Security Informatics
a SpringerOpen Journal

RESEARCH Open Access

Belief manipulation and message meaning for protocol analysis

Aaron Hunter

Abstract

Agents often try to convince others to hold certain beliefs. In fact, many network security attacks can actually be framed in terms of a dishonest that is trying to get an honest agent to believe some particular, untrue claims. While the study of belief change is an established area of research in Artificial Intelligence, there has been comparatively little exploration of the way one agent can explicitly manipulate the beliefs of another. In this paper, we introduce a precise, formal notion of a belief manipulation problem. We also illustrate that the meaning of a message can be parsed into different communicative acts, as defined in discourse analysis theory. Specifically, we suggest that each message can be understood in terms of what it says about the *world*, what it says about the *message history*, and what it says about *future actions*. We demonstrate that this kind of dissection can actually be used to discover the goals of an intruder in a communication session, which is important when determining how an adversary is trying to manipulate the beliefs of an honest agent. This information will then help prevent future attacks. We frame the discussion of belief manipulation primarily in the context of cryptographic protocol analysis.

Keywords: Belief change; Deception; Goal discovery

Introduction

In message passing systems involving two-way communication between agents, it is often the case that the behaviour of one agent may lead to changes in the beliefs of another agent. In fact, it is frequently the case that one agent is explicitly interested in convincing another agent to hold some particular beliefs. This is the case, for example, in authentication protocols and online negotiation. In this paper, we are interested in formalizing the manner in which one agent may explicitly try to manipulate the beliefs of another through the exchange of a sequence of messages.

While the study of belief change operators has a long history, there has been comparatively little research exploring how these operators can be used to model the manner in which one agent can manipulate the beliefs of another. In this paper, we introduce a general definition of a *belief manipulation problem*, which can be applied accross a wide range of application domains. We illustrate that belief manipulation problems can be used to model a dishonest agent in protocol analysis, and we consider the dangers and implications for widely accessed networks.

In order to address the notion of belief manipulation, we need to be explicit about the way that an agent's beliefs change when a message is received. Messages exchanged over open communication lines are frequently encrypted in order to conceal the contents from malicious intruders. However, information hiding is just one goal that is achieved through encryption. Messages are frequently encrypted as part of a larger *cryptographic protocol* to achieve higher level goals such as commitment [1] or authentication [2]. The fact that encryption is critical to achieving higher level communicative goals is well known, but there has been little attempt to specify the precise semantic impact of encryption. We suggest that, in oder to understand the meaning of a message, it is often important to consider what the sending agent is trying to achieve. From this perspective, it can be useful to ask questions of the following form:

- Why has the sender encrypted the given message with the given key?
- Does the encryption tell us anything about the goals or intentions of the sender?

Correspondence: aaron_hunter@bcit.ca
British Columbia Institute of Technology, 3700 Willingdon Avenue, Burnaby, Canada

We use a logical framework to reason about these problems. First, following [3], we define a formal framework to represent the perspective of all parties involved in an exchange of messages. Second, we analyze the meaning of encrypted messages in terms of *speech act theory* [4,5]. Third, we consider how this information can be used to hypothesize about the goals of the sender, thereby improving security.

Motivation

Cryptographic protocols are sequences of encrypted messages that are exchanged to achieve communicative goals, such as authentication or fair exchange. Since many protocols involve goals that amount to “convincing” another agent to believe some claim, logics of knowledge and belief have been used extensively for the verification of cryptographic protocols [6-9]. In many cases, logics of knowledge are used together with a formal model of a multi-agent system and a precise definition of a message trace [10,11]. An agent can then be said to *know* some fact is true, if it is true in every possible trace. The Scyther tool provides an illustrative example of this approach to protocol verification [12]. For this tool, there are three possible outcomes of protocol verification. First, an attack may be discovered. Second, it might be determined that no attack exists. Third, it might be determined that no attack was found in the search space, but the search space was not complete.

An alternative approach to protocol analysis is to view the traces simply as vehicles to convince the participants that certain facts are true. This is the approach that we have taken in our previous work, where we have encoded the messages exchanged in a protocol in a variant of first-order logic [13]. This approach to protocol verification can be framed as a belief manipulation problem.

In general, a cryptographic protocol has the following structure.

Generic protocol

1.	send M_1
2.	receive N_1
$\vdots$	
2n-1.	send M_n
2n.	receive N_n

Suppose this is an authentication protocol for some agent A. For an intruder, the goal is to convince the honest agent B to believe that they are communicating with some agent $C \neq A$. In general, the way that this is accomplished is for A to *make B believe* that A has K_C, where K_C is some piece of secret information held by C.

For our purposes, what is important is that an attack on this kind of protocol is directly concerned with *manipulating the beliefs* of another participant. For example, one agent might want to convince another agent to believe that they hold a certain peice of secret information, or a particular encryption key.

The standard assumption when analyzing this kind of protocol is that a malicious agent is able to read, block and re-direct any message that is sent. Therefore communication is essentially *anonymous*, because the recipient of a message is never aware of the identity of the sender. Communication is also *unreliable* in the sense that there is no guarantee a sent message will ever be received.

We introduce a simple example, using standard notation from the protocol verification literature. In this tradition, A and B are used to denote agents and the notation $A \rightarrow B : M$ is used to express the fact that A sends B the message M. A message encrypted with key K is written $\{M\}_K$. Finally, we use N (possibily with subscripts) to denote a random number generated by an agent during the execution of a protocol. A random number generated in this manner is normally called a *nonce*, which is short for “number used once”.

Example The following describes a simple cryptographic protocol. The underlying assumption is that the key K is shared by A and B, but no other agents.

The challenge-response protocol
1. $A \rightarrow B : \{N\}_K$
2. $B \rightarrow A : N$

The goal of this protocol is to convince the agent A that the agent B is alive on the network. It turns out that this protocol is susceptible to a *mirror attack*, which we discuss later.

Note that the key K in the example is a *symmetric key*. While we focus primarily on protocols involving symmetric keys, our approach can be applied equally well in the setting of public key cryptography.

Protocol verification involves searching for attacks on protocols, or constructing proofs that no attacks exist. This has been studied extensively using a variety of methods, including formal logics [14,15], inductive theorem proving [16], planning formalisms [17] and process algebras [18]. While we are not directly concerned with proofs of correctness in this paper, we employ the notation and terminology from the protocol verification community in our discussion of encrypted communication.

Preliminaries

Belief revision

Let **F** be a set of propositional symbols, informally representing different properties of the world. A *state* is a propositional interpretation of **F**. A *belief set* is a finite set of formulas over **F**, representing the formulas that some particular agent believes to be true. We can assume without loss of generality that a belief set includes a single formula, due to the finiteness assumption.

In practice, the most important difference between belief and knowledge is that beliefs can be proved false through observations or experience. Formally, given some initial belief set ϕ, it is possible for an agent to obtain new information γ that conclusively demonstrates that the actual state is not consistent with ϕ. Resolving this kind of problem is the domain of *belief revision theory*. Roughly, an agent would like to adopt a new belief set that is "close" to the original belief set, while supporting the new information. The influential AGM approach to belief revision specifies a well-known set of postulates dictating the manner in which beliefs should be revised [19]. Briefly, a belief change operator $*$ is a function that takes two inputs: a formula ϕ representing the initial beliefs, and a formula γ representing some new peice of information. A belief change operator that satisfies the AGM postulates is called an *AGM belief revision operator*. The AGM postulates can be formulated as follows [20]:

[R1] $\phi * \gamma$ implies γ.
[R2] If $\phi \wedge \gamma$ is satisfiable, then $\phi * \gamma \equiv \phi \wedge \gamma$.
[R3] If γ is satisfiable, then $\phi * \gamma$ is satisfiable.
[R4] If $\phi_1 \equiv \phi_2$ and $\gamma_1 \equiv \gamma_2$, then $\phi_1 * \gamma_1 \equiv \phi_2 * \gamma_2$.
[R5] $(\phi * \gamma) \wedge \beta$ implies $\phi * (\gamma \wedge \beta)$.
[R6] If $(\phi * \gamma) \wedge \beta$ is satisfiable, then $\phi * (\gamma \wedge \beta)$ implies $(\phi * \gamma) \wedge \beta$.

Several semantic characterizations of these postulates have been given in terms of orderings over states [20-22]. Roughly, AGM belief change operators implicitly require a plausibility ordering over possible states; new information is incorporated by believing the "most plausible" states consistent with the new information.

Speech acts

In our analysis of encrypted communication, we will be interested in determining the precise meaning of each message sent during communication. Our analysis relies on the theory of *speech acts*, as developed by Austin [4] and Searle [5]. Briefly, the intuition behind the theory of speech acts is that a single utterance has several kinds of meaning. The *locutionary force* of an utterance is the direct meaning of the utterance, taken at face value. The *perlocutionary force* is the indirect meaning of the message, roughly characterized by what the utterance makes another agent believe. The *illocutionary force* of a message is any change in the world that is directly caused by the utterance itself. The notion of illocutionary force is most easily understandable in terms of examples, such as making a promise. The act of uttering a promise changes the world in the sense that it creates a commitment to do something.

Although we will be studying the speech acts implicit in cryptographic protocols, it is easier to introduce the subject in the traditional setting of discourse analysis.

Example Consider an air traffic controller at a small military airport that receives the following message from an unknown commercial plane: "I am experiencing serious engine trouble." In terms of speech act theory, this utterance can be understood as follows.

- Locutionary force: The controller becomes aware that the plane is having engine trouble.
- Perlocutionary force: The controller comes to believe that the pilot is concerned about the safety of flying, and is in need of assistance.
- Illocutionary force: A request to land at the airport.

The exact details of the Austin-Searle tradition will not be critical for our purposes. However, we will be interested in partitioning the meaning of an utterance with respect to the terms defined above.

Disjoint belief domains

The are two different domains of information involved in anonymous communication over a network. First, there is information about the world. This information is obtained from the literal meaning of the messages exchanged. We refer to this kind of information as *world information*. The second kind of information is related to the communication session. This kind of information includes the text of each message sent, the text of each message received, the agent that sent each message, and the agent that received each message. We refer to this kind of information as *exchange information*. Formal symbolic approaches to cryptographic protocol verification tend to focus exclusively on exchange information.

Example Suppose that Alice receives a message with the text "13:00-12-13-2008" encrypted with a key that is shared with Bob, and no one else. In terms of world information, this might communicate a time and date for some particular event. The relevant event may be known to Alice through context, or it might not. In terms of exchange information, the message indicates that Bob sent a message at some point including this date. Note that Alice is not justified in concluding that Bob sent the message recently. She is, however, justified in concluding that the message was intended for her because no one else shares the given key with Bob.

The preceding example highlights precisely the kind of reasoning we would like to formalize in the analysis of encrypted communication. The important point to highlight is that world information and exchange information are related, but disjoint domains. This is not always explicit in literature on cryptographic protocol verification. In most logical work on verification, for example, there is no useful notion of world information. As a result, it is difficult to formalize the actual communicative content of many messages. By contrast, in work on Zero Knowledge Protocols [23], there is clearly a need to discuss both world information and exchange information.

Belief manipulation

Intuition

Consider a domain involving two agents, which we call the *believer* and the *manipulator*. The believer holds some beliefs about the state of the world, including an initial belief set as well as a mechanism for revision. The manipulator would like to make the believer believe some distinguished formula ψ is true. The way the manipulator tries to bring this about is by specifying a sequence of formulas to be provided to the believer as information to be incorporated. Hence, the manipulator is trying to provide information that causes the believer to perform a suitable sequence of revisions. Note that the manipulator is not providing an argument in the sense of argumentation theory; each piece of information is treated as an independent item by the believer, with no justification. Informally, we describe the manipulator to be "sending messages" to the believer. We assume that the believer receives all messages, but the receiver is not aware of the sender or the sender's goals. A preliminary version of the framework presented in this section appeared in [24].

Given this simple description of the problem, there is an obvious solution for the manipulator: the formula ψ should be provided as information for the believer. If the believer uses an AGM revision operator, then the success postulate guarantees that the believer will believe ψ after revision. Even if the believer does not use an AGM revision operator, any rational approach to belief revision should at least treat ψ as "evidence" that supports ψ. However, there are important examples where it is either impossible or undesirable to simply send ψ.

1. *Secret motive*: The manipulator does not want the believer to be aware that someone is trying to convince them to believe ψ. In this case, indirectly convincing the believer may be more appropriate.
2. *Pursuasiveness*: In the case where the manipulator stands to gain directly from the truth of ψ, it is unlikely that the message ψ would be very convincing evidence to the believer.
3. *Restrictive medium*: The communication channel for sending messages constrains the information that can be sent to the believer.

In the following sections, we formalize belief manipulation subject to these restrictions.

Formalization

The following definition provides a model of the dynamics of belief for the opponent that is to be manipulated.

Definition 1. *A* definite opponent model *is a pair* $\langle \Phi, * \rangle$, *where* Φ *is a belief set and* $*$ *is a belief revision operator.*

An opponent model gives the initial beliefs of the believer, as well as the revision operator to be used. As stated previously, however, there might be uncertainty about this information.

Definition 2. *An* opponent model *is a set of definite opponent models.*

Intuitively, an opponent model consists of all definite opponent models considered possible by the manipulator.

Define a *message constraint* to be a finite set of propositional formulas. Message constraints will be used to represent the set of all formulas that can be sent in a particular domain. We are now able to define the basic setting for belief manipulation.

Definition 3. *A* belief manipulation problem *is a triple* $\langle O, \overline{C}, \psi \rangle$, *where O is an opponent model,* $\overline{C} = C_1, \ldots, C_n$ *is a finite sequence of message constraints, and* ψ *is a formula (called the goal). The number n is the length of the scenario.*

A belief manipulation problem encodes the dynamics of belief for some opponent, along with a set of constraints on the messages that the opponent may be sent.

We define two kinds of solutions for belief manipulation problems.

Definition 4. *A* credulous solution *for* $\langle O, \overline{C}, \psi \rangle$, *is a sequence of formulas such that* $\phi_1, \ldots, \phi_n$ *such that:*

1. $\phi_i \in C_i$, *for* $i \leq n$
2. $\Phi * \phi_1 * \cdots * \phi_n \models \psi$ *for some* $\langle \Phi, * \rangle \in O$.

Definition 5. *A* skeptical solution *for* $\langle O, \overline{C}, \psi \rangle$, *is a sequence of formulas such that* $\phi_1, \ldots, \phi_n$ *such that:*

1. $\phi_i \in C_i$, *for* $i \leq n$
2. $\Phi * \phi_1 * \cdots * \phi_n \models \psi$ *for every* $\langle \Phi, * \rangle \in O$.

The appropriate solution depends on the application under consideration. A credulous solution means that an individual is able to convince at least one opponent to believe ψ, whereas a skeptical solution means that all opponents can be convinced. In attacks exploiting a "weak" link, the credulous solution is all that is required.

Note that the implementation of a belief manipulation solver is straightforward, given an implementation of the appropriate belief revision operators. Most systems that automate the calculation of belief revision operators must also be capable of checking entailments, or at least membership of a formula in a set. As such, the implementation of a belief manipulation solver simply requires the application of n revision operations, as well as n simple constraint checks. This can be carried out, for example, in the COBA system for consistency-based belief change [25].

Protocol verification

Finding an attack on a cryptographic protocol can now be formulated as a belief manipulation problem. Consider, for example, the Challenge Response protocol from the introduction. This can be formalized as $\langle O, \overline{C}, \psi \rangle$, defined as follows:

- O consists of a set of key assignments believed by B, as well as a conservative revision operator $*$.
- $\overline{C}$ is a sequence of constraints on the format of each message, which follows the protocol structure.
- ψ is a suitable logical formulation of the statement A has K_C.

We remark that formulating the verification problem in this manner has several advantages. First of all, by specifying an opponent model, we are able to make the assumptions about the beliefs of B explicit. This is not always the case in existing approaches to protocol verification, despite that fact that most attacks on communication protocols rely on exploiting faulty assumptions.

The second advantage of this formulation is the fact that we focus on the *intruder goals* as opposed to the *protocol goals*. In logic-based approaches to verification, proofs are typically produced in an abstract message passing environment. As such, given a proof of correctness for a logical representation, it is difficult to conclude that the "actual" protocol is correct. Given this state of affairs, it is not clear what can be concluded from existing proofs of correctness. By contrast, it is normally easy to translate a logical formalization of an attack into an actual attack.

Formalizing belief in message passing systems

Thus far, we have only considered belief manipulation in the very general setting of propositional logic. We remark, however, that this approach can be greatly improved in the context of a message passing system. When information is exchanged through encrypted messages, the believer will not just have simple factual beliefs about the state of the world. In addition, the believer will also have beliefs about the messages that have been exchanged and the current state of any relevant communication protocol. In this section, we set up a formal framework for representing and reasoning about these different kinds of beliefs. In the next section, we demonstrate how to dissect the meaning of an encrypted message, following our approach in [26].

Vocabulary

We introduce a formal, logical framework that is suitable for analyzing the meaning of encrypted messages. Our framework is essentially an extension of the message passing systems of [3], but we explicitly focus on the representation of sequences of messages that may involve encryption. We require the following non-empty sets of primitive symbols to describe messages exchanged:

- $\mathbf{A}$ is a set of *agents*
- $\mathbf{T}$ is a set of *texts*
- $\mathbf{K}$ is a set of *encryption keys*

We remark that we have not taken the notion of a "message" as a primitive concept. Instead, we define messages in terms of texts and keys. Specifically, we define the set of messages $\mathbf{M}$ to be the smallest set satisfying:

$$T \subseteq \mathbf{M}.$$
$$\text{If } m \in \mathbf{M} \text{ then } \{m\}_k \in \mathbf{M} \text{ for all } k \in \mathbf{K}$$

By using a separate domain for encrypted messages, we essentially rule out the possibility of guessing the encrypted value of a number. This kind of assumption is common in logical work on protocol verification. The significance of such assumptions is discussed in [27].

We require a set $\mathbf{F}$ of propositional symbols to describe the state of the world. Each element of $\mathbf{F}$ describes some aspect of the world, and takes the value *true* or *false*. For example, a propositional symbol *Raining* might be used to specify if it is raining or not. An *interpretation* is a function that assigns a value to every element of $\mathbf{F}$. Interpretations are understood to represent possible states of the world, and sets of interpretations are frequently used to represent the beliefs of an agent.

Finally, we require a set of action terms designating the activities that agents may perform. For our purposes, the only action terms are of the form *send(A,m,B)* and *receive(A,m)*. We let $\mathbf{E}$ denote the set of all such action terms, and we give the semantics of these actions below.

The following example illustrates how these sets of primitive symbols are used.

Example We define a message passing system suitable for discussing the Challenge-Response Protocol. Define the set of agents and the set of messages as follows:

$$\mathbf{A} = \{A, B, P\}$$
$$\mathbf{T} = \mathbf{N} \text{ (the set of natural numbers)}$$
$$\mathbf{K} = \mathbf{N}.$$

Note that we are following a convention in which A and B represent honest agents, and P is the *perpetrator*, which is a dishonest intruder. We will use this notation each time we return to this example. Note also that we will often say that P receives a message from A or B, which really means that P has intercepted the message. An honest agent will not send a message to a dishonest agent in the framework we are currently developing. The set $\mathbf{F}$ consists of propositional variables of the form $HasKey(i, k)$ where $i \in \mathbf{A}$ and $k \in \mathbf{K}$. Informally, $HasKey(i, k)$ is true if agent i has key k. Note that the set of texts is typically larger than the set of keys, but in this example we make the simplifying assumption that the set of natural numbers serves both roles.

Message passing

In this section, we give a series of formal definitions that are useful for describing message passing systems with cryptographic functions. First, we need to define a *message exchange*, which is the analogue of an utterance in discourse analysis.

Definition 6. *An exchange is a triple $\langle A_1, m, A_2\rangle$, where $A_1, A_2 \in \mathbf{A}$ and $m \in \mathbf{M}$.*

We call p_1 the sender of the message m and we call p_2 the recipient. A sequence of *message exchanges* defines a history.

Definition 7. *An exchange history (of length n) is an n-tuple of message exchanges.*

Let $\mathbf{H}$ denote the set of all exchange histories.

An agent is typically not aware of the messages that are exchanged privately between other agents. Therefore we need to introduce a formal notion of a *believed history* for a particular agent.

Definition 8. *A believed history is a set of exchange histories.*

Informally, a believed history is the set of all global histories that some agent believes to be possible. There is always some uncertainty about such histories, since senders and recipients are anonymous.

We are now interested in defining an appropriate notion of the local state of an agent in a message passing system. Roughly, the local state should include three things: an assignment of values to all propositional symbols, a history of messages exchanged, and a queue of actions to be executed. Formally, we have the following definition.

Definition 9. *A local state is a triple $\langle s, h, e\rangle$ where s is an interpretation of $\mathbf{F}$, $h \in \mathbf{H}$ and e is an action symbol (representing the next action to be executed).*

Again, since agents typically do not have complete information, we define a believed local state as follows.

Definition 10. *A local belief state is a triple $\langle S, H, E\rangle$ where S is a set of interpretations, H is a set of histories, and E is a set of action symbols.*

We are now in a position to say something about action effects. Following the tradition of [28], we give the effects of actions by specifying a *transition system* over local states. Formally, a transition system is just a directed graph where the nodes are labelled with local states and the edges are labelled with action symbols. If $\langle s, h, e\rangle$ is a local state for some agent A, then the outcome of executing $send(A, m, B)$ is a state of the form $\langle s, h \cdot X, e\rangle$, where $h \cdot X$ is the result of appending $\langle A, m, X\rangle$ to the exchange history h. Hence, the effects of a *send* action are non-deterministic, but straightforward in the sense that the exchange history is the only thing that changes. The effects of a *receive* action are more difficult to specify, as an agent needs to dissect the meaning of the message.

Dissecting the meaning of an encrypted utterance

Every message exchanged in a cryptographic protocol may serve three related purposes, roughly corresponding to the three kinds of speech act:

1. The content of the message may contain a statement about the world. (*locutionary act*)
2. The message may convince the recipient to hold some new beliefs. (*perlocutionary act*)
3. The message may satisfy a step in a protocol, and simultaneously request an action from the recipient. (*illocutionary act*)

We suggest that, in order to understand the meaning of an encrypted message, one must consider all three dimensions.

In order to provide a more comprehensive understanding of meaning in cryptographic protocol analysis, we introduce three *force functions*: Φ_L, Φ_P, and Φ_I. Each function takes a message M as an argument, and it returns the "meaning" of M in terms of locutionary force, perlocutionary force, and illocutionary force. Specifically, we have the following:

$$\Phi_L(M) \subseteq 2^{\mathbf{F}}$$
$$\Phi_P(M) \subseteq 2^{\mathbf{H}}$$
$$\Phi_I(M) \subseteq 2^{\mathbf{E}}$$

Here we are using the standard logical notation in which 2^X denotes the set of subsets of X. Hence, Φ_L maps M to a set of interpretations of $\mathbf{F}$, Φ_P maps M to a set of histories and Φ_I maps M to a set of actions.

Let $\langle S, H, E\rangle$ be a local belief state. When a message is received, each component might need to change. Therefore, we will need three revision operators $*_L$, $*_P$ and $*_I$ in order to incorporate all of the information contained in a single message. Suppressing subscripts on these operators for readability, receiving a message M should lead to the new local belief state $\langle S', H', E'\rangle$, where:

$$S' = S * \Phi_L(M)$$
$$H' = H * \Phi_P(M)$$
$$E' = E * \Phi_I M$$

In the next sections, we describe the three force functions individually. We then briefly discuss the associated revision operators.

Locutionary force

The locutionary force of a message M is the unencrypted contents of M.

Definition 11. *For $M \in \mathbf{M}$:*

1. *If $M \in \mathbf{T}$, then $\Phi_L(M) = \Phi_L(M)$.*
2. *If $M = \{N\}_k$, then $\Phi_L(M) = \Phi_L(N)$.*

For some cases, we need to translate the message M into a meaningful proposition in the appropriate vocabulary. In other cases, the message M really does not have a locutionary force. This is the case, for example, in the Challenge Response Protocol: a message consisting of a single random number does not make any statement about the configuration of the world.

Perlocutionary force

The perlocutionary force of a message is anything that the message makes a recipient believe related to the exchange history. Hence, the perlocutionary force of a message is the collection of implicit inferences that we make regarding the identity of the sender, based on the structure of the message.

Before defining Φ_P, we make a couple of remarks about the nature of our revision operator. We would like to revise by a set of possible histories, so $\Phi_P(M)$ should be the set of histories that are possible given that someone sent the message M. Therefore, a message that gives no indication of the sender would correspond to the set of all histories where that message was sent. By contrast, a message that could only be sent by a particular individual would corresond to a smaller set of histories. For any message M, let H_M denote the set of histories where the message M is sent at some point.

Definition 12. *For $M \in \mathbf{M}$:*

1. *If $M \in \mathbf{T}$, $\Phi_P(M) = H_M$.*
2. *If the previous history $H \in H_M$, then $\Phi_P(M) = H_M$.*
3. *If the previous history $H \notin H_M$:*
    - *If $M = \{N\}_{k_1}$ and $\{N\}_{k_2}$ was previously sent, then $\Phi_P(M) = H'$ where H' is the set of histories where some agent A holding k_1 previously received $\{N\}_{k_2}$ and then sent M.*

Condition (3) is a more general version of the so-called *message meaning* postulate of BAN logic [14]. Basically, our version of the postulate deals with sequences where A *sends* $\{N\}_{k_1}$ and then A *receives* $\{N\}_{k_2}$. If neither of these messages occurs elsewhere in the history, we are justified in concluding that some agent holding both k_1 and k_2 has seen the message N. Of course this may not always be correct; we may have been incorrect about the assignment of keys and the honesty of other agents. Nevertheless, the conclusion is the intended conclusion under “normal” circumstances.

Illocutionary force

The illocutionary force of a message is defined in terms of what it makes the recipient do. In our framework, there are two kinds of actions that an agent can perform: send actions and receive actions. We make the assumption that a receiving action is never a security risk, as receiving a message simply amounts to obtaining information. Certainly this assumption is not true in some applications, such as computer networks where messages might contain executable code. In such a context, it is important to be aware if an adversary is trying to get an honest agent to perform a receive action. However, this is not the target domain that we have in mind. We are concerned with agents communicating text messages. As such, the only kind of action with which we are concerned is a send action.

The following definition gives a straightforward procedure for determining the illocutionary force of a message.

Definition 13. *Let PROT be the set of protocols available on the network. For a received message M, $\Phi_I(M)$ is the determined by following this procedure:*

1. *For each $p \in PROT$, check if M matches step n in some protocol for which steps 1 up to $n-1$ have been performed. If not, $\Phi_I(M) = \emptyset$. If so, proceed to step 2.*
2. *If the sender of M in every possible history matches the sender from steps 1 up to $n-1$ and step $n+1$ is a send action for A, then $\Phi_I(M)$ is the next send action in the protocol. Otherwise, $\Phi_I(M) = \emptyset$.*

Revision operators

The force functions specify precisely what an encrypted message asserts, but the force functions alone do not indicate how an agent should interpret the new information. We also need to specify the behavior of the three revision operators $*_L$, $*_P$ and $*_I$. Giving a complete specification of these operators is beyond the scope of this paper, because it is highly dependent on the application. However, in this section, we give a basic idea.

The locutionary revision operator $*_L$ is basically a "standard" belief revision operator, in the sense that it deals with incorporating new information about the world. In order to define such an operator, it is both necessary and sufficient to define a total pre-order over interpretations of $\mathbf{F}$ that represents the relative plausibility of each possible state of the world. Given such an ordering, new information is incorporated by believing the most plausible states that are consistent with the new information.

The perlocutionary revision operator $*_P$ is also a standard belief revision operator, but it operates on histories. Therefore, we can define $*_P$ by specifying an ordering over histories. One suitable ordering can be defined as follows. Let A be a fixed agent with local history H_A. We define a total pre-order $<$ over $\mathbf{H}$ that partitions the set of histories into four disjoint classes:

- *MIN*: The set H_A.
- *BACK*: The set of histories obtained by extending elements of H_A with message exchanges that do not involve A.
- *PERMUTE*: The set of histories obtained from *BACK* by changing senders or recipients on some messages.
- *INIT*: The set of histories initially differing from the initial beliefs of A.

We can define $<$ according to the following coarse ordering:

$$MIN < BACK < PERMUTE < INIT.$$

Within each subclass, the ordering $<$ can be refined further. For example, histories that postulate very few new message exchanges might be preferred over those that postulate a large number of message exchanges. There are many ways to fill out this ordering, we have just provided one plausible initial construction. The levels in this ordering are intuitively plausible because each higher level requires an agent to abandon beliefs with stronger empirical support.

Finally, we need to define the illocutionary revision operator $*_I$. We are referring to this operator as a revision operator, but this is not a true revision operator in the sense of the AGM revision theory because there is no clear notion of "inconsistency" for sets of actions. However, the function $*_I$ is used to modify the set of actions to be executed in response to messages received. This can be done by simply adding all elements of $\Phi_I(M)$ to the queue of actions to be executed.

Goal discovery

Basic algorithm

Dissecting the meaning of a message in terms of speech acts is useful for analyzing the goals of an adversary. In this section, we present an algorithm for automating the goal discovery process. The basic algorithm takes three inputs:

- An exchange history H.
- An agent A trying to uncover the goals of an adversary.
- An agent P representing a believed adversary.

The output of the algorithm is a pair (G_{done}, G_{out}). The first component of the output is a list of messages that A has already sent in response to the adversary's requests. The second component is a list of messages that the adversary is trying to get A to send, but A has not yet sent.

Let H be an exchange history of length n. For $i \leq n$, let $H(i)$ denote the i^{th} message exchanged in this history. The following algorithm can be used for automated goal discovery.

```
GOAL DISCOVERY
Set $Perf = \emptyset$.
Set $Queue = \emptyset$.
    1. Set $i = 1$.
    2. Let $H(i) = \langle A_1, M, A_2 \rangle$.
    3. If $A_1 = A$, then add $M$ to $Perf$.
    4. If $A_1 = B$ and $A_2 = A$,
    then add $\Phi_I(M)$ to $Queue$.
    5. If $i < n$, set $i = i + 1$ and goto 2.
    6. Return $(Queue \cap Perf, Queue - Perf)$.
```

This goal discovery algorithm requires a complete exchange history as input, which is not generally a plausible assumption. Instead, the agent A is more likely to have a set of possible exchange histories as input. In this case, the goal discovery algorithm can be run on each exchange history in pointwise fashion, giving a set of (G_{done}, G_{out}) pairs. We can then describe the goals of the adversary in terms of *skeptical* and *credulous* reasoning.

A skeptical approach to achieved goals (resp. outstanding goals) would identify a message as a goal if it is in *every* G_{done} set (resp. G_{out} set). By contrast, a credulous approach identifies a goal if it is in *any* G_{done} (resp. G_{out}) set.

Note that the purpose of the goal discovery algorithm is really to identify what the intruder is trying to make another agent belief. Hence, we can perform the goal discovery algorithm to identify the belief manipulation problem the intruder is trying to solve. Once this problem is identified, we can then try to determine if it has a solution; this would correspond to an attack on the protocol that must be prevented.

A concrete example

In this section, we present a concrete example of our approach. Consider the following brief exchange history on a network where the Challenge-Response Protocol is used to check for liveness. Recall that K is a key shared by A and B.

1. $\langle A, \{21312\}_K, P\rangle$
2. $\langle P, \{21312\}_K, A\rangle$

The initial local belief state for A is $\langle S, \emptyset, \emptyset\rangle$ where S is the set of states where no other agent has the message 21312. Hence, there are no actions in the initial action queue for A. After step one of the algorithm, we add $send(A, \{21312\}_K, B)$ to *Perf* and we do not change *Queue*. In step two of the algorithm, we see that A has received $\{21312\}_K$ and the illocutionary force of this message is a request to send 21312. Hence, $send(A, 21312)$ is added to *Queue*. The algorithm now terminates, and we can see that the adversary is trying to get A to send the message 21312.

What is the significance of this result? Basically, we do not want to provide the adversary with any information that the adversary is trying to get. This is the point of the goal discovery algorithm: if we can find a message that the adversary wants us to send, we can then avoid sending it. Hence, an automated system could simply specify that the message 21312 must not be sent by A for the remainder of the session. In this example, we can also hypothesize about the intentions of the adversary. For example, we can hypothesize that the adversary would like to receive the message 21312 in order to send it back to someone else. Furthermore, by looking at the perlocutionary force of the message, we can see that receiving 21312 would convince A that the agent B is alive on the network. This analysis leads us to conclude that the adversary is requesting 21312, because the adversary wants to impersonate B.

This example is far more simple than the situations encountered in practice, but it illustrates the basic idea. By monitoring the goals of an adversary in terms of message requests, we are able to block an adversary from receiving those messages.

Conclusion

In this paper, we have discussed the notion of a belief manipulation problem, specifically in the context of cryptographic protocol verification. Formally, a belief manipulation attack can occur in any environment where intelligent agents have fallible, dynamic beliefs. In order to manipulate the beliefs of an agent, you must respect certain constraints and you must understand the way that your opponent's beliefs change. In this paper, we have represented the constraints as sets of propositional formulas and we have represented the belief change process in terms of AGM belief revision operators. The result is a simple, concrete formulation of a belief manipulation problem that can be applied accross a wide range of applications.

In order to understand how beliefs can be manipulated in a message passing system, we have also presented a high-level approach to specifying the meaning of encrypted utterances in terms of speech act theory. We have presented a formal framework that dissects the content of a message into three distinct communicative components. We have argued that dissecting the meaning of encrypted messages in this manner can be useful in the analysis of encrypted communication over anonymous networks. In particular, we have shown that this kind of analysis can be used to uncover the goals of an adversary. In many cases, this corresponds to identifying the belief manipulation problem that the adversary is trying to solve.

By explicitly uncovering the goals of an adversary, we are able to improve security in two different ways. First, we are able to avoid sending the messages that the adversary wants to receive. Second, we are able to analyze potential uses for this information.

Future work

This has been primarily an expository paper, as we are introducing a new form of attack to be considered in a message passing system. But this kind of attack is common in practice; particularly in the domain of protocol verification discussed in this paper. There are several different directions for future research.

One direction to consider is theoretical: exploring the properties of belief change operators that are suitable for modelling belief manipulation effectively. For example, we have recently explored the relationship between belief revision and trust [29], and it is clear that the manner in which trust is modelled will impact the way that beliefs can be manipulated. We are currently exploring a variety of formal belief change operators that incorporate a general model of trust.

The second direction to consider is directly related to cryptographic communication over message passing systems. We illustrated that many attacks on cryptographic protocols can be formulated as belief manipulation problems. As there are many tools available for finding attacks on protocols, it is possible that some of these tools could be extended to solve belief manipulation problems in a more general context. We suggest, for example, that this is the case for our own protocol verification tool presented in [13].

The third direction for future research is to explore further applications of belief manipulation. Our particular interest is the use of belief manipulation to model communication on the Smart Grid. It is well-known that the Smart Grid introduces a number of new security issues that must be addressed [30]. In this domain, agents need to communicate and make decisions about how to use, store, and trade electrical power. As such, an agent must have two pieces of information for each point in time: the amount of power needed, and the price of power. In order to define specific belief manipulation attacks on the Smart Grid, we need to use propositional variables to encode both historical records of power usage as well as future predictions. In this general setting, there are several obvious cases where belief manipulation would be a risk.

- An opponent might convince an honest agent to believe that prices will be inflated/deflated at a later time.
- An opponent might convince an honest agent to believe certain amounts of power will be needed/available later.

These examples are relatively mundane, in that the risk is primarily an economic risk to an agent on the Smart Grid. However, analyzing this kind of attack is important for several reasons. First, analyzing this kind of attack forces us to formalize the way that beliefs change due to message passing. Second, these attacks make us consider the perspective of an adversary. When we look at the actions of an adversary in terms of their effects on our beliefs, then it becomes more evident where the risks are. Third, from a bargaining perspective, an intelligent Smart Grid user will want to exploit strategies based on belief manipulation to their own economic gain. We are not aware of any work that focuses on Smart Grid violations and attacks that are based explicitly on the manipulation of the beliefs of other agents. In future work, we intend to address this problem more precisely, following our existing approach to modelling and verifying communication protocols.

Competing interests
The author declares that he has no competing interests.

Acknowledgements
The author would like to acknowledge the support of NSERC and the VP Research Seed Fund at BCIT. The author would also like to thank the reviewers from both Security Informatics and the Pacific Asia Workshop on Intelligence and Security Informatics for helpful suggestions on earlier versions of this manuscript.

Received: 17 January 2014 Accepted: 29 September 2014
Published online: 10 October 2014

References
1. G Brassard, D Chaum, C Crepeau, Minimum disclosure proofs of knowledge. J. Comput. Syst. Sci. **37**, 156–189 (1988)
2. R Needham, M Schroeder, Using encryption for authentication in large networks of computers. Commun. ACM. **21**(12), 993–999 (1978)
3. R Fagin, J Halpern, Y Moses, M Vardi, *Reasoning About Knowledge* (MIT Press, Menlo Park, California, 1995)
4. J Austin, *How To Do Things With Words* (Harvard University Press, Cambridge, Mass, 1962)
5. J Searle, *Speech Acts* (Cambridge University Press, Cambridge, England, 1969)
6. L Aiello, F Massacci, Planning attacks to security protocols: case studies in logic programming, in *Computational Logic: Logic Programming and Beyond, Essays in Honour of Robert A. Kowalski*, ed. by A Kakas, F Sadri, vol. 1, (2001)
7. A Armando, L Compagna, Y Lierler, Automatic compilation of protocol insecurity problems into logic programming, in *Proceedings of JELIA*, (2004), pp. 617–627
8. M Burrows, M Abadi, R Needham, A logic of authentication. ACM Trans. Comput. Syst. **8**(1), 18–36 (1990)
9. A Hunter, JP Delgrande, Belief change and cryptographic protocol verification, in *Proceedings of AAAI*, (2007), pp. 427–433
10. J Halpern, R Pucella, On the relationship between strand spaces and multi-agent systems. CoRR. **cs.CR/0306107** (2003)
11. J Thayer, J Herzog, J Guttman, Strand spaces: Proving security protocols correct. J. Comput. Secur. **7**(2–3), 191–230 (1999)
12. C Cremers, The Scyther Tool: Verification, falsification, and analysis of security protocols, in *Computer Aided Verification, 20th International Conference*. (2008), pp. 1–30
13. A Hunter, JP Delgrande, R McBride, Protocol verification in a theory of action, in *Proceedings of the Canadian Conference on Artificial Intelligence*. (2013), pp. 52–63
14. M Burrows, M Abadi, R Needham, A logic of authentication. ACM Trans. Comput. Syst. **8**(1), 18–36 (1990)
15. P Syverson, P van Oorschot, A unified cryptographic protocol logic. Technical Report 5540-227, Naval Research Lab (1996)
16. L Paulson, The inductive approach to verifying cryptographic protocols. J. Comput. Secur. **6**, 85–128 (1998)
17. L Aiello, F Massacci, Verifying security protocols as planning in logic programming. ACM Trans. Comput. Logic. **2**(4), 542–580 (2001)
18. F Crazzolara, G Winskel, Events in security protocols, in *Proceedings of the 8th ACM Conference on Computer and Communication Security*. (2001), pp. 96–105
19. C Alchourron, P Gardenfors, D Makinson, On the logic of theory change: Partial meet functions for contraction and revision. J. Symbolic Logic. **50**(2), 510–530 (1985)
20. H Katsuno, AO Mendelzon, On the difference between updating a knowledge base and revising it, in *Belief Revision*, ed. by P Gardenfors. (Cambridge University Press, 1992), pp. 183–203
21. A Grove, Two modellings for theory change. J. Philos. Logic. **17**, 157–170 (1988)
22. W Spohn, Ordinal conditional functions: a dynamic theory of epistemic states, in *Causation in Decision, Belief Change, and Statistics*, vol. 2, (1988), pp. 105–134
23. O Goldreich, S Micali, A Wigderson, Proofs that yield nothing but their validity. J. ACM. **38**(3), 690–728 (1991)
24. A Hunter, Belief manipulation: a formal model of deceit in message passing systems, in *Proceedings of the Pacific Asia Workshop on Intelligence and Security Informatics*. (2013)

25. JP Delgrande, A Hunter, T Schaub, COBA: A consistency-based belief revision system, in *Proceedings of the 8th European Conference on Logics in Artificial Intelligence (JELIA-02).* (2002), pp. 509–512
26. A Hunter, Dissecting the meaning of an encrypted message: An approach to discovering the goals of an adversary, in *Proceedings of the European Conference on Intelligence and Security Informatics.* (2008)
27. M Abadi, P Rogaway, Reconciling two views of cryptography (the computational soundness of formal encryption). J. Cryptology. **15**(2), 103–127 (2002)
28. M Gelfond, V Lifschitz, Action languages. Linkoping Electron. Articles Comput. Inf. Sci. **3**(16), 1–16 (1998)
29. A Hunter, Belief revision and trust, in *Proceedings of the International Workshop on Nonmonotonic Reasoning.* (2014)
30. P McDaniel, S McLaughlin, Security and privacy challenges in the smart grid. Secur. Privacy. **7**(3), 75–77 (2009)

doi:10.1186/s13388-014-0015-3
Cite this article as: Hunter: **Belief manipulation and message meaning for protocol analysis.** *Security Informatics* 2014 **3**:15.