SAND2010-6150C

Formal Analysis of Device Authentication Applications in Ubiquitous Computing

ABSTRACT

Authentication between mobile devices in ad-hoc computing environments is a challenging problem. Without pre-shared knowledge, existing applications rely on additional communication methods, such as out-of-band or location-limited channels for device authentication. Much of the focus in development of new applications in this area seeks to reduce or eliminate the impact of this additional requirement. However, no formal analysis has been conducted to determine whether out-of-band channels are actually necessary. We seek to answer this question through formal analysis of authentication protocols in mobile device applications. Specifically, we use BAN logic to show that device authentication using a single channel is not possible, and propose a BAN logic extension to help prove correct existing authentication protocols. We further demonstrate this analysis on existing mobile device authentication applications.

1. INTRODUCTION

The field of ubiquitous computing has shown tremendous growth over the past several years, both in the sophistication and capabilities of devices, and in the variety of applications to which devices are being applied. Many people rely on *mobile devices*, such as smart phones, personal digital assistants (PDAs), tablets, and laptop computers to stay in touch with friends, family, co-workers, and other important contacts.

One core component of ubiquitous computing is communication with other devices, which usually occurs using broadcast radio frequency (RF) transmission. Consequently, problems may arise when communication between devices is expected to be private. Consider a credit-card transaction between a smart-phone and a movie-ticket kiosk; the consumer's credit card information should be protected from eavesdropping. The most common method for protecting these types of information exchanges is encryption. However, this raises additional issues. First, devices in ubiquitous computing are usually constrained in both power and computational capabilities, making some encryption techniques either too expensive in terms of power consumption, or undesirable in terms of time. Fortunately, advances in device hardware have addressed this concern very well. Second, in order to encrypt information, both the sender and receiver must agree on the method of encryption, as well as key information. While this is a relatively simple operation between two machines operating in a wired, trusted environment, accomplishing key establishment between *wireless* devices in *untrusted* environments presents several challenges. The greatest of these challenges is *device authentication*, which means ensuring that only the intended devices are those actually communicating. Put another way, it is ruling out the possibility that an unknown device, or "man-in-the-middle," is present and intercepting data between the devices. This challenge is especially difficult because devices in ubiquitous computing are not assumed to possess a priori knowledge of each other.

Several applications exist to address this problem, proposing authenticated information exchange between mobile devices using an array of methods other than the standard broadcast channel. These are sometimes called *out-of-band*, *side-channels* or *location-limited channels* (LLCs) [1], and include audio, visual, infrared, ultrasound, and other forms of transmission [7, 9, 15, 19]. While not necessarily providing confidentiality, they allow the receiver of the device to physically verify the source of the transmission. Using the information received, devices can be authenticated, and secure key establishment can occur.

In standard computing environments, authentication protocols are subject to rigorous performance analysis and verification. One tool for doing this is known as BAN Logic [2]. This work presents a logic of *belief*, which is basically the process of coupling assumptions about current states with a simple set of steps and reasoning, to arrive at a conclusion regarding the soundness of an authentication protocol. BAN Logic has been used extensively to identify flaws in new and existing security protocols, and is often employed by authors to prove new proposals sound.

However, applying such formal analysis to device authentication protocols in mobile computing applications has not been studied to date. Important and interesting questions arise such as: can device authentication protocols that use location-limited channels be shown correct using analysis techniques such as BAN logic? More importantly, are location-limited channels a necessary component of device authentication between mobile devices in ubiquitous computing?

In this paper, we seek to answer these questions. Specifically, we show that device authentication between previously unknown devices in ubiquitous computing is not possible using only a single broadcast channel for communication.

Next, we attempt a formal analysis of device authentication protocols that use LLCs. However, as originally proposed, BAN logic is insufficient to address the characteristics of such protocols, so we propose two extensions to BAN logic to support them. To demonstrate our extension, we show the analysis of two existing device authentication applications.

The remainder of this paper is organized as follows. Section II discusses background material and related works. Section III considers device authentication using a single broadcast channel only and extends BAN logic to support analysis of device authentication protocols. Section IV demonstrates the use of these extensions, and Section V concludes the paper with a discussion of future work.

2. BACKGROUND

Mobile computing environments generally do not assume the presence of a trusted authority (TA) to provide verified authentication information, such as public keys. Therefore, to authenticate public keys, another solution is necessary, and several approaches have been developed to address this problem. Solutions requiring some sort of additional knowledge, passed between users either in the form of a password or via some other communication channel besides the standard channel make up the bulk of the work on this subject.

2.1 Location-Limited Channels

One of the most widely referenced techniques for verifying key establishment between mobile devices is proposed by [1] and introduces three components: *location-limited channels*, *demonstrative identification*, and *pre-authentication*. Location-limited channels (LLC) are a means of communication between two devices with the property that the operators of the devices have control over which devices are communicating. Unlike radio frequency (RF), where the sending and receiving devices are not easily identifiable, LLCs could include non-RF communication. Demonstrative identification describes the process by which the sending device is authenticated simply by sending information via the LLC. For instance, a kiosk showing a video display of a two-dimensional barcode can be verified as the source of the data simply by looking at it - it demonstrates that the information being shown originated from that device. Although this seems rather simple and intuitive, it plays a key role in key establishment protocols between mobile devices. Pre-authentication is the process of identifying the devices to communicate, and exchanging information over the LLC between them. Using that information, key establishment can occur.

A protocol for device authentication using these two concepts, shown in Table 1 using a protocol trace developed in [21], has served as the basis for most of the later work on the subject [1]. Using this protocol and an LLC, devices exchange information - specifically, the network addresses of the devices and the hash values of the devices' public keys. Then, using the network addresses exchanged, the devices establish communication over the normal channel and request the others' public key. The key received is hashed and compared to the value received via the LLC. If the values match, then the public key is authenticated.

Many works build upon the foundations established by [1], including approaches based on visual channels proposed in [7, 8, 15, 19]. While most approaches use two-way LLC transmission, some of them achieve device authentication using only one LLC transmission [9, 19, 21].

#	Ch	Alice	Bob
1	LL	$-Addr_{Alice}, H\{K_{Alice}\} \rightarrow$	
2	LL	$\leftarrow Addr_{Bob}, H\{K_{Bob}\}-$	
3	RF	$\leftarrow$ Key exchange protocol $\rightarrow$	

Table 1: Basic Protocol for Exchanging Keying Information via a Location-Limited Channel (Ch: Communication Channel, RF: Radio Frequency, LL: Location-Limited)

2.2 BAN Logic

We can turn to formal methods as a tool in evaluating the soundness of proposed authentication protocols. An approach to formalizing the logic associated with authentication, called "BAN Logic," is presented in [2]. This work presented a logic of *belief*, which is basically the process of coupling assumptions about current states with a simple set of steps and reasoning, to arrive at a conclusion regarding the soundness of an authentication protocol. It has been used extensively to identify flaws in new and existing security protocols, and is often employed by authors to prove new proposals sound.

BAN logic is not without limitation, however. Various issues, most often a result of ambiguous assumptions [12, 13], have been identified with the original BAN logic. However, by noting and carefully addressing these concerns, the original BAN logic continues to be used by authors to verify authentication protocols, generally due to the simplicity of the approach [6, 14, 17, 18, 20].

Constructs.

We must first cover several basic components of BAN logic, including its constructs and postulates. The constructs are represented as follows [2]:

$A \mid\equiv X$: A **believes** X. A may act as though X is true.

$A \triangleleft X$: A **sees** X. A has received a message containing X.

$A \mid\sim X$: A **said** X. A once said X, though when X was said is unknown.

$A \Rightarrow X$: A **controls** X (A **has jurisdiction over** X). A is an authority on X and should be trusted to provide a correct X.

$\sharp(X)$: X is **fresh**. X has not been sent in any previous message during the current protocol run. X is typically a *nonce*.

$A \overset{K}{\leftrightarrow} B$: K is a shared key A and B may use to communicate

$\overset{K}{\mapsto} A$: K is the public key of A, with a matching secret key K^{-1} that remains secret to A or any principal trusted by A.

$A \overset{X}{\rightleftharpoons} B$: X is a *secret* formula, such as a password, known only to A and B.

$\{X\}_K$: The formula X encrypted under K.

$\langle X \rangle_Y$: X combined with formula Y. Usually, Y is a secret, and its presence proves the identity of whoever utters $\langle X \rangle_Y$.

Postulates.

The postulates which manipulate these constructs are as follows:

Message-Meaning Rule If A believes K is the shared key with B, and A sees X encrypted under K, then A believes B once said X:

$$\frac{A \textbf{ believes } A \overset{K}{\leftrightarrow} B,\ A \textbf{ sees} \{X\}_K}{A \textbf{ believes } B \textbf{ said } X}$$

Or, written using the construct notation (as all future postulates will be):

$$\frac{A \models A \overset{K}{\leftrightarrow} B,\ A \triangleleft \{X\}_K}{A \models B \mid\!\sim X}$$

Nonce-Verification Rule This rule checks to see if the message was sent recently, that is, the sender still believes in the message. If A believes X is fresh, and A believes B once said X, then A believes B believes X.

$$\frac{A \models \sharp(X),\ A \models B \mid\!\sim X}{A \models B \models X}$$

Jurisdiction Rule If A believes that B has jurisdiction over X, and A believes B believes X, then A trusts B on the truth of X, thus A believes X:

$$\frac{A \models B \Rightarrow X,\ A \models B \models X}{A \models X}$$

3. OUR APPROACH

First, we must establish the conditions by which the problem of device authentication in ubiquitous computing is bound. We assume the only channel of communication is a wireless broadcast channel (RF), available to all devices, including those of an attacker. We assume no a priori knowledge exists between the two devices wishing to communicate securely. Furthermore, we assume the existence of an active attacker, who is able to intercept and modify the contents of RF transmissions between our two devices - this is commonly referred to as the "man in the middle."

We will use BAN logic to analyze a basic question of authentication protocols for mobile computing applications: is authentication possible without the use of a side-channel (LLC, human interaction, etc.)? In considering this question, we make the following claim: *If* a protocol exists that allows device authentication without use of LLCs (i.e. using RF communication only), *then* it can be verified to be correct using BAN logic.

It is important to note that we assume both parties participating in the authentication protocol are trustworthy. That is, neither device knowingly provides incorrect information. This is consistent with [2], which does not deal with authentication of untrustworthy principals, stating "We focus on the beliefs of trustworthy parties involved in the protocols and on the evolution of these beliefs as a consequence of communication." Other security protocol analysis works also make this assumption [5,10,11,16].

3.1 Analyzing Authentication Between Unknown Mobile Devices

Let us consider the following definitions regarding our environment and goals:

DEFINITION 1. *A device authentication protocol is a procedure for identifying the identity of another device on a network.*

DEFINITION 2. *Ad-hoc computing environments exhibit the following characteristics:*

- *No pre-shared, or a priori, knowledge exists between devices*
- *Device location is not fixed, nor is device proximity assumed*
- *Only a broadcast method of communication is used for normal data transfer*

We must establish these definitions for several reasons. First, it is important to be very clear about our definition of "ad-hoc computing," as interpretations have been proposed which differ from ours [4]. Next, we must remove any ambiguity regarding the meaning of "device authentication" protocols. It is essential that we be clear about these definitions in order to proceed with the following proposition regarding device authentication in ad-hoc computing:

PROPOSITION 3.1. *Key-based device authentication between two previously unknown mobile devices in an ad-hoc computing environment is not possible using only a single broadcast communication channel.*

PROOF. We will prove this proposition by contradiction. To do so, we will assume that there *is* a protocol for device authentication between two previously unknown devices in an ad-hoc environment that does *not* use additional information, such as that provided by a demonstrative side-channel, and attempt to prove this assumption correct.

Let us first establish the goals of device authentication. The goals of authentication, according to [2], can vary, but generally consist of a shared session key between two entities, that is:

$$A \models A \overset{K}{\leftrightarrow} B$$
$$B \models A \overset{K}{\leftrightarrow} B$$

Because we assume no a priori knowledge between devices, the shared key $A \overset{K}{\leftrightarrow} B$ must be established via the authentication protocol between A and B, it cannot be assumed to exist beforehand. There are two possible cases for authentication protocols that are key-based: those that use symmetric keys and those that use asymmetric keys.

Case 1: Symmetric Keys.

Device authentication protocols using symmetric keys, by the very definition of symmetric keys, assume that information is already shared between both devices - specifically a shared symmetric key. It is trivial to show, because this violates the constraints we have established for key-based device authentication in mobile computing, that this family of protocols is not possible.

Case 2: Asymmetric Keys.

For device authentication protocols utilizing asymmetric keys, it is required that each device *know* the public key of the other device. Doing so requires that each device, at some point during the mutual device authentication protocol, *receives* the public key of the other device. To put it formally, at some point, the following must occur:

$$A \rightarrow B : \overset{K_A}{\mapsto} A$$
$$B \rightarrow A : \overset{K_B}{\mapsto} B$$

However, simply receiving a public key does not assure a device that it is the authentic public key of the intended target device. Therefore, the goal in this protocol is more stringent, specifically, it is assurance the public key of the sending device is authentic, or to put it formally:

$$A \mid\equiv \overset{K_b}{\mapsto} B$$
$$B \mid\equiv \overset{K_a}{\mapsto} A$$

Note that both components are required for mutual device authentication to occur. We will therefore begin with an analysis of the first component, $A \mid\equiv \overset{K_b}{\mapsto} B$.

Case 2.1: Authenticating B to A..

Typically, when proving a protocol using BAN logic, assumptions are established first. We assume the following for our hypothetical device authentication algorithm:

$$\begin{array}{ll}
A \mid\equiv \overset{K_a}{\mapsto} A & B \mid\equiv \overset{K_b}{\mapsto} B \\
A \mid\equiv A \mid\Rightarrow \overset{K_a}{\mapsto} A & B \mid\equiv B \mid\Rightarrow \overset{K_b}{\mapsto} B \\
A \mid\equiv B \mid\Rightarrow \overset{K_b}{\mapsto} B & B \mid\equiv A \mid\Rightarrow \overset{K_a}{\mapsto} A \\
A \mid\equiv \sharp(\overset{K_a}{\mapsto} A) & B \mid\equiv \sharp(\overset{K_b}{\mapsto} B) \\
A \mid\equiv \sharp(\overset{K_b}{\mapsto} B) & B \mid\equiv \sharp(\overset{K_a}{\mapsto} A) \\
\neg(A \mid\equiv \overset{K_b}{\mapsto} B) & \neg(B \mid\equiv \overset{K_a}{\mapsto} A)
\end{array}$$

Most of these assumptions are obvious. However, the assumptions in the last row require a brief explanation. Normally, we only state what we assume to be true, rather than what we assume is *not* true. However, in this case, these statements of negation are necessary because of the ad-hoc computing environment. It is not enough to omit the assumption that we believe a specific public key belongs another device, we must explicitly state that we do *not* believe a specific public key belongs to another device, because an ad-hoc environment explicitly excludes this type of prior knowledge.

Working backwards from our goal, by the jurisdiction rule, we see that for $A \mid\equiv \overset{K_b}{\mapsto} B$ to be true, we must establish $A \mid\equiv B \mid\Rightarrow \overset{K_b}{\mapsto} B$ and $A \mid\equiv B \mid\equiv \overset{K_b}{\mapsto} B$. We assume $A \mid\equiv B \mid\Rightarrow \overset{K_b}{\mapsto} B$, that is, B has control of its own public key. To establish the second part of this rule, $A \mid\equiv B \mid\equiv \overset{K_b}{\mapsto} B$, we must establish $A \mid\equiv \sharp(\overset{K_b}{\mapsto} B)$ and $A \mid\equiv B \mid\sim \overset{K_b}{\mapsto} B$.

Considering our assumptions, we see that only $A \mid\equiv \sharp(\overset{K_b}{\mapsto} B)$ is shown. To establish $A \mid\equiv B \mid\equiv \overset{K_b}{\mapsto} B$, we would need to show that $A \mid\equiv B \mid\sim \overset{K_b}{\mapsto} B$. This is achieved either by assumption (which we do not have) or by the message-meaning rule, which stipulates $A \mid\equiv \overset{K_b}{\mapsto} B$ and $A \triangleleft \{X\}_{K_b^{-1}}$ must be true for $A \mid\equiv B \mid\sim \overset{K_b}{\mapsto} B$. However, this contradicts our assumption $\neg(A \mid\equiv \overset{K_b}{\mapsto} B)$, and therefore, Case 2.1 is not possible.

Case 2.2: Authenticating A to B..

By symmetry, Case 2.2 is also not possible.

Final Steps.

By analyzing the necessary steps for key-based device authentication protocols between two previously unknown mobile devices, we have demonstrated that such protocols under the constraints of ad-hoc computing are not possible using only a single broadcast communication channel. Because our example is representative of any key-based device authentication protocol operating under the constraints of ad-hoc computing, and we have shown that such an protocol is not possible, we have successfully proven our proposition. □

In essence, what BAN logic tells us about the constraints of ad-hoc computing is that it is necessary to have some sort of basis to believe a message was sent from a particular device. Because we use a broadcast method of communication, we cannot make any statements about the origin of a particular message simply because we suppose it came from a specific device, or the message claims to have originated from a specific device. To *prove* that a message originated from a specific device, BAN logic tells us that the message must have been encrypted (or otherwise encoded) using a key (or secret) that A believes is either shared with B, or belongs to B. There is no other way, using the BAN logic constructs and postulates we specified, to form a belief about the origin of a message.

3.2 Extending BAN Logic

If we cannot achieve device authentication using only a single broadcast channel, suppose data X is transmitted from B to A using a side-channel, or LLC. This allows the users of each device to identify the other device, establishing the origin of the data received (*demonstrative identification* [1].) This additional information exchanged supplies the conditions necessary to establish our goals. While the data can also be seen by an attacker, an attacker is not able to modify the data transmitted. The receiving device knows with certainty the information was said by the sending device.

Doing so allows us to conclude the following:

$$A \triangleleft X$$
$$A \mid\equiv B \mid\sim X$$

The last statement is of particular importance. Under the message-meaning rule and normal communication, to establish $A \mid\equiv B \mid\sim X$ would require us to establish $A \mid\equiv \overset{K_b}{\mapsto} B$ and $A \triangleleft \{X\}_{K_b^{-1}}$. However, since we are using another method which allows A to verify the origin of X, we can establish $A \mid\equiv B \mid\sim X$ without using the message-meaning rule.

This is an extension to BAN logic afforded by the properties of LLCs, and to assist in our proofs of device authentication protocols, we will denote this extension using the following construct:

$A \overset{X}{\dashleftarrow} B$: A receives X from B via a channel by which A can verify the origin of X is B.

The following postulate will also be added:

Side-Channel-Communication (SCC) Rule If A receives X from B via a side-channel, then A sees X, and A is entitled to believe B said X:

$$\frac{A \overset{X}{\dashleftarrow} B}{A \triangleleft X, A \mid\equiv B \mid\sim X}$$

3.3 Handling Hash Functions

The device authentication protocol we seek to prove correct involves the transmission of hash values via the side-channel. However, this raises serious issues; specifically, how do we treat hash values with respect to the constructs and postulates of BAN Logic? The authors of BAN logic, in an extended technical report [3], propose a postulate for handling a hash function H, as follows:

$$\frac{A \mid\equiv B \mid\sim H(X), A \triangleleft X}{A \mid\equiv B \mid\sim X}$$

However, it is clear from their work that $H(X)$ refers to signed hashes, not an arbitrary hash function, as the authors state, "If H is an arbitrary function, nothing convinces one that when A has uttered $H(m)$ he must have also uttered m. In fact, A may never have seen m."

How then are we to deal with the use of arbitrary hash functions in our protocols? According to [3] we cannot assume that because B said $H(X)$, B also said X. However, we can deduce the following: if A believes $H(X)$, and sees some message Y (not necessarily from B), and confirms that $H(Y)$ equals $H(X)$, then A can reasonably assume that Y equals X, and can say that A sees X[1]. However, what we really need to determine is: did B *say* X?

To help with this determination, we must first reveal an implicit assumption within the constructs of BAN logic, specifically, the notion of *uniqueness*. In all postulates involving a key K, it is assumed K is unique. For instance, we assume $\overset{K}{\mapsto} A \neq \overset{K}{\mapsto} B$. In other words, without the assurance that K is unique, the message-meaning rule would not be true, if it were possible that $\overset{K}{\mapsto} A = \overset{K}{\mapsto} B$. Uniqueness of keys is a safe assumption to make in the case of sufficiently large cryptographic keys, and for secret data that is generated prior to protocol execution.

However, in the case of hash values, it is not safe to say that all values generated are unique. Virtually any cryptographic hash function will generate an infinite number of collisions for an infinite input set. Additionally, two entities hashing the same value, using the same hash function, will return identical hashes. Consider the case where A generates the hash value of the word "cat," and B also generates the value of the word "cat". If they both use the same hashing function, they will both return the same value, and it will be impossible to determine which entity performed the hash. This is the problem stated by [3] above.

However, if we make a stronger assumption about the value being hashed, that is, that the X which is used to generate $H(X)$ is unique, then we can also assume that $H(X)$ is unique. We will use this belief in comparing values and their hashes during our communication protocol. We can say that if we have a hash value $H(X)$, and we receive another value Y, that we may determine $Y = X$ if we can show $H(Y) = H(X)$, as long as we believe X is unique.

Combining the notion of uniqueness with the BAN logic construct of jurisdiction, we may conclude the following. If A believes B is an authority on X, and A believes X is unique, then A may believe B created X. More importantly, A may conclude that no other entity could have independently created X. So, once A establishes that it sees X (realizing that A has no foreknowledge of the value of X, so it is still incumbent on the protocol to establish the value of X, as well as its origin in B), A may conclude that at some point in time, X was said by B.

To show that we consider a value to be unique, we propose the following additional construct:
$\dagger(X)$: X is considered to be sufficiently unique that two entities independently generating the same X is considered computationally unlikely.
The applications of this construct extend beyond hash functions. For instance, uniqueness could be used to describe newly generated symmetric or asymmetric keys. If A and B independently generate K_A and K_B, they should be unique.

[1]Although we recognize that collisions exist in hash functions, we assume the use of strong cryptographic hash functions for this step. Assuming an attacker is unable to produce a collision (i.e. generate $Y \neq X$ such that $H(Y) = H(X)$) in this case is similar to assuming an attacker is unable to decrypt encrypted messages in other BAN logic postulates

#	Ch	Alice	Bob
1	LL	$-Addr_A, H(K_A) \rightarrow$	
2	LL		$\leftarrow Addr_B, H(K_B)-$
3	RF	$-K_A \rightarrow$	
4	RF		$\leftarrow K_B-$
5		Calc $H(K_B)_{\#4}$	Calc $H(K_A)_{\#3}$
6		$H(K_B)_{\#4} \overset{?}{=} H(K_B)_{\#2}$	$H(K_A)_{\#3} \overset{?}{=} H(K_A)_{\#1}$

Table 2: Simple Protocol for Authenticating Public Keys via a Location-Limited Channel

Now we can address the question of establishing which entity *said* X, based on our beliefs about X and $H(X)$. We do this by adding some constraints to the hashing postulate proposed by [3], making it correct for arbitrary hash functions. Our proposed addition is as follows:

Hash Analysis Rule If A believes B controls X, and if A believes X is unique, and if A believes B said $H(X)$, and A sees X, then A believes that B said X.[2]

$$\frac{A \mid\equiv B \mid\Rightarrow X,\ A \mid\equiv \dagger(X),\ A \mid\equiv B \mid\sim H(X),\ A \triangleleft X}{A \mid\equiv B \mid\sim X}$$

Next, we consider why each component in this rule is necessary:

$A \mid\equiv B \mid\Rightarrow X$: A has to believe B controls the value X, in our case, B's public key.

$A \mid\equiv \dagger(X)$: As mentioned above, for this to hold, X cannot be independently generated anywhere else (within computational allowances); combined with the previous component, this means A believes only B created X.

$A \mid\equiv B \mid\sim H(X)$: This is necessary to verify $A \triangleleft X$. Otherwise, A has no way to know whether or not the message it sees is actually X. Despite the assumptions above, A does not initially know the value of X, **but** A does know the value of $H(X)$.

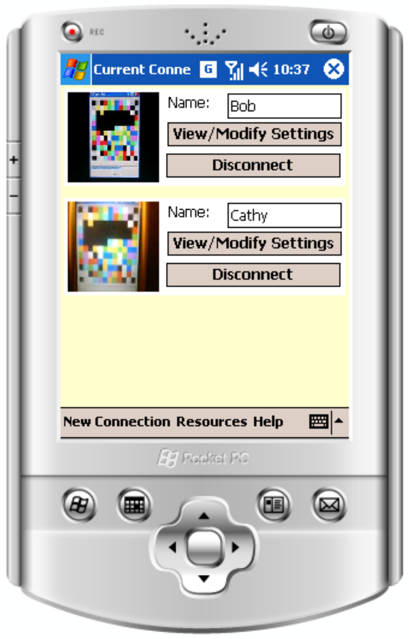
$A \triangleleft X$: This is actually determined using another component in the same rule. A cannot know it has seen X until it compares $H(X)$ to the value contained in $A \mid\equiv B \mid\sim H(X)$.

4. APPLICATIONS

We will demonstrate the application of the SCC and Hash Analysis Rules by applying them to the formal analysis of authentication protocols in two mobile computing applications. The first, shown in Figure 1, uses a basic bi-directional protocol based on [1]. The second uses a uni-directional protocol requiring only one LLC transmission.

4.1 Proving the Basic Device Authentication Protocol Secure

[2]For simplicity we assume the following trivial rule:
$$\frac{B \mid\Rightarrow X}{B \mid\Rightarrow H(X)}$$



Name: Bob

Name: Cathy

Figure 1: Application Using LLCs for Device Authentication [8]

The basic mobile device key authentication protocol is shown in Table 2. This protocol is used by applications such as the one shown in Figure 1; in this case, colorized two-dimensional barcodes are exchanged via device displays and digital cameras. The first step in analyzing the protocol is to *idealize* it, that is, to change it into a form more easily analyzed by the logic. (See [2] for more details on this process.) The idealized form of the protocol is as follows:

Message 1	$: A \rightarrow B : H(\overset{K_A}{\mapsto} A)$
Message 2	$: B \rightarrow A : H(\overset{K_B}{\mapsto} B)$
Message 3	$: A \rightarrow B : \overset{K_A}{\mapsto} A$
Message 4	$: B \rightarrow A : \overset{K_B}{\mapsto} B$

The assumptions we make are as follows:

$A \mid\equiv \overset{K_A}{\mapsto} A$	$B \mid\equiv \overset{K_B}{\mapsto} B$
$A \mid\equiv A \Rightarrow \overset{K_A}{\mapsto} A$	$B \mid\equiv B \Rightarrow \overset{K_B}{\mapsto} B$
$A \mid\equiv \sharp(\overset{K_B}{\mapsto} B)$	$B \mid\equiv \sharp(\overset{K_A}{\mapsto} A)$
$A \mid\equiv \sharp(H(\overset{K_B}{\mapsto} B))$	$B \mid\equiv \sharp(H(\overset{K_A}{\mapsto} A))$
$A \mid\equiv B \Rightarrow \overset{K_B}{\mapsto} B$	$B \mid\equiv A \Rightarrow \overset{K_A}{\mapsto} A$
$A \mid\equiv B \Rightarrow H(\overset{K_B}{\mapsto} B)$	$B \mid\equiv A \Rightarrow H(\overset{K_A}{\mapsto} A)$
$A \mid\equiv \dagger(\overset{K_B}{\mapsto})$	$B \mid\equiv \dagger(\overset{K_A}{\mapsto})$

Why is $\sharp(\overset{K_A}{\mapsto} A), \sharp(H(\overset{K_A}{\mapsto} A)), \sharp(\overset{K_B}{\mapsto} B), \sharp(H(\overset{K_B}{\mapsto} B))$ assumed in this protocol? This may seem to be a poor assumption to make. The notion of *freshness* is BAN logic is meant to assure that the messages being exchanged are recently generated (during the current run of the protocol), and not being replayed by an attacker. It could be argued that the properties of side-channels prevent a replay attack of the data transmitted (A knows that the information came immediately from B, therefore, it is fresh). We use this reasoning in our assumptions above. However, a more complete notion of freshness would be assured if some nonce were included in the side-channel data, or if the side-channel data were generated uniquely for each instance of communication [7]. This would assure the sending device, B, that subsequent protocol messages from A were, in fact, fresh.

Note that both A and B trust the other on the correctness of their respective public keys. This is a staple of BAN logic, we assume our counterpart is trustworthy, that is, it has not been compromised to provide incorrect information. We do not, however, assume that anything purported to be received by A from B is actually from B; deciding that is the purpose of the message-meaning rule.

The main steps of our proof are as follows:

B receives message 1. The SCC rule yields that:

$$B \triangleleft H(\overset{K_A}{\mapsto} A), B \mid\equiv A \mid\sim H(\overset{K_A}{\mapsto} A)$$

Likewise, receiving message 2 yields for A:

$$A \triangleleft H(\overset{K_B}{\mapsto} B), A \mid\equiv B \mid\sim H(\overset{K_B}{\mapsto} B)$$

Since we have the assumptions

$$A \mid\equiv \sharp(H(\overset{K_B}{\mapsto} B)) \text{ and } B \mid\equiv \sharp(H(\overset{K_A}{\mapsto} A))$$

We can apply the nonce-verification rule to get

$$A \mid\equiv B \mid\equiv H(\overset{K_B}{\mapsto} B) \text{ and } B \mid\equiv A \mid\equiv H(\overset{K_A}{\mapsto} A)$$

Recall we also assume

$$A \mid\equiv B \Rightarrow H(\overset{K_B}{\mapsto} B) \text{ and } B \mid\equiv A \Rightarrow H(\overset{K_A}{\mapsto} A)$$

Next, we apply the jurisdiction rule to get

$$A \mid\equiv H(\overset{K_B}{\mapsto} B) \text{ and } B \mid\equiv H(\overset{K_A}{\mapsto} A)$$

Next, we consider messages 3 and 4, where we find A and B receive messages purporting to be K_B and K_A, respectively. However, since these messages do not come from known sources, we will call them Y and Z for now.

$$A \triangleleft Y \text{ and } B \triangleleft Z$$

We use our hash analysis reasoning to check that $H(Y)$ equals $H(K_B)$ and $H(Z)$ equals $H(K_A)$. (This is step 5 and 6 in the protocol listed in Table 2.) Because they match, we now assume

$$A \triangleleft \overset{K_B}{\mapsto} B \text{ and } B \triangleleft \overset{K_A}{\mapsto} A$$

Next, we recall our assumptions

$$A \mid\equiv B \Rightarrow \overset{K_B}{\mapsto} B \text{ and } B \mid\equiv A \Rightarrow \overset{K_A}{\mapsto} A \text{ and}$$
$$A \mid\equiv \dagger(\overset{K_B}{\mapsto}) \text{ and } B \mid\equiv \dagger(\overset{K_A}{\mapsto})$$

We couple those assumptions with the previously determined statements and apply them to our Hash Analysis Rule to achieve

$$A \mid\equiv B \mid\sim \overset{K_B}{\mapsto} B \text{ and } B \mid\equiv A \mid\sim \overset{K_A}{\mapsto} A$$

Using the assumptions

$$A \mid\equiv \sharp(\overset{K_B}{\mapsto} B) \text{ and } B \mid\equiv \sharp(\overset{K_A}{\mapsto} A)$$

We can apply the nonce-verification rule to get

$$A \mid\equiv B \mid\equiv \overset{K_B}{\mapsto} B \text{ and } B \mid\equiv A \mid\equiv \overset{K_A}{\mapsto} A$$

Next, we apply the jurisdiction rule to get

$$A \mid\equiv \overset{K_B}{\mapsto} B \text{ and } B \mid\equiv \overset{K_A}{\mapsto} A$$

Which is the goal of our protocol.

4.2 One-way LLC for device authentication

In [9], a simplified the key establishment protocol, based on [21], is presented. This protocol is shown in Table 3. The simplified protocol begins with both Alice and Bob selecting new public keys, g^a and g^b. Alice sends her public key, g^a, to

Table 3: UbiSound Key Establishment Protocol [9] (RF: Radio Frequency Channel, LL: Location-Limited Channel, PB: Manual User Interaction (Push-Button))

#	Ch	Alice		Bob
1		Chooses g^a		
2	RF		$-g^a \rightarrow$	
3				Chooses g^b
4				Chooses random R_b
5				$H_b = H(g^a \mid g^b \mid R_b)$
6	RF		$\leftarrow H_b -$	
7	LL		$\leftarrow R_b -$	
8	RF		$\leftarrow g^b -$	
9		$H'_b =$		
		$H(g^a \mid g^b \mid R_b)$		
10		Verifies		
		$H_b = H'_b$		
11	PB		$-verify \rightarrow$	

Bob using the unsecured wireless channel. Bob then chooses a random number, R_b, of sufficient size to prevent guessing by Marvin, the adversary. Next, Bob calculates a hash value, H_b, as the concatenation of g^a, g^b, and R_b, and sends H_b to Alice using the unsecured channel.

The next step involves the LLC. R_b is encoded and transmitted over this channel, followed by Bob's public key, g^b, which is sent over the unsecured wireless channel. After receiving g^b, Alice has all the information she needs to calculate $H'_b = H(g^a|g^b|R_b)$, and verify that $H_b = H'_b$. Because Alice can verify that R_b came from Bob, using the demonstrative identification [1] of the LLC, she can verify that H_b was also generated by Bob. Assuming that an adversary has not compromised Bob's device, this confirms to Alice that the information she received from Bob is authentic, verifying his device, his public key g^b, and allowing key establishment to commence.

How does Bob establish that he is communicating with Alice, though? This question is addressed in [21], and the answer is reasoned as follows. Bob does not receive any communication from Alice via an LLC, which may lead to the conclusion that Bob cannot demonstratively identify Alice's device. While this would be true for completely automated devices, we have the advantage of user interaction to complete the protocol. When Alice verifies she is communicating with Bob, she implicitly verifies to Bob that H_b is correct. Because H_b contains g^a, Alice's verification to Bob also confirms to Bob that he has used the correct values in calculating H_b, and those values can be trusted to establish a secure channel. Even if Bob were a kiosk device, he could receive confirmation from Alice via a push-button device, which only Alice would be able to press. ([21] points out that it would take an extremely sophisticated attacker to develop a button-pushing device that could not be detected by the kiosk user.)

4.2.1 *Proving the Protocol Correct*

We will use BAN logic to analyze the correctness of this protocol. To do so, we first establish the following assumptions:

$A \mid\equiv g^a$	$B \mid\equiv g^b$
$A \mid\equiv A \Rightarrow g^a$	$B \mid\equiv B \Rightarrow g^b$
$A \mid\equiv B \Rightarrow g^b$	$B \mid\equiv A \Rightarrow g^a$
$A \mid\equiv \sharp(R_b)$	$B \mid\equiv \sharp(R_b)$
$A \mid\equiv \dagger(g^a)$	$B \mid\equiv \dagger(g^b)$
$A \mid\equiv \dagger(g^b)$	$B \mid\equiv \dagger(g^a)$
$A \mid\equiv \dagger(R_b)$	$A \mid\equiv B \Rightarrow R_b$

Next, we idealize the protocol. According to [2], idealization of protocols involves removing any messages not encrypted. Doing so would leave us with the following idealized protocol:

Message 1 : $B \rightarrow A : H(g^a, g^b, R_b)$

While this is technically correct, according to the BAN logic analysis process, we must point out that we require additional information to be transmitted for the Hash verification step of the protocol. [2] argues that cleartext messages can be forged. However the hashing element of the protocol prevents cleartext messages, such as g^a and g^b, from being forged. Effectively, the hash component verifies the correctness of these values, and therefore they should be included in the protocol idealization. We also add the fifth message, the manual user interaction step "verify" between A and B, because this step also serves to validate information sent in cleartext during the protocol.

Message 1 : $A \rightarrow B : g^a$
Message 2 : $B \rightarrow A : H(g^a, g^b, R_b)$
Message 3 : $B \rightarrow A : R_b$
Message 4 : $B \rightarrow A : g^b$
Message 5 : $A \rightarrow B :$ *verification*

Now we can prove the correctness of the protocol, the goal of which is for each device to believe the public key of the other device, so secure key establishment may occur. Formally, this is:

$$A \mid\equiv g^b \text{ and } B \mid\equiv g^a$$

We begin by proving $A \mid\equiv g^b$. In message 2, we see $H(g^a, g^b, R_b)$ sent via LLC from B to A. By the side channel communication (SCC) rule, we obtain:

$$A \triangleleft H(g^a, g^b, R_b) \text{ and } A \mid\equiv B \mid\sim H(g^a, g^b, R_b)$$

Next, we apply the hash analysis rule. Because we assume $A \mid\equiv B \Rightarrow g^b$, $A \mid\equiv B \Rightarrow R_b$, $A \mid\equiv \dagger(g^b)$, $A \mid\equiv \dagger(g^a)$, and $A \mid\equiv \dagger(R_b)$, and we have deduced that $A \mid\equiv B \mid\sim H(g^a, g^b, R_b)$, knowing that $A \mid\equiv A \Rightarrow g^a$, we combine these with message 4, $A \triangleleft g^b$ to get:

$$A \mid\equiv B \mid\sim g^b$$

Next, we consider another component of BAN logic not yet mentioned, a freshness rule that states: if A believes one part of a formula is fresh, then the entire formula must also be fresh [2]. This is shown as follows:

$$\frac{P \mid\equiv \sharp(X)}{P \mid\equiv \sharp(X, Y)}$$

Using this rule, we can establish that since we assume $\sharp(R_b)$, we can deduce:

$$\sharp(g^a)$$
$$\sharp(g^b)$$

Knowing that $\sharp(g^b)$ and $A \mid\equiv B \mid\sim g^b$, we can apply the nonce-verification rule to obtain:

$$A \mid\equiv B \mid\equiv g^b$$

And because we assume $A \mid\equiv B \Rightarrow g^b$, we can use the jurisdiction rule to obtain the desired result:

$$A \mid\equiv g^b$$

This shows the first half of the protocol results. What about $B \mid\equiv g^a$? Actually, A establishes this for B. Consider what happens during step 10 of the protocol (shown in Table 3). A verifies that the information from B is correct, according to what A sent and received using the LLC. So message 5 of the idealized protocol verifies for B that it has the correct information, allowing B to conclude the g^a it received in message 1 was actually said by A ($A \mid\sim g^a$). This also provides the belief in a fresh g^a for B, since A must have established $\sharp(g^b)$ prior to the verification step (thus implying $\sharp(g^a)$ by the freshness rule). We can apply these beliefs to the nonce-verification rule and jurisdiction rule to obtain the second half of the desired result:

$$B \mid\equiv g^a$$

5. CONCLUSION AND FUTURE WORK

We have shown that device authentication in mobile computing applications in ad-hoc computing environments is not possible using only a single broadcast communication channel. However, using LLCs, we can achieve device authentication. Proving such a protocol correct requires extensions of BAN logic, which we proposed and demonstrated by analyzing two different device authentication applications. Future work will include more detailed analysis of existing solutions, and the potential generation of new device authentication protocols for mobile computing applications.

Acknowledgement

This work was partially supported at the Secure Computing Laboratory at New Mexico Tech by the grant from the National Science Foundation (NSF-IIS-0916875)

6. REFERENCES

[1] D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *In Symposium on Network and Distributed Systems Security (NDSS Š02)*, 2002.

[2] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. *ACM Trans. Comput. Syst.*, 8(1):18–36, 1990.

[3] M. Burrows, M. Abadi, and R. Needham. A logic of authentication. Technical Report SRC Research Report 39 - Revised, Digital Equipment Corporation Systems Research Center, February 1990.

[4] S. Capkun, M. Cagalj, R. Rengaswamy, I. Tsigkogiannis, J.-P. Hubaux, and M. Srivastava. Integrity codes: Message integrity protection and authentication over insecure channels. *IEEE Trans. Dependable Secur. Comput.*, 5(4):208–223, 2008.

[5] I. Cervesato, C. Meadows, and D. Pavlovic. An encapsulated authentication logic for reasoning about key distribution protocols. In *CSFW '05: Proceedings of the 18th IEEE Workshop on Computer Security Foundations*, pages 48–61, Washington, DC, USA, 2005. IEEE Computer Society.

[6] C.-W. Chang, H. Pan, and H.-Y. Jia. A secure short message communication protocol. *International Journal of Automation and Computing*, 5:202–207, 2008.

[7] W. Claycomb and D. Shin. Using a two-dimensional colorized barcode solution for authentication in pervasive computing. In *Proceedings of the IEEE Int'l Conf. on Pervasive Services 2006*, June 2006.

[8] W. Claycomb and D. Shin. Towards secure resource sharing for impromptu collaboration in pervasive computing. In *Proceedings of the 22nd ACM Symposium on Applied Computing*, March 2007.

[9] W. Claycomb and D. Shin. Secure device pairing using audio. In *Proceedings of the 43rd IEEE Int'l Carnahan Conference on Security Tech.*, Oct 2009.

[10] A. Datta, A. Derek, J. C. Mitchell, and D. Pavlovic. A derivation system and compositional logic for security protocols. *J. Comput. Secur.*, 13(3):423–482, 2005.

[11] A. Datta, A. Derek, J. C. Mitchell, and A. Roy. Protocol composition logic (pcl). *Electron. Notes Theor. Comput. Sci.*, 172:311–358, 2007.

[12] L. Gong, R. Needham, and R. Yahalom. Reasoning about belief in cryptographic protocols. In *Proc. 1990 IEEE Symposium on Research in Security and Privacy*, pages 234–248. IEEE Comp. Soc. Press, 1990.

[13] R. Kailar, V. D. Gligor, and L. Gong. On the security effectiveness of cryptographic protocols. In *In Proceedings of the 4th IFIP Working Conference on Dependable Computing for Critical Applications*, pages 139–157, 1994.

[14] H. Li, S. Guo, K. Zheng, Z. Chen, J. Cui, and X. Wu. Improved adoptable scheme for authentication and key agreement. In *Proceedings of the International Conference on Management and Service Science*, Sept 2009.

[15] J. McCune, A. Perrig, and M. Reiter. Seeing-is-believing: using camera phones for human-verifiable authentication. In *The 2005 IEEE Symposium on Security and Privacy*, May 2005.

[16] C. Meadows and D. Pavlovic. Deriving, attacking and defending the gdoi protocol. In *Computer Security - ESORICS 2004, 9th European Symposium on Research Computer Security*, volume 3193, pages 53–72. Lecture Notes in Computer Science, 2004.

[17] S.-S. Park, J.-H. Lee, and T.-M. Chung. Authentication analysis based on certificate for proxy mobile ipv6 environment. In *Proceedings of Computational Science and Its Applications*, 2009.

[18] C. Qingling, Z. Yiju, and W. Yonghua. A minimalist mutual authentication protocol for rfid system and ban logic analysis. In *Proceedings of ISECS International Colloquium on Computing, Communication, Control, and Management*, volume 2, pages 449 –453, aug. 2008.

[19] N. Saxena, J.-E. Ekberg, K. Kostiainen, and N. Asokan. Secure device pairing based on a visual channel (short paper). In *SP '06: Proceedings of the 2006 IEEE Symposium on Security and Privacy*, pages 306–313. IEEE Computer Society, 2006.

[20] J. M. Sierra, J. C. Hernandez, A. Alcaide, and J. Torres. Validating the use of ban logic. In *Proceedings of the Internet Communication Security Workshop*, volume 3043 of *Lecture Notes in Computer Science*, pages 851–858. Springer, April 2004.

[21] F.-L. Wong and F. Stajano. Multi-channel protocols. In *Int'l Workshop in Security Protocols*, April 2005.