a difficult task. Informal qualitative arguments by themselves are not reliable or acceptable, thus a formal analysis to verify the claim made by a protocol is needed.
Computer assisted formal methods for verifying security protocols can be divided into two major categories:

- Model Checking: considers a finite number of possible protocol behaviors and allows checking if that satisfy a set of *correctness conditions*. This method works well for finding attacks on a protocol, rather than proving their correctness [12, 28, 33].
- Theorem Proving: considers all possible protocol behaviors, and checks that they satisfy a set of *correctness conditions*. This method works well for proving protocol correctness, rather than finding attacks on protocols [32, 34, 41].

Both model checking and theorem proving methods require computer assistance to aid with the analysis. However, methods based on theorem proving are less automated than those based on model checking.
A useful feature of model checking methods is that they can prove an attack when a protocol does not satisfy a correctness condition. The failure to find an attack implies that the protocol is correct. However, model checkers do not provide a symbolic proof that can explain why a protocol is correct and thus are uninformative when checking a valid protocol. Another important limitation of model checking methods is that they only guarantee correctness of a scaled down version of the protocol.
Theorem proving mechanisms have their own strengths and limitations. One of the strengths of theorem proving methods is that they can provide a symbolic proof when a protocol is found to be valid. Their main limitation is that they generally

require more expert human guidance than methods based on model checking.
Another mechanism to verify that a protocol is secure is to use a mathematical proof [9]. Problems with using mathematical proofs include:

- With each small change in the protocol a new proof needs to be constructed.
- Security proofs are complex and involve long mathematical reasoning and are difficult to understand to the average practitioner.
- There are relatively few protocols with mathematical security proofs.
- As systems become more complex, constructing mathematical proofs becomes more challenging.

A combination of informal verification, machine analysis (either using model checking, or theorem proving), and mathematical proofs is important to gain assurance on the security of the protocol.
The following section describes how Gentic Design Methodology (GDM) is used when examing complex systems and the correctness of security protocols.

III. Use of GDM in Complex Systems

The common techniques used to verify that a protocol is correct does not easily scale to a complex system. Using the above techniques to validate a system with hundreds of nodes, and many different protocols, is almost infeasible. To further complicate matters, there are inherent restrictions such as, the formal verification will need to be repeated for every minor change in the system.
When proving that a protocol is secure, the proof relies on a number of assumptions made about the environment where the protocol is run. This may be assumptions such as secure time synchronizations between the parties, or that the physical security of the communication medium. Showing that a protocol is secure in a complex system may be considered as a two step process. The first step is proving that the protocol is secure based upon some assumptions. The second step is to show that the assumptions are valid and consistent in a complex system. The GDM has recently been used as a tool to prove and/or validate the overall correctness of complex systems.
Sitherasenan et al. [40] have used GDM to check the correctness of the 802.11i wireless security protocol. The requirements of the protocol was placed into a number of Requirement Behaviour Trees. The requirements were then verified by integrating them into a single Integrated Behaviour Tree. Thereafter, the Behaviour Tree model was translated into SAL formal notations for theorem proving. This mechanism shows that both model checking and theorem proving approaches can be performed using the same analysis tool. The checks performed was mainly focused on the protocol correctness and not on the assumptions made by the protocol. We will show that the GDM analytical tool can effectively perform model checking on the correctness of protocol assumptions in a complex system.
When using GDM, systems are designed out of the requirements as opposed to methods that produces designs to meet the requirements. A major advantage of GDM is that it produces graphical models that are derived and integrated from the original requirements. The models can easily be used to verify that security protocols correctly work in a complex system.
An example of a complex system is the home health care system. For instance, in a home health care system it can become difficult to track how sensed data is used at different stages in the system. When the sensed data is also used in key establishment protocols, tracking the various uses of sensed data becomes even more important. For example, some key establishment protocols require that the sensed data never to be sent in the clear or to an untrusted third party, whereas other protocols do not need such restrictions. The complex system and the protocols can be defined in requirement behavior trees using GDM.
Each requirement can be represented as a behavior tree, this representation is specifically called a Requirement Behaviour Tree (RBT). An important part of the genetic design methodology is constructing the behavior trees. Dromey [15, 47] defined Behaviour Trees as: *a formal, tree–like graphical form that represents behavior of individual or networks of entities which realize or change states, make decisions, respond–to/cause events, and interact by exchanging information and/or passing control.*
Behaviour trees provide a direct and clearly traceable relationship between what is expressed in the natural language representation and its formal specification. Conventional software engineering method applies the underlying design strategy of constructing a design that will satisfy its set of functional requirements. Whereas, a clear advantage of the behavior tree notation is that it allows us to construct a design out of its set of functional requirements, by integrating the behavior trees for individual functional requirements (RBTs), one–at–a–time, into an evolving design behavior tree (DBT).
The RBTs are integrated based on the precondition of the tree that must be satisfied in order for the behavior encapsulated in a functional requirement to be accessible or applicable or executable. If there is no matching post–condition embodied in the evolving DBT then a defect is identified and needs to be rectified. In which case, either the requirement is invalid, or there is a missing requirement. Integrating RBTs is an important feature when showing that security requirements in the system are valid or if there is a missing security requirement.
Once the RBTs are integrated, and any missing or invalid requirements are dealt with, we can then generate other models from the evolved DBT. SAL code can be generated, allowing the creation of theorems that also checks the security requirements of our system.
Behavior trees can in turn be used to generate SAL code [40]. A model checker can then be used to verify the SAL code and thus verify the protocol in the sensor environment. The main steps in the GDM are: translation of requirements to behavior trees; integration of behavior trees; architecture transformation; component behavior projection; and component design. When modelling the entire system, genetic design has signif-

icant advantages over Unified Modeling Language (UML), state charts or other methods [14]. The advantages include:

- Allows designers to focus on the complexity and design of individual requirements while not having to simultaneously worry about the details in other requirements. The requirements can be dealt with one at a time (both for translation and integration).
- The component architecture and the component behavior designs of the individual components are emergent properties of the design behavior tree.
- The methodology concentrates on discovery of behavior gaps, which in turn discovers requirement and security gaps. The focus of direct translation of requirements to design, makes it easier to see and find gaps either manually or using automated tools.
- Presents an automated method of mapping changes in requirements to changes in design.

A. Notation

The behavior tree described in this paper will use the standard notation. There is no standard notation to describe security protocols, however, we will use a commonly used format [7] as shown in Table 1.

Table 1: Notations used in Security Protocols

Notation	Description
A, B	The two nodes who wish to share a new session key
S	A trusted server
N_A, N_B	Random numbers generated by nodes A and B respectively
V	A SEV read from the environment by a sensor
V_i	The i'th SEV read from the environment by a sensor
$[[M]]_K$	Encryption of message M with key K to provide confidentiality
$[M]_K$	One–way transformation of message M with key K to provide integrity
K_{AB}, K'_{AB}	The long–term key initially shared by A and B and the new session key respectively
K_{AS}, K_{BS}	Long–term keys initially shared by A and S, and B and S respectively
X, Y	The concatenation of data strings X and Y
$A \xrightarrow{m} B$	A sends a message m to B
$\oplus$	Exclusive–or function

IV. Analysis of the Home Health Care System

The home health care system, described in this paper, is relatively complex. When security mechanisms are incorporated, the importance of modelling technique becomes more apparent. This section examines in detail a complex security protocol that secures a hand–held device, such as a PDA or mobile phone with the home health care system.
Table 2 depicts an overview of four of the major components in home health care system : mobile phone; home sensors; home health controller; body sensors. The table shows the different types of communication protocols and technology each of the components may have, as well as the physical security of the component. The connectivity column describes when or how often the component is able to obtain patient information and the replacements column indicates how often the patient will replace that particular component. The operations of each component is described below.

A. Body Sensors

Body sensors measure the vital physiological signs of the patient. Body sensors use a low powered communication medium such as 802.15.4. The implanted sensors' physical security is very high and they should be rarely replaced. Sensors that are strapped onto the patient are less secure and can be replaced more frequently. For the remainder of this paper we will assume that the patient has at least one implanted sensor. The implanted sensor can be used to safely store security keys within the home health care system.
The sensors will need to send the information securely. This produces an inherit requirement of establishing session keys for the sensors. After sensors gather the data they will need to send that data to a central computer via a mobile phone or the nearest component within the system, as described in Figure 1.

B. Mobile Phone

The mobile phone is a mobile gateway between the body sensors and the rest of the network. It can be incorporated to use many different communication technologies. It can display the status of all sensors and the vital signs for the patient. Facilitating the patient to feel in control of the entire system. Its mobility causes it to be easily lost or misplaced. Hence the physical security of this device is low. Therefore, an important limitation placed on this device is that we must not store any cryptographic session keys in stateful memory. Therefore, the mobile phone will need to be able to quickly establish secure session keys with the body and home sensors when it is turned on.
For privacy reasons when the patient goes outside their home, the phone should be able to send data back to the home health controller. The home health controller can decide, based on a set of pre–determined rules, the salient information that needs to be sent to the hospital staff.
The mobile phone also has a sensor attached to it. Hence if the patient picks up the phone, the phone will be able to read the physiological data from the body. This is a convenient method to have redundant sensors on the body to accommodate the case if one of the sensors becomes faulty. By having a sensor on the phone, the system can also quickly detect any faulty sensors.

C. Home Sensors

The home sensors can augment the body sensors and supply more information, such as temperature and movement of patients. The home sensors may not always be switched on, and may only be turned on if the home health system determines that there may be an emergency and requires more information. This saves power and also enhances the privacy of the patient. Especially if some of the home sensors are cameras. The home sensors need to establish session keys with the mobile phone and body sensors. The mobile phone can be used by the patient to know the status of the other sensors in the

Table 2: Components in the Health Care System

Component	Communication	Physical Security	Connectivity	Replacements
Body Sensors Implanted	802.15.4	Very High	Always	Very seldom
Body Sensors Strapped	802.15.4	Medium	Always	Frequently
Mobile Phone	802.11, 802.15.4 plus many others	Low	When turned on & close to patient	Frequently
Home Sensors	802.11 and/or 802.15.4	High	When patient is at home	Frequently
Home Health Controller	802.11, 802.15.4 plus many others	High	When turned on	Seldom

system. The body sensors should directly be able to communicate with other components within the system, allowing the patient to freely move within their home without needing to always carry their mobile phone.
Since the home sensors are located inside a building they are physically secure and have a lower likely–hood of being stolen. Home sensors will only be replaced if they are found to be faulty.
In this paper we have assumed that the security between the home sensors and the health controller is achieved by other known protocols, such as defined in the IEEE standards for 802.11.This paper focuses mainly on securely establishing keys with body sensors.

D. Health Controller

The health controller coordinates the entire home health system. It contains heuristics to determine if the patient is in any danger. If the health controller determines that the patient is in some danger it may then power on some of the home sensors to gather more data. It may also notify the patient or the hospital depending on the type of danger. The home health controller is a key component for the privacy of the patient. The patient is more likely to have cameras installed in their home with the assurance that they will only be turned on in case of an emergency.
The health controller has an interface allowing the patient to enter extra data, such as, going for a jog, cleaning, etc. This enables the health controller to obtain an informed judgement on whether the patient should go to the hospital for an early check-up, or if the patient is doing some exercise.
The health controller is the central hub within the home, hence it will need to have established session keys between all of the other components. The health controller also has a built–in body sensor similar to the mobile phone. This allows the patient to place their hand on the sensor to confirm that the readings from the body sensors are accurate or whether any of the sensors is faulty and needs to be replaced.

E. Other Components

Other entities within the system include the hospital and the patient, as shown in Figure 1. The patient originates the physiological data, which the body sensors can capture. The patient can also input information to the system, such as exercising or eating, via a mobile phone or PDA.
The hospital will be supplied salient information about the patient. However, the patient's privacy will not be compromised by supplying non-vital data (such as, camera footage), unless there is a justified medical reason for it. The hospital will be notified of any irregularities and hence the hospital can inform the patient to have an additional appointment with a doctor, etc.

V. Proposed Mechanisms for Key Establishment

There are several scenarios where the body sensors and the mobile phone do not have any keys established with the remainder of the home health care system. This may be because the mobile phone has recently been turned on, and no keys are stored in the mobile phone's permanent memory. Hence it will need to establish keys with the rest of the system a fresh. The body sensors may have expired keys, if the patient was away from their home for an extended period of time. For operational efficiency reasons, keys in body sensors are small, hence the keys have a small life–time and need to be updated more frequently.
There are four major steps when first establishing keys, the steps are as follows:

- Initial Setup :- This step describes how the system is initially setup.
- Home Health Controller and Body Sensor :- This step describes how the Home Health Controller establishes a key with the body sensor.
- Body Sensors and Home Sensors :- This step describes how the body sensors establish keys with the home sensors.
- Mobile Phone :- This step describes how the mobile phone establishes keys with the other components.

Each step is described in detail in the following sections.

A. Initial Setup

In this paper, we assume that the session keys between the Home Health Controller and the Home Sensors have already been established [20]. The assumption is that existing industry standards, such as 802.11, was used to set the system up. The standard 802.11 is well understood and found in many higher resource environments, such as, mobile phones, cameras, laptops, and personal computers.

B. Home Health Controller and Body Sensor

The next step is to establish a session key between the Home Health Controller and a body sensor. An implanted body sensor is desirable since they have the highest level of physical security. That body sensor can be used in the future to hold session keys for most of the other components in the healthcare system.

Ease of use is an important requirement for our system. A complex system where a user needs to install security certificates is infeasible. Setting passwords on tens to hundreds of devices is technically difficult due to lack of a user interface such as keyboards or monitors on many of the sensors. Forcing a patient to remember a new password or PIN will be a deterrent on the take up of a home health care system. It is envisioned that the elderly could benefit from a home health care system. However, they would have the greatest difficulty in managing and handling complex technology. As technology improves new devices will be added to an already complex system. It is infeasible to have users learn new technology when changing components in an ever changing environment. The security setup should be simple so any new devices can be easily added.

An example of a simple method to establish a key between two devices is for the patient to place their palm on a sensor (such as an ECG reader) built into or attached to the Home Health Controller. If an ECG reader is used then authentication between the Home Health Controller and the patient can also be performed [43, 10].

Singh et al. [39] has supplied an overview of key establishment within a BSN and have shown different methods to establish keys between sensors and components that obtain the same SEV :

- Use a SEV as a one–time pad.
- Use a RSA–based Diffie–Hellman password protocol, where the password is the SEV.
- Use a ECC–based Diffie–Hellman password protocol, where the password is the SEV.

The ECC–based password protocols have major efficiency problems, which are shown in our implementation, explained in Section VII. Hence in this section we concentrate on the first two approaches.

The simplest method is to use the SEV as a one–time pad. Venkatasubramanian et al. [45] used a single message to send a new key to the neighbouring sensor node, by using the SEV as a one–time pad, as shown in *Protocol 1*. The home health controller, C_h, initiates the protocol by sending a message to the implanted sensor, S_b. The new session key is sent encrypted by the SEV, $K_{C_hS_b} \oplus V$; where $K_{C_hS_b}$ is the newly created session key and V is the SEV read by each of the sensors. Venkatasubramanian et al. noted that finding additional cryptographically sound physiological values is still an open research problem. Another problem is that all the protocols developed with physiological values require all the sensor nodes to be able to measure the same phenomena. Only cryptographically strong physiological values, such as IPI (Inter–Pulse Interval) and HRV (Heart Rate Variance), can be used. Also, modern wireless technology (ultra wideband – UWB, radar [42]) may be used to remotely capture the heart rate and could cause security risks when using IPI and HRV to only secure the communication. Other cryptographically weaker physiological values, such as blood pressure, and iron count, are less susceptible to those remote attacks.

Protocol 1 Venkatasubramanian BSN protocol

$C_h \rightarrow S_b : N_A, [N_A]_{K_{C_hS_b}}, K_{C_hS_b} \oplus V$

The new key $K_{C_hS_b}$ is encrypted with the physiological value V, which is only known to sensors on a particular person. Sensor node B validates that $K_{C_hS_b}$ is correct by verifying the MAC of N_A.

One of the major limitations of this protocol is the length of time it will take to generate a sufficiently random physiological value V. For example, a sufficiently random value based on IPI will take approximately 30 seconds [44]. Another limitation is that the Venkatasubramanian et al. protocol has a requirement that the sensed data should never be sent in the clear or to an untrusted third party.

A more complex method is to use a RSA–based password protocol [39]. Singh et al. showed how the EKE protocol can handle small entropy secrets, so that off–line and on–line dictionary attacks are infeasible for an adversary. The smaller entropy secrets do not require a long time to generate. If we have an EKG/PPG peak every 300–500 ms, a secret can be generated in less than a second. Another useful feature is that even if the SEV is compromised or available freely after the running of the key establishment protocol, the new session key will remain secure and safe. Several other RSA password protocols were also developed, but we will concentrate on the EKE password protocol.

The EKE protocol is chosen because other variants of password protocols require exponents of size 1024 bits. The EKE protocol is diagrammatically shown in *Protocol 2*, where the home health controller, C_h, initiates the key establishment protocol with the implanted sensor, S_b. A drawback of the EKE protocol is that it cannot use ECC [39].

Protocol 2 Diffie–Hellman–based EKE protocol

Shared Information: Generator g of G where $p - 1 = qr$

C_h		S_b
$r_A \in_R \mathbb{Z}_p$		
$t_A = g^{r_A}$	$\xrightarrow{A,[[t_A]]_{V_1}}$	$r_B \in_R \mathbb{Z}_p$
		$K_{C_hS_b} = t_A^{r_B}$
		$t_B = g^{r_B}$
$K_{C_hS_b} = t_B^{r_A}$	$\xleftarrow{[[t_B]]_{V_2},[[n_B]]_{K_{C_hS_b}}}$	
	$\xrightarrow{[[n_A,n_B]]_{K_{C_hS_b}}}$	Verify n_B
Verify n_A	$\xleftarrow{[[n_A]]_{K_{C_hS_b}}}$	

The EKE protocol contains four messages. The home health controller C_h sends the first message to the implanted sensor S_b, the message contains the location specified by A (the location value is in the clear), and the first part of Diffie–Hellman, t_A, is encrypted by the weak key V_1. After the first message is sent, the implanted sensor S_b will calculate the second part of the Diffie–Hellman scheme and hence be able to calculate the session key $K_{C_hS_b}$. The implanted sensor

S_b then sends the second part of the Diffie–Hellman scheme encrypted by the weak key V_2 to the home health controller C_h. The nonce n_B is also sent, encrypted by the session key $K_{C_hS_b}$. The last two messages authenticate both C_h and S_b, as well as confirming that they have the session key $K_{C_hS_b}$. The encryption of t_A, t_B, n_A, and n_B can be implemented with an XOR, as originally described by Bellovin [5].
Depending on which environmental value is measured, and how long the protocol will run, different SEVs may be used for the request and response. However, if the SEV stays constant throughout the running of the protocol, then both V_1 and V_2 will be the same. The EKE protocol is designed for a constant password throughout the running of the protocol, so similar or same data for both V_1 and V_2 will not adversely affect the protocol.
The EKE protocol was originally designed to handle small entropy secrets, so that off–line and on–line dictionary attacks are infeasible for an adversary. Another useful feature is that even if the secrets V_1 or V_2 are compromised or available freely after the running of the key establishment protocol, the session key $K_{C_hS_b}$ will remain secure and safe.
Both nonces n_A and n_B are cryptographically strong random numbers, allowing the XOR function to be used for encryption. If any nonce was not cryptographically strong then either $n_A \oplus K_{C_hS_b}$ or $n_B \oplus K_{C_hS_b}$ operation would allow an adversary to significantly reduce the number of valid $K_{C_hS_b}$ values. A characteristic of the EKE protocol is that the nonces are never sent out in the clear, since the nonces are used to encrypt the new key $K_{C_hS_b}$. The EKE protocol has the requirement that the sensed data should not be sent in the clear or to an untrusted third party, while the protocol has not completed. However, once the protocol is completed the sensed data that the protocol used can be made available.
The EKE protocol was proven to be secure [4]. However, one of the assumption was that the shared secret is only known by the parties that wish to establish a key, and is never sent out in the clear before or during the key establishment phase.

1) Performance Analysis of RSA and Elliptic Curves in BSNs

Previous research in sensors suggested that RSA is not suitable, and elliptic curves should be used [26].In this section we show that RSA can be used and is more efficient than an elliptic curve implementation. However, if the application does contain an elliptic curve implementation and there is not enough memory for an RSA implementation, we show suitable protocols that can be used with elliptic curves.
In traditional networks, password protocols can use different encryption algorithms, thus leading to several variants of the original EKE protocol. The variant protocols include the PPK protocol, PAK–R protocol [29], and the SPEKE protocol [21].
Common sizes used by these protocols are: p is 1024 bits, r is 864 bits and q is 160 bits. The technique used by the PAK protocol for its RSA based algorithm has the following steps:

1. $P = H(A, B, V_1)^r$ calculation is performed to map V_1 into the group. The hash of A, B, and V_1 is taken to the power r.

2. $r_A \in_R \mathbb{Z}_q$ a random value is found in the $\mathbb{Z}_q$ field.

3. $t_A = g^{r_A}$ the ephemeral key is created for the Diffie–Hellman operation.

4. $m = t_A P$ the message m is created, to decrypt the message the other node can divide P, m/P. An adversary will not be able to discover the value for t_A unless they know V_1. Also, the adversary will not find any invalid decrypted values, which removes the partition attack problem.

This protocol does suffer from the extra computational costs attributed to the larger exponents, and thus leading to much larger message sizes. In traditional networks, the design of protocols err on the side of caution. Sensor networks can not afford this luxury.
Another technique is to use $P = V_1^2$, where V_1 is interpreted as an element of $\mathbb{Z}_p^*$ [21]. The shared secret is $P^{r_A r_B}$. This is used by the SPEKE protocol.

1. $P = V_1^2$. Map V_1 into an element of $\mathbb{Z}_p^*$ of prime order $q = (p-1)/2$.

2. Select a random exponent value r_A. [21] suggested that the size of r_A to be 160 bits, however, no security proof could be defined. [30] provided a security proof, however, the exponent needs to be of order $\mathbb{Z}_q$, so that $r_A \in_R \mathbb{Z}_q$. This causes the size of the exponent to be 1024 bits.

3. $t_A = P^{r_A}$. The ephemeral key is created for the Diffie–Hellman algorithm.

A problem with the above constructs is that the size of the messages will be over 1024 bits. In an energy constrained and low bandwidth environment, this is not suitable. However, when using the [21] method of generating the exponent, this will limit the computational expense. To limit the message sizes caused by RSA, elliptic curve cryptography has been proposed for sensor networks [46, 31].
Many RSA based password protocols can be generalised to use elliptic curves [29]. Using a straightforward conversion of RSA to elliptic curves on the EKE protocol, we can create *Modified Protocol 1*.

Modified Protocol 1 EKE protocol with elliptic curve

Shared Information: Generator g of G where $y^2 = x^3 + ax + b$

A		B
$r_A \in_R \mathbb{Z}_p$		
$t_A = r_A g$	$\xrightarrow{A,[[t_A]]_{V_1}}$	$r_B \in_R \mathbb{Z}_p$
		$K_{AB} = r_B t_A$
		$t_B = r_B g$
$K_{AB} = r_A t_B$	$\xleftarrow{[[t_B]]_{V_2},[[n_B]]_{K_{AB}}}$	
	$\xrightarrow{[[n_A,n_B]]_{K_{AB}}}$	Verify n_B
Verify n_A	$\xleftarrow{[[n_A]]_{K_{AB}}}$	

The *Modified Protocol 1* is similar to the EKE protocol, except that instead of exponentiations, we propose the use of elliptic curve point multiplications. However, encryption of

an elliptic curve point using a low randomness is not as simple as encrypting an element of $\mathbb{Z}_p^*$. The problem is that an adversary guessing V_1 can attempt to decrypt $[[t_A]]_{V_1}$ and examine whether the resulting plaintext is a valid point on the curve. A symmetric key algorithm matched to the elliptic curve group is required.
To convert the above RSA implementation, the V_1 will need to be mapped to an elliptic curve point. A general procedure for this can be found in *IEEE P1363.2: Standard Specifications for Password–Based Public–Key Cryptographic Techniques.*A simplified version of the procedure is shown below.

1. Set $i = 1$.
2. Compute $w = h(A, B, V_1, i)$.
3. Set $\alpha = w^3 + aw + b$.
4. If $\alpha = 0$ then the point is $(w, 0)$.
5. Find the minimum square root of α, and call it β. Can use the method found in *IEEE P1363*(Appendix A.2.5).
6. If no square root exists, set $i = i + 1$, and go to Step 2.
7. The elliptic curve point is (w, β).

The above algorithm is non–deterministic for different V_1 values. If a square root is not found then the algorithm will loop back to the second step to compute a new value for β. In an environment where a sensor scavenges energy to perform an operation, it is not suitable to have a non–deterministic algorithm.
Mapping a variable into a point on an elliptic curve allows the conversion of many RSA password protocols to an elliptic curve password protocol.

1. Mapping SEV into the field. When using RSA, the SEV can be naturally mapped into the field. This could either be a direct modulus, or a modulus of the hash of the SEV. However, mapping the SEV onto a point requires more work as shown by the following equation $P = f(A, B, V_1)r$. The function f is the non–deterministic method to map a number onto a point in an elliptic curve, as described above. The point is then multiplied by r, to place it into the correct group.
2. Both the RSA and the ECC implementation require a random value in the field $\mathbb{Z}_q$ to obtain an r_A as shown by $r_A \in_R \mathbb{Z}_q$.
3. The ephemeral value for Diffie–Hellman is calculated, for ECC, that is $t_A = r_A g$. The RSA algorithm involves an exponent.
4. Finally, the message is created. In the RSA case, P is multiplied to t_A, whereas in the ECC case the following equation is used $m = t_A + P$. The message m is created, to decrypt the message the receiver can subtract P, since $t_A = m - P$. An adversary will not be able to discover the value for t_A unless they know V_1. Also, the adversary will not find any invalid decrypted values, which removes the partition attack problem.

However, when converting this into an elliptic curve–based construct, we face the same problems as shown for the PAK family of products. The SEV will need to be mapped onto an elliptic curve point, which is non–deterministic.
We have developed a technique to make the protocol deterministic, where the elliptic curve point $(x, y) = (h_1(V_1, R_1), h_2(V_1, R_1))$. After this calculation, a valid elliptic curve is found where this point is valid $y^2 = x^3 + ax + b$, where the value for a is predefined. However, this does require extra encrypted information to be sent from A to B, $[[R_1]]_{V_1}, [[b]]_{V_1}$. In a resource constrained environment, extra information sent by a sensor will require additional energy to be consumed by the sensor.

C. Body Sensors and Home Sensors

The implanted body sensor can now obtain all the session keys via the Home Health Controller. Since the Home Health Controller has a secure connection between the home sensors and the implanted body sensor, it can be used as a trusted third party. Singh et al. [38] surveyed and proposed trusted third party protocols for multi-tiered sensor networks.
In our case we will use the Singh et al. trusted third party protocol, as shown in *Protocol 3* [39], since it was shown to have some advantages over other protocols for the sensor environment.

Protocol 3 Singh et al. Trusted Third Party Protocol

$m = A, B, K_S$
$AUTH_A = [m]_{K_{AS}}$, $MASK_A = [[AUTH_A]]_{K_{AS}}$
$AUTH_B = [m]_{K_{BS}}$, $MASK_B = [[AUTH_B]]_{K_{BS}}$

$M1$	$A \rightarrow S:$	A, B, N_A
$M2$	$S \rightarrow B:$	$A, N_A, AUTH_B, MASK_B \oplus K_S$
$M2'$	$S \rightarrow A:$	$AUTH_A, MASK_A \oplus K_S$
$M3$	$B \rightarrow A:$	$[N_A]_{K_{AB}}, N_B$
$M4$	$A \rightarrow B:$	$[N_B]_{K_{AB}}$

There are only two messages that contain the $MASK$; the message $M2$ – server S sending to node B, and the message $M2'$ – server S sending to node A. Neither A nor B ever send out the $MASK$. A possible attack on our protocol is for an adversary to try and obtain the $MASK$ value by interrogating S. If an adversary pretends to be A, it does not matter what locations and nonce gets passed to S, because S should produce a new K_S, and therefore a new $MASK$ and a new $MASK \oplus K_S$.
As shown in Equation (1), if two exclusive–ors produce the same value, and the keys are different, then the $MASK$ will have to be different.

$$MASK \oplus K_S = MASK' \oplus K'_S \tag{1}$$

If the $MASK$ is the same, as shown in Equation (2), then no extra information about K_S can be obtained, as long as a strong MAC is used. It is assumed that the adversary does not know the long term key K_{AS}.

$$[A, B, K_S]_{K_{AS}} = [A, B, K'_S]_{K_{AS}} \tag{2}$$

Since B does not initiate the protocol, it has no input into the creation of $MASK$. So an adversary who pretended to be B has less input in the value of $MASK$ than if they pretended

to be A. The integrity of the key is also assured since key modification requires simultaneous modification of $AUTH$ as well as $MASK \oplus K_S$.
The key K_S can be used in the future to create or renew a session key between A and B. However, that relies on the assumption that the key K_S has not been compromised. Since K_S is never used as a session key or used to encrypt any plaintext, the keys K_{AS} and K_{BS} should be compromised before K_S. The sensor nodes should regularly refresh K_{AS}, K_{BS} and K_S with the base station.
A variant of the above protocol was proven to be secure [6]. One of the assumption was that the long–termed key is not compromised. In a complex system it is important to know which components have high physical security and which devices can be easily obtained by an adversary.

D. Mobile Phone

Establishing keys between the mobile phone and body and home sensors is the last step. The mobile phone can have a sensor attached via a USB connection or have it built-in. This allows the mobile phone to establish a key with the implanted sensor in the same way that the Home Health Controller established a key with the implanted sensor.
After the key is established with the mobile phone, the implanted sensor can be used as a trusted third party to establish a key between the mobile phone and the Home Health Controller. The mobile phone can then use the Home Health Controller to establish keys between itself and all the home sensors.

VI. Modelling of the Home Health Care System using GDM

As described previously, the protocols by themselves have been proven to be secure. However, there has been no check to validate that the protocols are secure within a complex home health care system. The validation of the security protocol in our home health care system is performed by using GDM. The modelling was completed after several stages. The initial stage placed the Venkatasubramanian et al. protocol into a behavior tree.
From the properties of the key establishment we developed the Requirement Behaviour Trees (RBTs). While developing the RBTs, we found that the previous definitions and properties of the protocols did not have a consistent method to define the need for the sensor to sense the physiological data. The RBT is designed for, and has built–in syntax for, external events, so this requirement was easily added to our RBTs. The feature for quickly adding external events makes RBTs suitable for modelling and analysis of a sensor environment. There are three major components in the Behaviour Tree: C_h; Body; S_b. Two requirements were put into the behavior tree, but space restrictions limited the display of the home sensors in this paper. While we specified the first requirement we realized that there was a missing section which was not specified in the original protocol, and that was to remove the physiological value V that was used to encrypt the new session key. So we placed that at the end of the requirement $R1$. After the value V is destroyed we then attached the requirement $R2$ onto the behavior tree in Figure 2.

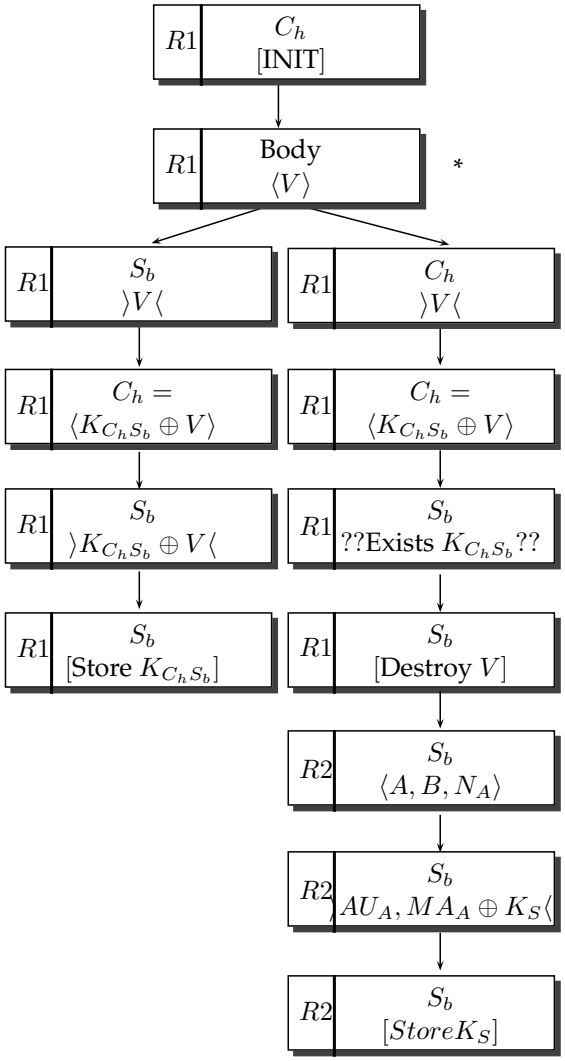

Figure. 2: Behaviour Tree Representation of the Health Care System

The Venkatasubramanian et al. protocol properties require that the physiological value V needs to be cryptographically strong, and the physiological value V that was used during key establishment should never be sent to a third party.
In this case we add another requirement as shown in Figure 3, that if the mobile phone obtains the physiological value V then it can calculate the new session key between the implanted sensor and the Home Health Controller. We can also do the same for all of the other components within the household.

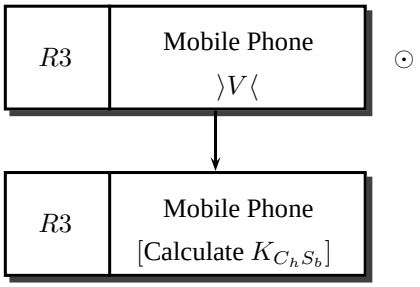

Figure. 3: Intruder Represented as a Behaviour Tree

There is no integration point between the first behavior tree and the second behavior tree. So the system is secure. We then add a new requirement, as shown in Figure 4. In this requirement we have the mobile phone having the ability to read the physiological value V from the body.

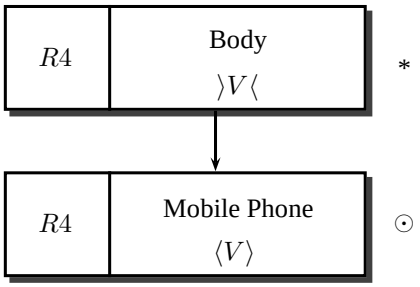

Figure. 4: Mobile Phone Represented as a Behaviour Tree

After adding this requirement into our system, an integration point does exist. In Figure 2 we placed a $*$ next to the behavior that will be integrated with the same behavior found in Figure 4. Then we can also have the requirement, shown in Figure 3, be integrated. We have denoted that integration point with the symbol, $\odot$. With the behavior tree, shown in Figure 3, integrated into our system, we have shown the system to be insecure.
As our system grows and new requirements for the system are found, past assumptions may prove to have become wrong, as shown in this case. Behaviour trees is a good tool to keep track of past assumptions, confirming that any new requirements placed on the system does not cause the system to become insecure.
Our next step is to find a protocol where there are not as many restrictions. The EKE protocol is a good candidate and has the following properties:

- Sensor nodes only possess a secret of small entropy,
- Off–line dictionary attacks are not feasible,
- On–line dictionary attacks are not feasible, and
- The key must have forward secrecy.

The protocol is modelled into a behavior tree, as shown in Figure 5. Also, the behavior tree (showing an intruder), shown in Figure 3, is no longer a valid possibility. If any component obtains the SEV they will not be able to calculate the key using the new protocol. There still exists an integration point between the mobile phone requirement, $R4$ and the implanted sensor key establishment requirement, new $R1$.

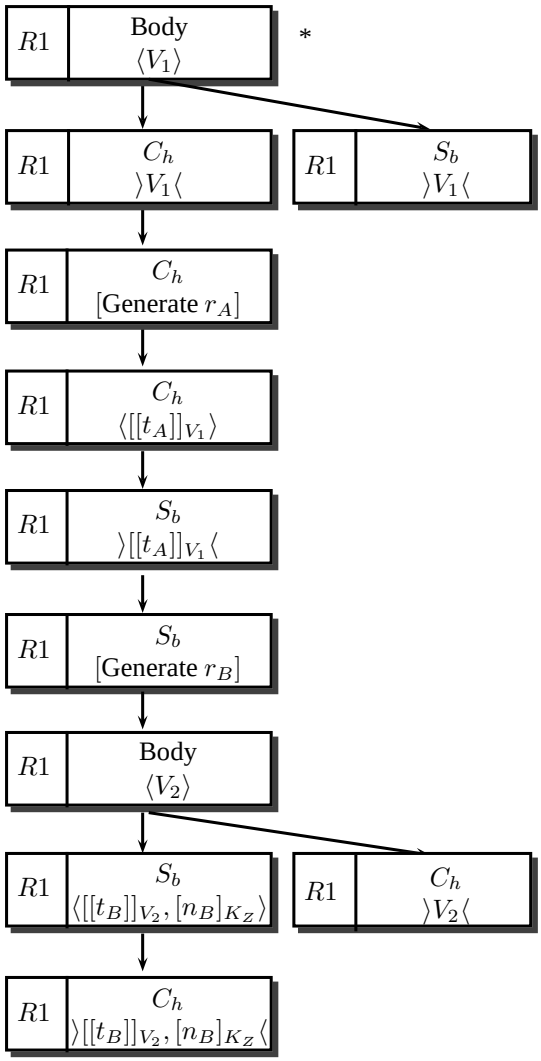

Figure. 5: Updated Behaviour Tree of a Home Health Care System

By using behavior trees, we were quickly able to find all of the possible inputs and outputs that a sensor can obtain, either through wireless communication or through their sensing devices. This also helps us to verify that each component that we are developing has the needed features to run in our environment. When there are a large number of sensors, this requirement becomes difficult to track. Singh et al. [39] showed the following by first generating SAL code and then creating theorems. We have shown that instead of generating SAL code, validation of the security protocols can be accomplished while integrating the Behaviour Trees.
We have shown that protocol assumptions about an environment can be validated using GDM. A protocol assumption

was defined as an RBT mimicing an intruder. If the intruder RBT is able to be integrated with the final IBT there is a security flaw with the system. One solution to the problem is to remove the requirement that allowed the intruder RBT to be attached to the final IBT. If that is not feasible, another solution is specifying a different security protocol with different protocol assumptions. We used the second solution to show that a security protocol can be used even if the initial secret becomes known to an intruder.

VII. Implementation and Performance Analysis of Protocols

In this section, we describe the implementation of the cryptographic algorithms and some of the selected security protocols discussed in this paper. We implemented and compared different cryptographic primitives that can be used in body sensor security protocols on a Crossbow mica2 MPR2600 mote.A number of comparisons were performed to check the viability of the protocols in a sensor environment. We investigated the length of time it takes to perform different cryptographic operations, as well as the total time to run different protocols. A comparison between results from implementations executed on hardware and results from implementation executed on simulators. We also check the memory sizes of the application using different cryptographic operations.

A. Timing and Instruction Size

Before comparing the different cryptographic primitives, and the benefits that one implementation has over another, we created skeleton code based on TinyOS 2.x.The skeleton code initializes the sensor node and after the sensor is initialized we obtained the time in milliseconds. The next step is to execute a cryptographic primitive in a loop for 2000 iterations, and then we obtain a new time. We subtracted the new time from the initial time to obtain the elapsed time in milliseconds to run our cryptographic primitive for 2000 attempts. The elapsed time was then sent via the serial connection to a PC running a Linux® distribution where we have a Java® application reading the TinyOS packet from the serial port and report that data to the user.

The configuration we used is shown in Figure 6. One of the sensors is attached to the computer with a USB cable. The computer registers the connection as a serial port. The communication between the computer and the sensor is achieved through the serial port.

The key establishment protocols use exclusive–or (XOR) with a random number to encrypt the new session key. We compare this method with other methods of encrypting the new session key for body sensor networks. We have implemented RC5, SKIPJACK, HMAC–MD5, RSA, and ECC cryptographic primitives on the mica2 MPR2600 motes using TinyOS 2.x. When comparing ECC password protocol with an RSA password protocol an important distinct primitive within the ECC protocol is the square root function. We have separated the times of the square root function with the ECC computation.

Table 3 shows the ratio of the application size for each of the algorithms compared to exclusive-or algorithm. When we ran the algorithm on the mica2 mote, over the 2000 iterations it took approximately one millisecond. In the ATEMU simulator it took approximately 7000 instructions.

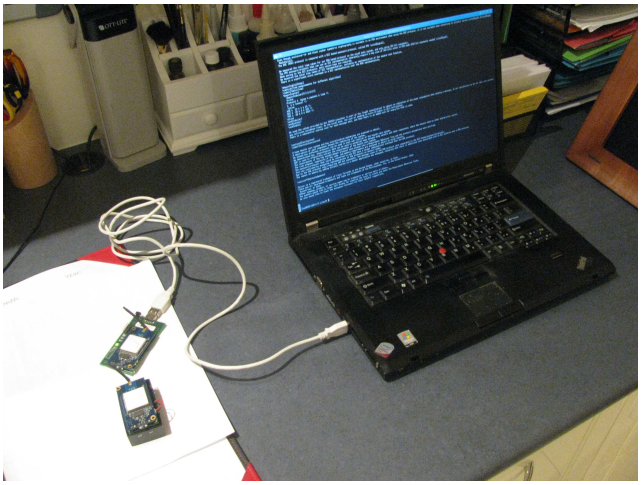

Figure. 6: Reading from the Sensor using the Serial Port

Table 3: Comparison of Application Size: As Ratio to exclusive-or

Algorithm	Mica2	ATEMU
RC5	453	456
SKIPJACK	739	741
HMAC–MD5	18400	18500
RSA	41600	41900
SQRT	87800	88400
ECC	4820000	4920000

We found little difference between the simulation results and the amount of time an operation takes when put on the mica2 mote. The most notable difference in results was for the ECC algorithm where there was a two percent difference. Both the time and number of instructions suggest that for the same size key the RSA algorithm is significantly better than the ECC algorithm.

B. Memory Size

The size of the application both with number of lines of code and the size in bytes is important when choosing an algorithm. Table 4 list the number of lines of code and the size in bytes of the application that we used to run our original time and instruction measurements.

Table 4: Memory size for different algorithms

Algorithm	Code Lines	Size (bytes)	Stack (bytes)	RAM (bytes)
XOR	80	5600	158	432
RC5	506	6776	172	466
SKIPJACK	697	8138	190	496
HMAC–MD5	507	15540	523	918
RSA	1456	7062	213	624
SQRT	3366	7662	230	748
ECC	5038	14020	760	2066

The *Code Lines* indicates lines of code and thus the complexity of the code for a developer to implement the application. The *Size (bytes)* indicates the size in bytes of the application. The *Stack (bytes)* indicates the maximium size of the stack for the application. The *RAM (bytes)* is the maximium

amount of RAM the application will need. The figures are obtained from the stack analysis tool found in tinyos.
The RC5 application took considerably more time to implement than the XOR application. We found an RC5 implementation for TinyOS 1.x in the TinySEC library [23], however, it needed to be ported to TinyOS 2.x. We ported the code to the new platform.
The SKIPJACK application had similar problems as the RC5 application. Where there was an implementation for TinyOS 1.x in the TinySEC library but there was not one for TinyOS 2.x. Once again, significant effort was put on porting the code to the platform.
For HMAC–MD5 application we could not find any previous implementations of HMAC–MD5 in any version of TinyOS. In this case we obtained code from RFC1321and RFC2104and ported the code to first the nesc language and then to the TinyOS application. This needed considerably more time to implement then either RC5 or SKIPJACK implementations. The code for RC5 and SKIPJACK were ported from one TinyOS version to another. Whereas, the HMAC-MD5 application needed to be rewritten into the nesc language.
The RSA application also had similar problems as the RC5 and SKIPJACK implementations. We found code in the Deluge System [16], however, the RSA code was based off TinyOS 1.x. Effort was required to port this code to TinyOS 2.x. We used a 160 bit exponent as required by the EKE protocol.
The SQRT application had the most difficulties since we implemented it from pseudo–code rather than porting any code. We used Newton's Method [36] for finding square roots to implement the SQRT application.
The ECC application also had similar problems to the RSA, RC5 and SKIPJACK implementations. We ported an ECC library [26] developed for TinyOS 1.x to TinyOS 2.x. The ECC application used a 160 bit points, since password protocols that could be converted to use ECC require stronger keys [39].
The XOR application is the quickest by several orders of magnitude compared to the other cryptographic primitives. But the size of the application is smaller, and the number of lines is less then the other applications. The XOR application is the quickest, whereas the ECC application is the slowest. This verifies existing research into the differences in speed for password protocols of RSA and ECC implementations in TinyOS simulators [39]. The HMAC–MD5 application is the largest, however the application was a straight port from the RFCs, where the code was not intended for sensors.
We also examine the memory requirements of the application, as shown in Table 5. The figures were obtained using the tool avr–size. The combination of *.bss* and *.data* segments use SRAM, and the combination of *.text* and *.data* segments use ROM. The *.text* contains the machine instructions for the application. The *.bss* contains uninitialized global or static variables, and the *.data* section contains the initialized static variables.

C. Protocol Times

Even though exclusive–or and block cipher symmetric cryptography is suitable in an RSA environment when using the EKE protocol, it is not suitable when converting to elliptic curves [7].The EKE (RSA) protocol is compared with a ECC based password protocol, called PPK [8].
We measured the total time taken for our RSA based protocol on the mica2 mote system, and only using 160 bit exponents. The PPK password protocol that can use an ECC implementation inherently require a key size of 160 bits since in RSA mode 1024 bit exponents needed [7]. When moving to the ECC protocols, more secure keys are required. We measured the total time taken of ECC implementation, including an implementation of the square root function. There is a significant extra overhead in a ECC implementation over the RSA implementation.

Table 5: Memory Overhead In Bytes On MICA2 Platform

Memory	RSA	ECC
ROM	1942	9720
RAM	177	859
.data	60	8
.bss	117	851
.text	1882	9712

Table 6: Time measurements for different algorithms

Protocol	bytes	packets	time (msecs)
XOR	10	1	15
EKE	45	4	102
PPK	69	6	4910

We used the values provided by the TOSSIM simulator (a part of the TinyOS installation) to obtain an indication of the power consumption when sending a message. In our calculations we do not take into account any collision avoidance times. On the mica2 mote, the cost of sending an extra 20 bytes is 28.1 microjoules. There is a substantial startup cost for each message sent, and then there is an added cost for every bit that is sent.

VIII. Conclusions

A home health care system and key initialization mechanisms are examined in detail. A description of each component showed the complexity of the system. We showed how different protocols can be used in each sub–section of the health care system. We demonstrated how physiological data can be used to establish keys between body sensors and other components, where the sensors have no other shared prior secret. GDM is used to effectively extract the requirements of the health care system. The requirements of the key establishment protocol were placed into a Requirement Behaviour Tree and the protocol assumptions were verified. The time elapsed, complexity of the code, and memory requirements are analysed in detail on mica2 sensors. The password protocols that could be converted to use ECC have a larger computational overhead than the EKE protocol. This was confirmed by analyzing implementations of the protocols on sensors nodes. Due to the EKE protocol only requiring 160 bit exponents, the message sizes of the EKE protocol were comparable to the ECC–based password protocols. The impact on memory by adding elliptic curves to a sensor application was analyzed, revealing that there is additional cost associat-

ed with an ECC solution over a RSA solution. Future work includes creating behaviour trees for the entire health care system, and analysis of the security properties for the entire system.

Acknowledgments

The authors would like to thank the Late Prof. Geoff Dromey for suppling the tools to create and check the Requirement Behaviour Trees. We also would like to thank the anonymous reviewers for their helpful suggestions.

References

[1] Omer Aziz, Benny Lo, Ara Darzi, and Guang-Zhong Yang. Introduction. In Guang-Zhong Yang, editor, *Body Sensor Networks*. Springer–Verlag, 2006.

[2] Shu-Di Bao, Yuan-Ting Zhang, and Lian-Feng Shen. Physiological signal based entity authentication for body area sensor networks and mobile healthcare systems. In *27th Annual International Conference of the Engineering in Medicine and Biology Society, 2005*, pages 2455–2458. IEEE Press, 2005.

[3] Shu-Di Bao, Yuan-Ting Zhang, and Lian-Feng Shen. A design proposal of security architecture for medical body sensor networks. In *BSN '06: Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*, pages 84–90, Washington, DC, USA, 2006. IEEE Computer Society.

[4] Mihir Bellare, David Pointcheval, and Phillip Rogaway. Authenticated key exchange secure against dictionary attacks. pages 139–155. Springer-Verlag, 2000.

[5] Steven M. Bellovin and Michael Merritt. Encrypted key exchange: Password-based protocols secure against dictionary attacks. In *IEEE Symposium on Research in Security and Privacy*, pages 72–84. IEEE Computer Society Press, 1992.

[6] Colin Boyd, Kim kwang Raymond Choo, and Anish Mathuria. An extension to bellare and rogaway (1993) model: Resetting compromised long-term. In *Information Security and Privacy, 11th Australasian Conference, ACISP*, 2006.

[7] Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer Berlin / Heidelberg, 2003.

[8] Victor Boyko, Philip MacKenzie, and Sarvar Patel. Provably secure password-authenticated key exchange using Diffie-Hellman. In B. Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 156–171. Springer-Verlag, 2000.

[9] Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In *EUROCRYPT 2001: Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, pages 453–474, London, UK, 2001. Springer-Verlag.

[10] A.D.C Chan, M.M. Hamdy, A. Badre, and V. Badee. Wavelet distance measure for person identification using electrocardiograms. *IEEE Transactions on Instrumentation and Measurement*, 57(2):248–253, 2008.

[11] Haowen Chan and Adrian Perrig. PIKE: Peer intermediaries for key establishment in sensor networks. In *Proceedings of IEEE Infocom*. IEEE Computer Society Press, March 2005.

[12] E. M. Clarke, S. Jha, and W. Marrero. Verifying security protocols with brutus. *ACM Transactions Software Engineering Methodology*, 9(4):443–487, 2000.

[13] Jing Deng, Richard Han, and Shivkant Mishra. Sensor–network security, privacy, and fault tolerance. In N. Bulusu and S. Jha, editors, *Wireless Sensor Networks: A Systems Perspective*. Artech House, 2005.

[14] R. Geoff Dromey. From requirements to design: Formalizing the key steps. In *SEFM*, pages 2–, 2003.

[15] R.G. Dromey. From requirements to design: Formalizing the key steps. *sefm*, 00:2, 2003.

[16] Prabal K. Dutta, Jonathan W. Hui, David C. Chu, and David E. Culler. Securing the deluge network programming system. *In the Fifth International Conference on Information Processing in Sensor Networks (IPSN'06)*, April 2006.

[17] Javier Espina, Thomas Falck, and Oliver Mülhens. Network topologies, communication protocols, and standards. In Guang-Zhong Yang, editor, *Body Sensor Networks*. Springer–Verlag, 2006.

[18] Panu Hämäläinen, Mauri Kuorilehto, Timo Alho, Marko Hännikäinen, and Timo D. Hämäläinen. Security in wireless sensor networks: Considerations and experiments. In *SAMOS*, pages 167–177, 2006.

[19] Arun Hampapur, Lisa Brown, Jonathan Connell, Norman Haas, Max Lu, Hans Merkl, Sharat Pankanti, Andrew Senior, Chiao-Fe Shu, and Yingli Tian. S3-r1: the ibm smart surveillance system-release 1. In *ETP '04: Proceedings of the 2004 ACM SIGMM workshop on Effective telepresence*, pages 59–62, New York, NY, USA, 2004. ACM Press.

[20] IEEE. *IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks — Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 802.11-2007 edition, June 2007.

[21] David Jablon. Strong password–only authenticated key exchange. *ACM Computer Communication Review*, 26(5):5–26, October 1996.

[22] Aman Kansal and Mani Srivastava. Energy–harvesting–aware power management. In N. Bulusu and S. Jha, editors, *Wireless Sensor Networks: A Systems Perspective*. Artech House, 2005.

[23] Chris Karlof, Naveen Sastry, and David Wagner. Tinysec: a link layer security architecture for wireless sensor networks. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 162–175, New York, NY, USA, 2004. ACM Press.

[24] D.T.H. Lai, R. Begg, and M. Palaniswami. *Healthcare Sensor Networks: Challenges Toward Practical Implementation*. Taylor & Francis, 2011.

[25] Yuan-Hsiang Lin, I-Chien Jan, P.C.I Ko, Jau-Min Wong, and Gwo-Jen Jan. A wireless pda–based physiological monitoring system for patient transport. *IEEE Transactions on Information Technology in Biomedicine*, 8(4):439–447, 2004.

[26] An Liu and Peng Ning. Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks. In *Proceedings of the 7th international conference on Information processing in sensor networks*, IPSN '08, pages 245–256, Washington, DC, USA, 2008. IEEE Computer Society.

[27] Donggang Liu and Peng Ning. *Security for Wireless Sensor Networks*. Springer Berlin / Heidelberg, 2007.

[28] Gavin Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In *TACAs '96: Proceedings of the Second International Workshop on Tools and Algorithms for Construction and Analysis of Systems*, pages 147–166, London, UK, 1996. Springer-Verlag.

[29] Philip MacKenzie. More efficient password-authenticated key exchange. In D. Naccahe, editor, *Topics in Cryptology — CT–RSA 2001*, volume 2020 of *Lecture Notes in Computer Science*, pages 361–377. Springer-Verlag, 2001.

[30] Philip MacKenzie. On the security of the SPEKE password-authenticated key exchange protocol. Technical Report 2001/057, 2001.

[31] D. J. Malan, M. Welsh, and M. D. Smith. A public–key infrastructure for key distribution in tinyos based on elliptic curve cryptography. In *Proc. 1st IEEE Communications Society Conference on Sensor and Ah Hoc Communications and Networks (SECON '04)*, pages 71–80, Santa Clara, CA, USA, October 2004. IEEE Computer Society Press.

[32] Catherine A. Meadows. The nrl protocol analyzer: An overview. *Journal of Logic Programming*, 26:113–131, 1996.

[33] J. C. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using mur/spl phi/. In *SP '97: Proceedings of the 1997 IEEE Symposium on Security and Privacy*, page 141, Washington, DC, USA, 1997. IEEE Computer Society.

[34] Lawrence C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1-2):85–128, 1998.

[35] Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, and J. D. Tygar. SPINS: Security protocols for sensor networks. In *Seventh Annual International Conference on Mobile Computing and Networks (MobiCOM 2001)*, Rome, Italy, July 2001.

[36] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. Root finding and nonlinear sets of equation. In William H. Press, editor, *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, 2007.

[37] K. Singh and V. Muthukkumarasamy. Using physiological signals for authentication in a group key agreement protocol. In *Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on*, pages 720 –725, april 2011.

[38] Kalvinder Singh and Vallipuram Muthukkumarasamy. Performance analysis of proposed key establishment protocols in multi–tiered sensor networks. *Journal of Networks*, 3(6), 2008.

[39] Kalvinder Singh and Vallipuram Muthukkumarasamy. Key establishment protocols using environmental and physiological data in wireless sensor networks. *Inderscience International Journal of Sensor Networks IJSNet*, 8(1), 2010.

[40] Elankayer Sithirasenan, Saad Zafar, and Vallipuram Muthukkumarasamy. Formal verification of the ieee 802.11i wlan security protocol. In *Australian Software Engineering Conference (ASWEC '06)*, Sydney, Australia, 2006.

[41] Dawn Xiaodong Song. Athena: a new efficient automatic checker for security protocol analysis. In *CSFW '99: Proceedings of the 12th IEEE workshop on Computer Security Foundations*, page 192, Washington, DC, USA, 1999. IEEE Computer Society.

[42] E. M. Staderini. Uwb radars in medicine. *IEEE Aerospace and Electronic Systems Magazine*, 21:13–18, January 2002.

[43] Fahim Sufi, Ibrahim Khalil, and Ibrahim Habib. Polynomial distance measurement for ecg based biometric authentication. *Security and Communication Networks*, 9999(9999):n/a, 2008.

[44] Krishna K. Venkatasubramanian, Ayan Banerjee, and Sandeep K. S. Gupta. Pska: Usable and secure key agreement scheme for body area networks. *IEEE Transactions on Information Technology in Biomedicine*, 14(1):60–68, 2010.

[45] Krishna K. Venkatasubramanian and Sandeep K. S. Gupta. Security for pervasive health monitoring sensor applications. In *ICISIP '06: Proceedings of the 4th International Conference on Intelligent Sensing and Information Processing*, pages 197–202, Bangalore, India, December 2006. IEEE Press.

[46] Ronald Watro, Derrick Kong, Sue fen Cuti, Charles Gardiner, Charles Lynn, and Peter Kruus. Tinypk: securing sensor networks with public key technology. In *SASN '04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 59–64, New York, NY, USA, 2004. ACM Press.

[47] K. Winter, I.J. Hayes, and R. Colvin. Integrating requirements: The behavior tree philosophy. In *Software Engineering and Formal Methods (SEFM), 2010 8th IEEE International Conference on*, pages 41 –50, sept. 2010.

Author Biographies

Kalvinder Singh Kalvinder Singh obtained a BSc with 1st Class Hons and a universtiy medal from James Cook University, Australia in 1996. He is currently completing his PhD at Griffith University, Australia. His current research areas include investigation of security issues in sensor networks, key establishment protocols and medical sensor networks.

Elankayer Sithirasenan Elankayer Sithirasenan received his PhD and Master of Software Engineering degrees from Griffith University, Australia in 2009 and 2004 respectively and the BSc degree in Electrical and Electronic Engineering from University of Peradeniya, Sri Lanka in 1991. He is currently a lecturer in Information and Communication Technology at Griffith University, Gold Coast, Australia. His current research interests include wireless/wired network security, authentication and access control, disaster recovery, intrusion detection and prevention, outlier detection on multilevel, multivariate data sets and software requirements analysis. He was a lecturer at the University of Peradeniya, Sri Lanka form 2001 to 2003 and the director of Integrated Digital Systems, Sri Lanka from 1998 to 2008.

Vallipuram Muthukkumarasamy Dr Muthukkumarasamy obtained BScEng with 1st Class Hons from University of Peradeniya, Sri Lanka and obtained PhD from Cambridge University, England. He is currently attached to School of Information and Communications Technology, Griffith University, Australia as Senior Lecturer. His current research areas include investigation of security issues in wireless networks, sensor networks, trust management in MANETs, key establishment protocols and medical sensor networks. He is currently leading the Network Security research Group at the Institute for Integrated and Intelligent Systems at Griffith University. He has also received number of best teacher awards.