Rethinking about Guessing Attacks

Zhiwei Li
Department of SIS
UNC Charlotte
Charlotte, NC 28223
zli19@uncc.edu

Weichao Wang
Department of SIS and CyberDNA
UNC Charlotte
Charlotte, NC 28223
weichaowang@uncc.edu

ABSTRACT

Although various past efforts have been made to characterize and detect guessing attacks, there is no consensus on the definition of guessing attacks. Such a lack of generic definition makes it extremely difficult to evaluate the resilience of security protocols to guessing attacks.

To overcome this hurdle, we seek a new definition in this paper to fully characterize the attacker's guessing capabilities (i.e., guessability). This provides a general framework to reason about guessing attacks in a symbolic setting, independent of specific intruder models. We show how the framework can be used to analyze both passive and active guessing attacks.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Protocol verification*; D.2.4 [**Software**]: Software/Program Verification—*Formal methods*

General Terms

Security, Theory, Verification

1. INTRODUCTION

Many security protocols are vulnerable to guessing attacks, which aim to obtain a poorly chosen password or data by trying every possible value for it. Let us consider a simple one-way authentication protocol:

$$\begin{array}{ll} \text{Message 1.} & A \rightarrow B : \{N_A\}_{K_{AB}} \\ \text{Message 2.} & B \rightarrow A : \{\mathtt{f}(N_A)\}_{K_{AB}} \end{array}$$

Here N_A is a fresh nonce generated by A and K_{AB} is the symmetric key shared between A and B, and $\mathtt{f}$ is a given function (e.g., $\mathtt{f}(N_A) = N_A + 1$). An attacker may obtain K_{AB} by trying to decrypt both messages with a guessed key k and then to compare the results, say r_1 and r_2: if r_2 equals $\mathtt{f}(r_1)$, then k is the correct guess. Such attacks become more feasible when one chooses a low entropy secret.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ASIACCS '11, March 22–24, 2011, Hong Kong, China.
Copyright 2011 ACM 978-1-4503-0564-8/11/03 ...$10.00.

Starting from the early work of Gong et al. [32, 31], a lot of efforts have been made either to formulate guessing attacks or to detect them. Many approaches focus on heuristics to explore ways of validating a guess [17, 42, 33]. This is usually done by enumerating rules to determine whether a guess can be "verified", a term widely accepted to characterize a correct guess. These rules are used to derive an inference system modeling the guessing capabilities [22], by extending the standard Dolev-Yao model [25]. Realizing the "incompleteness" of such an inference system in a sense that it may fail to capture some guessing attacks, Drielsma et al. [26] develop a precise formalization of off-line guessing attacks, which is independent of any particular intruder model. However, no automatic procedure is given in [26] and, more importantly, it only allows guessing atomic values. In [17], Corin et al. first use static equivalence from the applied pi calculus [2] to characterize guessing attacks, which is then used to derive a procedure for detecting guessing attacks [3]. More recently, Blanchet and Abadi [7] refine the definition by imposing the observational equivalence condition.

Up to now, there is still no clear consensus regarding the general definition of guessing attacks, which explains why some protocol previously shown resistant to guessing attacks turns out to be vulnerable [40, 32]. There are two main reasons for this lack of generality.

First, the term "verifiable" is not fully understood or formalized, while being used implicitly as a synonym for "guessable" in all previous approaches. It is fair to mention here that several definitions regarding verifiability do exist, although none of them is general enough to be independent of protocol modeling and/or specific intruder models. For instance, Lowe [42] presents a group of rules to verify a guess. Indeed, these rules correctly identify verifiable guesses. It is unclear whether or not the rule set can completely cover all guesses that can actually be verified somehow, even under the Dolev-Yao intruder model. Similarly, Corin et al. [17] define a "verifiable" guess based on two conditions of a "verifier". However, without any intuitive appeal, this definition can fail to capture some practically verifiable guess. Besides, the verifier itself can be very difficult to find. Corin et al. [16] then formulate a new definition of verifying a guess using static equivalence [2], which elegantly captures the essence of verifying a guess. Nonetheless, this definition may require the modeling of security protocols by the applied pi calculus. Moreover, it only considers guesses of atomic messages.

Second, guessing attacks have been studied from two different perspectives: (1) the process perspective [16, 3, 7], which relies on the modeling of security protocols; and (2) the attacker's perspective [17, 22, 42], which emphasizes the guessing capabilities from a logical point of view. Neither provides a unified view towards guessing attacks.

Our work is therefore geared towards a unified framework for the study of guessing attacks. The primary goal is to establish an intimate understanding of guessing, which is intuitive, yet provides a rigorous basis for guessing attacks. In other words, the new framework should be

- **faithful** (i.e., fits the common sense of guessing attacks),
- **expressive** (i.e., accounts for multiple guesses), and
- **complete** (i.e., captures all guessing attacks in a symbolic setting).

Unlike most previous work, we treat "guessing" and "attack" separately, because guessing relates closely to the attacker's ability to reason about its knowledge, whereas attack further exploits the vulnerability of security protocols. It is worthwhile to reveal the dominant factor of a guessing attack — the attacker's guessing capabilities or the interactions between entities.

Contributions.

In this paper, we propose a new definition to fully characterize the attacker's guessing capabilities and then show how it relates to finding guessing attacks in security protocols. Specifically, this paper makes the following contributions:

- To uncover relationship between "verifiable" and "guessable", we formalize the idea of verifying a message in terms of recognizability [39] — the ability to distinguish a message from noise. To our best knowledge, this is the first definition of verifiability that is independent of security protocols and/or intruder models. We show, surprisingly, that a guessable message needs NOT to be verifiable. In other words, even though some message is not verifiable, it can still be guessed correctly by the intruder.
- We propose a weaker notion of verifiability to recover the intuitive understanding of guessing — a message can be guessed if and only if it is weakly verifiable. This weaker notion thus provides a faithful, expressive, and complete framework for the study of guessing attacks.
- We introduce a novel way to evaluate the hardness of guessing. While some guessing attack turns out to be (computationally) infeasible, the new metric provides an accurate way to discriminate between feasible and infeasible guessing attacks, reducing the gap between formal methods and real implementation. To our best knowledge, this is the first explicit measurement about guessing.
- As a case study, we apply our methodology to find passive guessing attacks under the standard Dolev-Yao intruder model and discuss how to extend this methodology to analyze active attacks.

Additional Related Work.

Our work is inspired by previous research efforts on applying knowledge to security problems, which is often divided into two groups: deducibility [41] and indistinguishability [1].

Deducibility is one kind of algorithmic knowledge [35], in which "knowing what" can be determined by an algorithm. The BAN [9] logic, proposed by Burrows, Abadi and Needham, is probably the first extensively studied logic in protocol analysis based on knowledge.

The concept of indistinguishability comes from the classical possible-worlds approach to model knowledge [30], in which the actual world is considered to be one of many possible worlds. Recently, Cohen and Dam [13] provide a generalized Kripke semantics for studying this type of knowledge in security protocol analysis. They use a static equivalence [2] to capture the indistinguishability for agents. Abadi and Cortier [1] examine the decidability of these two notions of knowledge by studying the underlying equational theories for deduction and static equivalence.

Organization.

In Section 2, we introduce some background material. In Section 3, we formalize the idea of verifying a guess and explain why (strong) verifiability is not a necessary condition for guessing. After presenting a new knowledge model that accounts for the attacker's guessing capabilities in Section 4, we introduce a weaker notion of verifiability that fully characterizes guessing capabilities in Section 5. In Section 6, we present our metric to gauge the hardness of guessing. In Section 7, we move our attention to finding guessing attacks. Section 8 concludes the paper.

2. PRELIMINARIES

In this section, we briefly review the basic definitions of term rewriting systems and the deducibility. We mainly follow the notations in [23].

2.1 Term Algebra

A signature is a finite set of function symbols $\mathcal{F}$ and a possibly infinite set of constants $\mathcal{A}$. Each function symbol has an associated arity. We discriminate two types of function symbols, namely, *public* and *private* function symbols, the sets of which are denoted by $\mathcal{F}^+$ and $\mathcal{F}^-$, respectively. Public functions are used to describe operations that can be freely performed by a principal, such as encryption and decryption. We need to point out that a decryption operation is conducted by calling a decryption algorithm even without the proper decryption key and can be applied to any message. It can be different from the decryption of a ciphertext.

We define the *term algebra* $\mathcal{T}(\mathcal{F}, \mathcal{A}, \mathcal{X})$ as the smallest set containing $\mathcal{X}$ and $\mathcal{A}$ such that $f(t_1, \cdots, t_n) \in \mathcal{T}(\mathcal{F}, \mathcal{A}, \mathcal{X})$ whenever $f \in \mathcal{F}$ with arity n, and $t_1, \cdots, t_n \in \mathcal{T}(\mathcal{F}, \mathcal{A}, \mathcal{X})$. Elements of the set $\mathcal{T}(\mathcal{F}, \mathcal{A}, \mathcal{X})$ are called *terms*. We will use l, r, s, t to denote terms and x, y, z to denote variables. To avoid confusion, syntactic equality of two terms t_1 and t_2 will be denoted by $t_1 =_s t_2$. We tend to use the words "term" and "message" interchangeably in the rest of this paper.

We say s is a *subterm* of t, written $s \subseteq t$, if either $s =_s t$ or $t =_s f(t_1, \cdots, t_n)$ and s is a subterm of t_i for some i. We also write $s \subset t$ to mean $s \subseteq t$ and $s \neq_s t$. A term s *occurs* in a term set T if $s \subseteq u$ for some $u \in T$. As usual, $fv(t)$ is defined as the set of variables that occur in term t. A term is *ground* if $fv(t) = \emptyset$.

We define the *length* of a term t, notation $|t|$, as the length of its binary representation. For convenience, we will use $ff(t)$ and $sub(t)$, respectively, to denote the outmost function symbol of t and the immediate subterm set of a term t[1]. A *context* C is a term with exactly a "hole" $\Box$. Then the term $C[t]$ is C except $\Box$ is replaced by t. Abusing notation slightly, we refer to $t[u \mapsto v]$ as t except that *every* occurrence of u in t is replaced by v.

A *substitution* is a finite tuple $[t_1/x_1, ..., t_n/x_n]$ mapping from variables x_i to terms t_i. The *domain* and *range* of a substitution σ are defined by $Dom(\sigma) \overset{def}{=} \{x | x\sigma \neq_s x\}$ and $Ran(\sigma) \overset{def}{=} \bigcup_{x \in Dom(\sigma)} \{x\sigma\}$, respectively. We write $\sigma = \theta$ if $Dom(\sigma) = Dom(\theta)$ and $x\sigma =_s x\theta$ for all x. We define the *composition* of substitutions σ and θ as a new substitution $\sigma \circ \theta$ such that $t\sigma \circ \theta =_s (t\sigma)\theta$.

2.2 Term Rewriting Systems

An *equation* is a pair of terms, written $s = t$ and an *equational theory* E is presented by a finite set of equations. We write $t_1 =_E t_2$ when equation $t_1 = t_2$ is a logical consequence of E.

As is commonplace, the reflexive transitive closure of a binary relation $\to$ is denoted by $\to^*$. A *term rewriting system* R consists of a set of rules, $l \to r$. A term rewriting system R defines a *term rewriting relation* $\to_R$ in a standard way: $C[l\sigma] \to_R C[r\sigma]$ where C is a context, $l \to r \in R$, and σ is a substitution. Relation $\leftrightarrow_R$ is defined by $s \leftrightarrow_R t$ iff $s \to_R t$ or $t \to_R s$.

We say that an element p is *reducible* for $\to$ if there is an element q such that $p \to q$ and *irreducible* otherwise. We write $p \to^! q$ if $p \to^* q$ and q is irreducible. If $s \to^!_R t$, then t is called an *R-normal form* of s. $\to_R$ is *terminating* if there exists no infinite derivation $t_0 \to_R t_1 \to_R \cdots$ and $\to_R$ is *confluent* if there is a term t such that $t_1 \to^*_R t$ and $t_2 \to^*_R t$ whenever $t_0 \to^*_R t_1$ and $t_0 \to^*_R t_2$. A term rewriting system R is *convergent* if $\to_R$ is terminating and confluent. Given an equational theory E, we define term rewriting system $R_E \overset{def}{=} \{l \to r | l = r \in E\}$. When $\to_{R_E}$ is convergent, $t_1 =_E t_2$ iff t_1 and t_2 have the same R_E-normal form[4, 23].

A substitution σ is *R_E-normal* if all terms in $Ran(\sigma)$ are R_E-normal. We write $\sigma_1 =_E \sigma_2$ to indicate that $Dom(\sigma_1) = Dom(\sigma_2)$ and $x\sigma_1 =_E x\sigma_2$ for all $x \in Dom(\sigma_1)$.

2.3 Modeling Standard Adversaries

The most straightforward way to model the attacker's knowledge is in terms of message deducibility [25, 41]. That is, given an equational system E and some messages T one might be able to compute another message t from T under equational theory E. Formally,

$$\boxed{\vdash^{(n)}} \qquad \text{(R1)} \qquad \frac{t \in T}{T \vdash^{(1)} t}$$

$$\text{(R2)} \qquad \frac{T \vdash^{(n_1)} t_1 \cdots T \vdash^{(n_k)} t_k}{T \vdash^{(1+\max_{1 \le i \le k} n_i)} f(t_1, \cdots, t_k)} \; f \in \mathcal{F}^+$$

$$\boxed{\vdash^{(n)}_E} \qquad \text{(R3)} \qquad \frac{T \vdash^{(n)} s \quad s =_E t}{T \vdash^{(n+1)}_E t}$$

[1]We let $sub(t) = \{t\}$ and $ff(t) = \emptyset$ if $t \in \mathcal{X} \cup \mathcal{A}$.

We say that t can be deduced from T, written $T \vdash t$, if $T \vdash^{(n)} t$ for some n. Likewise, t is deduced from T under E, notation $T \vdash_E t$, if $T \vdash^{(n)}_E t$ for some n. We say that S and T are *equivalent* (under E), denoted as $S \equiv_E T$, if $S \vdash_E t$ for every $t \in T$ and $T \vdash_E s$ for every $s \in S$. As we can see, both $\vdash$ and $\vdash_E$ are closed under substitution. We informally refer to all terms in T as the attacker's *explicit knowledge* and all terms deducible from T under E as its *implicit knowledge.*

Lemma 2.1. *Let T be a term set and σ be a substitution. Then, $T\sigma \vdash t$ if and only if $T \vdash t'$ for some t' such that $t'\sigma =_s t$.*

The following equational theory E_{dy} is used to model the standard Dolev-Yao intruder.

Public function symbols	$\texttt{pair}, \texttt{fst}, \texttt{snd}, \texttt{enc}, \texttt{dec}$
Private function symbols	$\texttt{kp}$
Equations E_{dy}	$\texttt{fst}(\texttt{pair}(x, y)) = x$
	$\texttt{snd}(\texttt{pair}(x, y)) = y$
	$\texttt{dec}(\texttt{enc}(x, y), \texttt{kp}(y)) = x$
	$\texttt{dec}(\texttt{enc}(x, \texttt{kp}(y)), y) = x$

Figure 1: Equational Theory E_{dy} modeling the standard Dolev-Yao intruder.

The equational theory E_{dy} contains two public constructive function symbols for encryption and concatenation, three destructive function symbols for decryption and split, and one private function symbol for key pair. Our analysis does not rely on the actual cryptosystem being used. Rather, we would use $\texttt{kp}(k)$ to denote the pair key of an encryption key k. So, for symmetric encryption key k, we implicitly assume that $\texttt{kp}(k) = k$. To reduce notational clutter, we will often use $\{s\}_t$, $s \cdot t$, and k^- as shorthands for $\texttt{enc}(s, t)$, $\texttt{pair}(s, t)$, and $\texttt{kp}(k^+)$, respectively.

3. FORMALIZING THE IDEA OF VERIFYING A GUESS

As mentioned in the introduction, although the intuitive idea of verifying a guess has been extensively used to analyze guessing attacks in security protocols, it has not been adequately formalized. The purpose of this section is to formalize the meaning of "verifying a guess".

It is crucial to note that verifiability requires one to distinguish useful information (a correct guess) from noise — an ability that is independent of security protocols. For instance, as seen in the example in the introduction, the attacker who knows $\{N_A\}_{K_{AB}}$ and $\{\texttt{f}(N_A)\}_{K_{AB}}$ can easily test whether a message g is the correct guess of K_{AB}. And the test can be done off-line by checking

$$\texttt{dec}(\{\texttt{f}(N_A)\}_{K_{AB}}, g) \overset{?}{=}_{E_{dy}} \texttt{f}(\texttt{dec}(\{N_A\}_{K_{AB}}, g))$$

Some may argue, however, that for more complicated protocols (e.g., simplified LGSN protocol [24]) the attacker do need to communicate with other parties to verify a guess. We adopt a cognitive point of view here: verifying a guess is a process of using its knowledge, whereas communication is a way for protocol participants to exchange knowledge.

It is desirable to formalize verifiability independent of intruder models and security protocols. This demand resonates with a recent work on type-flaw attacks [39], which introduces "recognizability" to characterize the fact that a message could not be type-flawed, i.e., the incoming message can never be replaced by another message without detection. Although their concern appears to be different from here, the methodology is exactly the same: using one's knowledge to distinguish a message from another. We thus build our work on the concept of "recognizability".

Definition 3.1 (Operational Equivalence [39]). *Let T be a term set and σ_1 and σ_2 be two ground substitutions such that $Dom(\sigma_1) = Dom(\sigma_2) = fv(T)$. They are* operational equivalent *in equational theory E w.r.t. term set T, written $\sigma_1 \approx_{E,T} \sigma_2$, if for all terms u and v such that $T \vdash \{u, v\}$ we have $u\sigma_1 =_E v\sigma_1 \Leftrightarrow u\sigma_2 =_E v\sigma_2$.*

Intuitively, operational equivalence establishes the fact one can never discriminate two ways of instantiating messages by exploiting the difference with what he knows. More specifically, we use term set T to model one's deductive knowledge with variables denoting possibly ambiguous (or informally unverified) messages.

The following lemma gives some useful characterizations of operational equivalence.

Lemma 3.2 (Transformation Lemma [38]).

(i). Suppose that $T \vdash_E t$. Then, $\sigma_1 \approx_{E,T} \sigma_2$ iff $\sigma_1 \approx_{E,T\cup\{t\}} \sigma_2$;

(ii). Suppose that $T \vdash s$, $s\sigma_1 =_E w_1$, $s\sigma_2 =_E w_2$, and x never occurs in T. Then, $\sigma_1 \approx_{E,T} \sigma_2$ iff $\sigma_1' \approx_{E,T\cup\{x\}} \sigma_2'$, where $\sigma_1' = \sigma_1 \circ [w_1/x]$ and $\sigma_2' = \sigma_2 \circ [w_2/x]$.

Example 1. *Consider again the one-way authentication protocol presented in the introduction. Assume a passive attacker can eavesdrop on communication links and save all the messages. Then, we can use $T_0 = \{\{N_A\}_{K_{AB}}, \{f(N_A)\}_{K_{AB}}\}$ to represent the attacker's knowledge. Here and hereafter, whenever needed, we implicitly add the public unary function symbol f into the term algebra presented in Figure 1.*

Suppose that the attacker wants to guess the value of N_A and we use variable x to signify the guess. Let $T = T_0 \cup \{x\}$, $\sigma_1 = [N_A/x]$, and $\sigma_2 = [N_B/x]$. Clearly, $x\sigma_1$ is a correct guess, but $x\sigma_2$ is not. Then, it can be shown that $\sigma_1 \approx_{E_{dy},T} \sigma_2$. In other words, the attacker is unable to check whether a guess (of N_A) is correct or not.

We now suppose that the attacker wants to guess the value of K_{AB}. Again, we use x to signify the guess, and let $\sigma_3 = [K_{AB}/x]$ and $\sigma_4 = [N_B/x]$. We choose

$$u =_s \mathtt{dec}(\{f(N_A)\}_{K_{AB}}, x)$$
$$v =_s f(\mathtt{dec}(\{N_A\}_{K_{AB}}, x))$$

Then,

$$u\sigma_3 =_s \mathtt{dec}(f(\{N_A)\}_{K_{AB}}, K_{AB})$$
$$v\sigma_3 =_s f(\mathtt{dec}(\{N_A\}_{K_{AB}}, K_{AB}))$$
$$u\sigma_4 =_s \mathtt{dec}(\{f(N_A)\}_{K_{AB}}, N_B)$$
$$v\sigma_4 =_s f(\mathtt{dec}(\{N_A\}_{K_{AB}}, N_B))$$

Consider now, $T \vdash \{u, v\}$, $u\sigma_3 =_{E_{dy}} v\sigma_3 =_{E_{dy}} f(N_A)$, and $u\sigma_4 \neq_{E_{dy}} v\sigma_4$. By the definition of operational equivalence, we have $\sigma_1 \not\approx_{E_{dy},T} \sigma_2$.

In the above example, we see that the attacker can discriminates a correct guess of K_{AB} from N_A by investigating the operational equivalence relation between two guesses (described by two substitutions): if the two different substitutions (resp. a correct and an incorrect guess) do not satisfy operational equivalence, then the guess can be verified; otherwise, the attacker cannot capture any nuance and the guess is not verifiable.

Definition 3.3 (Recognizability [39]). *Let T be a ground term set, t be a ground term, and $\sigma_0 = [t/x]$. We say that t is* recognizable *by T under equational theory E, and write $T \rhd_E t$, if the following condition holds:*

$$\sigma \approx_{E,T\cup\{x\}} \sigma_0 \textit{ iff } \sigma =_E \sigma_0$$

With this hindsight, we say a guess of t is *(strongly) verifiable* by T under equational theory E if $T \rhd_E t$. As in the previous example, we have $T \not\rhd_{E_{dy}} N_A$ and $T \rhd_{E_{dy}} K_{AB}$, which confirm that the protocol is vulnerable to off-line guessing attack.

Remark. It should be noticed that operational equivalence is closely related to that of static equivalence [2, 1]. The main difference is that operational equivalence is from a *cognitive* perspective, whereas static equivalence is from a *process* point of view. Nonetheless, deciding recognizability and deciding static equivalence are significantly different. For recognizability, we concern with the problem: given a message m whether there exists another message m' that is indistinguishable from m by the observer. In other words, we need to consider all possible message m' that is relevant to the operational equivalence relation. Consequently, deciding recognizability can be much harder than deciding static equivalence.

Example 2. *We extend the equational theory E_{dy} to model probabilistic encryption scheme by adding two public function symbols $\mathtt{renc}$ and $\mathtt{rdec}$, and the following two equations:*

$$\boxed{\begin{array}{r} \mathtt{rdec}(\mathtt{renc}(x, y, r), \mathtt{kp}(y)) = x \\ \mathtt{rdec}(\mathtt{renc}(x, \mathtt{kp}(y), r), y) = x \end{array}}$$

We use $\{s\}_t^r$ to denote $\mathtt{renc}(s, t, r)$ and E_{dyr} to represent the new extended equational theory.

Let us consider the Encrypted Password Transmission (EPT) protocol [34]

Message 1.	$S \to U : N_S \cdot K_S^+$
Message 2.	$U \to S : \{N_S \cdot P\}_{K_S^+}^r$

Here, we use P to denote the secret password memorized by the user U and shared with the server S[2]*. Now, suppose that a passive attacker knows N_S, K_S^+, and wants to guess P. Let $T = \{N_S, K_S^+, \{N_S \cdot P\}_{K_S^+}^r\}$ and $\sigma_0 = [P/x]$.*

Since the encryption scheme is randomized, the attacker does not know r and thus it is not able to compute $\{N_S \cdot P\}_{K_S^+}^r$ by the guess of P, say P'. It is not hard to see that for all u, v such that $T \cup \{x\} \vdash \{u, v\}$ we have $u\sigma_0 =_{E_{dyr}} v\sigma_0$ iff $u =_{E_{dyr}} v$. Similarly, for all u, v such that $T \cup \{x\} \vdash \{u, v\}$ we have $u[P'/x] =_{E_{dyr}} v[P'/x]$ iff $u =_{E_{dyr}} v$. Hence, $u\sigma_0 =_{E_{dyr}} v\sigma_0$ iff $u[P'/x] =_E v[P'/x]$. Because $\sigma_0 =_{E_{dyr}} [P'/x]$ needs not to be true, using the definition

[2]In implementation, the secret password is either stored in plain text or hashed under some one-way function.

of recognizability we get $T \not\triangleright_{E_{dyr}} P$. This confirms the claim that this protocol is resistant to guessing attacks [34, 16].

However, if the protocol uses deterministic encryption, that is the second message is replaced by $\{N_S \cdot P\}_{K_S^+}$, then the value of P can actually be guessed. Let $T' = \{N_S, K_S^+, \{N_S \cdot P\}_{K_S^+}\}$. Towards a contradiction, suppose that $\sigma \approx_{E_{dy}, T \cup \{x\}} \sigma_0$ and $\sigma \neq_{E_{dy}} \sigma_0$.

Let $u =_s \{N_S \cdot x\}_{K_S^+}$ and $v = \{N_S \cdot P\}_{K_S^+}$. Clearly, $T \cup \{x\} \vdash \{u, v\}$ and $u\sigma_0 =_E v\sigma_0$. By the definition of operational equivalence, we get $u\sigma =_{E_{dy}} v\sigma$. That is, $\{N_S \cdot P'\}_{K_S^+} =_{E_{dy}} \{N_S \cdot P\}_{K_S^+}$. So, $P' =_{E_{dy}} P$ and thus $\sigma =_{E_{dy}} \sigma_0$, a contradiction. Therefore, $\sigma \approx_{E_{dy}, T \cup \{x\}} \sigma_0$ implies $\sigma =_{E_{dy}} \sigma_0$ and thus $T' \triangleright_{E_{dy}} P$.

Indeed, (strong) verifiability implies the ability to guess. Nonetheless, we claim that this notion may fail to fully capture all possible guesses. Here's an example to show why.

Example 3. *Let $T = \{N_A, \{N_A \cdot P\}_{K_B^+}\}$ denotes the attacker's knowledge. Suppose that the attacker wants to guess the value of P, say P'. Note that the attacker does not know K_B^-. It is not hard to see that for all u, v such that $T \cup \{x\} \vdash \{u, v\}$ we have $u\sigma =_{E_{dy}} v\sigma$ iff $u =_{E_{dy}} v$. So, $u[P'/x] =_{E_{dy}} v[P'/x]$ iff $u[P/x] =_{E_{dy}} v[P/x]$. Since $P' =_{E_{dy}} P$ does not necessarily need to be true, using the definition of recognizability we know $T \not\triangleright_{E_{dy}} P$. In other words, P is not strongly verifiable by T under E_{dy}.*

Now, we suppose that the attacker first tries to guess K_B^-. Let $\sigma_0 = [K_B^-/x]$. Towards a contradiction, suppose that $\sigma \approx_{E_{dy}, T \cup \{x\}} \sigma_0$ and $\sigma \neq_{E_{dy}} \sigma_0$. Let $u =_s$ $\mathtt{fst}(\mathtt{dec}(\{N_A \cdot P\}_{K_B^+}, x))$ and $v =_s N_A$. Clearly, $T \cup \{x\} \vdash \{u, v\}$ and $u\sigma_0 =_E v\sigma_0$. By the definition of operational equivalence, we get $u\sigma =_E v\sigma$. That is, $\mathtt{fst}(\mathtt{dec}(\{N_A \cdot P\}_{K_B^+}, x))\sigma =_{E_{dy}} N_A$. So, $\sigma =_{E_{dy}} \sigma_0$, a contradiction. Therefore, $\sigma \approx_{E_{dy}, T \cup \{x\}} \sigma_0$ implies $\sigma =_{E_{dy}} \sigma_0$ and thus $T \triangleright_{E_{dy}} K_B^-$. Then, with the correct guess of K_B^-, the attacker can easily get P.

We thus close this section by remarking that a complete characterization of guessing attacks requires a more general notion than strong verifiability.

4. ACCOUNTING FOR THE ATTACKER'S GUESSING CAPABILITIES

Before proceeding any further with a more general notion to characterize guess, we introduce a new knowledge model to account for the attacker's guessing capabilities.

4.1 Explicit Guess and Implicit Guess

We have already seen in Example 3 that a guessable term is not necessarily a term that the attacker actually guesses. To avoid confusion, we use "explicit guess" to refer to the actual guess that the attacker makes; and "implicit guess" to refer to new terms deducible from the attacker's updated knowledge (i.e., knowledge plus explicit guess(es)). Besides, when we say a term is "guessable" or "can be guessed", we always refer to implicit guess. In this terminology, we say P is guessable by making explicit guess of K_B^- in Example 3. We tend to omit "implicit" or "explicit" when it is clear from the context.

As we will see, such a distinction between explicit and implicit guesses is important to understand the innate nature of guessing attacks. Let us consider some other examples that highlight this distinction.

Example 4. *Let $T = \{N_A, K_B^+, \{N_A \cdot P\}_{K_B^+}\}$ denotes the attacker's knowledge. Suppose that the attacker aims to obtain P. There are two possible ways: First, the attacker can explicitly guess P by using*

$$\{N_A \cdot x\}_{K_B^+}\sigma =_{E_{dy}} \{N_A \cdot P\}_{K_B^+}$$

Second, it can explicitly guess K_B^- by using

$$\mathtt{fst}(\mathtt{dec}(\{N_A \cdot P\}_{K_B^+}, y))\sigma =_{E_{dy}} N_A$$

These two methods differ in their explicit guesses. Clearly, the one with the shorter binary length is easier to be guessed.

The above example shows that to launch a guessing attack, there might be several ways for the attacker to make explicit guess; The following example illustrates the situation involves multiple explicit guesses.

Example 5. *Let $T = \{N_A, K_B^+, \{N_A \cdot K_{AB}\}_{K_A^+}, \{N_A \cdot \{P\}_{K_{AB}}\}_{K_B^+}\}$ denotes the attacker's knowledge. Suppose that the attacker aims to obtain P (i.e., implicitly guess P). One straightforward way is by explicitly guessing K_A^- and P. Let x and y signify the two guesses, respectively. At first, the attacker can use*

$$\mathtt{fst}(\mathtt{dec}(\{N_A \cdot P\}_{K_B^+}, x))\sigma =_{E_{dy}} N_A$$

to obtain the correct guess of K_A^-. Then, it gets K_{AB} by decrypting $\{N_A \cdot K_{AB}\}_{K_A^+}$. Finally, it can use

$$\{N_A \cdot \{y\}_{K_{AB}}\}_{K_B^+}\sigma =_{E_{dy}} \{N_A \cdot \{P\}_{K_{AB}}\}_{K_B^+}$$

to obtain the correct guess of P.

Remark. An explicit guess might turn out to be an implicit one, due to the redundancy in explicit guesses. For example, suppose the attacker knows $\{N_A, \{N_A \cdot P\}_{K_{AS}}\}$ and it makes explicit guesses of K_{AS} and P. Note that

$$\mathtt{snd}(\mathtt{dec}(\{N_A \cdot P\}_{K_{AS}}, K_{AS})) =_{E_{dy}} P$$

It is not hard to see that P can be derived from the explicit guess of K_{AS}. So, there is no need to make explicit guess of P. We postpone to Section 6 some further discussion of the redundancy in explicit guesses.

4.2 A New Knowledge Model

We now define a new notion to describe the attacker's knowledge that accounts for the attacker's guessing capabilities.

Definition 4.1 (Markup Term Set)**.** *A markup term set, notated as $\vec{T}$, is a pair $\langle T, \sigma \rangle$, where σ is a ground substitution such that $Dom(\sigma) = fv(T)$.*

Here, all ground terms and free variables in T correspond to its explicit knowledge and explicit guesses, respectively. We use the substitution σ to indicate either the correct guess value or a possible guess value. In the analysis of type-flaw attack [36, 43], free variables in T correspond to potentially ambiguous incoming messages.

This definition accords with the possible worlds semantics for knowledge [30], in which the true state resides in one of the many possible states. More specifically, if σ and σ' correspond to the correct guess and a possible guess, respectively, then $\langle T, \sigma\rangle$ and $\langle T, \sigma'\rangle$ are the actual state and a possible one. It is worth pointing out that the "true" state does not account for the attacker's true knowledge state, but rather describes the expected state from the attacker's point of view. To avoid confusion, we will informally refer to σ and σ' as the *expected substitution* and *possible substitution*, respectively. Similarly, $\langle T, \sigma\rangle$ (resp. $\langle T, \sigma'\rangle$) is regarded as the *expected* (resp. *possible*) *markup term set* (or *state*).

As usual, we require the possible worlds to be indistinguishable to an agent from the true world/state. This means that the attacker should not be able to distinguish the expected state from all possible states. In our terminology, they comply with the operational equivalence relation, i.e., they fit into one equivalence class. Therefore, both the expected state and possible state suffice to model the knowledge of an attacker with guessing capabilities.

For instance, in Example 4, the attacker's knowledge can be modeled by

$$\langle\{N_A, K_B^+, \{N_A \cdot P\}_{K_B^+}, x\}, [P/x]\rangle$$

and

$$\langle\{N_A, K_B^+, \{N_A \cdot P\}_{K_B^+}, y\}, [K_B^-/y]\rangle$$

corresponding the two explicit ways of guessing.

5. A COMPLETE CHARACTERIZATION OF GUESSING

In this section, we introduce a weaker notion of verifiability to fully characterize the intuitive understanding of guessing.

5.1 Weak Verifiability

The possible-worlds semantics lends more sense to recognizability: a term t (indicated by x) is recognizable if and only if x indicates t (i.e., $x\sigma =_E t$) in all possible states. This suggests a more general definition of recognizability, which extends Definition 3.3 to the case of multiple free variables (indicating potentially ambiguous messages [39] or unchecked guesses).

Definition 5.1 (Weak Recognizability). *Let $\vec{T} = \langle T, \sigma_0\rangle$ be a markup term set and t be a ground term. We say that t is* weakly recognizable *by $\vec{T}$ under equational theory E and write $\vec{T} \rhd_E t$ if $x\sigma =_E t$ for all σ satisfying $\sigma \approx_{E,T\cup\{x\}} (\sigma_0 \circ [t/x])$ where x is a fresh variable.*

Example 6. *Consider again Example 5. The attacker's knowledge is modeled by*

$$\vec{T} = \langle\{N_A, \{N_A{\cdot}K_{AB}\}_{K_A^+}, \{N_A{\cdot}\{P\}_{K_{AB}}\}_{K_B^+}, x, y\}, [K_B^-/x, P/y]\rangle$$

in which x and y correspond to two distinct explicit guesses made by the attacker. Then, $\vec{T} \rhd_{E_{dy}} P$. However, if the attack only makes a single guess, either K_B^- or P, then $\vec{T}' \not\rhd_{E_{dy}} P$, where $\vec{T}'$ is either

$$\langle\{N_A, \{N_A \cdot K_{AB}\}_{K_A^+}, \{N_A \cdot \{P\}_{K_{AB}}\}_{K_B^+}, x\}, [K_B^-/x]\rangle$$

or

$$\langle\{N_A, \{N_A \cdot K_{AB}\}_{K_A^+}, \{N_A \cdot \{P\}_{K_{AB}}\}_{K_B^+}, y\}, [P/y]\rangle$$

At this point, one may be tempted to conjecture that this weaker notion of recognizability suffices to describe the desired new notion of verifiability, as the stronger notion (Definition 3.3) does. Unfortunately, this is not the case, because in Definition 5.1 $[t/x]$ is composed with σ_0, introducing a new explicit guess of t, as shown by the following example.

Example 7. *Let*

$$\vec{T} = \langle\{N_A, \{(N_A \cdot N_B) \cdot \{N_A\}_{K_B^+}\}_{K_{AS}}, x\}, [K_{AS}/x]\rangle$$

denotes the attacker's knowledge. Suppose that the attacker wants to obtain K_B^+. Note that the attacker only makes one explicit guess of K_{AS}. It is not hard to see that the attacker indeed can correctly guess K_{AS}. Then, the attacker's knowledge becomes $\vec{T}' = \langle\{N_A, N_B, K_{AS}, \{N_A\}_{K_B^+}\}, \phi\rangle$. Now, it is not hard to see that, without any further guess(es), the attacker is still not able to obtain K_B^+. On the other hand, however, it can be shown that $\vec{T} \rhd_{E_{dy}} K_B^+$.

There is one simple fix to avoid adding the new explicit guess. As explained earlier, an explicit guess may turn out to be an implicit one by exploiting the redundancy in explicit guesses. The trick is that we impose condition(s) to ensure that the newly added explicit guess becomes an explicit one.

Definition 5.2 (Weak Verifiability). *Let $\vec{T} = \langle T, \sigma_0\rangle$ be a markup term set and t be a ground term. We say that t is* weakly verifiable *by $\vec{T}$ under equational theory E and write $\vec{T} \blacktriangleright_E t$ if $\vec{T} \rhd_E t$ and $T\sigma_0 \vdash_E t$.*

The condition $T\sigma_0 \vdash_E t$ implies that $T \vdash s$ and $s\sigma_0 =_E t$ for some s. In other words, the explicit guess can be exactly described by using T, obviating the need to explicitly guess t. The following lemma states this formally.

Lemma 5.3. *Let $\vec{T} = \langle T, \sigma_0\rangle$ be a markup term set and t be a ground term. If $\vec{T} \blacktriangleright_E t$, then there exists a term s such that $T \vdash s$ and $s\sigma_0 =_E s\sigma =_E t$ for all $\sigma \approx_{E,T} \sigma_0$.*

Proof. By Definition 5.2, we have $\vec{T} \rhd_E t$ and $T\sigma_0 \rhd_E t$. Then, it follows from Lemma 2.1 that there exists a term s such that $T \vdash s$ and $s\sigma_0 =_E t$. It remains to show that $s\sigma =_E t$ for all $\sigma \approx_{E,T} \sigma_0$.

Let $s\sigma =_E t'$ and x be a fresh variable. Since $\sigma \approx_{E,T} \sigma_0$, we get $\sigma \circ [t'/x] \approx_{E,T\cup\{x\}} \sigma_0 \circ [t/x]$. Moreover, since $\vec{T} \rhd_E t$, we thus have $x\sigma \circ [t'/x] =_E t$ by Definition 5.1. Hence, $t' =_E t$. This completes the proof. □

Recall the example given at the end of Section 4.1, where the attacker knows N_A and $\{N_A \cdot P\}_{K_{AS}}$. Suppose that it only makes one explicit guess of K_{AS} and aims to obtain P. Then, his knowledge is represented by

$$\vec{T} = \langle\{N_A, \{N_A \cdot P\}_{K_{AS}}, x\}, [K_{AS}/x]\rangle$$

Moreover, it can be shown that $\vec{T} \rhd_{E_{dy}} P$ and $T[K_{AS}/x] \vdash_{E_{dy}} P$. That is, P is weakly verifiable by $\vec{T}$. Here, the attacker needs not to explicitly guess P.

On the contrary, in Example 7, we notice that

$$\{N_A, \{(N_A \cdot N_B) \cdot \{N_A\}_{K_B^+}\}_{K_{AS}}, x\}[K_{AS}/x] \nvdash_{E_{dy}} K_B^+$$

Thus, as noted before, the attacker has to make other explicit guess(es) (e.g., a guess of K_B^+) to obtain K_B^+.

5.2 Guessability

Finally, we coin the term guessability (i.e., the attacker's ability to guess) in terms of weak verifiability.

Definition 5.4 (Guessability). *Let $\vec{T}$ be a markup term set representing the attacker's knowledge. We say that a ground term t is* guessable *by the attacker if $\vec{T} \blacktriangleright_E t$.*

This provides the last step to formalize and justify the long held intuition between "guess" and "verify".

Noticing that the attacker's knowledge should be updated to $\langle T \cup \{t\}, \sigma\rangle$ if $\langle T, \sigma\rangle \blacktriangleright_E t$, one may reasonably think that we need to recursively add new guessable terms into the attacker's knowledge until no new guessable term can be found. It seems probable that Definition 5.4 fails to account for this dynamics.

Somewhat surprisingly, we find that adding t into the attacker's knowledge makes no difference in terms of guessability. The following theorem states this formally and justifies the Definition 5.4.

Theorem 5.5. *Suppose that $\langle T, \sigma_0\rangle \blacktriangleright_E s$. Then, $\langle T, \sigma_0\rangle \blacktriangleright_E t$ if and only if $\langle T \cup \{s\}, \sigma_0\rangle \blacktriangleright_E t$.*

6. HARDNESS OF GUESSING

Until now we have mainly focused on the possibility of guessing. In this section, we concern ourselves with the hardness of guessing, that is, how much computational efforts are required to obtain a guessable term t, provided $\vec{T} \blacktriangleright t$.

It should be noted that different guessing problems incur different computational cost. For example, (explicitly) guessing a 128-bit symmetric key is significantly harder than guessing a poorly chosen password. In fact, there is a physical argument [37] that implies that guessing a 128-bit symmetric key is "practically infeasible". Moreover, even for the same guessing problem, the efforts can vary considerably in different ways of (explicit) guessing. For instance, in Example 4, the attacker can either explicitly guess P or explicitly guess K_B^- to obtain P. Let us assume K_B^- is a 1024-bit private key and P is a poorly chosen password. Then, guessing P could be much easier than guessing K_B^-.

Thus, despite the guessability results, we also need a new notion to characterize the hardness of guessing. One may think of using the binary length of all the explicit guesses. Unfortunately, this simple way may fail to faithfully characterize the hardness, as the following examples show.

Example 8. *Let us consider two scenarios, in which the attacker's knowledge is, respectively, represented by*

$$\vec{T_1} = \langle\{N_A, \{N_A \cdot P\}_{K_{AB}}, \{N_A \cdot K_A^+\}_{K_{AS}}\}, x, y\},\\ [K_{AB}/x, K_{AS}/y]\rangle$$

and

$$\vec{T_2} = \langle\{N_A, \{\{N_A \cdot P\}_{K_B^+}\}_{K_{AB}}, \{K_B^-\}_{K_{AS}}, x, y\},\\ [K_{AB}/x, K_{AS}/y]\rangle$$

Suppose that the attacker wants to obtain $\{P\}_{K_A^+}$ in the first scenario and P in the second. In both cases, these can be done by explicitly guessing K_{AB} and K_{AS}. It is tempting to conclude that guessing $\{P\}_{K_A^+}$ and P is equally difficult.

However, a closer examination reveals the difference.

In the first scenario, the attacker can use

$$\mathtt{fst}(\mathtt{dec}(\{N_A \cdot P\}_{K_{AB}}, x))\sigma =_{E_{dy}} N_A \qquad (1)$$

to obtain the correct guess of K_{AB}. Note that Equation (1) does not involve the guess of K_{AS}. So, the attacker can correctly guess K_{AB} without guessing K_{AS}. Similarly, we see that the attacker can also correctly guess K_{AS} without guessing K_{AB}. After correctly guessing K_{AB} and K_{AS}, the attacker can easily get P and K_A^+, and thus derive $\{P\}_{K_A^+}$. To sum up, the maximum number of times the attacker has attempted to obtain $\{P\}_{K_A^+}$ is $2^{|K_{AB}|} + 2^{|K_{AS}|}$.

On the contrary, in the second scenario, the attacker can only use

$$\mathtt{fst}(\mathtt{dec}(\mathtt{dec}(\{\{N_A \cdot P\}_{K_B^+}\}_{K_{AB}}, x), y))\sigma =_{E_{dy}} N_A \qquad (2)$$

to obtain the correct guesses of K_{AB} and K_{AS}, and thus derive P. This means the attacker has to guess K_{AB} and K_{AS} simultaneously. Hence, the maximum number of times it has attempted to obtain P is $2^{|K_{AB}|+|K_{AS}|}$.

Therefore, guessing in the second scenario is considerably harder than in the first scenario.

Example 9. *Let*

$$\vec{T} = \langle\{N_A, \{N_A \cdot P\}_{K_{AB}}, \{K_{AS}\}_P, \{N_A \cdot K_B^+\}_{K_{AS}}, x, y\},\\ [K_{AB}/x, K_{AS}/y]\rangle$$

denotes the attacker's knowledge. Suppose that the attacker wants to obtain $\{P\}_{K_B^+}$. Similar to the first scenario in the previous example both explicit guesses (of K_{AB} and K_{AS}) can be made independently. But we have to be careful not to conclude that the maximum number of times the attacker has attempted to obtain $\{P\}_{K_B^+}$ is also $2^{|K_{AB}|+|K_{AS}|}$.

Let us take a closer look at $\vec{T}$. We notice that after obtaining the correct guess of K_{AB} the attacker can use $\mathtt{snd}(\mathtt{dec}(\{N_A \cdot P\}_{K_{AB}}, K_{AB})) =_{E_{dy}} P$ to derive P, which can be further used to derive K_{AS} as $\mathtt{dec}(\{K_{AS}\}_P, P) =_{E_{dy}} K_{AS}$. So, the attacker can derive K_{AS} only by a single explicit guess of K_{AB}. In other words, the maximum number of times the attacker has attempted is just $2^{|K_{AB}|}$.

As noted in the above examples, the number of bits that the attacker has to guess might be less than that of all explicit guesses. There are two main reasons for this: (i) some explicit guess(es) can be readily made without dealing with other guesses, dividing an overall hard guess problem into several easier ones; and (ii) the redundancy inherent in all the explicit guesses makes it possible to derive useful information between them.

We thus propose to use the search space, rather than the number of bits of the explicit guesses, to characterize the hardness of guess.

Definition 6.1 (Hardness). *We define $\boldsymbol{minmax}(\vec{T} \blacktriangleright t)$ as the minimum maximum number of times one might attempt to obtain t. Moreover, we say that the hardness of $\vec{T} \blacktriangleright t$ is* in order of n *(or* n-bit hard*) if $n = \lceil \log_2 \boldsymbol{minmax}(\vec{T} \blacktriangleright t)\rceil$.*

Now, it is not hard to see that $\vec{T_1} \blacktriangleright \{P\}_{K_A^+}$ and $\vec{T_2} \blacktriangleright P$ in Example 8 are in order of $\log_2(2^{|K_{AB}|} + 2^{|K_{AS}|})$ and $|K_{AB}| + |K_{AS}|$, respectively; $\vec{T} \blacktriangleright \{P\}_{K_B^+}$ in Example 9 is in order of $|K_{AB}|$.

Remark. Although Definition 6.1 allows us to evaluate the hardness of guess accurately, it does not provide much insight into how to determine $\mathbf{minmax}(\vec{T} \blacktriangleright t)$ and thus the hardness of $\vec{T} \blacktriangleright t$. Obviously, much future work remains to be done for solving $\mathbf{minmax}(\vec{T} \blacktriangleright t)$. There are two issues to be considered in addressing this problem: first, to explore the redundancy in those explicit guesses, and second, to partition the explicit guesses into groups that can be done without involving others. We do not explore these issues further in this paper.

7. DETECTING GUESSING ATTACKS

In this section, we briefly discuss how the proposed framework can be used effectively in detecting guessing attacks.

7.1 A Cognitive Perspective

Before diving into the technical discussion, it helps to have a clear distinction between passive and active attacks (not just guessing attacks).

Passive attack.

The passive attacker does not interact with protocol participants; whether or not it can launch an attack solely based upon the eavesdropped data. We thus informally view the passive attack as a *computing* problem: given a set of observed messages, whether it is possible to "compute" confidential data.

In the literature, intruder deduction [15, 1, 21, 18] and static equivalence [2, 1, 7, 12] correspond to this computational view, where computing is regarded as a knowledge reasoning process.

Active attack.

Besides its ability to reason about knowledge as the passive attacker, the active attacker can also communicate with legitimate participants. Benefit from a cognitive perspective, this can be understood in two complementary ways:

1. (Communication view) we can think of communication with external entities as a way of gaining new information that cannot be deduced from its current knowledge.

2. (Computational view) we can regards the external entities as as an internal oracle that computes new information from its current knowledge.

Example 10. *Let us consider again the protocol presented in the introduction:*

Message 1. $A \to B : \{N_A\}_{K_{AB}}$
Message 2. $B \to A : \{f(N_A)\}_{K_{AB}}$

An active attacker can act in the role of A initiate communication with B. Assume that the attacker's knowledge is represented by term set $T_I = \{I, A, B, \{N_A\}_{K_{AB}}\}$.

From a communication point of view, the attacker does not know $\{f(N_A)\}_{K_{AB}}$ *(i.e.,* $T_I \nvdash_{E_{dy}} \{f(N_A)\}_{K_{AB}}$*) at first. Only after exchanging messages with B, it obtain message* $\{f(N_A)\}_{K_{AB}}$ *and thus its knowledge becomes*

$$T_I' = \{I, A, B, \{N_A\}_{K_{AB}}, \{f(N_A)\}_{K_{AB}}\}$$

Clearly,

$$T_I \not\equiv_{E_{dy}} T_I' \qquad (3)$$

From a computational point of view, the attacker is endowed with an oracle that takes t as input and outputs

$$\mathsf{g}(t) = \mathsf{enc}(f(\mathsf{dec}(t, K_{AB})), K_{AB}) \qquad (4)$$

where g *is a public function symbol that never occurs in the original term algebra* $\mathcal{T}$. *As the oracle is internal, we thus incorporate the above equation to equation theory* E_{dy} *and get* E'_{dy}. *Therefore,*

$$T_I \equiv_{E'_{dy}} T_I' \qquad (5)$$

In this light, we can categorize the security protocol models into two groups: one is based on communication view, such as Strand Space Model [28], CSP [45], and applied pi-calculus [2]; the other is based on computational view, such as multiset rewriting [10], constraint solving[44], Prolog rules [5], and Horn clauses [6].

We remark that a clear distinction between passive and active attack enables us to determine whether the attack is primarily due to the attacker's knowledge or its interaction with legitimate participants. Moreover, a thorough understanding of passive attacks will shed important light on the study of active attacks and security protocol design as well.

7.2 Passive Guessing Attacks

In terms of passive guessing attack, the knowledge reasoning problem is that, given a set of observed messages, whether it is at all possible to correctly guess any confidential data.

Our framework formulates the above knowledge reasoning problem accurately. We use term set T to describe the set of observed messages, term t to represent some confidential data, variables set X to correspond to all the guess made by the attacker, and substitution σ with $Dom(\sigma) = X$ to indicate the correct guesses. Because passive eavesdropping is performed over legitimate protocol sessions, observed messages must comply with the protocol specification and thus we can assume T to be a ground term set. Likewise, t is also ground. Then, markup term set $\langle T \cup X, \sigma\rangle$ models the passive attacker's knowledge. Finally, the problem of detecting passive guessing attacks is reduced to deciding $\langle T \cup X, \sigma\rangle \blacktriangleright_E t$.

At this point, detection of passive guessing attacks boils down to deciding guessability. The last missing step is to give a decision procedure for $\langle T \cup X, \sigma\rangle \blacktriangleright_E t$. Unfortunately, in general, this may be undecidable [1].

Deciding Guessability under standard Dolev-Yao intruder model.

Recently, Li and Wang [38] proposed a terminating procedure to determine recognizability under standard Dolev-Yao intruder model [25]. Here, we adopt this procedure to decide guessability under Dolev-Yao model.

Although the original procedure (i.e., algorithm **Solve**) is intended for deciding strong recognizability (Definition 3.3), it can be easily extended to weak recognizability, as required in Definition 5.4. At first, we extend the definition of markup term set to a triple $\langle T, \eta, \sigma\rangle$, which includes a second substitution η that account for partial solution. Then, algorithm returns a new triple $\langle T', \eta', \sigma'\rangle$ in solved form. More formally,

Theorem 7.1. *Let* $\langle T, \sigma\rangle$ *be a markup term set,* t *be a ground term, and* x *be a fresh variable. Suppose that* $T\sigma \cup$

$\{t\}$ does not contain function symbol `fst`*,* `snd`*, or* `dec`*. If $T\sigma \vdash_{E_{dy}} t$,* ***Solve***$(\langle T \cup \{x\}, \phi, \sigma \circ [t/x]\rangle)$ *returns $\langle T', \eta', \sigma'\rangle$, and $x\eta' =_s t$, then $\vec{T} \blacktriangleright_{E_{dy}} t$.*

Please refer to [38] for more details on the algorithm.

7.3 Extension to Active Guessing Attacks

To handle an active attacker, it is important to model security protocols. As mentioned in Section 7.1, existing formal methods for protocol modeling fall into two groups: communication based and computation based.

For simplicity, we adopt a computational view here: we regard the active attacker as a special passive attacker with an oracle. More specifically, we can add equations describing the oracle to the original equational theory. For instance in Example 10, we just add Equation 4 to equation theory E_{dy} (and obtain equational theory E'_{dy}). This method is similar to that of [3], which uses a set of second-order variables to keep track of the computations. In general, a symbolic trace [29, 8, 14] that describes the sequences of actions (receive or send) of a given protocol role brings about n distinct equations, where n is the number of messages sent by the role.

By extending the original equational theory, we get a new equational theory, say E', to model the active attacker's capabilities[3]. Therefore, the problem of detecting active guessing attack boils down to deciding guessability under the new equational theory E'.

It should be noted that deciding $\blacktriangleright_{E'}$ may be undecidable. After all, the our approach considers an unbounded number of sessions of the protocol [46, 11], for which protocol insecurity is undecidable [27]. Approximation techniques [20, 6] are usually employed to handle unbounded verification. Due to space limit, we do not pursue these further here.

Active guessing attack is passive guessing attack?.

Thanks to the clear distinction between passive and active attack, we find surprisingly that in many cases the enhanced capabilities of active attacker does not impact guessability at all; that is to say, active attacker is no more powerful than passive attacker in term of guessability.

For example, in the protocol given at the beginning of the introduction, if an attacker knows $\{\{N_A\}_{K_{AB}}, \{\mathtt{f}(N_A)\}_{K_{AB}}\}$ and makes explicit guess of K_{AB}, then all actively guessable terms are actually passively guessable, as the following proposition shows.

Proposition 7.2. *Let $\vec{T}$ be a markup term set and t be a ground term. Suppose that*

$$\vec{T} = \langle\{\{N_A\}_{K_{AB}}, \{f(N_A)\}_{K_{AB}}, x\}, [K_{AB}/x]\rangle$$

and t does not contain function symbol `g`*,* `dec`*,* `fst`*, or* `snd`*. Then, $\vec{T} \blacktriangleright_{E'_{dy}} t$ if and only if $\vec{T} \blacktriangleright_{E_{dy}} t$.*

8. CONCLUSION

In this paper, we present a general framework of guessing, which clarifies and formalizes the intuitive understanding of "verifying a guess". Thanks to its following innovative features

- independence of any specific adversary model,
- support of multiple (explicit) guesses, and
- definition to measure the hardness of guessing

this framework enables us to detect passive and active guessing attacks, both of which rely critically on the decision problem $\rhd_E$.

Apart from the technical contributions of this paper, other messages we want to convey are that passive attacks are as important as active attacks, especially in the study of guessing attacks; and that both communication and computational views of active attacks may offer new insight in security protocol analysis.

There are two major limitations of this study. First, the standard Dolev-Yao model considered in Section 7.2 assumes "perfect encryption", that is, $\{m\}_k =_{E_{dy}} \{m'\}_{k'}$ if and only if $m =_{E_{dy}} m'$ and $k =_{E_{dy}} k'$. Such an assumption is unrealistic for cryptographic primitives with visible algebraic properties such as exclusive or and homomorphic operator, see [19] for a survey. Second, our definition of hardness is too general to be practically useful and it is non-trivial to determine $minmax(\vec{T} \blacktriangleright t)$. Moreover, our analysis in Example 8 and 9 assumes a uniform distribution of the guessing value and thus there is no better way than brute force guessing. However, in reality, weak secret (say, n bits) usually has low entropy, making it easier to guess ($< n$-bit hard).

Our future work will be aimed at addressing these limitations. In particular, we plan to investigate the problem of detecting guessability under more general equational theory and develop automatic tools to detecting guessing attack.

[3]In fact, the original term algebra $\mathcal{T}$ is also extended to $\mathcal{T}'$, which includes several new public function symbols modeling the oracle computation.

9. REFERENCES

[1] M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theor. Comput. Sci.*, 367(1):2–32, 2006.

[2] M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *POPL '01*, pages 104–115. ACM, 2001.

[3] M. Baudet. Deciding security of protocols against off-line guessing attacks. In *CCS '05*, pages 16–25. ACM, 2005.

[4] G. Birkhoff. On the structure of abstract algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(04):433–454, 1935.

[5] B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *CSFW '01*, page 82, 2001.

[6] B. Blanchet. Automatic verification of correspondences for security protocols. *J. Comput. Secur.*, 17(4):363–434, 2009.

[7] B. Blanchet, M. Abadi, and C. Fournet. Automated verification of selected equivalences for security protocols. *Journal of Logic and Algebraic Programming*, 75(1):3 – 51, 2008.

[8] M. Boreale and M. G. Buscemi. A method for symbolic analysis of security protocols. *Theoretical Computer Science*, 338(1-3):393 – 425, 2005.

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

[10] I. Cervesato, N. A. Durgin, P. D. Lincoln, J. C. Mitchell, and A. Scedrov. A meta-notation for protocol analysis. In *CSFW '99*, page 55, 1999.

[11] Y. Chevalier and L. Vigneron. Automated unbounded verification of security protocols. In *CAV '02*, pages 324–337. Springer-Verlag, 2002.

[12] c. Ciobâcă, S. Delaune, and S. Kremer. Computing knowledge in security protocols under convergent equational theories. In *CADE-22*, pages 355–370. Springer-Verlag, 2009.

[13] M. Cohen and M. Dam. A complete axiomatization of knowledge and cryptography. In *LICS '07*, pages 77–88, 2007.

[14] H. Comon-Lundh and V. Cortier. Computational soundness of observational equivalence. In *CCS '08*, pages 109–118. ACM, 2008.

[15] H. Comon-Lundh and V. Shmatikov. Intruder deductions, constraint solving and insecurity decision in presence of exclusive or. In *LICS '03*, pages 271–280, June 2003.

[16] R. Corin, J. Doumen, and S. Etalle. Analysing password protocol security against off-line dictionary attacks. *Electron. Notes Theor. Comput. Sci.*, 121:47–63, 2005.

[17] R. Corin, S. Malladi, J. Alves-Foss, and S. Etalle. Guess what? here is a new tool that finds some new guessing attacks. In R. Gorrieri and R. Lucchi, editors, *IFIP WG 1.7*, pages 62–71, 2003.

[18] V. Cortier and S. Delaune. Deciding knowledge in security protocols for monoidal equational theories. In *LPAR*, pages 196–210, 2007.

[19] V. Cortier, S. Delaune, and P. Lafourcade. A survey of algebraic properties used in cryptographic protocols. *J. Comput. Secur.*, 14(1):1–43, 2006.

[20] C. J. Cremers. Unbounded verification, falsification, and characterization of security protocols by pattern refinement. In *CCS '08*, pages 119–128. ACM, 2008.

[21] S. Delaune. Easy intruder deduction problems with homomorphisms. *Information Processing Letters*, 97(6):213 – 218, 2006.

[22] S. Delaune and F. Jacquemard. A theory of dictionary attacks and its complexity. In *CSFW '04*, page 2, 2004.

[23] N. Dershowitz and D. A. Plaisted. Rewriting. In *Handbook of Automated Reasoning*, pages 535–610. MIT Press, 2001.

[24] Y. Ding and P. Horster. Undetectable on-line password guessing attacks. *SIGOPS Oper. Syst. Rev.*, 29(4):77–86, 1995.

[25] D. Dolev and A. Yao. On the security of public key protocols. *Information Theory, IEEE Transactions on*, 29(2):198–208, Mar 1983.

[26] P. H. Drielsma, S. Modersheim, and L. Vigano. A formalization of off-line guessing for security protocol analysis. In *Logic for Programming, Artificial Intelligence, and Reasoning*, volume 3452, pages 363–379. 2005.

[27] N. Durgin, P. Lincoln, J. Mitchell, and A. Scedrov. Multiset rewriting and the complexity of bounded security protocols. *J. Comput. Secur.*, 12(2):247–311, 2004.

[28] F. Fabrega, J. Herzog, and J. Guttman. Strand spaces: why is a security protocol correct? pages 160 –171, may 1998.

[29] F. J. T. Fábrega. Strand spaces: proving security protocols correct. *J. Comput. Secur.*, 7(2-3):191–230, 1999.

[30] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge*, volume 1 of *MIT Press Books*. The MIT Press, December 2003.

[31] L. Gong. Optimal authentication protocols resistant to password guessing attacks. In *CSFW '95*, page 24, 1995.

[32] L. Gong, M. Lomas, R. Needham, and J. Saltzer. Protecting poorly chosen secrets from guessing attacks. *Selected Areas in Communications, IEEE Journal on*, 11(5):648 –656, jun. 1993.

[33] B. Groza and M. Minea. A calculus to detect guessing attacks. In *ISC '09*, pages 59–67. Springer-Verlag, 2009.

[34] S. Halevi and H. Krawczyk. Public-key cryptography and password protocols. *ACM Trans. Inf. Syst. Secur.*, 2(3):230–268, 1999.

[35] J. Halpern, Y. Moses, and M. Vardi. Algorithmic knowledge. In *Proc. of 5th conference on Theoretical Aspects of Reasoning about Knowledge*, pages 255–266, 1994.

[36] J. Heather, G. Lowe, and S. Schneider. How to prevent type flaw attacks on security protocols. *J. Comput. Secur.*, 11(2):217–244, 2003.

[37] R. Landauer. Irreversibility and heat generation in the computing process. *IBM Journal of Research and Development*, 44(1.2):261 –269, jan. 2000.

[38] Z. Li and W. Wang. Deciding recognizability under dolev-yao intruder model. In *ISC '10*, to appear.

[39] Z. Li and W. Wang. Rethinking about type-flaw attacks. In *Global Telecommunications Conference, 2010. GLOBECOM 2010. IEEE*, to appear.

[40] T. Lomas, L. Gong, J. Saltzer, and R. Needhamn. Reducing risks from poorly chosen keys. *SIGOPS Oper. Syst. Rev.*, 23(5):14–18, 1989.

[41] G. Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In *TACAs '96*, pages 147–166, 1996.

[42] G. Lowe. Analysing protocols subject to guessing attacks. *J. Comput. Secur.*, 12(1):83–97, 2004.

[43] C. Meadows. A procedure for verifying security against type confusion attacks. In *CSFW 03*, pages 62–72, 2003.

[44] J. Millen and V. Shmatikov. Constraint solving for bounded-process cryptographic protocol analysis. In *CCS '01*, pages 166–175. ACM, 2001.

[45] S. Schneider. Security properties and csp. In *SP '96*, page 174, 1996.

[46] D. X. Song, S. Berezin, and A. Perrig. Athena: a novel approach to efficient automatic security protocol analysis. *J. Comput. Secur.*, 9(1-2):47–74, 2001.