CROO: A Generic Architecture and Protocol to Detect Identity Fraud*

D. Nali† P.C. van Oorschot

Abstract

Identity fraud (IDF) may be defined as unauthorized exploitation of credential information through the use of false identity. We abstract the problem of IDF by defining fundamental terms, identifying major stakeholders, and modeling the generic process of IDF. We then propose CROO, a generic architecture and protocol to either prevent IDF (by detecting attempts thereof), or limit its consequences (by identifying cases of previously undetected IDF). CROO is a Capture Resilient Online One-time password scheme, whereby each user must carry a personal trusted device used to generate and send encrypted one-time passwords (OTPs) verified by online trusted parties. OTPs are generated and verified at any desired user transaction, and can be used regardless of the transaction's purpose, associated credentials, and online or on-site nature; this makes CROO a generic scheme. OTPs are combined with hashed transaction information, in a manner allowing OTP-verifying parties to confirm the transaction information's correctness; this provides a certain level of user privacy, and prevents OTPs from being used for transactions other than those for which they were intended. Each OTP is generated from a PIN-encrypted non-verifiable key; this makes users' personal devices resilient to off-line PIN-guessing attacks.

1 Introduction

We informally define identity fraud (IDF)[1] as unauthorized exploitation of extracted credential information (e.g. identification passwords, driver's licence numbers, and credit card numbers) involving some form of impersonation or misrepresentation of identity. A 2005 survey [60] reported that over 9 million Americans (i.e. one in 23 American adults) were IDF victims in 2004, corresponding to a total annual cost of $51.4 billion and a median cost of $750 per victim.[2] The survey also reported that, while 67% of IDF victims did not suffer out-of-pocket losses, victims spent on average 28 hours resolving credit, financial and other problems caused by IDF. Financial losses due to existing credit card account fraud were higher than losses due to fraud on new accounts; 11.6% of misused information was obtained online, versus 68.2% from lost or stolen wallet, checkbook, or credit card. The mean loss due to phishing (a form of credential information extraction) was $2,320 per victim, while the mean loss due to theft of paper mail was $9,243 per victim.

The motivation behind IDF is multifaceted and the possible damages are diverse. Since IDF is a type of impersonation attack, *loss of privacy* is one type of damage; the side effects of *worry and fear* are difficult to quantify. Loss of privacy may facilitate *discrimination*, e.g. employment and insurance rate discrimination. (An employer may refuse to hire someone because of bad credit resulting from IDF; higher car insurance rates may result from improperly attributed criminal records resulting from IDF.) Two direct effects of financially motivated IDF are *financial loss* and *time loss*. *Denial of service* is another type of damage induced by IDF. For example, IDF victims have been arrested due to fraud committed by their impersonators under the victims' names [18]; in a survey published in May 2000, 15% of respondents reported a criminal investigation or warrant for their arrest due to IDF associated with them [36]. Moreover, *public discredit* is a potential consequence; many victims whose names and credentials have been used to forge new credit cards and to open new loan accounts are overwhelmed by the amount of personal effort, time and money required to restore the good credit record they once had [8].

In the academic literature, there are relatively few proposals addressing IDF. Most focus on prevention of credential information extraction. (See, e.g. [14, 19, 37, 47, 61], for countermeasures to phishing or key logging.) We are aware of only one non-application-specific academic proposal addressing general IDF [72], which, as presented, has limitations including restriction to on-site (vs. online) transactions and loss of user location privacy (users are geographically tracked).

In this paper, we focus on IDF (as defined above). More precisely, while such impersonation attacks may be committed under fictitious identities (e.g. identities of non-existing people to conceal involvement in illegal activity), we focus on IDF involving real people's identities, in large part because we wish to focus on IDF that

*December 20, 2006, School of Computer Science, Carleton University, Ottawa, Canada.

†Contact Author: deholo@ccsl.carleton.ca

[1] We prefer this term over "identity theft" (IDT), although both have often been used [24, 29, 42]. The term theft seems to suggest that victims are "deprived" of their identity, which is not always true, nor our focus.

[2] The mean total cost per victim was $5,686.

consumes real people's time. We also include consideration of IDF involving newly created credentials (e.g. credit, health, and building access cards) obtained by fraudsters in their victims' names, because this type of IDF currently seems difficult to detect. Our focus is on the *generic* IDF problem; we seek a solution that works for both remote and on-site transactions, and which is not application-specific. We also seek a solution which is not restricted to instances (of the IDF problem) associated with one class of credential tokens (e.g. credit cards). Credential-specific solutions are potentially what individual applications' (e.g. credit card) vendors are likely to propose; we believe end-users will find generic solutions both more usable (when considered across all applications), and less costly in terms of personal time. One might also argue that, for overall economic reasons, an IDF solution detecting driver's license and health/debit/credit card-based forms of IDF is more likely to be adopted and accepted by card bearers and state and financial institutions than solutions which only detect one of these forms of IDF.[3]

While we deal with architectural problems associated with the design of privacy-preserving credential management systems (cf. [12]), our primary focus is not the privacy aspect of such systems, but their fraud detection capability. Similarly, we do not aim to solve the bootstrap problem of human identification at the time when credentials are issued to people. Instead, we assume that trusted parties exist that can identify legitimate users (e.g. using out-of-band mechanisms), and we focus on the detection of fraudulent uses of credentials.

We propose a generic architecture and protocol for IDF detection, which we call `CROO` (`C`apture `R`esilient `O`nline `O`ne-time password scheme). Each user must carry a personal device used to generate encrypted one-time passwords (OTPs) verified by online trusted parties. These OTP generation and verification procedures are generic, in the sense that they can be associated with any user transaction, regardless of the transaction's purpose (e.g. user identification or financial payment), associated credentials (e.g. driver's license or credit card), and online or on-site (e.g. point-of-sale) nature. Before encrypting OTPs, user personal devices concatenate OTPs with hashed unique transaction information (e.g. list of bought items) allowing OTP-verifying parties to confirm the correctness of the associated transaction information; this prevents encrypted OTPs from being used for transactions other than those for which they were generated. The hashing provides a certain level of user privacy. Online OTP-verifying parties detect IDF when received OTPs or the associated transaction information do not have expected values. Each OTP is generated from a non-verifiable text [45] encrypted using a key derived from a user-chosen PIN; hence, possession of a user's personal device (or clone thereof) does not suffice to confirm guesses of the associated PIN, to recover the associated non-verifiable key, and generate correct OTPs. Since OTPs can only be verified by online parties, the proposed scheme turns off-line PIN guessing attacks against stolen or cloned personal devices into online OTP-guessing attacks that can be easily detected by online parties.

An interesting aspect of `CROO` is that provides means to both prevent IDF (by detecting IDF attempts), and limit its consequences when sophisticated IDF attacks have bypassed the aforementioned preventive measures. The latter goal is achieved by identifying cases in which undetected IDF victims use any of their credentials. Limiting the consequences of IDF is of use when a fraudster has acquired a user's PIN, stolen the user's personal device, and used the device to generate correct OTPs for unauthorized transactions. `CROO` relies on malware-free personal devices in which authentic copies of select authorities' public keys are stored. As others [28, 53], we believe that, in the near future, a subset of deployed personal devices will meet this requirement, possibly as a result of initiatives such as the Trusted Computing Group [3] (see §4.3).

Contributions. We abstract the IDF problem through the definition of fundamental terms, identification of major stakeholders, and presentation of a generic model of the IDF process. This model can serve as a framework to characterize and compare instances of IDF. Our main contribution is the proposal of `CROO`, a generic architecture and protocol for addressing IDF. (*Generic* here means designed to be simultaneously used with multiple classes of applications and credential tokens, in both online and on-site transactions.) We analyze `CROO` using a combination of informal and AVISPA[4]-based formal analysis.

Outline. §2 abstracts the generic IDF problem. §3 proposes `CROO`. §4 analyzes the proposed scheme. §5 discusses related work. §6 concludes the paper. An AVISPA-based security analysis of `CROO` is included in the Appendix.

2 Abstracting the ID Fraud Problem

This section abstracts the general IDF problem, providing basic definitions, identifying major classes of stakeholders, and presenting a model of the generic process of IDF.

[3] Both health cards (HCs) and drivers' licenses (DLs) can be used to commit IDF. Stolen or cloned HCs can be misused to gain access to expensive medical services, and DLs can be exploited to illegally obtain new credit cards, loans, and merchandise.

[4] AVISPA [68] is a verification tool for security protocols and applications.

2.1 Basic Definitions

We first provide definitions of IDF and related terms suitable for designing and describing technical solutions. Previous definitions do not meet our present purposes (see below). Wang et al. [74] define identity theft (IDT) as the action whereby "criminals use someone else's personal identity and other relevant information in unauthorized ways"; the meaning of term identity and phrase "other relevant information" are implied by example, but not formally characterized. Lacey and Cunagesan [42] and the ACPR [24] respectively define IDT as: the action that "involves an individual falsely representing him- or herself as another real person's identity"; and "gaining money, goods, services, benefits (or avoiding obligations) through use of a false identity". For our work, we prefer a definition characterizing or summarizing critical steps of the IDF process (see §2.3). Gordon and Willox [29] define IDF as "the use of false identifiers, fraudulent documents, or stolen identity in the commission of a crime", without formally defining the expressions "false identifiers", "fraudulent documents", "stolen identity", or "crime". For our work, we favor a descriptive approach of common critical steps of application-specific instances of IDF, and attempt to avoid words that have context-specific legal connotation (e.g. *illegal* and *criminal*).

We define *identity* (ID) as a collection of characteristics by which a person is known. An individual may have more than one identity; e.g. the combination of a person's legal name, profession and citizenship may constitute one of their identities. A person's identity is said to be *real* (resp. *fictitious*) if this person exists (resp. does not exist) in reality. For example, the identity of an imaginary character in a science fiction novel is fictitious, while the identities of this paper's authors are real. We use the term *identifier* to denote any label assigned to an identity to distinguish this identity from other identities. For example, someone's customer number is an identifier of this person. We use the phrase *credential information* (*cred-info*) to denote information (or a piece thereof) presented by a party to either gain privileges or goods, or to support the veracity of an identity-related claim made by this party. For example, the combination of a userid and a password is cred-info that may grant access to resources. Alternatively, information found on a passport can be shown to confirm someone's claim to a certain legal name. Similarly, we use the phrase *credential token* (*cred-token*) to refer to an object (either tangible, e.g. a bus pass, or intangible, e.g. a digital data structure) on which cred-info is recorded. Cred-tokens are normally issued by a *credential issuer* to a specific person. Someone holding a cred-token associated with one of their true identities (i.e. an identity that does not misrepresent the person) is called a *legitimate holder* of both this cred-token and the cred-info thereon. Cred-info recorded on a cred-token may be recorded either in *cleartext* (i.e. intelligible) form or in *encrypted* form (i.e. non intelligible without an associated code or secret key).[5] Finally, we define *IDF* as unauthorized exploitation of extracted and possibly propagated cred-info through the use of false (possibly fictitious) identity.

2.2 Stakeholders and Flow of Credential Information

We now categorize and specify the goals of parties involved in IDF. Wang et al. [74] identified five major stakeholders in the IDF problem: identity owners, issuers, checkers, protectors, and abusers. We include several others, as indicated in the next paragraph. Examples of concrete stakeholders are provided in the subsequent paragraph.

We identify 12 stakeholders in a generic IDF model; Fig. 1(a) shows 11 of these. A *user* is any legitimate holder of a cred-token. An *illegitimate credential holder* is any party that holds a cred-token or cred-info without being a legitimate holder thereof. An *ID issuer* is a party that assigns *identifiers* to people's identities (i.e. controls a name space). A *credential issuer* is a party authorized to produce and distribute credential tokens. A *reputation estimator* is a party that issues scores used to estimate the credibility or risk-worthiness of users. A *credential relying party* is one that relies on credential-related claims usually to provide services or goods. For this to happen, people must demonstrate either possession of specifiable cred-tokens or knowledge of cred-info thereon. A *credential safe* is a party or device trusted by a user to both store and control access to cred-tokens or cred-info. A *credential information broker* is a party that does not typically intend on committing IDF, but that collects and propagates or sells cred-tokens, cred-info, or personal information associated with users (e.g. lists of items bought). A *credential manager* is a party trusted by users to store, control access to, and manage cred-tokens or cred-info issued to users (e.g. for the purpose of a single-sign-on service). A *credential extractor* is a party that gathers cred-tokens, cred-info, or personal information with or without explicit authorization (e.g. by users). A *credential propagator* is a party that distributes cred-tokens, cred-info, or personal information, with or without explicit authorization. A *credential exploiter* is a party that exploits cred-tokens, cred-info, or personal information (extracted and possibly propagated), with or without proper authorization and through the use of false identity.

[5] Cred-info encoded in a non-intelligible form can be considered encrypted.

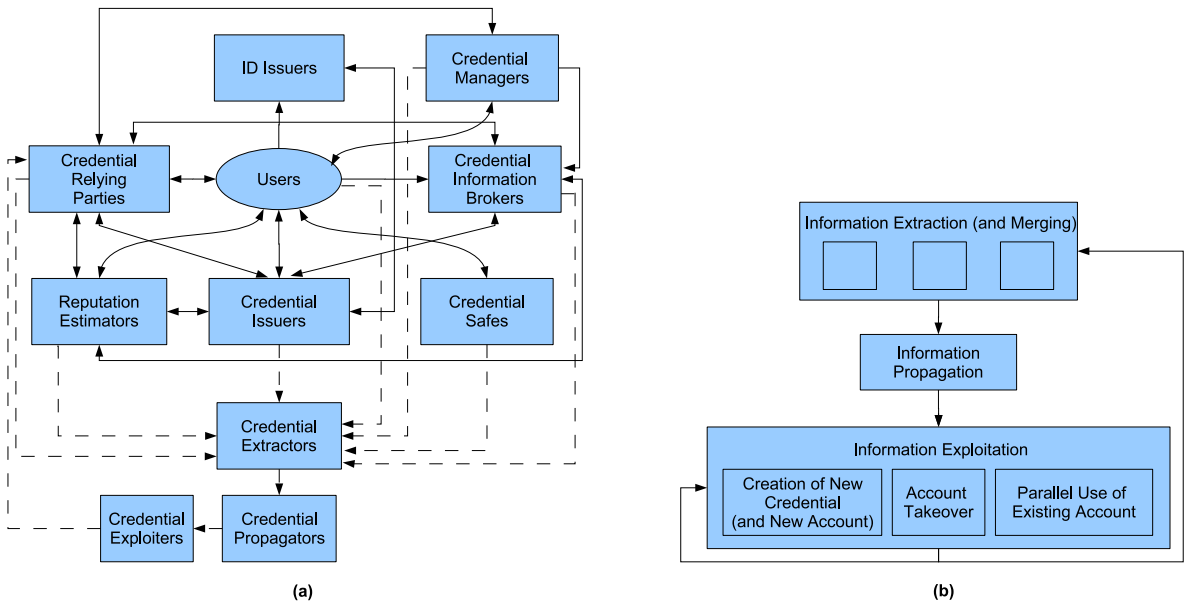

Figure 1: A Generic ID Fraud Model: (a) Parties Involved in ID Fraud; (b) Model of the ID fraud Process. Arrows indicate directional flow of information. Dashed lines are connected to malicious parties.

For example, in the case of credit card-based commerce: credit cards are cred-tokens; credit card numbers are pieces of cred-info; credit card companies are both ID issuers and credential issuers; people to whom legitimate credit cards have been issued are users; credit bureaus are reputation estimators;[6] merchants that deliver goods and services to customers that present credit cards are credential relying parties; data brokers[7] are credential information brokers; Microsoft Passport is a credential manager (e.g., it can use credit card numbers on behalf of users to facilitate a single-sign-on service); phishers that collect credit card numbers are credential extractors; crime rings that sell and distribute illicitly copied credit card numbers are credential propagators; and fraudsters who use credit card numbers not legitimately issued to them are credential exploiters.

Fig. 1(a) shows information flows between IDF stakeholders. For example, users may provide personal information to identity and credential issuers (e.g. credit card companies) in order to obtain credential tokens (e.g. credit cards). Users may also obtain reputation scores (e.g. FICO scores [56]) from reputation estimators such as credit bureaus, and request reputation score revisions by sending required information to reputation estimators. Users may also send and retrieve personal credential information to and from credential safes and credential managers. Credential issuers may send account activity reports to users. When interacting with credential relying parties, credential managers may provide credential information and obtain transaction receipts on behalf of users. Credential relying parties (e.g. merchants) may sell user buying patterns and profiles[8] to credential information brokers. Credential extractors may steal credential information from reputation estimators, credential issuers, credential managers, or credential safes. Credential extractors may also pose as legitimate credential relying parties (e.g. insurance providers) to obtain user profiles from credential information brokers. Credential propagators may sell, to networks of credential exploitors, credential information bought from credential extractors. As a final example from Fig. 1(a), credential exploiters may impersonate users by providing credential relying parties with credential information bought from credential propagators.

2.3 Model and Characterization of Generic ID Fraud Process

In this section, we provide a model and characterization of the IDF process. This is intended to be of use to analyze instances of IDF. The process of IDF can be modeled as a sequence of three phases, each having multiple characteristics and occurring under several possible circumstances. Fig. 1(b) depicts the three phases of the IDF process. List elements A1 through A9 below specify various characteristics and circumstances associated with each phase of this process.

[6] In North America, major credit bureaus (like Equifax, TransUnion, and Experian) use the Fair Isaac Company (FICO) score [56] to estimate and report on the financial credibility (e.g. entitlement to monetary credit) of people.

[7] Large U.S. data brokers include ChoicePoint, Lexis-Nexis and Acxiom.

[8] A user's profile may include the user's name, address, gender, and number children with corresponding ages. This information may be collected via forms filled by users, at retail stores or web sites, to participate in prize-winning contests.

The process of IDF (see Fig. 1(b)) starts with the extraction of cred-tokens, cred-info, or personal information. (Henceforth, *cp-info* will refer to "cred-info and personal information".) These objects or pieces of information are then communicated to exploiters who use them, under false identity, without proper authorization. When credential exploiters are not also credential extractors, we say that the cp-info obtained by the exploiters has been *propagated*. Cred-tokens or cp-info obtained by credential exploiters can be used (repetitively) for multiple purposes , including: (1) obtaining newly created credentials; (2) use of victims' existing credentials or accounts, in parallel with these victims; and (3) exclusive use, by ID fraudsters, of victims' cred-tokens or accounts. ID fraudsters can use their victims' cred-tokens and accounts in order to gain privileges or conceal their true identities.

Each phase of the IDF process (i.e. credential extraction, propagation, and exploitation) can be characterized by several attributes, including A1 through A9, which we derived from a diverse collection of reported ID fraud cases.

A1. *Acting Party.* The party that performs an extraction, propagation, or exploitation of cred-tokens or cp-info is referred to as the acting party. (More precisely, this party may be called the *extractor*, *propagator*, or *exploiter* depending on the phase they are associated with.) This party may be a single individual or a group thereof. The acting party may even be trusted by the victim, e.g. a family member, colleague, or neighbor; or alternatively, a competitor or attacker either of the party storing the victim's cred-tokens and cp-info, or of the party providing services or goods to the victim.

A2. *Action Target.* Action target refers to: the party or system from which cred-tokens or cp-info are extracted; the party or system to which cred-tokens or cp-info are sent; or the party or system targeted by an exploitation attack. An action target may be a computer system, person, or physical location. If it is a computer system, then the system may be private or publicly accessible. In the case of a private system (e.g. a company or home-based system), this system may be mobile (e.g. a cell phone) or fixed (e.g. a desktop PC). If the action target is a computer system, it may store cp-info about a single or multiple individuals.

A3. *Authenticity of Action Tokens and Information.* The cred-tokens or cp-info extracted, propagated, or exploited may sometimes be clones or forgeries of legitimate cred-tokens or cp-info.

A4. *Action Medium.* The channel used to extract, propagate or exploit cred-tokens or cp-info may be a person, traditional postal mailing system, telephone system, or computer system. In the case of a computer system, it may be privately or publicly accessible, and home- or company-based.

A5. *Action Technique.* The extraction, propagation, or exploitation of cred-tokens or cp-info may involve several techniques, including: social engineering, network-based intrusion, network-based traffic sniffing, physical attack (e.g. pickpocketing and dumpster diving), monitoring of user input (e.g. shoulder surfing and key logging), network traffic redirection (as in phishing and pharming attacks), parallel use of existing user accounts, obtaining of newly created cred-tokens or cred-info (either from legitimate or illegitimate parties), opening of new user accounts (e.g. opening credit card accounts in the name of victims), and taking over of existing user accounts (i.e. exclusive use of these accounts).

A6. *Application Context.* Relevant cred-tokens or cred-info may be associated with various contextual applications, including: banking, telecommunications, access control (e.g. building or computer system access control), health care, housing, employment, transportation, immigration, and club membership.

A7. *Factuality of Associated Identity.* The identity associated with a cred-token or cp-info is either real or fictitious. In the latter case, it may be completely fictitious or a modified version of a real identity.

A8. *Motivation.* The acting party may be motivated by various reasons, including: financial gain, material gain, gain of privileges, avoidance of financial obligations (e.g. using another's health card), concealment of identity, and framing of an individual [36].

A9. *Interaction with Issuer and Victim.* The acting party may sometimes be required to directly interact with the issuer or legitimate holder of the cred-token or cp-info in question.

This IDF process model – i.e. the aforementioned three phases and their attributes – is intended to be of use to better understand and analyze the ID fraud problem and instances thereof.

3 Proposed Architecture and Protocol for IDF Detection

This section proposes `CROO`, a generic architecture and protocol for IDF prevention and/or after-the-fact detection for consequence limitation. It is intended to be simultaneously usable with multiple classes of applications and cred-tokens, for both online and on-site transactions.

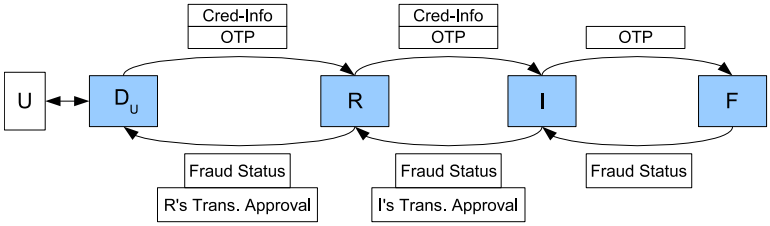

Figure 2: Overview of Proposed Solution
U is a user, D_U is a device carried by U, R is a credential relying party, I is a credential issuer, and F is a fraud detecting party. OTP stands for one-time password.

3.1 Overview

CROO uses knowledgeable online trusted parties to detect IDF when it is attempted, or when a undetected IDF victim uses any of her cred-tokens. These online trusted parties need not be the same for all users and users may use different such parties for different cred-tokens (see §3.5). We emphasize that the proposal may be implemented using various protocols; one representative protocol is presented in §3.2 and §3.3.

It is assumed that a user U has a trusted personal portable computing device D_U, able to communicate over a short range wireless channel (e.g., a Bluetooth-enabled cell phone or PDA). By a trusted device, we mean one that prevents unauthorized access to its internal states, if the device is not physically obtained by an attacker.[9] Below, we also explain how CROO deals with captured personal devices. It is assumed that a credential token C_U (e.g. a credit card) has been issued to U by a party I (e.g. a credit card company).[10] U must present C_U to a credential relying party R (e.g. a point-of-sale or web transaction site) to execute transactions whereby either U gains privileges, goods or services, or U is required to show convincing evidence of the truth of identity-related claims she has made. i is a positive integer indicating that C_U is used by U for the i^{th} time since some starting point. F is an online party that monitors the use of C_U and is trusted by both I and R.

The main idea behind CROO (see Fig. 2) is that each (say the i^{th}) transaction involving C_U (or one of its clones) requires F to validate a one-time password $k^{(i)}$ encrypted by D_U for F and sent to F via R and I. The only way to generate this one-time password is to access a non-verifiable plaintext [45] stored on D_U in encrypted form. To do so, an attacker must guess a PIN chosen by U and input to D_U each time U initiates a transaction. Since F is the only party that can verify the validity of one-time passwords, CROO *turns off-line attacks against a PIN into online attacks against one-time passwords.* If, for a small number of consecutive trials (e.g. 5 or 10), F finds that it has never accepted $k^{(i)}$ and $k^{(i)}$ has not an expected value, then F concludes that an attacker is attempting to impersonate U. (F may then temporarily block requests from U, notify U, e.g. via D_U, email, or a home phone, of the attack, and, if needed, request that the non-verifiable plaintext stored on D_U be reset.) F is thereby able to detect and address attacks targeting U. If $k^{(i)}$ has previously been accepted by F, then F concludes that U was impersonated the previous time $k^{(i)}$ was accepted.[11] Hence, F is able to detect cases in which IDF has been committed against U. Finally, if $k^{(i)}$ has never been accepted by F and has an expected value, then F accepts the password and concludes that the legitimate U has used D_U to generate $k^{(i)}$.

CROO is designed in such a way that, in the course of any transaction T, R only interacts with U (through D_U) and I, and I only interacts with R and F. Moreover, the scheme allows U to send cred-info to I via R, without R being able to access this information in intelligible form. This is achieved by having D_U encrypt the cred-info for I. Likewise, D_U encrypts $k^{(i)}$ for F and sends the ciphertext to F via R and I. R completes a transaction only if I indicates to R that cred-info sent by R for the transaction T has been approved. Moreover, I approves of T only if F approves of T (based on $k^{(i)}$) and if the cred-info that was sent by D_U to I is sufficient and correct. (For example, I may require that a credit card number sent by D_U to I be valid, that this number match a given name, and that this number correspond to an account with sufficient credit limit.) §3.2 and §3.3 describe a protocol instantiation of the proposal.

[9] Such unauthorized access may be used for unauthorized key logging, concealment of authorized output messages, output of unauthorized messages, or unauthorized secret key extraction and communication to other devices.

[10] For each application for which protection is to be provided, we assume such a cred-token.

[11] It is assumed that U does not maliciously submit two identical OTPs to F, and the odds that an attacker submits a valid OTP without being detected by F are negligible. Hence, two identical submitted OTPs associated with U essentially implies impersonation of U.

3.2 Architectural Components

Parties. Let I be a party that issues cred-tokens and authorizes, when needed, the execution of operations associated with cred-tokens issued by I. (For example, I may be a credit card company that issues credit cards and authorizes payments made with these cards.) Let F be a party that monitors the use of cred-tokens, and can assign identifiers to a person U. (In some practical instantiations, F may be a sub-component of party I, and/or the two may be co-located.) Assume that I issues a cred-token C_U to U, and let R be a party that provides goods or services to any person A, when the following conditions are satisfied: (1) A presents to R either certain cred-tokens (e.g. a credit card) or pieces of cred-info (e.g. a credit card number and a name); and (2) either the items presented to R grant A required privileges, or confirm that A has required attributes (e.g. is of a certain age).

Personal Device. U acquires a personal trusted computing device D_U equipped both with an input/output user interface and capability to communicate via a standard short range wireless (SRW) channel (e.g., a Bluetooth-enabled cell phone, if suitable as a trusted computing platform, or a small special-purpose device usable for multi-application IDF prevention and detection.) Any communication between D_U and F, R, or I is over the SRW channel. When R is an online party in a web-transaction (rather than a physically present point of sale), then communication between D_U and R combines SRW communication between D_U and a PC, and Internet-based communication between this PC and R.

3.3 Anti-Fraud Protocol

In what follows, the notation from Table 1 is used.

Public Key Setup. Based on trust and reputation, U selects an appropriate party I trusted by R, and an appropriate party F trusted by I and R. (U may select I and F independently from R; R may trust I and F on the basis of their popularity among users.) R acquires authentic copies of public keys of both I and F (e.g. by contacting I and F directly, or from trusted representatives), and I and F obtain an authentic copy of the other's public key. I, F, and R are assumed to safeguard the confidentiality of their respective private (signing and decryption) keys.

Secret Generation Setup. To generate initial secret keys, U goes to F, in person, and uses D_U to partially automate the setup procedure presented below. Then, U indicates to I (e.g. by phone) that F monitors C_U. More precisely, U appears before F to allow F to verify that U is who she claims to be.[12] This is done using standard (e.g. out-of-band) techniques and, if possible, the successful use of trusted cred-tokens that utilize the transaction protocol described below. (For example, U may be required to successfully complete a payment to F, with a credit card whose use requires the completion of the transaction protocol described below.) Once F is convinced of U's identity, the following take place:

1. U causes D_U to generate, and send to F, an identifier ID_{C_U} of C_U. (ID_{C_U} must be unique with respect to U's cred-token identifiers, but not necessarily to the identifiers of all cred-tokens monitored by F.)
2. F sends to D_U a global identifier ID_I of C_U's issuer, assigns to U a unique (permanent) identifier ID_U, and sends both ID_U and an authentic copy of F's public key to D_U. (ID_I is provided by a trusted global naming authority and allows (among other things) R to know whom to contact when U initiates a transaction associated with C_U. ID_I need not be established before R and F use it in the proposed system.)
3. U chooses and memorizes a d_1-digit secret PIN, p_U. D_U generates a d_2-bit secret random salt s_U. U inputs p_U into D_U, and D_U computes $k^{(0)}$, where $k^{(n)}$ is a random d_2-bit string generated by D_U, and $k^{(j)} = h(s_U, k^{(j+1)})$ for $j = n-1, n-2, \cdots, 0$.[13] Then, D_U sends $(ID_U, s_U, k^{(0)})$ to F (over a secure, i.e. integrity-protected, confidentiality-protected and authenticated, channel), and F securely stores this 3-tuple as an entry in a table B_F. (The size of B_F is linear in the number of transactions processed by F, and search in B_F can be done in constant time.)
4. D_U generates a (d_2-bit) nonce q, computes $\{s_U, k^{(n)}, q\}^{\hat{p_U}}$ (i.e. symmetrically encrypts the concatenation of s_U, $k^{(n)}$, and q with a key derived from p_U) and stores the result on D_U. D_U also sets to 0 a counter i, and erases p_U and $\hat{p_U}$ from its memory.
5. U indicates to I that F monitors C_U.

Transactions. Table 2 provides a protocol summary for reference, while the paragraphs below present detailed explanations of each protocol step. The notation from Table 1 is used. Suppose that U wants to use C_U in a

[12] Instead, U may visit a trusted representative of F. However, for simplicity, we henceforth assume that U visits F.

[13] The string concatenation of x and y is henceforth denoted by x, y or (x, y).

Symbol	Explanation
d_1, d_2, d_3	Length parameters, e.g. $d_1 \geq 4$, $d_2 = 160$, $d_3 = 128$.
n	System parameter limiting how many (e.g. 10,000) cred-tokens or pieces of cred-info are monitored by F.
$\{x\}^y$	Symmetric encryption of x using key y (e.g. using AES with d_3-bit keys).
$[x]_V$	Asymmetric signature (e.g. using RSA-PSS) of x using V's signature key. (For simplicity, it is assumed that x's size is small enough to allow for direct asymmetric signature. If x's size was too long, $[x]_V$ could be replaced by $[h(x)]_V$, where $h(x)$ is assumed to be small enough to be used directly for asymmetric signature.)
$[x]^V$	Asymmetric encryption (e.g. using RSA-OAEP) of x using V's encryption public key. (For simplicity, it is assumed that x's size is small enough to allow for direct asymmetric encryption. If x's size was too long, $[x]^V$ could be replaced by $(\{x\}^y, [y]^V)$, where y is a fresh randomly generated symmetric key assumed to be small enough to be used directly for asymmetric encryption.)
$[x]_V^W$	$[x]_V^W = [x, [x]_V]^W$ can be decrypted using W's decryption key, and it is then possible to verify whether $[x]_V^W$ was generated using V's signature key.
C_U	Cred-token issued to U by I.
m_{C_U}	Digital representation of cred-info associated with C_U (e.g. U's name and credit card number)
ID_U	Permanent identifier assigned to U by F.
ID_{C_U}	Identifier of C_U (e.g. a serial number) assigned by U (e.g. using D_U).
ID_I	Permanent identifier of I (provided by a trusted naming authority).
ID_R	Permanent identifier of R (provided by a trusted naming authority).
p_U	d_1-digit PIN chosen by U.
s_U	d_2-bit secret random salt generated by D_U.
$\hat{p_U}$	Symmetric key derived from p_U (e.g. first d_3 bits of $h(p_U)$).
h	Cryptographic hash function (e.g. SHA-1) with co-domain elements of same bit length as s_U.
$k^{(j)}$	j^{th} d_2-bit one-time password. $k^{(n)}$ is a random secret d_2-bit string. $k^{(j)} = h(s_U, k^{(j+1)})$ for $j = n-1, n-2, \cdots, 0$.
z	Transaction details (e.g. timestamped combination of merchant identifier and dollar value of goods or services).
S_z	Binary-valued fraud status issued by F for the transaction associated with z.
A_z	Receipt issued by I concerning the transaction associated with z.
G_z	Receipt issued by R concerning the transaction associated with z.
q_j	d_2-bit random nonce (generated by its user, right before its use).

Table 1: Notational Overview

U	D_U	R	I	F	Messages Sent
1.		$\leftarrow$			z
2.	$\leftarrow$				z
3.	$\rightarrow$				p_U
4.		$\rightarrow$			$(v, w, h(z), ID_I)$, where $v = [ID_U, ID_{C_U}, ID_I, k^{(i)}, h(z), q_1]^F$ and $w = [m_{C_U}, h(z), q_2]^I$
5.			$\rightarrow$		$[v, w, z, ID_R, q_3]^I$
6.				$\rightarrow$	$[v, h(z), q_4]_I^F$
7.				$\leftarrow$	$[b]_F^I$, where $b = (h(z), S_z, q_5)$
8.			$\leftarrow$		$[A_z, b, [h(z), S_z, q_5]_F, q_6]_I^R$
9.		$\leftarrow$			$(b, G_z, h(v), [b, G_z, h(v)]_R, [h(z), S_z, q_5]_F)$
10.	$\leftarrow$				S_z

Table 2: Transaction Fraud Verification Protocol

transaction T involving R. Assume that this transaction requires U to provide I with cred-info m_{C_U} (e.g. the combination of U's name and credit card number), and that R must receive from I a confirmation that m_{C_U} is correct, sufficient, and non-fraudulent. (For example, if m_{C_U} encodes a credit card number and a name, I may verify correctness by determining whether the number matches with the name.)

- *Step 1.* To initiate T, R generates a string z specifying transaction details, e.g. the current date and time, the identity of R, and a dollar value associated with a list of items. R also stores z in a temporary variable f_z, and sends z to D_U .
- *Step 2.* D_U displays z to U in an appropriate format.
- *Step 3.* If U does not agree with the terms of z, then U signals disagreement using D_U, and D_U relays this decision to R. If U agrees, then U conveys this decision to D_U and inputs p_U in D_U.
- *Step 4.* If this is the i^{th} time U uses C_U, then D_U decrypts $\{s_U, k^{(n)}, q\}^{p\hat{U}}$ with $p\hat{U}$, generates a nonce q_1, computes $k^{(i)}$ as described in Table 1, and D_U computes $v = [ID_U, ID_{C_U}, ID_I, k^{(i)}, h(z), q_1]^F$. v indicates to F that the cred-token identified by ID_{C_U} and issued by I to U is used for a transaction that can be identified using $h(z)$. v also provides F with an OTP $k^{(i)}$, and q_1 (as all the other nonces[14] henceforth used) is intended to prevent an attacker from guessing the content of v and verifying her guess by encrypting it with F's public key. D_U also generates a nonce q_2 and computes $w = [m_{C_U}, h(z), q_2]^I$. w provides I with m_{C_U} (credential information associated with C_U), indicates to I that m_{C_U} was encrypted for the purpose of T, and avoids that unauthorized parties obtain m_{C_U}. Then, D_U sends $(v, w, h(z), ID_I)$ to R (over an integrity-protected channel).
- *Step 5.* To detect replay attacks (e.g. of v or w) and for accounting purposes, R maintains a table B_R (whose entries' format shall be made clear in Step 9(A)). (The size of B_R is linear in the number of transactions processed by R, and search in B_R can be done in constant time.) If, upon receiving $(v, w, h(z), ID_I)$, B_R has an entry containing $h(v)$ or $h(w)$, then: (a) R generates a string G_z that specifies R's decision to cancel the transaction T because of a replay attack; and (b) R proceeds to Step 9(B). Otherwise, R executes the following 3 steps: (i) it retrieves, from f_z, the string z such that $h(z)$ is the third element of the received tuple $(v, w, h(z), ID_I)$; (ii) it generates a nonce q_3; and (iii) it computes $L = [v, w, z, ID_R, q_3]^I$, where ID_R is an identifier of R assigned by a naming authority. ID_R is later on used by I to retrieve the encryption public key of R, and (v, w, z, ID_R, q_3) is encrypted to avoid that any party except I obtain z and the associated information. R sends L to I.
- *Step 6.* Upon receiving L, I decrypts it. To detect replay attacks, I maintains a table B_I (whose entries' format is made clear below). (The size of B_I is linear in the number of transactions processed by I, and search in B_I can be done in constant time.) If B_I has an entry indexed by $h(z)$, then I sets b to be any short constant string (e.g. the empty string), and specifies, in a string A_z, that T failed due to a replay attack. Otherwise, I adds to B_I the entry $(h(v), h(w), z, b)$.[15] Then, I decrypts w, makes sure that the plaintext recovered from w includes $h(z)$ (in second position), and verifies that m_{C_U} is correct and sufficient. If not, I sets b to be any short constant string (e.g. the empty string), specifies, in a string A_z, that the cred-info provided by D_U failed, and proceeds to Step 8(B). Otherwise (i.e. if m_{C_U} is correct and sufficient), I generates a nonce q_4, computes $X = [v, h(z), q_4]^F_I$, and sends this ciphertext to F. The ciphertext provides F with v (the information given by D_U for F), and an identifier $h(z)$ of the transaction details listed in z. X also allows F to determine (via signature verification) whether X was issued by I (i.e. using I's signature key), and $h(z)$ can be compared with information provided by D_U in v (to make sure that the OTP contained in v was issued for the same transaction to which I refers in X).
- *Step 7.* Upon receiving X, F decrypts it, makes sure (through signature verification) that it was issued by I, decrypts v, and verifies whether the plaintext obtained from v (i.e. $(ID_U, ID_{C_U}, ID_I, k^{(i)}, h(z), q_1)$) satisfies the following: (1) ID_I is the identifier of the party that issued X;[16] (2) $h(z)$ matches the corresponding transaction detail identifier sent by I;[17] and (3) the received OTP $k^{(i)}$ is correct with respect to the entry indexed by ID_U in B_F. This last condition is verified as follows: if the entry of B_F indexed by ID_U has the form $(ID_U, s_U, k^{(0)})$, i.e., is a 3-tuple of appropriate length, then $k^{(i)}$ is correct if and only if $i = 1$ and $k^{(0)} = h(s_U, k^{(i)})$; otherwise, the entry of B_F indexed by ID_U has the form $(ID_U, s_U, (k^{(j)}, ID_{C_U}, ID_I, h(v_j))^{i-1}_{j=0})$ (where i is the index[18] of the last transaction accepted by F in

[14] The term *confounder* could be used instead of *nonce* (cf. [45]).

[15] Entries of B_I have this form.

[16] This is checked by signature verification.

[17] This prevents an attacker (e.g. a malicious I) from capturing v and sending it to F in association with another transaction than the one for which v was issued.

[18] The current value of i can be inferred by F from the length of the entry of B_F indexed by ID_U.

association with U, and v_j is information sent in the j^{th} transaction accepted by F as non-fraudulent), and $k^{(i)}$ is correct if and only if $k^{(i-1)} = h(s_U, k^{(i)})$. If $k^{(i)}$ is correct, then F sets the fraud status S_z of z to 0, where a value of 0 indicates that the transaction T (whose details are listed in z) is non fraudulent; otherwise, F sets $S_z = 1$ (to indicate that T is fraudulent). If the OTPs sent by (or on behalf of) ID_U have been incorrect for more than a small number of times (e.g. 5 or 10), within a specified time period, then F concludes that U's cred-tokens are currently under attack and F deals with this situation according to a predefined procedure. (For example, F may temporarily declare all uses of cred-info associated with ID_U as fraudulent.) If $k^{(i)}$ is incorrect for the current index i, but correct for a previous index j (i.e. $k^{(j)} = h(s_U, k^{(i)})$ for some j such that $0 \leq j \leq i-2$), then F concludes that U has been impersonated and deals with this case according to another predefined procedure. (For example, F may directly notify U accordingly, e.g. by calling D_U if D_U is a mobile phone.) Then, F generates a nonce q_5, and computes $[b]_F^I$, where $b = (h(z), S_z, q_5)$. b indicates that the transaction identified by $h(z)$ has fraud status S_z, and includes a signature by F of $(h(z), S_z)$. If $S_z = 0$, then F updates the entry of B_F indexed by ID_U to have the form $(ID_U, s_U, (k^{(j)}, ID_{C_U}, ID_I, h(v_j))_{j=0}^{i})$, where $v_i = v$. (For $0 \leq j \leq i-1$, v_j is the first component of the 3-tuple sent by D_U in Step 4, in the context of the transaction whereby F accepts $k^{(j)}$.) ID_{C_U}, ID_I, and $h(v_i)$ are stored by F so that F can indicate (to authorized parties, e.g. U, at any given time) which of U's cred-tokens have been successfully used (along with associated credential issuers and transaction identifiers). (If needed, F may also store the time and date of these transactions.) Note, however, that F does not need to receive private information recorded on C_U (ID_{C_U}, which is generated by D_U, does not need reveal such private information). If required, F may store $(ID_U, s_U, (k^{(j)})_{j=0}^{i})$, instead of $(ID_U, s_U, (k^{(j)}, ID_{C_U}, ID_I, h(v_j))_{j=0}^{i})$. In any case, $[b]_F^I$ is then sent to I by F.

- *Step 8.* This step consists of two parts, namely (A) and (B). (A) Upon receiving $[b]_F^I$, I decrypts it and makes sure it comes from F (through signature verification). (B) I then generates a nonce q_6, and computes $Y = [A_z, b, [h(z), S_z, q_5]_F, q_6]_I^R$, where A_z is a string that indicates: (1) whether I authorizes the transaction T (with respect to the transaction details listed in z); and (2) any required complementary information (e.g. reasons guiding I's authorization decision). (To compute Y, I uses R's encryption public key, obtained using ID_R, from a trusted representative of R.) Then, I sends Y to R.
- *Step 9.* This step consists of two parts, namely (A) and (B). (A) Upon receiving Y, R decrypts this ciphertext, and makes sure (through signature verification) that it comes from I. If the table B_R has an entry containing transaction details $\hat{z}$ such that $h(\hat{z}) = h(z)$, where $h(z)$ is extracted from b, then R generates a string G_z that specifies R's decision to cancel the transaction T because of a replay attack. Otherwise, R retrieves z from f_z, and stores $(z, A_z, b, [h(z), S_z, q_5]_F, q_6, [A_z, b, [h(z), S_z, q_5]_F, q_6], G_z, h(v), h(w))$ in B_R, where z is obtained from the temporary variable initialized in Step 1.[19] All these elements can be used either to detect replay attacks (e.g. of v, w, or Y) or provide detailed information of transactions involving R. (B) If R has completed Step 9(A), then R computes $Z = (b, G_z, h(v), [b, G_z, h(v)]_R, [h(z), S_z, q_5]_F)$; otherwise, R computes $Z = (z, G_z, h(v), [z, G_z, h(v)]_R)$. In either case, R sends Z to D_U, and zeros out f_z and any temporary variable storing v or w. Note that $[z, G_z, h(v),]_R$ is a signature that can be presented by U to an adjudicator J, if G_z indicates that v was not sent to I, while F indicates (and provides convincing evidence) that it has accepted v for a transaction identified by $h(z)$. In such a case, U provides J with $(ID_R, z, G_z, v, [z, G_z, h(v)]_R)$, and J declares R guilty if $[z, G_z, h(v)]_R$ was issued using R's signature key; otherwise, R is declared non-guilty.
- *Step 10.* Upon receiving Z, D_U forms $\hat{Z}$ by removing $h(v)$ from Z, adds $(\hat{Z}, v, w)$ to a table B_{D_U} (used by D_U to save transaction receipts), and then notifies U of S_z. (The size of B_{D_U} is linear in the number of transactions requested by U, and search in B_{D_U} can be done in constant time.) D_U also extracts $[h(z), S_z, q_5]_F$ from Z, verifies whether this signature is from F, and notifies U accordingly. If $S_z = 0$, then D_U increments, by one, the counter i stored in D_U. If R approves of T, then either R is convinced that a claim made by U is true, or R provides expected goods and services to U. Otherwise, T fails, U obtains nothing from R, and if U had made a claim, then R is not convinced that this claim is true.

Fraud Recovery. Upon suspecting that she has been impersonated,[20] U goes to F in person. F verifies that U's claimed identity (using out-of-band procedures), and the following take place:

1. D_U generates two d_2-bit secret random strings s_U and $k^{(n)}$. U inputs p_U in D_U, and D_U computes $k^{(0)}$, where $k^{(j)} = h(s_U, k^{(j+1)})$ for $j = n-1, n-2, \cdots, 0$. Then, D_U sends $(ID_U, s_U, k^{(0)})$ to F, and F replaces the previous entry of B_F indexed by ID_U by $(ID_U, s_U, k^{(0)})$.

[19] Entries of B_R have this form.

[20] Such suspicion may come to U from reviewing personal transaction reports.

2. D_U generates a new nonce q, computes $\{s_U, k^{(n)}, q\}^{p\hat{_U}}$, and stores this ciphertext. Then, D_U erases p_U and $p\hat{_U}$ from its memory.

3.4 Examples

Credit Card Fraud. A real-world instantiation of CROO could be as follows: I is a credit card company; R is an online merchant; U is a legitimate customer of I to whom I issues a credit card C_U; F is a credit bureau; m_{C_U} is a credit card number concatenated with C_U's expiration date and U's name; and D_U is a Bluetooth-enabled cell phone equipped with a software application facilitating web-based online commerce via PCs. The transaction phase of CROO would then look like this: U accesses R's web site from a PC, and selects a list of items to buy; R sends the associated transaction details to U's PC; these details are sent, via Bluetooth, to D_U; U examines the details, and if correct, inputs a PIN into D_U; D_U sends (v, w, ID_I) to her PC and this 3-tuple is sent to R via SSL; R contacts I via SSL and I contacts F in the same way; if C_U's credit limit exceeds the charges associated with the list of items to be bought, and if, according to F (as explained in the transaction protocol), the current transaction is non-fraudulent, then I sends R a promise of payment for an appropriate amount of money; R notifies D_U, via the web and U's PC, that the transaction has been approved, and D_U relays this information to U.

Driver's License Fraud. As a second example, CROO can be instantiated with the following parties: I is a state agency that issues drivers' licences; R is a bank; U is a person to whom I issues a driver's licence C_U; F is a state agency that specializes in the detection of fraud involving state-issued cred-tokens; and m_{C_U} is a driver's licence number concatenated with U's legal name (as written on C_U). (When validation of driver's licence information (e.g. for credit card issuing) does not currently involve online check with a trusted party, this second instantiation of our (online) proposal may be used to better detect driver's license-related IDF.)

Student ID Fraud. A third instantiation of CROO has the following parties: I is a university that issues facility access cards to its students; U is an employee of I; R is a university lab; $F = I$; C_U is a facility access card issued to U by I, and m_{C_U} is an identifier of C_U (e.g. a serial number).

3.5 Extensions

CROO is flexible with respect to the number of credential issuers I and the number of fraud detection parties F. In other words, U may have cred-tokens issued by different parties I, and these parties may rely on different fraud detecting parties F. For example, fraud detecting parties may be peculiar to particular applications or contexts (e.g. financial or government-oriented services). In some cases, however, it may be simpler to associate all the cred-tokens of a user with a single fraud detecting party, even though this party might not be the same for all users (e.g. for scalability purposes). The advantage of using a single fraud detecting party for all cred-tokens of a user is that when fraud is committed with any of this user's cred-tokens, this instance of fraud is detected the next time the user utilizes any of its cred-tokens. This is due to the fact that each one-time password is not bound with a particular cred-token, but with a user and the party that validates this OTP. In other words, one-time passwords are used across cred-tokens and cred-info thereon. Another extension of CROO consists in asking users (say U) to memorize different PINs for different groups of cred-tokens; if a PIN is guessed by an attacker, the cred-tokens associated with PINs that have not been guessed may still be used by U, and the OTPs associated with the non-guessed PINs are not temporarily declared as fraudulent. (Recall that if a fraud detecting party F has received, within a given time frame, a significant number of incorrect OTPs in association with ID_U, F may temporarily declare as fraudulent all received OTPs associated with ID_U.)

4 Analysis of Proposed Solution

This section discusses the usability, privacy-preserving capability, fraud detection capability, and communication security of the proposed protocol (henceforth referred to as S – for scheme). We are primarily interested (see §4.2) in conveying an understanding of S's usability, privacy, and security characteristics (using practical criteria presented in §4.1), rather than algebraically proving the security of S. "Proving" (algebraically) that S is "secure" in a general practical setting would be quite difficult because mathematical models tend to be different from the reality of deployed systems. To complement the analysis of §4.1 and §4.2, the Appendix includes a security analysis of a simplified version of S using AVISPA [68] – a tool for automated validation of internet security-sensitive protocols and applications. §4.3 discusses selected security and privacy-related design requirements of CROO. §4.4 discusses incentives which various parties might have to adopt CROO or IDF detection systems in general.

4.1 Comparative Evaluation Criteria for ID Fraud Solutions

The aim of this section is to provide criteria that can be used to evaluate the effectiveness of the proposed IDF detection scheme. We consider criteria under four categories: usability, privacy (i.e. ability of users to control access to their cp-info), fraud detection capability (i.e. capability to detect IDF attempts or cases in which IDF has been committed without being detected), and communication security (e.g. protection against man-in-the-middle attacks). Presented below, these criteria are not exhaustive, but rather what we hope is a useful first step towards an accepted set of criteria to evaluate IDF solutions in general.

The following notation is used: I is any legitimate credential issuer; U is a user (person); x_U is a cred-token or cred-info issued by I to a person believed to be U; $x_U{}^*$ denotes x_U and/or any clones thereof; and R is a (relying) party whose goal is either to verify claims made by, or provide goods/services to, any party A, provided A demonstrates knowledge of appropriate secret information, or shows possession of certain cred-tokens or cred-info that are both valid and not flagged as fraudulent. Moreover, terms denoted by † can further be qualified by "instantly" or "within some useful time period".

Usability Evaluation Criteria

U1. *No Requirement to Memorize Multiple Passwords.* S does not require U to memorize cred-token-specific or application-specific passwords.

U2. *No Requirement to Acquire Extra Devices.* S does not require U to acquire extra devices (e.g. computers, cell phones, memory drives).[21]

U3. *No Requirement for Users to Carry Extra Devices.* S does not require U to carry extra personal devices (e.g. cell phones).

U4. *Easy Transition from Current Processes.* S does not require U to significantly change current processes to which U is accustomed. For example, U is likely to be used to entering a PIN when using bank cards (vs. having an eye scanned).

U5. *Support for Online Transactions.* S detects instances of attempted and/or committed but previously undetected IDF for online (e.g. web) transactions.

U6. *Support for On-Site Transactions.* S detects instances of attempted and/or committed but previously undetected IDF for on-site (e.g. point-of-sale) transactions.

U7. *Convenience of Fraud Flagging Procedures.* When IDF has been detected (e.g. by a user or system), S provides a convenient mechanism to flag the appropriate cred-tokens as fraudulent. For example, S may enable U to interact with only one party to flag, as fraudulent, any of her cred-tokens.

U8. *Suitability for Fixed Users.* S can be used by fixed users (i.e. who carry out transactions from a constant geographic location).

U9. *Convenience of Fraud Recovery Procedures.* When U suffers IDF, S allows U to easily recover. For example, S may enable U to interact with only one party to obtain new cred-tokens that can be used thereafter, without having to obtain new cred-tokens from a number of credential issuers. Alternatively, S may enable U to interact with only one party that allows her to both continue to use her cred-tokens, and have the assurance that the use of any clones of her cred-tokens will be detected as fraudulent.

U10. *Support for Transactions Involving Off-line Relying Parties.* S detects instances of attempted and/or committed but previously undetected IDF even if R is not able to communicate, in real time, with other parties (e.g. I and F).

Privacy Evaluation Criteria

P1. *No Disclosure of User Location.* S does not disclose U's location information, e.g. to multiple entities, or an entity that shares it with other parties.

P2. *No Disclosure of User Activity.* S does not disclose transaction details regarding U's activity (e.g. what U has bought, and when or where this was done).

P3. *No Disclosure of User Capabilities.* S does not reveal what hardware or software capabilities (e.g. digital camera or printer) U has.

P4. *No Disclosure of User's Private Information.* S does not reveal private (e.g. medical or financial) user information.

[21] S may require U to load new software on an existing general-purpose device.

Fraud Detection Evaluation Criteria

D1. *Determination of Credential Use.* U and I know† when $x_U{}^*$ is used.

D2. *Control on Credential Use.* U and I can control† the use of $x_U{}^*$ (i.e. approve or reject each use thereof).

D3. *Detection of Illegitimate Credential Holder.* When $x_U{}^*$ is presented to R, then U, I, and R can determine whether $x_U{}^*$'s holder is authorized to hold $x_U{}^*$. This credential holder legitimacy check might be based on the possession of a specified token, the knowledge of a memorized secret, the presentation of inherent biometric features, the proof of current geographic location, or some other criterion.

D4. *Determination of Credential Use Context.* U and I can determine† in which context (e.g. R's identity, network location, and geographic location) $x_U{}^*$ is used.[22]

D5. *Verification of R's Entitlement to View Credential.* U and I can determine† whether R is a party to which $x_U{}^*$ is authorized to be shown for specified purposes (e.g. the delivery of cred-tokens, goods, or services).

D6. *Entitlement Verification of Credential Holder's Claimed ID.* R and I can determine† whether $x_U{}^*$ is associated with its holder's claimed identity.[23]

D7. *Fraud Flagging of Credentials.* S allows authorized parties (e.g. U and I) to flag x_U as fraudulent (i.e. indicate in a trusted accessible database that, for a specified period, all uses of $x_U{}^*$ are fraudulent).

D8. *Verification of Credential Fraud Flag.* R can know whether $x_U{}^*$ is currently flagged as fraudulent.

D9. *Detection of Clone Usage.* R (resp. I) can distinguish† x_U from its clones whenever the latter are presented to R (resp. I).

D10. *Detection of Credential Cloning.* U and I can detect† that $x_U{}^*$ is cloned.

D11. *Detection of Credential Theft.* U and I can detect† that x_U is stolen from U.

D12. *Determination of Malicious Fraud Claims.* I can determine† whether U is honest when claiming that $x_U{}^*$ has been used without proper authorization.

Communication Security Evaluation Criteria

C1. *Protection Against Physical Exposure of $x_U{}^*$.* S protects $x_U{}^*$ from being visually captured (e.g. via shoulder surfing) by unauthorized parties, without requiring U's conscious cooperation.

C2. *Protection Against Digital Exposure of $x_U{}^*$.* If $x_U{}^*$ is cred-info, S protects $x_U{}^*$ from being accessed by unauthorized parties using computer systems. For example, S may protect $x_U{}^*$ from being captured in an intelligible form when $x_U{}^*$ is communicated over untrusted channels (e.g. the Internet).

C3. *Protection Against Replay Attacks.* S prevents (or reduces to a negligible proportion) reuse of electronic messages sent to impersonate U.

C4. *Protection Against Man-In-The-Middle Attacks.* S prevents (or reduces to a negligible proportion) impersonation of U through tampering or injection of messages between parties used by S.

C5. *Protection Against Denial of Service Attacks.* S prevents (or reduces to a negligible proportion) denial of services against U.

Specific applications may require that subsets of the proposed criteria be met (as best as possible), but generic IDF solutions may be required to meet many or even all criteria. For practical purposes, instant detection of credential cloning and theft (see D10 and D11) might be optional for generic IDF solutions; the existence of cloned cred-tokens may be more difficult to detect (with current technologies) than their use.

4.2 Evaluation of Proposed Scheme

The following analysis uses the terminology of Tables 1 and 2. We assume that an ID fraudster (attacker) A seeks the successful completion of a transaction T requiring A to provide I with cred-info m_{C_U} (e.g. the concatenation of a credit card number and a name). It is also assumed that: m_{C_U} can be extracted from C_U or any clone thereof; A has either stolen C_U or has obtained a clone thereof; $i-1$ is the index of last use of C_U (i is therefore the index of the presently ongoing use of C_U). In order to initiate T, A interacts with a credential relying party R, and A provides R with a candidate one-time password k^* that is encrypted according to the procedure presented in Table 2. Notation ✓ (resp. ✗) indicates that S meets (resp. does not meet) the associated criterion. Notation ✓✗ indicates that the associated criterion is partially met.

Discussion of Usability.

[22] Note that this criterion may adversely affect user privacy.

[23] For example, $x_U{}^*$'s holder may claim to be *Joe Dalton* while $x_U{}^*$ was issued to *Lucky Luke*. This is different from the situation in which x_U's holder pretends to be Lucky Luke (see D3).

✓ **U1**: U is required to memorize only one PIN, instead of many issuer or application-specific passwords.

✓ **U2** and **U3**: If U carries a Bluetooth-enabled cell phone or PDA, then U does not need to acquire an extra portable device. Otherwise, U needs to acquire and carry D_U.

✓ **U4**: S requires U to enter a PIN when she uses C_U. This is assumed not to cause too drastic a change from current processes (cf. debit card use). Moreover, S uses a standard (wireless) communication channel for interaction between D_U and either R or a local PC communicating with R.

✓ **U5**, **U6**, and **U7**: If R or this local PC can communicate over the standard channel, then S can be used both for on-site and on-line transactions. (Recall that, as presented, [72]'s proposal is restricted to on-site transactions.) S is therefore suitable for both mobile and fixed users.

✓ **U8**: U can flag its cred-tokens as fraudulent by interacting with F (i.e. potentially one party per user).

✓ **U9**: Similarly, U needs to interact with only one party (i.e. F) in order to recover from IDF, and U does not then need to change her cred-tokens (she only needs to send to F new initial OTP information).

✗ **U10**: However, S does not work if R cannot interact with I and F at each transaction.

Discussion of Privacy.

✓✗ **P1**: S does not require the use of user-location information in order to detect IDF (unlike [72]'s proposal). This does not mean, however, that U's location information cannot be known (e.g. by D_U's call carrier if D_U is a GPS-enabled cell phone). Nonetheless, the fact that U's location information is not required to detect IDF may decrease the odds that this information be collected. (Hence, S is potentially less privacy invasive than the scheme in [72].)

✓✗ **P2**: I and F know identifiers of all cred-tokens used by (or on behalf of) U. Nevertheless, S is designed in such a way that m_{C_U} is revealed neither to R nor F.

✓✗ **P3** and **P4**: Similarly, S does not require R to reveal to I private information such as lists of items bought by U and private information recorded on monitored cred-tokens. Instead, R may include in transaction details only information that is necessary, e.g. a dollar value and an identifier of R.

Discussion of Fraud Detection Capability.

✓ **D1**: For T to be successfully completed, I must send its approval thereof to R. Consequently, I detects each use of C_U, and, if necessary, I notifies U accordingly (e.g. via D_U or postal mail).

✓ **D2**: For the same reason, I controls the use of m_{C_U} (i.e. can reject or approve of each use of m_{C_U}). If necessary, I may consult U (e.g. via D_U) before approving of T. Hence, S may be instantiated in such a way that U controls each use of m_{C_U}.

✓ **D3**: In order for F to accept k^* as non-fraudulent (i.e. in order for A to generate the correct k^*), A must have an authentic copy of $\{s_U, k^{(n)}, q\}^{\hat{p_U}}$ and a correct guess of both p_U and i. Since these three pieces of information are not easily obtained without successive guesses of p_U and the cloning or capture of the mobile device D_U, A is not likely to generate the correct k^* within an undetected number of guesses (e.g. fewer than 5 trials). If A is able to clone D_U, then i and $\{s_U, k^{(n)}, q\}^{\hat{p_U}}$ can be obtained, but any significant number of wrong guesses of p_U is detected by F. F notifies I of any fraudulent use of C_U (or m_{C_U}), and, if necessary, F also notifies U (e.g. via D_U). Hence, F, I, R, and potentially U can detect illegitimate credential holders.

✓✗ **D4** and **D5**: I is not necessarily able to reliably determine the context of R (e.g. R's geographic or network location). However, if I and R interact via a broker that is trusted by I, then this broker may be able to provide I with reliable contextual information about R. (Note that certain cred-tokens may not be used with relying parties associated with certain contextual attributes, e.g. foreign relying parties.) If necessary, I can communicate this contextual information to U (e.g. via D_U).

✓ **D6**: S uses m_{C_U} to encode ID_U (i.e. U's claimed identity). Since I obtains m_{C_U} in uncrypted form, I is able to determine whether v is associated with ID_U. Then, I may communicate the success of this match to R (e.g. via A_z). Note, however, that neither I nor F is able to determine, with full assurance, the true identity of A. R may verify A's claimed identity by requesting C_U from U, requiring that a photo of A appear on C_U, and making sure that F declares T as non-fraudulent. However, A may tamper C_U by substituting U's photo with A's. Nevertheless, A is not likely to be able to use C_U successfully, because A is not likely to issue a correct k^* before being detected (and blocked) by F.

✓ **D7**: When U suspects that any of her cred-tokens or any cred-info thereon is stolen or cloned, U notifies F accordingly. Then, F declares each transaction involving U's cred-info as fraudulent, until F and I complete the fraud recovery procedure (see the Fraud Recovery protocol in §3.3).

- ✓ **D8**: Since, at each transaction, R receives a fraud status notification from F, R can reject or approve of each transaction based on up-to-date fraud status information. (The faster messages are communicated between F and I, and I and R, the more up-to-date the information received by R is. If D_U, R, I, and F have synchronized clocks, then these parties may even include a time stamp in each message they encrypt or sign. Among other things, this would allow R to determine whether fraud status notifications it receives are outdated (using a predefined expiration rule).)
- ✓ **D9**: The need for A to provide F with a correct k^* helps F distinguish the presentation of m_{C_U} by U from an analogous presentation by A. Hence, S detects the fraudulent use of cred-tokens or cred-info.
- ✗ **D10** and **D11**: However, S does not detect the actual theft or cloning of cred-tokens or cred-info.
- ✓ **D12**: S does not detect instances of fraud that are intentionally committed by U.

Discussion of Communication Security.

- ✓ **C1**: The use of a small portable device D_U is partially intended to reduce the risk of shoulder surfing attacks when U enters p_U in D_U.
- ✓ **C2**: Moreover, S has been designed in such a way that plaintext cred-info is padded with nonces before being encrypted. This is intended to counter: unauthorized access to confidential cred-info (e.g. access to passwords via *phishing* or *key logging* at untrusted user terminals); and guessing attacks whereby guessed plaintext is encrypted and compared with ciphertext coming from D_U, R, I or F.
- ✓ **C3**: Replay attacks are countered through the storage (by R, I and F) of hashed pieces of received messages (as explained in Step 6 of §3.3).
- ✓ **C4**: The acquisition of public keys from trusted parties, and the use of asymmetric encryption and signature are intended to counter man-in-the-middle attacks (whereby portions of messages sent between D_U, R, I, and F are subtracted, injected, or tampered with).
- ✓ **C5**: A can intentionally generate (and send to F) incorrect OTPs. This would cause a denial of service against U (after a small number of failed transactions, all uses of U' cred-tokens will be temporarily considered as fraudulent). If, on the other hand, the information used by A as m_{C_U} is incorrect or insufficient, then I will not relay encrypted OTPs to F, preventing this attack.

Summary. Overall, `CROO` is expected[24] to provide usability benefits for both users and relying parties; to detect IDF attempts; to identify cases of committed yet previously undetected IDF; and to be resistant to a number of communication-based attacks (e.g. replay and man-in-the-middle attacks, including phishing and PC-based key logging). Two limitations of `CROO` are: its inability to detect cases in which legitimate users perform transactions, and later repudiate them; and susceptibility to denial of service attacks against specific users, by attackers who have gathered sufficient and correct credential information.

4.3 Relative Importance of Security and Privacy-Related Design Requirements

Below, we discuss the relative importance of some security and privacy-related design requirements of `CROO`.

1. *Trusted User Devices*: `CROO` makes use of trusted user devices. Currently, many widely-deployed PDAs and cell phones are arguably more trustworthy than commonplace home PCs, even though this may change in the future [41]. To increase the current relative trustworthiness of PDAs and cell phones, one may use formally verifiable OS kernels [70], or trusted virtual machine monitors [26], among other alternatives [3]. While implementing `CROO` with *any* existing cell phones or PDAs may not be appropriate, we believe that some smartphones (e.g. Blackberry 8700 series) could provide a sufficiently trustworthy computing platform for applications such as credit or debit card transactions.
2. *User Devices Performing Cryptographic Operations*: `CROO` relies on user devices performing common cryptographic computations such as hashing, and encryption, decryption, signing, and signature verification. These operations are provided by the Bouncy Castle Crypto API for CLDC/MIDP-enabled devices. MIDP 1.1 is supported by a large proportion of widely-deployed cell phones and PDAs (thereby making reasonable our assumption concerning the computing capability of user devices).
3. *One-Time (vs. Static) Passwords*: `CROO` uses OTPs instead of static passwords (shared secrets), because the latter do not allow to detect lack of transaction-counter synchrony between user devices and their associated fraud detecting parties (F).

[24] This design-level paper considers a number of theoretical and practical issues of IDF detection. We have not empirically confirmed our usability analysis through a prototype implementation, user lab, or field tests. This is left for future work.

4. *Identification of Previously Undetected IDF*: One may distinguish the detection of IDF attempts, from the identification of previously undetected IDF. The former primarily aims at preventing IDF from happening; the latter primarily aims at limiting the damages of IDF that has occurred. CROO provides means to detect IDF attempts, and, when these fails (e.g. due to more sophisticated attacks), provides means to identify previously undetected IDF.
5. *Transaction Approval by Credential Issuers*: The protocol presented in §3.3 requires I to approve each transaction associated with credential information issued by I. This may be suitable for applications such as credit or debit card transactions, but not for other applications (e.g. due to privacy concerns). For the latter, the protocol presented in §3.3 may be adapted, by removing steps involving I. (The details of such a customization are left for future work.)
6. *Disclosure of Transaction Details to Credential Issuers*: In the §3.3 protocol, I receives transaction details from R. To protect user privacy, care must therefore be taken to reveal only information required to authorize transactions (e.g. transaction date, time, and dollar value).
7. *Detection of Replay Attacks by Credential Relying Parties*: The protocol of §3.3 requires R to detect replay attacks (see *Step 5*). While this may be seen as proper input validation on R's part, the requirement may be removed since R relies on F, F is involved in each user transaction, and F detects replay attacks.
8. *Logical Distinction between Parties Involved in Protocol*: The protocol presented in §3.3 logically distinguishes U, R, I, and F based on their respective roles. However, these parties may be collocated or incarnated by single entities in concrete instantiations, e.g. with $I = F$ (as in the Student IDF scenario presented in §3.4), or $I = R$. When $R = I$ (resp. $I = F$), the protocol steps ensuring secure communication between R and I (resp. I and F) can be removed. While we do not find concrete applications with $R = F \neq I$, the analysis of this scenario is left for future work.
9. *Complexity of Overall Design*: While the fundamental ideas of CROO are relatively simple,[25] we acknowledge that the protocol of §3.3 seems relatively complex. This appears to be the price to pay to achieve generality, given the variety of potential threats and applications.
10. *Online Fraud Detection*: CROO is an online protocol. To *prevent* IDF using an offline protocol is a challenging task. Offline IDF prevention employing user-specific cryptographic keys generated by personal user devices seems to require user devices that are tightly bound to their legitimate users (i.e. which, in practice, cannot be used by impersonators). Tight user-device binding may be achieved through authentication of users via practically unforgeable biometric techniques. Devices providing such a guarantee are currently not widely deployed, and it is not clear if or when they will be. Consequently, we favor online fraud detection, and note that recent trends in computing and communications are towards online protocols (e.g. a significant proportion of widely deployed cell phones are able to connect to the Internet).

4.4 Incentives for Adoption of CROO or new IDF Detection Proposals

In this section, we discuss incentives which various parties might have to adopt CROO or new IDF detection systems in general.

1. *Incentives for Users*: Given the variety and severity of potential damages of IDF (cf. §1), and the potential security and usability benefits of CROO, users may find CROO acceptable. This, however, should be empirically verified in future work.
2. *Incentives for Credential Relying Parties*: It is not clear whether credential relying parties currently have direct financial motivation to adopt new IDF detection systems (including, e.g., CROO). Indeed, these parties sometimes receive money in exchange for services or merchandize, whether or not the money was provided by unauthorized impersonators of legitimate users. In certain scenarios, credential relying parties may nevertheless adopt IDF detection systems for other reasons, such as reputation and credibility (which indirectly might generate financial gain). This might be the case of data brokers who have lost credibility, after having sold information to unauthorized parties (e.g. parties impersonating legitimate clients). Countries whose geographic borders are porous to terrorists may also find IDF detection systems important (e.g. for public safety), and therefore mandate the adoption of these systems by classes of credential relying parties falling under their jurisdiction (e.g. customs offices).
3. *Incentives for Credential Issuers*: As in the case of credential relying parties, it is not clear whether credential issuers currently have direct financial motivations to adopt new IDF detection systems. Many credential issuers are not held legally or financially responsible, if credentials they issue do not provide user-acceptable levels of protection against misuse by unauthorized parties. While credential issuers may

[25] We essentially use non-verifiable text and a user-PIN to generate OTPs encrypted for and verified by online parties.

reimburse their clients for money lost in fraudulent transactions, the overall cost of these transactions is sometimes absorbed by clients through increased interest/insurance rates, and service fees. When credential issuers are also relying parties, they may be motivated to adopt IDF detection systems by incentives such as reputation, credibility, public safety, or reduction of financial losses. Government agencies may be interested in CROO because of its generality (the same fundamental system can be reused for various scenarios).

4. *Incentives for Fraud Detecting Parties*: Trusted financial corporations, such as reputation estimators, may have financial motivations to become backend fraud detecting parties (providing fraud detection services for a fee). Trusted government agencies such as passport issuing agencies may become fraud detecting parties for government-issued credential tokens.

5 Related Work and Comparison

Short-Range Wireless Communication Technologies. To support user authentication and/or authorization, personal electronic devices (e.g. cell phones) are sometimes equipped with the ability to communicate over short-range wireless (SRW) channels. These channels can be implemented using a number of available technologies including Bluetooth [10], IrDA [2], RFID [34], NFC [25], and digital cameras [53]. Bluetooth [10] is an SRW communication technology suitable for low powered devices. It features high throughput and 10 meter-wide broadcast capability with no line-of-sight requirement between communicating devices. IrDA [2] specifies how devices placed in each other's line of sight can communicate using a 5 meter-long wireless channel, supporting information exchange rates of 1 to 16 Mbps. Radio frequency identification (RFID) [34] uses electromagnetic or electrostatic waves to uniquely identify objects, e.g. by physically attaching a RFID *tag* to a target object, and using an RFID receiver to detect the broadcast signal sent by the tag. RFID handles variable weather conditions (e.g. cold and humid environments) and does not require communicating objects to be in the each other's line of sight. Near Field Communication (NFC) [25] is a wireless connectivity technology that enables high-speed 10 centimeter-wide broadcast communication between electronic devices. (Current NFC-enabled devices use RFID.) CROO can be implemented using any of these SRW communication technologies (e.g. Bluetooth, since it is supported by a significant proportion of widely deployed cell phones and PDAs).

Password-based Authentication. Static password schemes,[26] one-time password (OTP) schemes [43], password schemes resilient to shoulder surfing attacks [31, 44, 62, 71], and schemes generating domain-specific passwords from a combination of single user-chosen passwords and multiple domain-specific keys [61, 30] can all be used to authenticate users and thereby solve parts of the problem of phishing and/or IDF. System designers must carefully select and combine multiple cryptographic and non-cryptographic tools and techniques to meet various requirements, e.g. those presented in §4.1. Our scheme can be viewed as a careful combination of known and modified tools and techniques (e.g. cell phones, non-verifiable text [45], OTP-based authentication, and symmetric and asymmetric cryptography) to detect IDF.

Web Authentication using Limited-Use Credit Card Numbers. Rubin and Wright [63] propose a scheme for off-line generation of limited-use (e.g. one-time) credit card (CC) numbers. Each limited-use CC number is generated by a personal trusted device, using a secret key shared between this device and a credit card (CC) issuer. The number is manually input in lieu of a commonplace (static) credit card number, and verified by a credit card issuer using the aforementioned shared key. While similar in some ways, CROO is generic, uses high entropy one-time passwords (vs. 16-digit limited-use CC numbers), and is designed to counter device capture attacks (through the use of PIN-encrypted unverifiable keys). Singh and dos Santos [65] describe another scheme for off-line generation of limited-use credentials. In this second scheme, each limited-use credential is generated by a personal trusted device, using a secret string shared between the device and a credit card issuer. The secret string describes a context-free grammar used to generate expressions whose grammatical correctness can only be verified by parties having the associated grammar-describing string. Unlike our scheme, Singh and dos Santos' is not meant to be generic nor to counter device capture attacks. Shamir [64] describes a scheme to generate one-time credit card numbers via an online interactive procedure whereby CC holders obtain these numbers from CC issuers. A benefit of Shamir's scheme is that its number-generation procedure can be automated using a plugin installed on user PCs. This does not (and is not meant to) counter attacks whereby users' browsers or PCs are compromised, e.g. via PC-based virus infection or key-loggers.

Biometric-based Authentication. Kwon and Moon [40] discuss the use of multi-modal biometric authentication as a way to improve unimodal biometric user identification, and thereby decrease the risk of associated forms of IDF (e.g. immigration fraud). Biometric-based authentication has known limitations. (See, e.g., [17],

[26] including commonplace typed textual password mechanisms, and strengthened password schemes,[1, 30, 51].

Chap. 10, p. 104; in practice, it is difficult to instantiate systems using trusted biometric readers intended to be deployed on remote trusted user PCs.)

Smartcard-based Authentication. Lu and Ali [46] propose the use of network smart cards to secure web-based transactions. The idea is to assign a tamper-resistant smartcard to each user, to have each user be authenticated by her smartcard (e.g. via PIN input through a PC's keyboard), and to equip the smart card with networking capability, in such a way that the card is able to handle web-based transactions. As presented, the scheme does not identify cases of committed yet previously undetected IDF. A compromised PC equipped with a key logger may also maliciously exploit a user-input PIN to control the user's smartcard, when this card is in the PC's smartcard reader.

Detection of Compromised Cryptographic Signature Keys. Just and van Oorschot [35] describe a scheme to detect fraudulent client-based cryptographic signatures. The idea is to require that each client signature be countersigned by a trusted register (TR). Before countersigning a client signature, each TR verifies that the signer knows a key shared between the TR and the associated legitimate user, which is updated each time the client signs a document. To prevent illegitimate countersigned signatures from being accepted, the following mechanism is suggested [35]: (a) before countersigning a client signature, the associated TR time-stamps the client signature; (b) before validating a countersigned and time-stamped user signature, verifiers wait a specified period of time and check whether the signature has been revoked by the TR before the end of the time period; (c) during this period, the claimed client is required to contact her TR and update the shared key; (d) if the client contacts her TR and does not know the expected key, the TR revokes the signature issued with this key. This work focuses on cryptographic signature keys; we deal with the more general context of fraudulent uses of credential information.

Limiting the Effect of Cryptographic Key Exposure. Various public-key techniques have been proposed to limit the effect of key exposure. In threshold cryptosystems [13], each private key is split among n parties and any k of these can reconstruct it. Over time, however, k or more key shares may be compromised. This motivates proactive cryptosystems [32] whereby each private key is split into shares that are regularly renewed. A portion (e.g. any k out of n) of these are required to reconstruct the private key; the corresponding public key remains fixed. In forward-secure cryptosystems, time is divided in time periods and the compromise of a private key at a given time limits exposures to subsequent (as opposed to previous) time periods. Proactive forward-secure schemes [7] combine the benefits of proactive and forward-secure cryptosystems. In key-insulated cryptosystems [21], time is also divided in periods and private keys are split between two parties: the signer and the base. The user can use her private key share autonomously (i.e. without directly interacting with the base), but the base needs to provide a fresh secret to the user at each time period. If an attacker compromises the user's key share, the attacker cannot use private keys associated with previous or subsequent time periods. If an attacker compromises both key shares, she is not able to use private keys associated with previous time periods. Intrusion-resilient cryptosystems [20] combine the signer's autonomy and bi-directional time-security of key-insulated schemes, along with the resilience of proactive schemes with respect to non-simultaneous compromise of private key share holders. If at a given time period, an attacker simultaneously compromises all shares of a private key, then the attacker cannot use private keys associated with previous time periods. All these key-exposure-limiting techniques are designed to decrease the odds that unauthorized public-key signatures be issued; we deal with the general problem of IDF committed with cloned credential information.

SET and Certificate-Based PKIs. The Secure Electronic Transaction (SET) protocol [73] allows credit card (CC) holders to obtain goods or services from merchants without revealing their CC information to the latter. SET is not designed to be used for multiple classes of cred-tokens, nor does it specify methods to identify cases of committed yet undetected IDF. SET also employs user-specific (i.e. CC holder) private keys in a certificate-based public-key infrastructure (PKI); we favor the use of OTPs as user authentication secrets, mostly because their misuse can be subsequently detected and their misuse detection does not call for an associated notification to a potentially large population of parties relying on the validity of public-key certificates associated with compromised signing keys.

Device Capture Resilience. The idea of capture resilience was suggested by Mackenzie et al. [48, 49] to detect attempts of off-line password-guessing attacks on password-protected mobile devices, by requiring password-based user-to-device authentication to be mediated (and, *ergo*, detectable) by online servers. This mediation is implemented using the concepts of partial decryption; a similar proposal by Tang et al. [67] is implemented using static PINs partially stored on mobile devices and on an online server. In contrast, our proposal is implemented using one-time passwords generated from non-verifiable PIN-protected text stored on mobile devices.

Proximity-based Authentication. Corner and Noble [16] propose a proximity-based authentication scheme

to avoid the input of user passwords to access devices such as laptop computers. This can be seen as countermeasure to password capture via keyboards, and therefore as a credential information extraction countermeasure.

On-site Access Control using SRW Communication. McCune et al. [53] propose a secure device-to-device communication scheme usable for on-site user access control (a form of IDF detection seeking to identify unauthorized use of credential information to access restricted physical resources or facilities). This scheme employs user-carried camera-enabled mobile devices to capture 2D barcodes [50] affixed to (or displayed by) access control devices (e.g. electronic door locks). The 2D barcodes encode control devices' public keys. When a barcode is read, the reading device receives, via an independent SRW channel, an authentic copy of the control device's public key. If the two public keys are identical, either is used to secure communication between a user's mobile device and the control device; this can be used to facilitate user access to restricted resources or facilities. Otherwise, either the 2D-barcode or public key received via the SRW channel must have been (maliciously) modified or replaced; access is then denied. Bauer et al. [5] propose an on-site user access control system based on proof-carrying codes and camera-enabled smart phones communicating over SRW channels. Beaufour and Bonnet [6] propose a system for using cryptographic keys stored on Bluetooth-enabled mobile phones to open electronic door locks.

IDF Detection via Location Corroboration. Van Oorschot and Stubblebine [72] propose an IDF detection scheme, whereby users' identity claims are corroborated with trusted claims of these users' location. The scheme requires users to be geographically tracked; this may be a concern for user location privacy.[27] A benefit of the scheme is that it can be used regardless of transactions' purpose and associated credentials. It is designed for on-site transactions.

Remote Authentication using OTP-Generating Tokens. Various companies (e.g. Aladdin, RSA and Mastercard) have developed variants of a scheme whereby hardware tokens or mobile device software are used to generate OTPs which are then manually input into PCs, in cleartext form, for remote user authentication and/or transaction authorization. These variants are typically not (and not meant to be) simultaneously generic, usable with a widely-deployed hardware token (e.g. cell phone), resilient to device capture, and immune to phishing and PC-based key-logging attacks whereby OTPs are copied and then used for unintended transactions.

Web Authentication through Manual Input of OTPs Sent to Mobile Phone via SMS. Signify is a company providing a service, whereby each user gives a mobile phone number, userid, and password to an authentication server (auth-server), at registration time. To access restricted services or resources over the web, each user must manually input her userid and password into a PC, send these two pieces of data to a web-based auth-server, receive an OTP via SMS (at the phone number specified at registration), manually input this OTP into the PC, and send the OTP over the web to the auth-server. If the web-based auth-server receives the OTP that was sent to a user's mobile phone, this user is authenticated. This scheme is not (and not meant to be) resilient to PC-based key-logging attacks whereby userids, passwords and OTPs are captured and used without proper authorization.

Web Authentication Complemented with Mobile Caller ID Authentication. A scheme by Identrica[28] involves a registration procedure as follows: each user provides a mobile phone number, userid, and password to a web-based auth-server S, and receives the phone number of an associated auth-server S'. When a user wants to access web-based restricted services or resources, the user must manually input into a PC her userid-password pair, send this pair to S, and call S' using her mobile phone. If S receives, in a predetermined short time frame, both a userid-password pair and a trusted report, from S', of a phone call received by S' with matching caller ID, the associated user is authenticated and authorized to access restricted web-based services or resources. Using a PC-based key-logger, an attacker may capture a user's userid-password pair, and impersonate this user, provided the user calls S', within an acceptable time frame, using her mobile phone. This scheme is not (and not designed to be) resilient to device capture attacks. A similar scheme involves mobile phone calls to transaction-specific phone numbers provided to users, at transaction time (see `www.saintlogin.com`).

Web Authentication Complemented with PIN-based Response to Calls Sent to Mobile Phones. SecureCall is a scheme[29] whereby provision of web-based services or resources requires each user: (1) to send a correct userid-password pair over the web to a trusted auth-server; and (2) to respond, with a correct PIN, to an associated phone call sent by the aforementioned auth-server to the user's mobile phone. This scheme is not designed to counter mobile phone capture attacks. It is susceptible to PC-based key-logging attacks, whereby userid-password pairs are captured to impersonate users who provide correct PINs, and who answer calls issued to their mobile phones from auth-servers.

[27] One may argue that if `CROO` is instantiated with mobile phones as user devices, `CROO` is also susceptible to user location tracking by mobile phone call carriers.

[28] http://www.identrica.com/

[29] See Third Networks, `http://www.thirdnetworks.co.jp/`.

Web Authentication Assisted by Mobile Phone Call Carrier. Mizuno et al. [54] propose a scheme whereby access to restricted web-based services or resources is granted upon a user-server mutual authentication procedure involving 2D-barcodes (displayed by users' PCs) and credential information (sent through trusted mobile phone call carriers). User authentication is performed as follows. A web-based auth-server sends a session identifier (SID) to a user's PC. The PC displays this identifier using a 2D-barcode. This barcode is parsed using a camera-enabled mobile phone. The mobile phone sends this SID and a unique mobile phone identifier to a mobile phone carrier (MPC) trusted by the web-based auth-server. The MPC forwards this information to the web-based auth-server who authenticates the user if the SID was recently generated and is received from the trusted MPC. Server authentication is done as follows. The user's mobile phone sends a nonce to the aforementioned web-based auth-server, via the MPC. The auth-server sends back, to the user's PC, this nonce and an identifier of the auth-server. The PC displays both pieces of data, in the form of a 2D-barcode. The barcode is captured by the user's camera-enabled mobile phone, and, for the server to be authenticated, the barcode captured by the mobile phone must encode the nonce that was recently sent by this device. A benefit of Mizuno et al.' scheme is that users do not have to manually input information into PCs. On the other hand, this scheme is not designed to counter mobile phone capture attacks, or detect previously undetected instances of user impersonation.

Mobile Payment Schemes. MobileLime [55] is a payment system in which nomadic users are enrolled via a web interface. To be enrolled, each user provides her name and mobile phone number, and associates this information with a prepaid or credit card account. To pay for an item or service at an authorized point of sale (POS), a MobileLime user dials a toll free number using her cell phone, and enters both a PIN (provided at enrollment) and a number identifying the POS. When a payment is completed, a text receipt is sent to the associated user's mobile phone; this can later be used as proof of payment. Other mobile payment systems include Black Lab Mobile's system [9], Sonera's Shopper [22], and Parkit [33]. Some payment systems require physical proximity of POS terminals and buyers' mobile devices. These include Ravi et al.'s system [59], and sytems marketed by PayCircle [58], Encorus PaymentWorks Mobile [23], and Ilium Software [66]. None of these mobile payment systems are designed to identify previously undetected fraudulent financial payments (e.g. when users' mobile phones are stolen and misused for unauthorized CC transactions).

Web Authentication from Untrusted Terminals. Oprea et al. [57] propose a scheme whereby a remote application is accessed through a mobile trusted device. This mobile device (e.g. PDA) delegates to an untrusted terminal temporary access to the remote application. More precisely, the mobile device establishes an SSL session with the remote application and maintains a secure (confidential and authenticated) communication link with the untrusted terminal. The remote application also establishes an associated SSL session with the untrusted terminal. The user issues requests to the remote application via the trusted mobile device, and the remote application sends encrypted answers to these requests to the untrusted terminal. The mobile trusted device provides temporary decryption keys to the untrusted terminal, thereby enabling this terminal to access portions of the encrypted information sent by the remote application for the user to view on the untrusted terminal. This scheme grants to the untrusted terminal temporary access to confidential information. The scheme is not meant to detect misuse of stolen trusted mobile devices or cryptographic keys stored therein. Clarke et al. [15] propose a scheme enabling bidirectional authenticated communication between a user and an associated trusted proxy via an untrusted terminal. The trusted proxy may be granted the authority to act on behalf of the user (e.g. to utilize the user's passwords to establish SSL connections with trusted web sites). This scheme relies on camera-enabled trusted mobile devices carried by users, and works as follows. The trusted mobile device video-captures the information displayed by the untrusted terminal, and compares it with confidential information received from the trusted proxy via an untrusted communication channel. (This communication channel is used by both the untrusted terminal and the mobile device in order to communicate with the proxy.) When the information displayed by the untrusted terminal does not match the value expected by the mobile device, the user is notified. Clarke et al. [15] specify that access to the trusted mobile devices should be protected by PINs or biometrics, thereby partially addressing the possibility of mobile device theft. The scheme is not designed to identify cases in which users' mobile devices have been used to impersonate users without proper authorization. Wu et al. [75] described a scheme enabling a user U to access a remote web-server R via an untrusted kiosk K. U uses a trusted proxy P to mediate all communication with R and utilizes a mobile phone M to communicate with P via SMS. P keeps U's passwords and can securely communicate with R. The scheme works as follows: (a) U directs K to P, specifying both her username and R's identifier (e.g. web address); (b) P sends a session name to K (via the web) and sends the same session name to M via SMS; (c) If U finds that the two session names match, she allows[30] a secure session to be established between P and R. The scheme is not designed to counter mobile phone theft or cloning. Balfanz and Felten [4] propose a method to prevent misuse or disclosure

[30] This is done by pressing a button on M and thereby sending a web message to P.

of signing and decryption cryptographic keys stored on smartcards. The idea is to use a trusted PDA (instead of a smartcard) to request from its user the authorization to perform cryptographic operations with secret keys stored on the PDA. The scheme is not designed to detect misuse of secret keys stored on stolen or cloned PDAs.

Automated Trade of Credit Card Information. McCarty [52] shows how IRC channels can be used to mount automated exchange and validation of credit card information (e.g. for IDF purposes).

Statistical IDF Detection. Bolton and Hand [11] review statistical fraud detection methods.

6 Concluding Remarks

We have considered the general problem of IDF. We propose a model to characterize and compare instances of IDF, and a framework to evaluate IDF solutions by examining (from a system design perspective) the usability, privacy-preserving capability, fraud detection capability, and communication security of these solutions. We argue that complete IDF solutions should provide mechanisms that detect the use of compromised private credential information. Our proposed scheme implements this idea without requiring the collection of private behavioral information (in contrast to statistical anomaly-based fraud detection schemes used, e.g., by banks to detect credit card fraud). Our scheme associates each use of credential information with a one-time password verified by an online trusted party F. F need not be the same for all users (thus improving scalability). An important feature of the proposed scheme is its generic nature, i.e. it can simultaneously be used with multiple classes of applications and credential tokens, in both online and on-site transactions. The proposed scheme also allows each IDF victim to continue to use her credential tokens (e.g. credit cards) provided she uses her portable trusted device to send new one-time password setup information to F. This feature can be useful when it is preferable (e.g. for time efficiency, convenience, or lack of alternative options) to continue to use credential tokens, even though they have been cloned, rather than obtaining new ones. This is appealing in cases in which it takes less time to go in person to a single local party F (e.g. a trusted government agency's office) to give new OTP setup information, than having social security numbers replaced, or obtaining new credit cards by postal mail. While we have analyzed our proposal with respect to a variety of detailed practical and formal criteria (see the appendix for our AVISPA analysis), we have not implemented it. We encourage work on mathematical models that help evaluate IDF detection schemes, but note the challenge of generating *realistic* models (see [27, 38, 39] for discussions on the empirical adequacy of standard mathematical security proof techniques). We also encourage further exploration of the IDF problem, and the design of cryptographic and non-cryptographic schemes that detect fraudulent uses of compromised authentication keys.

References

[1] M. Abadi, T.M.A. Lomas, and R. Needham. Strengthening Passwords. Technical Report 1997 - 033, Digital Equipment Corporation, 1997.

[2] Infrared Data Association. `https://www.irda.org/`. Site accessed in Jan. 2006.

[3] B. Balacheff, L. Chen, S. Pearson, D. Plaquin, and G. Proudler. *Trusted Computing Platforms TCPA Technology in Context*. Prentice Hall, 2003.

[4] D. Balfanz and E. Felten. Hand-Held Computers Can Be Better Smart Cards. In *USENIX Security Symposium*, Washington, DC, August 1999.

[5] L. Bauer, S. Garriss, J. M. McCune, M. K. Reiter, J. Rouse, and P. Rutenbar. Device-Enabled Authorization in the Grey System. In *International Conference on Information Security (ISC '05)*, volume 3650 of *LNCS*, pages 431–445. Springer-Verlag, 2005.

[6] A. Beaufour and P. Bonnet. Personal Servers as Digital Keys. In *IEEE International Conference on Pervasive Computing and Communications*, 2004.

[7] M. Bellare and S.K. Miner. A Forward-Secure Digital Signature Scheme. In *CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 431–448. Springer-Verlag, 1999.

[8] Better Busineess Bureau (USA). Victims' Stories. `http://www.bbbonline.org/idtheft/stories.asp`. Site accessed in Jan. 2006.

[9] Black Lab Mobile. `http://www.blacklabmobile.com/index.php`. Site accessed in Jan. 2006.

[10] Bluetooth. `https://www.bluetooth.org/`. Site accessed in Jan. 2006.

[11] R. Bolton and D. Hand. Statistical fraud detection: A review. *Statistical Science*, 17(3):235–255, 2002.

[12] S. Brands. *Rethinking Public Key Infrastructures and Digital Certificates; Building in Privacy.* MIT Press, 2000.

[13] R. Canetti and S. Goldwasser. An Efficient Threshold Public-Key Cryptosystem Secure Against Adaptive Chosen Ciphertext Attack. In *EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 90–106. Springer-Verlag, 1999.

[14] N. Chou, R. Ledesma, Y. Teraguchi, D. Boneh, and J.C. Mitchell. Client-Side Defense Against Web-Based Identity Theft. In *Network and Distributed System Security Symposium (NDSS '04)*. The Internet Society, 2004.

[15] D.E. Clarke, B. Gassend, T. Kotwal, M. Burnside, M. van Dijk, S. Devadas, and R.L. Rivest. The Untrusted Computer Problem and Camera-Based Authentication. In *International Conference on Pervasive Computing*, volume 2414 of *LNCS*, pages 114–124. Springer-Verlag, 2002.

[16] M. D. Corner and B. D. Noble. Zero-Interaction Authentication. In *International Conference on Mobile Computing and Networking (MOBICOM '02)*, pages 1–11. ACM Press, 2002.

[17] L. Cranor and S. Garfinkel. *Security and Usability.* O'Reilly Media, Inc., August 2005.

[18] DataTex Engineering Corporation. Computer Crime, Cyber Crime, and Identity Theft Are you a Victim? `http://www.datatexcorp.com/html/cybercrimenews.htm`. Site accessed in Jan. 2006.

[19] R. Dhamija and J. D. Tygar. The Battle Against Phishing: Dynamic Security Skins. In *Symposium on Usable Privacy and Security (SOUPS '05)*, pages 77–88. ACM Press, 2005.

[20] Y. Dodis, M. Franklin, J. Katz, A. Miyaji, and M. Yung. Key-Insulated Public Key Cryptosystems. In *CT-RSA '03*, volume 2612 of *Lecture Notes in Computer Science*, pages 19–32. Springer-Verlag, 2003.

[21] Y. Dodis, J. Katz, S. Xu, and M. Yung. Key-Insulated Public Key Cryptosystems. In *EUROCRYPT '02*, volume 2332 of *Lecture Notes in Computer Science*, pages 65–82. Springer-Verlag, 2002.

[22] eFinland. Mobile Services. `http://e.finland.fi/netcomm/news/showarticle.asp?intNWSAID=7240`. Site accessed in Jan. 2006.

[23] Encorus. PaymentWorks Mobile. `http://www.wi-fitechnology.com/printarticle535.html`. Site accessed in Jan. 2006.

[24] Australian Center for Policing Research. Standardization of Definitions of Identity Crime Terms - Discussion Paper, Prepared by the Australian Center for Policing Research for the Police Commissioners' Australian Identity Crime Working Party and the AUSTRAC POI Steering Committee, 2005.

[25] NFC Forum. `http://www.nfc-forum.org/home`. Site accessed in Jan. 2006.

[26] T. Garfinkel, B. Pfaff, J. Chow, M. Rosenblum, and D. Boneh. Terra: A virtual machine-based platform for trusted computing. In *ACM Symposium on Operating Systems Principles (SOSP'03)*, 2003.

[27] O. Goldreich. On Post-Modern Cryptography. Version of December 5, 2006, IACR ePrint archive, 2006. `http://eprint.iacr.org/2006/461.pdf`. Site accessed in December 2006.

[28] M.T. Goodrich, M. Sirivianos, J. Solis, G. Tsudik, and E. Uzun. Loud and Clear: Human-Verifiable Authentication Based on Audio. In *IEEE International Conference on Distributed Computing Systems (ICDCS' 06)*. IEEE, 2006.

[29] G.R. Gordon and N.A. Willox. Identity Fraud: A Critical National and Global Threat. *Journal of Economic Crime Management*, 2(1):1–47, 2005.

[30] J.A. Halderman, B. Waters, and E.W. Felten. A Convenient Method for Securely Managing Passwords. In *International Conference on World Wide Web (WWW '05)*, pages 471–479. ACM Press, 2005.

[31] J.A. Haskett. Pass-algorithms: A User Validation Scheme Based on Knowledge of Secret Algorithm. *Communications of the ACM*, 27(8):777–781, 1984.

[32] A. Herzberg, M. Jakobsson, S. Jarecki, H. Krawczyk, and M. Yung. Proactive Public Key and Signature Systems. In *ACM Conference on Computer and Communications Security (CCS '97)*, pages 100–110. ACM Press, 1997.

[33] PARK IT. `http://www.payway.fi/eng_index.html`. Site accessed in Jan. 2006.

[34] RFID Journal. `http://www.rfidjournal.com/`. Site accessed in Jan. 2006.

[35] M. Just and P.C. van Oorschot. Addressing the Problem of Undetected Signature Key Compromise. In *Network and Distributed System Security (NDSS '99)*. The Internet Society, 1999.

[36] G. Kellette. Testimony of the Maryland Public Interest Research Group in Favor of HB 281 (Kagan) To Help Stop Identity Theft By Preventing Coercion of Social Security Numbers By Private Businesses. Maryland Public Interest Research Group, `http://www.pirg.org/consumer/mdssnfeb02.htm`. Site accessed in Jan. 2006.

[37] E. Kirda and C. Kruegel. Protecting Users Against Phishing Attacks with AntiPhish. In *Computer Software and Applications Conference '05*, pages 517–524, 2005.

[38] N. Koblitz and A.J. Menezes. Another look at provable security. Version of September 10, 2006, IACR ePrint archive, 2006. `http://eprint.iacr.org/2006/230.pdf`. Site accessed in December 2006.

[39] N. Koblitz and A.J. Menezes. Another look at provable security II. Version of July 3, 2006, IACR ePrint archive, 2006. `http://eprint.iacr.org/2006/229.pdf`. Site accessed in December 2006.

[40] T. Kwon and H. Moon. Multi-modal Techniques for Identity Theft Prevention. In *International Conference on Human.Society@Internet*, volume 3597 of *Lecture Notes in Computer Science*, pages 291–300. Springer-Verlag, 2005.

[41] Kaspersky Lab. Trojan targets mobile phones running Java applications., 2006. `http://www.kaspersky.com/news?id=180984542`. Site accessed in February 2006.

[42] D. Lacey and S. Cuganesan. The Role of Organizations in Identity Theft Response: the Organization-Individual Dynamic. *Journal of Consumer Affairs*, 38(2):244–261, 2004.

[43] L. Lamport. Password Authentication with Insecure Communication. *Communications of the ACM*, 24:770–772, 1981.

[44] S. Li and H.-Y. Shum. Secure Human-Computer Identification (Interface) Systems against Peeping Attacks. Cryptology ePrint Archive, Report 2005/268, 2005.

[45] T.M.A. Lomas, L. Gong, J.H. Saltzer, and R.M. Needham. Reducing Risks from Poorly Chosen Keys. *ACM SIGOPS Operating Systems Review*, 23(5), 1989.

[46] H.K. Lu and A. Ali. Prevent Online Identity Theft Using Network Smart for Secure Online Transactions. In *International Conference on Information Security (ISC '04)*, volume 3225 of *Lecture Notes in Computer Science*, pages 342–353. Springer-Verlag, 2004.

[47] M. Jakobsson. Modeling and Preventing Phishing Attacks, 2005. Phishing Panel in Financial Cryptography (FC '05).

[48] P. MacKenzie and M.K. Reiter. Networked Cryptographic Devices Resilient to Capture. In *IEEE Symposium on Security and Privacy*, pages 12–25. IEEE Computer Society, 2001.

[49] P. MacKenzie and M.K. Reiter. Delegation of cryptographic servers for capture-resilient devices. *Distributed Computing*, 16(4):307–327, 2003.

[50] Barcode Man. 2D Barcodes Explained. `http://www.barcodeman.com/faq/2d.php`. Site accessed in Jan. 2006.

[51] U. Manber. A Simple Scheme to Make Passwords Based on One-Way Functions Much Harder to Crack. *Computers and Security*, (2):171–176, 1996.

[52] B. McCarty. Automated Identity Theft. *IEEE Security & Privacy Magazine*, 1(5):89–92, 2003.

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

[54] S. Mizuno, K. Yamada, and K. Takahashi. Authentication using multiple communication channels. In *Workshop on Digital Identity Management (DIM '05)*, pages 54–62. ACM Press, 2005.

[55] MobileLime. `https://www.mobilelime.com/mobilelime/home.do?action=index`. Site accessed in Jan. 2006.

[56] Myers. What is a FICO score? `http://www.mtg-net.com/sfaq/faq/fico.htm`. Site accessed in Jan. 2006.

[57] A. Oprea, D. Balfanz, G. Durfee, and D. Smetters. Securing a Remote Terminal Application with a Mobile Trusted Device. In *Annual Computer Security Applications Conference (ACSAC '04)*, Phoenix, AZ, December 2004.

[58] PayCircle. `http://www.paycircle.org/`. Site accessed in Jan. 2006.

[59] N. Ravi, P. Stern, N. Desai, and L. Iftode. Accessing Ubiquitous Services Using Smart Phones. In *International Conference on Pervasive Computing and Communications (PerCom 2005)*, 2005.

[60] Javelin Strategy & Research. 2005 Identity Fraud Survey Report, 2005. `http://www.javelinstrategy.com/reports/2005IdentityFraudSurveyReport.html`.

[61] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J.C. Mitchell. Stronger Password Authentication Using Browser Extensions. In *USENIX Security Symposium*, pages 17–32, 2005.

[62] V. Roth, K. Richter, and R. Freidinger. A PIN-Entry Method Resilient Against Shoulder Surfing. In *ACM Conference on Computer and Communications Security (CCS '04)*, pages 236–245. ACM Press, 2004.

[63] A.D. Rubin and R.N. Wright. Off-line generation of limited-use credit card numbers. In *Financial Cryptography 2001 (FC '01)*, volume 2339 of *Lecture Notes in Computer Science*, pages 196–209. Springer Verlag, 2002.

[64] A. Shamir. Secureclick: A web payment system with disposable credit card numbers. In *Financial Cryptography 2001 (FC '01)*, volume 2339 of *Lecture Notes in Computer Science*, pages 232–242. Springer Verlag, 2002.

[65] A. Singh and A.L.M. dos Santos. Grammar based off line generation of disposable credit card numbers. In *ACM Symposium on Applied Computing 2002 (SAC '02)*, pages 221–228. ACM Press, 2002.

[66] Ilium Software. eWallet. `http://www.iliumsoft.com/site/ew/ewallet.htm`. Site accessed in Jan. 2006.

[67] J. Tang, V. Terziyan, and J. Veijalainen. Distributed PIN Verification Scheme for Improving Security of Mobile Devices. *Journal of Mobile and Network Application*, 8(2):159–175, 2003.

[68] The AVISPA Team. Automated Validation of Internet Security Protocols and Applications. `http://www.avispa-project.org/`. Site accessed in August 2006.

[69] The AVISPA Team. AVISPA v1.1 User Manual. `http://www.avispa-project.org/package/user-manual.pdf`. Site accessed in August 2006.

[70] H. Tuch, G. Klein, and G. Heiser. OS verification - now! In *Workshop on Hot Topics in Operating Systems (HotOS X)*, 2005.

[71] T. Valentine and M. Endo. Towards an Exemplar Model of Face Processing: the Effects of Race and Distinctiveness. *Quaterly Journal of Experimental Psychology*, 44:671–703, 1992.

[72] P.C. van Oorschot and S. Stubblebine. Countering Identity Theft through Digital Uniqueness, Location Cross-Checking, and Funneling. In *Financial Cryptography and Data Security 2005 (FC '05)*, volume 3570 of *Lecture Notes in Computer Science*, pages 31–43. Springer-Verlag, 2005.

[73] Mastercard & Visa. SET Secure Electronic Transaction Specification: Formal Protocol Definition, 1997. `http://www.hta-bi.bfh.ch/Projects/eftpos/extern/set_bk3.pdf`. Site accessed in December 2006.

[74] W. Wang, Y. Yuan, and N. Archer. A Contextual Framework for Combating Identity Theft. *IEEE Security & Privacy Magazine*, 4(2):30–38, 2006.

[75] M. Wu, S. Garfinkel, and R. Miller. Secure Web Authentication with Mobile Phones. In *DIMACS Workshop on Usable Privacy and Security Systems*, 2004.

7 Appendix: AVISPA-based Security Analysis

	U/D_U	R	I	F	**Messages Sent**
1.	$\mid \rightarrow \mid$				$(v, w, h(z), ID_I)$, where $v = [ID_U, k^{(i)}, h(z), q_1]^F$ and $w = [m_{C_U}, h(z), q_2]^I$
2.		$\mid \rightarrow \mid$			$[v, w, z, q_3]^I$
3.			$\mid \rightarrow \mid$		$[v, h(z), q_4]^F_I$
4.			$\mid \leftarrow \mid$		$[b]^I_F$, where $b = (h(z), S_z, q_5)$
5.		$\mid \leftarrow \mid$			$[A_z, b, [h(z), S_z, q_5]_F, q_6]^R_I$
6.	$\mid \leftarrow \mid$				$(b, G_z, h(v), [b, G_z, h(v)]_R, [h(z), S_z, q_5]_F)$

Table 3: Protocol Used for AVISPA Analysis. (For notation, see §3.)

This section uses the notation of §3.

AVISPA [68] is a set of tools for automated validation of Internet security-sensitive protocols and applications, including: (1) a constraint logic-based model checker (called CL-AtSe); (2) a model checker (called OFMC) for efficient, general protocol falsification (i.e. attack discovery) and correctness; (3) a SAT-based model checker (called SATMC); and (4) a model checker (called TA4SP) that computes either an over-approximation or an under-approximation of the intruder knowledge. AVISPA can be used to gain increased confidence in the security of protocols.

7.1 Limitations of our AVISPA-based Analysis

The current version (i.e. version 1.1) of AVISPA does not provide (ready-to-use) constructs for addition, subtraction, multiplication, set ordering, and recursive statements (see §2.1.2 of [69]). Consequently, we focus, in our AVISPA-based analysis of S, on the use of a single valid one-time password (instead of a recursively defined sequence thereof).[31] Moreover, our AVISPA-based analysis uses an idealized version of S in which U and D are merged (see Fig. 3).[32] This stems from the fact that the current version of AVISPA does not provide (ready-to-use) constructs to model cyphertext decryption with invalid keys (e.g. those used by D and derived from invalid user passwords).[33] Nevertheless, our AVISPA-based analysis provides increased confidence in the correctness of S, and its resistance to falsification of its cryptographic goals.

7.2 Goals of our AVISPA-based Analysis

The goals of our AVISPA-based security analysis are to test whether:

G1. the secrecy of q_1 and $k^{(i)}$ is maintained between D and F;
G2. the secrecy of q_2 and m_{C_U} is maintained between D and I;
G3. the secrecy of q_3 is maintained between R and I;
G4. the secrecy of q_4 is maintained between I and F;
G5. F can reliably assume that $k^{(i)}$ is issued by D for the purpose of T;
G6. I can reliably assume that m_{C_U} is sent by D for the purpose of T;
G7. I, R, and D can reliably assume that q_5 is issued by F for the purpose of T;
G8. R can reliably assume that A_z is issued by I for the purpose of T;
G9. D can reliably assume that G_z is issued by I for the purpose of T.

[31] §4.2 examines the case of successive uses of OTPs.
[32] Idealized versions, as required by AVISPA, may not always reflect the reality of deployed systems.
[33] See §4.2 for an analysis of this scenario.

7.3 Results

This section presents the output obtained from running the web-based interface of AVISPA 1.1 with the code presented in §7.4. The output indicates that, according to OFMC, CL-AtSe and SATMC, security goals G1 through G9 are met by S.

```
AVISPA Tool Summary

OFMC    : SAFE
CL-AtSe : SAFE
SATMC   : SAFE
TA4SP   : INCONCLUSIVE

--------------------------------------------

% OFMC
% Version of 2006/02/13
SUMMARY
  SAFE
DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
PROTOCOL
  /home/avispa/web-interface-computation/./tempdir/workfilex3mnlY.if
GOAL
  as_specified
BACKEND
  OFMC
COMMENTS
STATISTICS
  parseTime: 0.00s
  searchTime: 2.91s
  visitedNodes: 324 nodes
  depth: 12 plies

--------------------------------------------

SUMMARY
  SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS
  TYPED_MODEL

PROTOCOL
  /home/avispa/web-interface-computation/./tempdir/workfilex3mnlY.if

GOAL
  As Specified

BACKEND
  CL-AtSe

STATISTICS

  Analysed   : 20300 states
  Reachable  : 11968 states
  Translation: 0.06 seconds
  Computation: 4.32 seconds
```

```
------------------------------------------

SUMMARY
  SAFE

DETAILS
  STRONGLY_TYPED_MODEL
  BOUNDED_NUMBER_OF_SESSIONS
  BOUNDED_SEARCH_DEPTH
  BOUNDED_MESSAGE_DEPTH

PROTOCOL
  workfilex3mnlY.if

GOAL
  %% see the HLPSL specification..

BACKEND
  SATMC

COMMENTS

STATISTICS
  attackFound               false     boolean
  upperBoundReached         true      boolean
  graphLeveledOff           7         steps
  satSolver                 zchaff    solver
  maxStepsNumber            11        steps
  stepsNumber               8         steps
  atomsNumber               3677      atoms
  clausesNumber             25889     clauses
  encodingTime              5.12      seconds
  solvingTime               0.01      seconds
  if2sateCompilationTime    1.73      seconds

ATTACK TRACE
  %% no attacks have been found..

------------------------------------------

SUMMARY
   INCONCLUSIVE

DETAILS
   TIME_OUT

PROTOCOL
   /home/avispa/web-interface-computation/./tempdir/workfilex3mnlY.if

GOAL
   SECRECY

BACKEND
   TA4SP

COMMENTS
   Computation broken
```

7.4 Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role mobiledevice (
    Device,RelyingParty,Issuer,FraudMonitor     : agent,
    Hash                                        : hash_func,
    PubR,PubI,PubF                              : public_key,
    TransD,PwdDF,PwdDI  : text,
    SND,RCV                                     : channel (dy) )

played_by Device

def=

local   State           : nat,
        Ki,Tref,Vref    : message,
        Q1,Q2,Q5        : text,
        A,G       : text,
        X               : message.text.text,
        V               : {agent.message.message.text}_public_key,
        W               : {agent.text.message.text}_public_key

init    State := 1

transition

1.  State   =1      /\  RCV(start)
    =|>
    State'  :=2     /\  Ki':= Hash(PwdDF)
                    /\  Tref':= Hash(TransD)
                    /\  Q1':= new()
                    /\  Q2':= new()
                    /\  V':= {Device.Ki'.Tref'.Q1'}_PubF
                    /\  W':= {Device.PwdDI.Tref'.Q2'}_PubI
                    /\  SND(V'.W'.Tref'.Issuer)

                    /\  secret(Ki',ki,{Device,FraudMonitor})
                    /\  secret(Q1',q1,{Device,FraudMonitor})
                    /\  secret(PwdDI,pwddi,{Device,Issuer})
                    /\  secret(Q2',q2,{Device,Issuer})

                    /\  witness(Device,FraudMonitor,ki,Ki')
                    /\  witness(Device,Issuer,pwddi,PwdDI)


2.  State   = 2     /\  RCV(X'.G'.Vref'.{X'.G'.Vref'}_inv(PubR).{X'}_inv(PubF))
                    /\  X' = Tref.success.Q5'
  /\ Vref'= Hash(V)
    =|>
    State'  :=3     /\  request(Device,FraudMonitor,q5,Q5')
                    /\  request(Device,RelyingParty,g,G')

end role
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role crp (
    Device,RelyingParty,Issuer,FraudMonitor : agent,
    Hash                                    : hash_func,
    PubR,PubI,PubF                          : public_key,
    TransD   : text,
    SND_DR,RCV_DR                           : channel (dy),
    SND_RI,RCV_RI                           : channel (dy) )

played_by RelyingParty

def=

local   State               : nat,
        Tref,Vref           : message,
        Q3,Q5,A,G,Q6      : text,
        V                   : {agent.message.message.text}_public_key,
        W                   : {agent.text.message.text}_public_key,
        X                   : message.text.text

init    State := 1

transition


%1.  State   = 0     /\  RCV_DR(start)
%    =|>
%    State'  :=1     /\  SND_DR(TransD)

1.  State   = 1     /\  RCV_DR(V'.W'.Tref'.Issuer)
                    /\  Tref' = Hash(TransD)
    =|>
    State'  :=2     /\  Q3':= new()
                    /\  SND_RI({V'.W'.TransD.Q3'}_PubI)
                    /\  secret(Q3',q3,{RelyingParty,Issuer})

2.  State    =2     /\
                        RCV_RI({A'.X'.{X'}_inv(PubF).Q6'.{A'.X'.Q6'.{X'}_inv(PubF)}_inv(PubI)}_PubR)
                    /\  X' = Tref.success.Q5'
    =|>
    State'  :=3     /\  G' := new()
  /\ Vref':= Hash(V)
                    /\  SND_DR(X'.G'.Vref'.{X'.G'.Vref'}_inv(PubR).{X'}_inv(PubF))

                    /\  request(RelyingParty,FraudMonitor,q5,Q5')
                    /\  request(RelyingParty,Issuer,a,A')

                    /\  witness(RelyingParty,Device,g,G')

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


role credissuer (
```

```
    Device,RelyingParty,Issuer,FraudMonitor : agent,
    Hash                                    : hash_func,
    PubR,PubI,PubF                          : public_key,
    PwdDI                                   : text,
    SND_RI,RCV_RI                           : channel (dy),
    SND_IF,RCV_IF                           : channel (dy) )

played_by Issuer

def=

local   State                   : nat,
        Tref                    : message,
        Q2,Q3,Q4,Q5,Q6,TransD,A  : text,
        V                       : {agent.message.message.text}_public_key,
        W                       : {agent.text.message.text}_public_key,
        X                       : message.text.text

init    State := 1

transition

1.  State    =1 /\  RCV_RI({V'.W'.TransD'.Q3'}_PubI)
                /\  W' = {Device.PwdDI.Tref'.Q2'}_PubI
                /\  Tref' = Hash(TransD')
    =|>
    State'  :=2 /\  Q4':= new()
                /\  SND_IF({V'.Tref'.Q4'.{V'.Tref'.Q4'}_inv(PubI)}_PubF)

                /\  secret(Q4',q4,{Issuer,FraudMonitor})
                /\  request(Issuer,Device,pwddi,PwdDI)


2.  State    =2 /\  RCV_IF({X'.{X'}_inv(PubF)}_PubI)
                /\  X'= Tref.success.Q5'
    =|>
    State'  :=3 /\  A':= new()
                /\  Q6':=new()
                /\  SND_RI({A'.X'.{X'}_inv(PubF).Q6'.{A'.X'.Q6'.{X'}_inv(PubF)}_inv(PubI)}_PubR)

                /\  request(Issuer,FraudMonitor,q5,Q5')

                /\  witness(Issuer,RelyingParty,a,A')

end role


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


role fraudmonitor (
    Device,RelyingParty,Issuer,FraudMonitor : agent,
    Hash                                    : hash_func,
    PubR,PubI,PubF                          : public_key,
    OTP                                     : message,
    SND,RCV                                 : channel (dy) )
```

```
played_by FraudMonitor

def=

local   State       : nat,
        Ki,Tref     : message,
        Q1,Q4,Q5    : text,
        V           : {agent.message.message.text}_public_key,
        X           : message.text.text

init    State   :=1

transition

1.  State    =1 /\ RCV({V'.Tref'.Q4'.{V'.Tref'.Q4'}_inv(PubI)}_PubF)
                /\  V' = {Device.Ki'.Tref'.Q1'}_PubF
                /\  OTP = Hash(Ki')
    =|>
    State' :=2  /\  OTP':=Ki'
                /\  Q5':= new()
                /\  X':= Tref'.success.Q5'
                /\  SND({X'.{X'}_inv(PubF)}_PubI)

                /\  request(FraudMonitor,Device,ki,Ki')

                /\  witness(FraudMonitor,Issuer,q5,Q5')
                /\  witness(FraudMonitor,RelyingParty,q5,Q5')
                /\  witness(FraudMonitor,Device,q5,Q5')

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role session (
    Device,RelyingParty,Issuer,FraudMonitor         : agent,
    Hash              : hash_func,
    PubR,PubI,PubF    : public_key,
    TransD,PwdDF,PwdDI : text)

def=

local   OTP             : message,
  SND_DR,RCV_DR   : channel (dy),
        SND_RI,RCV_RI   : channel (dy),
        SND_IF,RCV_IF   : channel (dy)


init    OTP := Hash(Hash(PwdDF))

composition
    mobiledevice(Device,RelyingParty,Issuer,FraudMonitor,Hash,PubR,PubI,PubF,TransD,PwdDF,
PwdDI,SND_DR,RCV_DR)
 /\ crp(Device,RelyingParty,Issuer,FraudMonitor,Hash,PubR,PubI,PubF,TransD,SND_DR,RCV_DR,
SND_RI,RCV_RI)
 /\ credissuer(Device,RelyingParty,Issuer,FraudMonitor,Hash,PubR,PubI,PubF,PwdDI,
SND_RI,RCV_RI,SND_IF,RCV_IF)
 /\ fraudmonitor(Device,RelyingParty,Issuer,FraudMonitor,Hash,PubR,PubI,PubF,OTP,
```

```
SND_IF,RCV_IF)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

role environment()

def=

const   device,relyingparty,issuer,fraudmonitor: agent,
        myhash                      : hash_func,
        pubr,pubi,pubf              : public_key,
        transd1,transd2      : text,
        pwd_df,pwd_df2 : text,
        pwd_di,pwd_di2 : text,
        success                     : text,
        q1,q2,q3,q4,q5,ki,pwddi,a,g : protocol_id

intruder_knowledge =
    {device,relyingparty,issuer,fraudmonitor,myhash,pubr,pubi,pubf,pwd_df2,pwd_di2,transd2}


composition
     session(device,relyingparty,issuer,fraudmonitor,myhash,pubr,pubi,pubf,transd1,pwd_df,pwd_di)
 /\ session(i,relyingparty,issuer,fraudmonitor,myhash,pubr,pubi,pubf,transd2,pwd_df2,pwd_di2)

end role

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

goal    secrecy_of q1
        secrecy_of q2
        secrecy_of q3
        secrecy_of q4
  secrecy_of ki
        secrecy_of pwddi

        authentication_on ki
        authentication_on pwddi
        authentication_on q5
        authentication_on a
        authentication_on g
end goal

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

environment()
```