# The Foul Adversary: Formal Models

Naipeng Dong[1] and Tim Muller[2]

[1] National University of Singapore, Singapore
[2] University of Oxford, UK

**Abstract.** In classical notions of privacy in computer security, users attempt to keep their data private. A user that is bribed, extorted or blackmailed (i.e., *coerced*) may not do so. To get a general model of coercion, we strengthen the Dolev-Yao adversary with the ability to coerce others, to the *foul adversary*. We show that, depending on the setting, subtly different abilities should be assigned to the adversary – whereas existing approaches are one-size-fits-all. The variations of the foul adversary are formalised and we provide a hierarchical relation in their strength. We further interpret the adversary models using several examples.

## 1 Introduction

Privacy is increasingly important in internet-based services [5, 9, 20]. A new privacy notion - *enforced privacy* - arose, which assumes users (are forced to) reveal private information due to, e.g., bribery or extortion (*coercion*). Vote-buying, bribed doctors and rigged auctions are real-life examples of voters/doctors/bidders revealing information that should be private. In domains like e-voting, e-auction and e-health, coercion must, therefore, be prevented [3, 12, 19]; the systems should enforce a user's privacy even when the user reveals his private information. The basic idea is that if a system provides a way for the coerced user to mislead the attacker, then the system enforces privacy of the user [14, 24][1]. In Section 2, we further explain the mechanism mitigating coercion.

There are cryptographic protocols that ensure enforced privacy [3, 12, 19, 26, 27]. As the design of cryptographic protocols is well-known to be error-prone and flaws in such protocols are often subtle and counter-intuitive, formal verification is an important step before implementation. There are multiple ways to formalise enforced privacy. Currently, a standard method (proposed by Benaloh and Tuinstra [7] and later symbolically formalised by Delaune et al [14]) is to encode a privacy property as the formal equivalent of "even if the user gives up his private information, the (Dolev-Yao [15]) adversary cannot be sure that this really is his private information". This method does not generalise to security properties other than privacy. We propose an alternative method, which is to keep the security/privacy property unaltered ("the adversary cannot know the

[1] Note that bribed, extorted or blackmailed users differ from compromised users (e.g., [6]) - a coerced user is assumed to lie to the attacker if possible whereas a compromised user is assumed to be totally controlled by the attacker.

user's private information"), but to verify it under an adversary that has the power to coerce; the *foul* adversary.

Our approach philosophically differs from the existing approaches. The existing approaches enhance the security requirements (privacy becomes enforced privacy) of the system in question. Our approach is the first to allow reasoning about coercion even in absence of a concrete security system or protocol. We give the attacker the ability to (try to) coerce whenever he desires. Like any attacker, he has the capability of reasoning about his knowledge, and inserting it into a protocol; the only difference is that there may now be coerced data in his knowledge. Our approach is rooted in similar formal techniques, meaning automated verification is also feasible.

The main advantage of the standard method [14] is that it already has some tool support (e.g., ProVerif [8]), and there are various case-studies using the method [4, 16, 17]. The advantages of our proposed method are: 1) it generalises to security properties other than privacy, 2) it allows a greater degree of fine-tuning, and 3) it makes the assumptions of coercion explicit. To illustrate why it pays to have assumptions explicit: The voting protocol in [14] makes an implicit assumption that it suffices to keep the vote of the user enforced private. However, as Küsters et al [24] point out, the fact that a person voted at all may need to be enforced private. Küsters et al [24] have an alternative proposal for enforced privacy. We discuss both methodologies in Section 2.

To motivate our alternative approach to coercion, take a frivolous example (more technical and relevant examples are given later) where residents are to be protected against potentially violent burglars. At a burglary, a burglar may threaten a resident to enter a code to disable the alarm. A duress code (or panic code) is a code that disables the sirens and lights, making it appear the alarm is disabled, but in reality notifies the police that a burglar is coercing a resident. When the system ensures that the residents have the code, then the alarm is coercion resistant[2] [10]. The precise details matter when the system allows cases where the residents never received the duress code, or where the residents never configured the duress code. We refer to this issue as the user knowledge aspect, as the crucial question is whether it is sufficient when a user could know something, or whether he actually needs to know it. Moreover, if the burglars are sufficiently notorious, then residents may forgo using their duress code, as they fear retribution from the burglar even after their arrest. We refer to this issue as the dynamics aspect, as the crucial question is how potential futures influence the present. The user knowledge aspect and the dynamics aspect are orthogonal issues, which can be individually fine-tuned in our approach.

*Contributions* We formalise and investigate a family of adversaries – *foul adversaries* – that extend the Dolev-Yao adversary with the ability to coerce. There

[2] Or at least somewhat coercion resistant. If the burglar is aware of the existence of a duress code, he could elicit two codes that turn off the alarm, knowing that one of the two must be the real code, and decrease the odds of a silent alarm from 100% to 50%. A burglar would typically still be deterred with a 50% probability of a silent alarm going off. We do not further investigate probabilistic scenarios.

are two orthogonal aspects that determine the strength of the foul adversaries, the user knowledge aspect and the dynamic aspect. We prove a hierarchy of these 8 ($2 \times 4$) different foul adversaries, and illustrate the foul adversaries using practical examples of protocols.

*Paper Organization* In Section 2, we introduce the context of our approach. In Section 3, we define the core of our foul adversary in the form of knowledge and reasoning. In Section 4, we introduce the notion of security systems and further formalise the variants of the foul adversary. Then we introduce examples to illustrate our approach and to concretely link it to security systems, in Section 5. Finally, we conclude in Section 6.

## 2 Coercion

Coercion involves an adversary forcing a user to say (or do) something against their will. However, unlike a controlled user, these coerced users may say (or do) something else without the adversary noticing. All definitions explicitly deal with the fact that users only actually say (or do) what the adversary demands when the adversary can distinguish if the user does not comply.

Currently, there is research on coercion in the literature and defending against coercion in protocol design. We discuss these approaches and their successes below, in *Existing Methods*. The approaches have in common that they see coercion as part of the security requirements. In *Foul Adversary*, we discuss the exact differences resulting from making coercion part of the adversary's abilities.

*Existing Methods* The requirement to prevent coercion was first proposed in e-voting systems [7]. Cryptographic e-voting protocols have been proposed to meet this requirement (e.g., [27, 26]). To formally verify these protocols, formalisations of enforced privacy in e-voting were proposed to capture the requirements, for instance quantitative receipt-freeness and coercion-resistance [22], coercion-resistance in game-based provable security style [25], coercion-resistance using epistemic approach [24], and receipt-free and coercion-resistance using process algebras, e.g., in the applied pi calculus [12] and in CSP [21]. Later, the enforced privacy requirements have been found in other domains, such as e-auctions [3] and e-health [12]. Formalisations of enforced privacy properties in e-auctions and e-health have also been proposed, following the framework in the applied pi calculus [22, 16, 17]. Thus, systems wherein coercion may occur are a growing phenomenon, occurring in many new security domains.

The definition of enforced privacy by Delaune et al (in [13]) is particularly influential, as it is the first symbolic formal definition of enforced privacy that is generalised over protocols. The definition, however, did not generalise nicely over different domains, as it was specifically intended for e-voting. Voters may be bribed to vote for a certain candidate, and receive benefits only if they can prove that they voted for that candidate. Thus, it is not sufficient that the adversary cannot invade your privacy and obtain a proof of your vote, but the protocol must prevent users from providing the proof to the adversary.

Assume that there is a user that honestly forwards all data honestly and correctly to the adversary. Presumably the user's privacy is broken if he actually does this. If there exists an alternative behaviour for the user, that looks exactly the same to the adversary but now it does not break the user's privacy, then the user can "cheat" the adversary. By behaving in the alternative way, the user does not break its privacy, but the adversary cannot tell that the user is not being honest. Therefore, the user cannot prove that it broke its own privacy, and the adversary has no reason to believe that the user actually broke his own privacy. When this is the case, Delaune et al say that enforced privacy holds. As pointed by Backes et al [4] and Küsters et al [24], the definition by Delaune et al does not capture certain protocols (such as [26, 23]) and certain attacks (abstention attacks). To tackle the problem, Backes et al [4] improved the definition. However, these definitions depend on specific protocol structures (as pointed out by Küsters et al [24]). Küsters et al [24] proposed a more general epistemic definition following the same basic idea. This approach requires reasoning on voter's goals and strategies. In addition, the above mentioned work focuses on specific a specific domain - e-voting.

*Foul Adversary* We have the following assumption: Users only want to cooperate with the foul adversary when it is impossible to merely pretend to cooperate. This is a high-level assumption shared by all of the variations of the foul adversary model (and shared by the existing enforced properties [4, 14, 16, 17]). The exact meaning of that abstract assumption is difficult to pin down. In fact, we argue that the precise interpretation depends on the context, and that a variety of models is necessary.

Another assumption is a standard assumption, namely that a system is secure if and only if no attacker can perform an attack. This means that we can ignore, without loss of generality, those attackers that are strictly weaker than some other attackers. Concretely, we can ignore attackers that coerce at the wrong time, for the wrong data, or do not realise they can coerce for data. For example, a foul adversary may not know whether a user knows some coercible data, but choose to try to coerce anyway, and gain knowledge if the user does (and punish unfairly if he does not).

There are two aspects on which we divide the foul adversaries: On the requirements for a user to cheat, and on the role of time in coercion. First, we use the duress code example from the introduction to illustrate the first aspect. Then, we introduce an informal example to illustrate the time aspect (see 5.1). We formalise the distinctions between the foul adversaries in the following sections.

## 3 Knowledge

In modern formalisms (such as Tamarin [29], the applied pi calculus [2], etc.), for the analysis of security properties under the Dolev-Yao adversary, it is possible but not necessary to reason explicitly about the knowledge of users (or even about users at all). Here, we explicitly model the users and their knowledge, to make the assumptions explicit.

Moreover, the fact that we explicitly reason about adversary knowledge is a core concept in our approach. We argue that some subtleties simply cannot be captured by a model that does not take knowledge into account. In this section, we create a model of knowledge and reasoning using coercion. We do not take learning (dynamic knowledge) into account, until Section 4. Our model of knowledge is similar to other definitions in symbolic security (e.g., [29, 11]).

### 3.1 Preliminaries

A common way to reason about knowledge is epistemic modal logic. However, an epistemic agent has perfect reasoning capabilities, allowing him to solve computationally hard problems. The (Dolev-Yao or foul) adversary is not capable of solving computationally hard problems. Hence, in the context of security protocol modelling, we need an alternative model of knowledge. We take Cortier and Kremer [11]'s model of the Dolev-Yao knowledge and reasoning as our starting point. In their model, a user knows something iff he can derive it from one of the facts in his core knowledge.

We adopt a symbolic approach [2, 15, 29], meaning that we adopt the ideal properties of the cryptographic primitives. Messages that a user and an adversary know can be modelled as the following (e.g. in [2, 11]):

- There exists a countable set of *names* $\mathcal{N}$, an countable set of *variables* $\mathcal{V}$, and a countable set of *signatures* $\Sigma$ – a set of function symbols with arities.
- A *term* in $\mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma)$ is either a name from $\mathcal{N}$, a variable from $\mathcal{V}$, or $\mathsf{f}(M_1, \ldots, M_n)$ where $\mathsf{f}$ is an $n$-ary function in $\Sigma$ and all $M_i$ $(0 \leq i \leq n)$ are terms in $\mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma)$.
- The variables in a term $M$ are denoted by $\delta(M)$. A term is *ground* when $\delta(M) = \emptyset$. Ground terms are called *data* and denoted as $\mathcal{T}(\mathcal{N}, \Sigma)$. Replacing a variable $x$ in term $M$ with a ground term $d$ is an *instantiation*, denoted by $M\{d/x\}$. We use $\theta$ to represent instantiation of a set of variables, $\delta(\theta)$ to denote the variables, and $\varphi(\theta)$ to denote the data to replace the variables[3].
- Properties of cryptographic primitives are captured by an *equational theory* $E$, where $E$ is a set of equations of terms of the form $M =_E N$, where $M, N \in \mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma)$.

The derivation rules are provided by an axiomatization, such as found in [30] and in [1, 11]. We axiomatize the reasoning of the users (the Dolev-Yao adversary in particular) with the rules in Figure 1. We may refer to axiom $A$ as the axiom of core knowledge, to axiom $B$ as the axiom of equality and to axiom $F$ as the axiom of function application. If there exists a derivation with premise $X$ and conclusion $y$ under axioms $A$, $B$ and $F$, and a specified equational theory $E$, then we may write $X \vdash_{DY} y$. The statement $X \vdash_{DY} y$ means that an agent with core knowledge $X$ has $y$ in his knowledge. $\vdash_{DY}$ models the reasoning ability of the Dolev-Yao adversary. Let $\mathcal{X}$ be a set of sets of knowledge, and $Y$ be a set of knowledge, we may write $\mathcal{X} \vdash_{DY} Y$ to mean $\forall_{y \in Y} \exists_{X \in \mathcal{X}} (X \vdash_{DY} y)$.

[3] Only our notion of instantiation differs from the standard, as we disallow names to be substituted, and we disallow variables to be substituted into a formula.

$$A\frac{x \in X}{X \vdash x} \quad B\frac{X \vdash y \quad x =_E y}{X \vdash x} \quad F\frac{X \vdash x_0, \ldots, x_n \quad f \in \Sigma}{X \vdash f(x_0, \ldots, x_n)}$$

**Fig. 1.** Standard knowledge reasoning rules.

### 3.2 Weak Coercion

As mentioned before, one of the two aspects on which we distinguish the adversary's power, is user knowledge. The distinction between the weak and strong variants is that the former uses weak coercion, and the latter uses strong coercion. The definitions of weak coercion and strong coercion are similar, but weak coercion is simpler. Here, we define weak coercion, and in Section 3.3 we show how strong coercion differs.

Weak coercion is based on a notion of verifiability and a notion of elicitation. Given an equation which can only be satisfied with data $d$, then $d$ is called verifiable under that equation. Elicitation models obtaining information by coercion. In particular, if $d$ is verifiable under an equation that the foul adversary can construct, and a user has $d$, then the foul adversary can ask the user to provide $d$, which he must provide as the foul adversary can verify it. Elicitation is modelled as a derivation rule, where the adversary elicits data whenever necessary.

*Verifiability* Verifiability is a property of data. For example, if you receive a hashed message, then there is only one original message that would give you that hash (assuming an idealised hash function without collision). In this case, we say that the original message is verifiable under the hashed message. More precisely, let the hashed message be $h(m)$ and $m$ the original, then the equation $h(x) =_E h(m)$ can only be satisfied when $x =_E m$. Thus $h(x)\theta =_E h(m)\theta$ holds, only if $\theta$ replaces $x$ by $m$ (note that $m$ is not a variable, and cannot be instantiated). This forms the bases of the definition.

We obtain the following formal definition of verifiability of $D$ (a set of data) under $M, N$:

$$V^{M,N}(D) \text{ iff } \exists_{\theta:\varphi(\theta)=D \wedge M\theta =_E N\theta} \left( \nexists_{\theta':\delta(\theta')=\delta(\theta) \wedge D \neq_E \varphi(\theta')} \left( M\theta' =_E N\theta' \right) \right).$$

The formula states that $D$ is verifiable under $M, N$, when there exists an instantiation $\theta$ (of $D$ onto variables that occur in $M, N$), such that $M$ and $N$ are equivalent and there is no instantiation $\theta'$ (of other data than $D$ onto the same variables) that equates $M$ and $N$. Thus, if the user is challenged to give the correct data to equate $M$ and $N$, then the user cannot provide any other data than $D$ (or data that equates to $D$)[4].

[4] The domain of $\theta$ has not been restricted in the formula. Note that we can add a condition $\delta(\theta) \subseteq \delta(M) \cup \delta(N)$ without loss of generality. If there were a variable $x \in \delta(\theta), x \notin \delta(M) \cup \delta(N)$, then $M\theta =_E N\theta$ implies $M\theta' =_E N\theta'$, for all $\theta'$ that are equal to $\theta$ except on where $x$ maps to. In that case, the condition $\nexists \ldots$ is trivially false. Therefore, we can limit our $\theta$ to those with only variables also in $M$ or $N$.

$$C\frac{\mathcal{C}_{\mathbf{K}}(D),\ A \vdash_{DY} M,\ A \vdash_{DY} N,\ V^{M,N}(D)}{(A, \mathbf{K}) \vdash_{\zeta} D}$$

**Fig. 2.** Axiom concerning elicitation.

*Elicitation* The foul adversary can gain knowledge by coercion – elicitation. A set of data $D$ is elicitable if the coercible users can derive it (i.e., when the users *know* it). Formally, given the set of core knowledges $\mathbf{K}$ of coercible users, elicitable of a set of data $D$ (denoted as $\mathcal{C}_{\mathbf{K}}(D)$) is defined as:

$$\mathcal{C}_{\mathbf{K}}(D) \text{ iff } \forall d \in D, \exists_{K \in \mathbf{K}}(K \vdash_{DY} d).$$

When $D$ is elicitable and the adversary can derive some terms under which $D$ is verifiable, then the user has no choice but to provide $D$ truthfully. This is the intuition behind the elicitation rule, which is modelled as a derivation rule, meaning that elicitation is just a way for the adversary to gain knowledge.

Since the question of whether $D$ is elicitable depends on the knowledge of the users, it is unavoidable that the elicitation rule does not merely depend on the adversary's (core) knowledge. The premises of elicitation are the core knowledges of the users, and the core knowledge of the adversary. If the adversary can construct terms $M, N$, such that data $d$ is verifiable under $M, N$, then the adversary can coerce $d$ from users that know the data $d$. This is directly codified in the coercion rule, in Figure 2.

In Figure 2, $\mathcal{C}_{\mathbf{K}}(D)$ ensures that $D$ can actually be provided by the coerced users, $A \vdash_{DY} M$ (or $N$) ensures that the adversary can actually construct $M$ (or $N$) from his core knowledge $A$ – note that he uses variables here – and finally $V^{M,N}(D)$ ensures that misrepresenting $D$ is impossible for the coerced users. Note that as mentioned in Section 2, to coerce for $d$, the adversary need not know that the coerced user knows $d$.

### 3.3 Strong Coercion

Strong coercion is highly similar to weak coercion. The verification rule is liberalised, and some data which is not verifiable in weak coercion may now be verifiable. The verification rule now takes into account the core knowledge(s) of the user(s) that need to cheat the adversary. If the users do not actually know the data needed to cheat the adversary, then they cannot cheat the adversary, meaning the data remains verifiable.

More precisely, data $D$ may not be verifiable under $M, N$, due to the existence of some $D' \neq_E D$ that fits the same equation. In reality, the existence of such $D'$ may not help the user, if he is unable to construct it. For example, if $h(k) =_E h(k')$, then the adversary, who saw the hash $h(k)$, may construct $M = h(k)$ and $N = h(x)$, and is not able to coerce for $k$ this way, since $k'$ satisfies the equation too. However, if the user cannot actually derive $k'$, then he still has no choice but to provide $k$ to satisfy the equation. In this section, we make minimal changes to weak coercion to obtain strong coercion.

$$C_s \frac{\mathcal{C}_{\mathbf{K}}(D),\ A \vdash_{DY} M,\ A \vdash_{DY} N,\ V_{\mathbf{K}}^{M,N}(D)}{(A, \mathbf{K}) \vdash_{\zeta_s} D}$$

**Fig. 3.** Axiom concerning strong coercion.

*Verifiability* Using the notion of instantiation, we obtain the formal definition of verifiability of $D$ under $M, N$ for coerced users with core knowledges $\mathbf{K}$:

$$V_{\mathbf{K}}^{M,N}(D) \text{ iff } \exists_{\theta:\varphi(\theta)=D \wedge M\theta =_E N\theta} \left( \nexists_{\theta':\delta(\theta')=\delta(\theta) \wedge D \neq_E \varphi(\theta')} \left( \mathbf{K} \vdash_{DY} \varphi(\theta') \wedge M\theta' =_E N\theta' \right) \right).$$

Strong verifiability is identical to weak verifiability, except for the additional expression $\mathbf{K} \vdash_{DY} \varphi(\theta')$ in the not-exists, which expresses the additional requirement that the user actually knows how to construct the deception.

**Proposition 1.** *$V^{M,N}(D)$ implies $V_{\mathbf{K}}^{M,N}(D)$.*

*Proof.* If $\theta'$ exists in strong verifiability, then it exists in weak verifiability.

*Elicitation* Strong coercion is a simple adaptation from weak coercion, where we use strong verifiability rather than weak verifiability; see Figure 3.

## 4 Behaviour

In this section, we use a crude model of the dynamics of the systems. We assert that all users follow some protocol, which determines what actions they may perform. The adversary can also perform actions, depending on his knowledge. The effect of the actions is deterministic, meaning that the consequences of an action are fixed. This allows us to use the extensive form representation of a system, where traces and states are equivalent notions. We formalise this representation in Section 4.1.

In Section 4.2, we introduce the four variations of the dynamic aspect: static, conservative, aggressive and extended foul adversaries. We show the relationships between the different adversaries that we have introduced, in Section 4.3.

### 4.1 Preliminaries

A system is $(S, \mathcal{A}, \mathcal{I}, s^0, U, \mathsf{K}_0)$ where $S$ is a set of states, $\mathcal{A}$ is a set of actions of the form $\mathsf{keyword}(u, v, d)$ where $\mathsf{keyword} \in \{\mathsf{public}, \mathsf{private}, \mathsf{block}, \mathsf{insert}\}$, $u$ is the (alleged) sender, $v$ is the (alleged) receiver, $d : \mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma)$ is the communicated data, $\mathcal{I} : S \times \mathcal{A} \times S$ is a deterministic[5] set of transitions forming a tree[6], $s^0$ is

[5] Given state $s$ and action $a$, there is at most one state $t$ such that $s \xrightarrow{a} t$.

[6] Due to knowledge monotonicity, it is important that the system is represented in the extensive form of a tree.

the initial state at the root of the tree, $U$ is a set of users (coerced users $U_{\mathcal{C}}$ are a subset of $U$) and $\mathsf{K}_0 : U \cup \{e\} \to \wp(\mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma))$ is an assignment of initial core knowledge to the users. As a consequence of this definition, every state is uniquely identified by the sequence of actions leading to it. Hence, we may simply write $[a_1, \ldots, a_n]$ to refer to the state $s_n$ which has the property that $s^0 \xrightarrow{a_1} s_1, \ldots, s_{n-1} \xrightarrow{a_n} s_n$.

Users behave according to some protocol specification, which dictates their actions. Users can send $\mathsf{s}(w, d)$ and receive $\mathsf{r}(w, d)$ public messages $d$ to/from $w$, and send $\mathsf{ps}(w, d)$ and receive $\mathsf{pr}(w, d)$ privately messages $d$ to/from $w$ under certain circumstances. Users do not introduce variables, only terms, if a user sends a variable, it is a variable it received by the adversary. Let $\pi_u(s)$ be a projection of the global state to the user state, and $\rho^s_u(a)$ be that action $a$ is enabled at user state $\pi_u(s)$. A transition

- $s \xrightarrow{\mathsf{public}(u,v,\tau)} t$ exists iff $\rho^s_u(\mathsf{s}(v, \tau))$, $\rho^s_v(\mathsf{r}(u, \tau))$,
- $s \xrightarrow{\mathsf{private}(u,v,\tau)} t$ exists iff $\rho^s_u(\mathsf{ps}(v, \tau))$, $\rho^s_v(\mathsf{pr}(u, \tau))$,
- $s \xrightarrow{\mathsf{block}(u,v,\tau)} t$ exists iff $\rho^s_u(\mathsf{s}(v, \tau))$, and
- $s \xrightarrow{\mathsf{insert}(u,v,\tau)} t$ exists iff $\rho^s_v(\mathsf{r}(u, \tau))$ and the adversary knows $\tau$ ($s \Vdash \tau$, where $\Vdash$ depends on the adversary model), with $\tau \in \mathcal{T}(\mathcal{N}, \mathcal{V}, \Sigma)$.

Thus, in the extensive form representation, a private communication can happen only if both parties can privately communicate. Similarly for public communication. However, in addition, the adversary can block a public communication – pretending to be the receiver – or insert a public communication (provided the adversary knows the content of the communication) – pretending to be the sender. These are standard assumptions in the Dolev-Yao model, which is our starting point.

At every state, the users and the adversary have some knowledge. The knowledge consists of a core knowledge, and the ability to reason. We define $\kappa^u(s)$ as a function that gives the core knowledge of $u$ in state $s$.

- $\kappa^u(s^0) = \mathsf{K}_0(u)$, for all users;
- $\kappa^v(t) = \kappa^v(s) \cup \{\tau\}$ when $s \xrightarrow{\mathsf{public}(u,v,\tau)} t$, $s \xrightarrow{\mathsf{private}(u,v,\tau)} t$ or $s \xrightarrow{\mathsf{insert}(u,v,\tau)} t$;
- $\kappa^e(t) = \kappa^e(s) \cup \{\tau\}$ when $s \xrightarrow{\mathsf{public}(u,v,\tau)} t$ or $s \xrightarrow{\mathsf{block}(u,v,\tau)} t$; and
- $\kappa^v(t) = \kappa^v(s)$ for all other users.

Due to the fact that we use extensive form representation, we have uniquely defined the knowledge of all users in all states. We write $\xrightarrow{a}_{\mathsf{D}}$ if the message in $a$ is data; i.e. if it does not contain variables.

### 4.2 Formal Models of Foul Adversaries

Here we introduce the static, conservative, aggressive and extended foul adversaries, that differ in how they treat the dynamic aspect.

*Static Foul Adversary* The static foul adversary models a situation in which the foul adversary only has power over the coerced users in the present. An example is a street robber that wants to obtain your PIN code, if you manage to cheat

the street robber, then he cannot punish you later. All that matters is that the data is not currently verifiable.

Let $\Vdash^{S}_{S}$ be the weakest relationship satisfying,

1. in state $s \in S$, for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\})$ and $A = \kappa^{e}(s)$, if $(A, \mathbf{K}) \vdash_{\zeta_s} d$, then $s \Vdash^{S}_{S} d$; and
2. $\forall s \to s' \in \mathcal{I}$, $s \Vdash^{S}_{S} d \implies s' \Vdash^{S}_{S} d$.

The strong static foul adversary (SSFA) is an adversary that uses $\Vdash^{S}_{S}$ as derivation relation.

Rule 1 simply encodes that the static foul adversary can elicit information in a state that allows him to elicit information. Rule 2 is a modelling trick. Without rule 2, it is possible that data $d$ becomes unverifiable due to the user learning a cheat. However, we can assume without loss of generality that the adversary had sufficient foresight to elicit $d$ when it was possible. We address this issue by simply defining the reasoning to be monotonic.

The relation $\Vdash^{W}_{S}$ is defined similarly, using $\vdash_{\zeta}$ rather than $\vdash_{\zeta_s}$. The weak static foul adversary (WSFA) is an adversary that uses $\Vdash^{W}_{S}$ as derivation relation. Here, the monotonicity rule (rule 2) is superfluous, as the adversary knowledge is trivially monotonic using only rule 1 (since data cannot become unverifiable).

*Conservative Foul Adversary* The conservative foul adversary models a situation in which the foul adversary is not willing to coerce unless it is sure it can follow up on its threats. An example is a mafioso who values his reputation of following up on threats more than breaking the security property. This typically occurs in scenarios where the stakes of the individual users are relatively low.

Let $\Vdash^{S}_{C}$ be the weakest relationship satisfying,

1. in state $s \in S$, for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\}$ and $A = \kappa^{e}(s)$, if $(A, \mathbf{K}) \vdash_{\zeta_s} d$, then $s \Vdash^{S}_{C} d$;
2. $\forall s \to s' \in \mathcal{I}$, $s \Vdash^{S}_{C} d \implies s' \Vdash^{S}_{C} d$; and
3. $\forall s \to_{\mathsf{D}} s' \in \mathcal{I}$ and $\mathcal{C}_{\mathbf{K}}(\{d\})$ for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\}$, $s' \Vdash^{S}_{C} d \implies s \Vdash^{S}_{C} d$.

The strong conservative foul adversary (SCFA) is an adversary that uses $\Vdash^{S}_{C}$ as derivation relation.

Rule 3 states that if for all (non-imaginary) futures, the foul adversary can verify data $d$, then the user has no choice to surrender $d$, provided he has $d$. The subscript $\mathsf{D}$ (in $\to_{\mathsf{D}}$) ensures that the future is not imaginary, as it disallows variables in the messages – restricting to communications with actual data.

The relation $\Vdash^{W}_{C}$ is defined similarly, using $\vdash_{\zeta}$ rather than $\vdash_{\zeta_s}$. The weak conservative foul adversary (WCFA) is an adversary that uses $\Vdash^{W}_{C}$ as derivation relation. Again, monotonicity is superfluous here.

*Aggressive Foul Adversary* The aggressive foul adversary models a situation in which the user wants to avoid crossing the foul adversary at all costs. This is the typical dynamic version of the foul adversary, applicable to voting systems, where the foul adversary may punish users after the results came in.

Let $\Vdash^{S}_{A}$ be the weakest relationship satisfying,

1. in state $s \in S$, for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\})$ and $A = \kappa^e(s)$, if $(A, \mathbf{K}) \vdash_{\zeta_s} d$, then $s \Vdash {}^{S}_{A} d$;
2. $\forall s \to s' \in \mathcal{I}$, $s \Vdash {}^{S}_{A} d \implies s' \Vdash {}^{S}_{A} d$; and
3. $\exists s \to_{\mathsf{D}} s' \in \mathcal{I}$ and $\mathcal{C}_{\mathbf{K}}(\{d\})$ for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\}$, $s' \Vdash {}^{S}_{A} d \implies s \Vdash {}^{S}_{A} d$.

The strong aggressive foul adversary (SAFA) is an adversary that uses $\Vdash {}^{S}_{A}$ as derivation relation.

Rule 3 is changed to an existential property, which states that if in some (non-imaginary) futures, the foul adversary can verify data $d$, then the user has no choice to surrender $d$, provided he has $d$. Again, we are only considering real data, not imaginary communications.

The relation $\Vdash {}^{W}_{A}$ is defined similarly, using $\vdash_{\zeta}$ rather than $\vdash_{\zeta_s}$. The weak aggressive foul adversary (WAFA) is an adversary using $\Vdash {}^{W}_{A}$ as derivation relation.

*Extended Foul Adversary* The aggressive foul adversary also models a situation in which the user wants to avoid crossing the foul adversary at all costs, but furthermore, the adversary cares more about not being cheated than about the actual security property at hand. In particular, it involves scenarios where the adversary coerces for data which he can only verify because he coerced for the data in the first place.

Let $\Vdash {}^{S}_{E}$ be the weakest relationship satisfying,

1. in state $s \in S$, for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\})$ and $A = \kappa^e(s)$, if $(A, \mathbf{K}) \vdash_{\zeta_s} d$, then $s \Vdash {}^{S}_{E} d$;
2. $\forall s \to s' \in \mathcal{I}$, $s \Vdash {}^{S}_{E} d \implies s' \Vdash {}^{S}_{E} d$; and
3. $\exists s \to s' \in \mathcal{I}$ and $\mathcal{C}_{\mathbf{K}}(\{d\})$ for $\mathbf{K} = \{\kappa^{u_i}(s) | u_i \in U_{\mathcal{C}}\}$, $s' \Vdash {}^{S}_{E} d \implies s \Vdash {}^{S}_{E} d$.

The strong extended foul adversary (SEFA) is an adversary that uses $\Vdash {}^{S}_{E}$ as derivation relation.

Rule 3 is changed to allow imaginary futures. In addition to non-imaginary future states, it may be useful for the adversary to send a variable (imaginary communication). The users process the received variable, and output a function of that variable. The adversary can then use that output term to construct an equation to verify the data.

The relation $\Vdash {}^{W}_{E}$ is defined similarly, using $\vdash_{\zeta}$ rather than $\vdash_{\zeta_s}$. The weak extended foul adversary (WEFA) is an adversary using $\Vdash {}^{W}_{E}$ as derivation relation.

### 4.3 Hierarchy

The relations between the foul adversaries are shown in Figure 4. We say adversary $A$ is stronger than adversary $B$ (denoted as $B \to A$), if a protocol satisfies a property w.r.t. $A$, then protocol satisfies the property w.r.t. $B$, i.e., $B \vdash d \implies A \vdash d$. In Figure 4, from left to right the adversary is getting stronger, because the ability of a stronger adversary contains all the ability of a weaker adversary. The adversaries in the second row is stronger than the corresponding one in the first row. The theorems in the figure and their proofs can be found in [18].

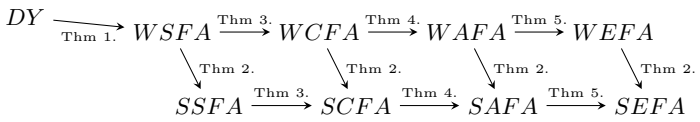


**Fig. 4.** Relations of adversary models.

## 5 Example Systems

The notions that we have introduced were, by design, of a high level of abstraction. In this section, we introduce examples to make the ideas more concrete, and to link our approach to security systems and protocols. Our notion of coercion allows other security properties than privacy, in Section 5.1, we use the common property of secrecy. We also apply out approach to privacy, in Section 5.2. In this section, we do not encode all properties formally into the formalism, for brevity's sake. We merely codify the relevant elements of the examples, and rely on common sense for the details. Even fairly simple systems and protocols would require pages of specification, when defined rigorously.

### 5.1 Examples on (Enforced) Secrecy

*Special Symmetric Encryption* Let the equational theory support a special symmetric encryption, meaning that $\mathsf{enc}(\mathsf{dec}(m,k),k) =_E m$ and $\mathsf{dec}(\mathsf{enc}(m,k),k) =_E m$ are in the equational theory. There are two honest (and coercible) users, the sender $u$ and the receiver $v$ communicating on public channels, and a foul adversary $e$. The user $u$ contains states $s_u, s'_u$ and the transition $(s_u \xrightarrow{\mathsf{s}(v,\mathsf{enc}(m,k))} s'_u)$ and user $v$ contains at least the states $s_v, s'_v$ and the transition $(s_v \xrightarrow{\mathsf{r}(u,x)} s'_v)$. The system at least contains the transition $s \to s'$ where in state $s'$, $v$ gains knowledge $\mathsf{enc}(m,k)$. Furthermore, a state $s''$ exists with $(s \to s'')$ where in state $s''$, the adversary gains knowledge $\mathsf{enc}(m,k)$. The reasoning abilities of $u$ and $v$ are $\vdash_{DY}$ and that of the adversary depends on the foul adversary model, e.g., $\Vdash_S^S$ in the case of static foul adversary. The initial knowledge is $\mathsf{f}$, such that $\mathsf{f}(u) = \{m,k\}$, $\mathsf{f}(v) = \{k\}$ and $f(e) = \emptyset$.

We are interested in the secrecy of $m$, meaning that the correctness of the protocol is determined by the reachability of a state $t$ where $t \Vdash_S^S m$.

The core knowledge of the adversary is initially empty, and the adversary receives at most one message, $\mathsf{enc}(m,k)$ in state $s''$. The largest core knowledge that the adversary can achieve is, therefore, $\{\mathsf{enc}(m,k)\}$. We can neither coerce for $m$ nor for $k$, since the user can generate $m'$ and $k'$, such that $\mathsf{enc}(m,k) =_E \mathsf{enc}(m',k')$. In particular for arbitrary $k'$, let $m' = \mathsf{dec}(\mathsf{enc}(m,k),k')$, in which case $\mathsf{enc}(m',k') =_E \mathsf{enc}(\mathsf{dec}(\mathsf{enc}(m,k),k'),k') =_E \mathsf{enc}(m,k)$. Formally, $m$ or $k$ cannot be verified: since encryption and decryption are the only functions, whenever two terms $M\theta =_E N\theta$ holds for $m$ and $k$, it also holds for some $m'$ and $k'$, due to the equational theory.

*Encryption of Natural Language* Take the same scenario as sketched in the previous example. We add a constant $\mathsf{c}$ and a unary function $\mathsf{e}$ to the equational theory, with $\mathsf{e}(m) =_E \mathsf{c}$ only for a subset of terms $T$ ($m \in T$), representing those messages that are valid English texts. Dissimilar to the previous example, we cannot conclude that the protocol that sends $\mathsf{enc}(m, k)$ (with $m \in T$) is safe under the foul adversary, as the fact that $\mathsf{enc}(m, k) =_E \mathsf{enc}(\mathsf{dec}(\mathsf{enc}(m, k), k'), k')$ is no longer sufficient to prevent coercion, due to that an arbitrary $k'$ leads to non-readable messages, assuming the probability of $k' \in T$ is negligible. The foul adversary can add a test $\mathsf{e}(\_) =_E \mathsf{c}$, which holds for $m$, but not for an arbitrary $m' =_E \mathsf{dec}(\mathsf{enc}(m, k), k')$. Formally, there exists $\mathsf{e}(\mathsf{dec}(\mathsf{enc}(y, x), x))\{k/x\}\{m/y\} =_E \mathsf{c}$ (serving as the relation $M\theta =_E N\theta$ in rule $C$ in Figure 2 or rule $C_s$ in Figure 3), that only holds for $m$ and $k$, but not holds for arbitrary $m'$ and $k'$.

Interestingly, together the last two examples imply that the same encryption method is coercion resistant when containing random data, but not coercion resistant when it contains natural language.

Note that under Dolev-Yao adversary, secrecy of $m$ is satisfied, because there is no way for the Dolev-Yao adversary to obtain the key. Thus, this example shows that static foul adversary is strictly stronger than Dolev-Yao adversary.

*Coercion with Delayed Verification* Take the same scenario as in the previous example, but let the adversary initially know $m'$. Furthermore, upon receiving the first term $x$, the receiver will respond with $\mathsf{dec}(x, k)$. In an honest run, the received term will be $\mathsf{enc}(m, k)$, meaning that the response is $m$.

Suppose the adversary wants to replace $m$ in the message with $m'$. That is, the adversary needs to insert the message $\mathsf{enc}(m', k)$ to the receiver, and thus he needs to know $\mathsf{enc}(m', k)$ before the receiver outputs anything. Since the static foul adversary cannot look ahead, the adversary cannot know $k$ (and thus $\mathsf{enc}(m', k)$) before the receiver sending the response, as the adversary does not have enough information to verify them. It means that the static foul adversary cannot insert the message $\mathsf{enc}(m', k)$ to the receiver before the receiver outputs anything. The dynamic foul adversary, however, can look ahead. Since there is a trace where the adversary will know $m$, and be able to verify $k$ using $\mathsf{enc}(m, \_) = \mathsf{enc}(m, k)$. That, in turn, means that the adversary can coerce for $k$ in the initial state. Hence, the adversary knows $k$ in the initial state, and can construct and insert $\mathsf{enc}(m', k)$ before the receiver outputs anything.

### 5.2 Examples on (Enforced) Privacy

Enforced privacy properties such as receipt-freeness and coercion-resistance are important requirements in e-voting. We use a simplified well-known e-voting protocol to show how enforced privacy properties can be formalised with respect to the foul adversary. This simplified protocol is the voting phase of the Okamoto e-voting protocol [28][7].

[7] For the simplicity of presentation, we ignore some functionalities, such as signature, registration and verifiability, and focus only on the critical part for enforced privacy.

Two voters $V_1$ and $V_2$ have initial knowledge $\{V_1, V_2, C, v_1, r_1\}$ and $\{V_1, V_2, C, v_2, r_2\}$ respectively, where $v_1$ and $v_2$ are their votes, $r_1$ and $r_2$ are two random numbers for the commitment, $C$ is the vote-collector. The two voters send the committed votes first over public channels, then send privately the opening information ($r_1$ and $r_2$ respectively), and finally receive the voting result; modelled as follows.

$$s^0_{V_1} \xrightarrow{\mathsf{s}(C,(\mathsf{com}(v_1,r_1)))} s_{V_1} \xrightarrow{\mathsf{ps}(C,(\mathsf{com}(v_1,r_1),r_1))} s'_{V_1} \xrightarrow{\mathsf{r}(C,((v_1,v_2),(m_1,m_2)))} s''_{V_1},$$
$$s^0_{V_2} \xrightarrow{\mathsf{s}(C,(\mathsf{com}(v_2,r_2),r_2))} s_{V_2} \xrightarrow{\mathsf{ps}(C,(\mathsf{com}(v_2,r_2),r_2))} s'_{V_2} \xrightarrow{\mathsf{r}(C,((v_1,v_2),(m_1,m_2)))} s''_{V_2}.$$

The vote-collector $C$'s initial knowledge is $\{V_1, V_2, C\}$. $C$ reads in the votes and the opening information, and sends out the voting results. One possible trace of $C$ is as follows.

$$s^0_C \xrightarrow{\mathsf{r}(V_1,\mathsf{com}(v_1,r_1))} s^1_C \xrightarrow{\mathsf{pr}(V_1,(\mathsf{com}(v_1,r_1),r_1))} s^2_C \xrightarrow{\mathsf{r}(V_2,(\mathsf{com}(v_2,r_2))} s^3_C$$
$$\xrightarrow{\mathsf{pr}(V_2,(\mathsf{com}(v_2,r_2),r_2))} s^4_C \xrightarrow{\mathsf{s}(V_1,((v_1,v_2),(m_1,m_2)))} s^5_C \xrightarrow{\mathsf{s}(V_2,((v_1,v_2),(m_1,m_2)))} s^6_C.$$

Assuming $s^0$ is the initial state, with $\kappa^e(s^0) = \{V_1, V_2, C\}$. The following transitions are eligible: $s^0 \xrightarrow{\mathsf{block}(V_1,C,(\mathsf{com}(v_1,r_1)))} s^1 \xrightarrow{\mathsf{insert}(V_1,C,(\mathsf{com}(v_1,r_1)))} s^2 \xrightarrow{\mathsf{private}(V_1,C,(\mathsf{com}(v_1,r_1),r_1))} s^3 \xrightarrow{\mathsf{public}(V_2,C,(\mathsf{com}(v_2,r_2)))} s^3 \xrightarrow{\mathsf{private}(V_2,C,(\mathsf{com}(v_2,r_2),r_2))} s^5 \xrightarrow{\mathsf{block}(C,V_1,((v_1,v_2),(m_1,m_2)))} s^6$. The adversary knowledge in state $s^1$ is $\kappa^e(s^1) = \kappa^e(s^0) \cup \{\mathsf{com}(v_1, r_1)\}$, since the public message from $V_1$ to $C$ is $\mathsf{blocked}$ by $e$.

Vote privacy is formalized as $V_1\{c_1/v_1\}|V_2\{c_2/v_2\} \sim V_1\{c_2/v_1\}|V_2\{c_1/v_2\}$, where '$\sim$' is indistinguishability of left side ($V_1$ votes for $c_1$ and $V_2$ votes for $c_2$) and right side ($V_1$ votes for $c_2$ and $V_2$ votes for $c_1$) and '|' denotes parallel composition (following the definition in [14]). When $v_1 \neq v_2$, the property is satisfied - the left situation and right situation of '$\sim$' lead to the same voting result, and the adversary cannot distinguish.

However, if the adversary can coerce $V_1$ for the vote $v_1$ and the random number $r_1$, then the property does not hold anymore. In state $s^1$ we have ($\mathbf{K} = \{V_1, V_2, C, v_1, r_1\}$, $\kappa^e(s^1) = \{V_1, V_2, C, M_1\}$ and $M_1 = \mathsf{com}(v_1, r_1)$),

$$\frac{\mathcal{C}_{\mathbf{K}}(\{v_1, r_1\}), A \vdash_{\mathcal{D}Y} M_1, \exists_{\{v_1/x\}\{v_2/y\} \wedge M_1 =_E \mathsf{com}(x,y)} \nexists_{v',r'}(M_1 =_E \mathsf{com}(v', r'))}{(A, \mathbf{K}) \vdash_{\zeta} \{v_1, r_1\}}.$$

Hence, in the left side situation, the adversary can verify that $V_1$ votes for $c_1$, since $v_1$ is substituted with $c_1$, whereas at the right side, the adversary verifies that $V_1$ votes for $c_2$ (since $\{c_2/v_1\}$). That is, enforced privacy is broken.

By replacing bit commitment with trap-door bit commitment $\mathsf{tdcom}(v, r, td)$, the rule does not hold anymore, because for $M_1 =_E \mathsf{com}(v_1, r_1, td), \exists r' : M_1 =_E \mathsf{com}(v_1, r', td')$. When the adversary coerces for both $r_1$ and $td_1$, although the adversary can elicit $r_1$ and $td_1$ for the case of $V_1$ voting for $c_1$, since $V_1$ can also derive $r'$ and $td'$ such that $M_1 =_E \mathsf{com}(v_2, r', td')$, the adversary can also elicit $r'$ and $td'$ for the case of $V_1$ voting for $c_2$. Hence, $M_1$ can be opened as $c_1$ and $c_2$. This holds on both sides of the equations; thus enforced privacy is not broken

in this way. Of course, to prove the enforced privacy property is satisfied in this case, one needs to consider all branches and all states, often using tool support.

## 6 Conclusions

In the paper, we propose the idea of modelling an adversary with the ability to coerce – the foul adversary. This contrasts the standard approach of modelling coercion resistance as a security requirement. Knowledge and reasoning are key points in the foul adversary, which is highlighted by the fact that elicitation is the main power of the foul adversaries. Elicitation is built upon the notion of verification, if only one piece of data fits the equation, then the foul adversary obtains the data. We show that reasoning about coercion itself can be just as important, as we have shown by example that different contexts may require different models of the adversary. In particular, we show a hierarchical relationship between the possible foul adversaries. The powers of the adversaries are divided along two aspects: user knowledge and dynamics. The most powerful foul adversary requires users to know how to cheat, and uses the dynamic nature of the system maximally to his advantage. Finally, we also provide a collection of examples of security systems to help place and interpret our theoretical results.

## References


1. M. Abadi and V. Cortier. Deciding knowledge in security protocols under equational theories. *Theoretical Computer Science*, 367(1-2):2–32, 2006.
2. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In *Proc. 28th Symposium on Principles of Programming Languages*, pages 104–115. ACM, 2001.
3. M. Abe and K. Suzuki. Receipt-free sealed-bid auction. In *Proc. 5th Conference on Information Security*, volume 2433 of *LNCS*, pages 191–199. Springer, 2002.
4. M. Backes, C. Hritcu, and M. Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *Proc. 21st IEEE Computer Security Foundations Symposium*, pages 195–209. IEEE Computer Society, 2008.
5. M. Barbaro and T. Zeller Jr. A face is exposed for AOL searcher no. 4417749.
6. D. A. Basin and C. J. F. Cremers. Modeling and analyzing security in the presence of compromising adversaries. In *Proc. 15th European Symposium on Research in Computer Security*, volume 6345 of *LNCS*, pages 340–356. Springer, 2010.
7. J. Benaloh and D. Tuinstra. Receipt-free secret-ballot elections (extended abstract). In *Proc. 26th Symposium on Theory of Computing*, pages 544–553. ACM, 1994.
8. B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proc. 14th IEEE Computer Security Foundations Workshop*, pages 82–96. IEEE CS, 2001.
9. Ch. Caldwell. A pass on privacy?, July 2005. 17.
10. Jeremy Clark and Urs Hengartner. Panic passwords: Authenticating under duress. *HotSec*, 8:8, 2008.
11. V. Cortier and S. Kremer. Formal models and techniques for analyzing security protocols: A tutorial. *Foundations and Trends in Programming Languages*, 1(3):151–267, 2014.

12. B. de Decker, M. Layouni, H. Vangheluwe, and K. Verslype. A privacy-preserving eHealth protocol compliant with the Belgian healthcare system. In *Proc. 5th European Workshop on Public Key Infrastructures, Services and Application*, volume 5057 of *LNCS*, pages 118–133. Springer, 2008.
13. S. Delaune, S. Kremer, and M. Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *Proc. 19th IEEE Computer Security Foundations Workshop*, pages 28–42. IEEE Computer Society, 2006.
14. S. Delaune, S. Kremer, and M. D. Ryan. Verifying privacy-type properties of electronic voting protocols. *Journal of Computer Security*, 17(4):435–487, 2009.
15. D. Dolev and A. C.-C. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–207, 1983.
16. N. Dong, H. L. Jonker, and J. Pang. Analysis of a receipt-free auction protocol in the applied pi calculus. In *Proc. 7th Workshop on Formal Aspects in Security and Trust*, volume 6561 of *LNCS*, pages 223–238. Springer, 2011.
17. N. Dong, H. L. Jonker, and J. Pang. Formal analysis of privacy in an eHealth protocol. In *Proc. 17th European Symposium on Research in Computer Security*, volume 7459 of *LNCS*, pages 325–342. Springer, 2012.
18. N. Dong and T. Muller. The foul adversary : Formal models. https://sites.google.com/view/foul-adversary/home.
19. A. Fujioka, T. Okamoto, and K. Ohta. A practical secret voting scheme for large scale elections. In *Proc. Advances in Cryptology – AUSCRYPT'92*, pages 244–251, 1992.
20. D. Goodin. Defects in e-passports allow real-time tracking, January 2010. 26.
21. J. Heather and S. Schneider. A formal framework for modelling coercion resistance and receipt freeness. In *Proc. Formal Mehtods – FM'12*, volume 7436 of *LNCS*, pages 217–231. Springer, 2012.
22. H. L. Jonker, J. Pang, and S. Mauw. A formal framework for quantifying voter-controlled privacy. *Journal of Algorithms in Cognition, Informatics and Logic*, 64(2-3):89–105, 2009.
23. A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *Proc. 4th ACM Workshop on Privacy in the Electronic Society – WPES'05*, pages 61–70. ACM, 2005.
24. R. Küsters and T. Truderung. An epistemic approach to coercion-resistance for electronic voting protocols. In *Proc. 30th IEEE Symposium on Security and Privacy*, pages 251–266. IEEE CS, 2009.
25. R. Küsters, T. Truderung, and A. Vogt. A game-based definition of coercion-resistance and its applications. In *Proc. 23rd IEEE Computer Security Foundations Symposium*, pages 122–136. IEEE CS, 2010.
26. B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *Proc. Information Security and Cryptology – ICISC'03*, pages 245–258, 2003.
27. T. Okamoto. An electronic voting scheme. In *Proc. IFIP World Conference on IT Tools*, pages 21–30, 1996.
28. T. Okamoto. Receipt-free electronic voting schemes for large scale elections. In *Proc. Security Protocols Workshop*, pages 25–35, 1997.
29. B. Schmidt, S. Meier, C. J. F. Cremers, and D. A. Basin. Automated analysis of diffie-hellman protocols and advanced security properties. In *Proc. 25th IEEE Computer Security Foundations Symposium*, pages 78–94. IEEE CS, 2012.
30. S. Schneider. Security properties and csp. In *Proc. IEEE Symposium on Security and Privacy*, pages 174–187. IEEE Computer Society, 1996.