Marlboro College Plan of Concentration

Authentication: Techniques and Theory

Samuel Judson

—

COMPUTER SCIENCE & MATHEMATICS\Cryptography

May of 2016

Plan Sponsor	-	Jim Mahoney	-	Professor of Computer Science at Marlboro College
Plan Sponsor	-	Matt Ollis	-	Professor of Mathematics at Marlboro College
Outside Evaluator	-	Michael Fischer	-	Professor of Computer Science at Yale University

Contents

Introduction

Plan Description

A study of modern cryptographic algorithms and protocols focused on their supporting algebraic and computational theory and the limits of mathematical reasoning on cryptographic problems, especially entity authentication.

Component	%
Creation of a flexible entity authentication protocol for medium-scale systems	30
A framework for consideration of entity authentication schemes and a survey of various approaches in use	25
An introduction to elliptic curves and their use in cryptography for encryption and authentication	25
Examinations in supporting mathematics and computer science material	20

Mathematics can be quite insidious. By my own natural proclivity for the study of history and the broader humanities, abetted by the exhortation to "study broadly" at Marlboro, the word truth does not carry the implication of objectivity. Instead, it is a matter for historiographic debate - to be settled and not discovered, inabsolute in its perfection. Hence the characterization of the clarity of mathematics as insidious. The correctness of mathematical thought is thoroughly tempting while it provides the comfort of rigor and a refined structure often labeled clean, elegant, and beautiful. If derived by valid logic from the axioms taken, a mathematical statement simply is true. This absolutism is a welcome respite from the theater of incertitude and the bias and contention it harbors.

But of course, this truth - or more correctly, these truths - have limitation. They hold exactly, but only within those consistent universes which agree with the axioms from which they were derived. In consideration of applicability their aura is dimmed, and it is further diminished by the possible existence of systems immune to formalization but with great questions to be answered. For these, even if we accept the correctness of mathematical arguments they can only be a tool - a means and not an end. Although whether systems of human interaction are of this informal type is a question of considerable debate and investigation, it is certainly the case that they are for now effectively so. We have no foundation and cannot resolve social exchange to a set of formulae.

How then to secure communication and informational systems - which facilitate interaction amongst human actors or their deterministic proxies - is just such a great question, and one for which immense strides have been made by the recent ready use of the mathematical tooling of cryptography. Like any prodigious technique its application has become instinctive, and it has become so not without reason nor sign of diminishing effectiveness. Yet its limitations still remain. Certain problems within the space of cryptographic applicability may be approached - at least in theory - fully within the bounds of mathematical reasoning. The implementation of that theory is not protected, but at least a clean and demarcated boundary as to the effective realm of an argument can be stated and observed by theorist and practitioner. However, many of the questions to which cryptographers attempt to provide an answer - or at least an understanding - do not contribute even this paltry assistance. Instead, they require direct consideration of the human relationships and interactions which frame the system.

It is not that mathematical approaches have no utility for these problems. If sufficient assumptions are made as to the behavior of the actors within the system they may even achieve the same status as arguments for those problems with mathematical isolation. Further, the importance of those assumptions are rarely understated. Whether they are justified or not in practice is recognized and often formal mechanisms are introduced to lessen the risk from their possible exposure.

However, this mentality is constricting as much as it is liberating. Its effectiveness provides avenues of exploration and argument, but they do not spread in every direction. Instead, it promotes those techniques and that understanding accessible through the generalized and formalized lens of the mathematical attitude, and rewards the existence of a formal proof of correctness over explicit utility. This is not to say those proofs are overvalued - they are in fact strictly necessary to establish a guarantee of security demanded. Rather, it is to say that in the pursuit of them their form can often be expanded and imposed to the entire context. A proof must be structured, it must be anchored in established theorems and assumptions stated to their limits of generality so as not to invite misunderstanding, and it must proceed with rigor and clarity. However there is no reason the problem it solves must be the same way.

Although each component of this Plan of Concentration is designed to stand alone, there is still a common thread through them. My goal was to study the limitations of mathematical approaches to one of these problems with a direct human aspect - entity authentication - without losing sight of the techniques and value of the mathematics itself. This general objective was grounded in more immediate ones for each paper, to develop my own cryptographic protocol, to survey current approaches of theory and practice, and to directly relate cryptographic constructions to their mathematical foundations.

The first of these, entitled Ad Hoc Keyless Authentication, is the introduction of an entity authentication protocol which demands unlikely behavior from communities of narrow demographics. In eschewing the general, it gains utility for its specific purpose. The second component, A Framework for Discussion of Entity Authentication Schemes, is an attempt to provide definitions and discussion for analysis of such schemes, as well as an application of that framework to a survey of those in current use. The final paper, Elliptic Curves and their Cryptographic Applications, explains itself through its title. Since elliptic curve cryptography has been of particular use for entity authentication, the common subject is a lessened but still present focus as well. Lastly, examinations in mathematical and computer science supporting material conclude this work.

Samuel Judson

Ad Hoc Keyless Authentication

1 Introduction

In most cryptographic contexts identities - and the entities which underlie them - are either trusted or untrusted. This binary status is conceptually simple and provides the basis for a rigorous, general approach to cryptographic problems. A trusted actor is given full access to the information and privileges to which they are entitled, whereas any capability for an untrusted actor to gain the same is a violation of the security guarantee a solution is attempting to provide. That an actor may be partially trusted, if considered, is often managed by authenticating them as a lesser version of themselves, restricting their access rights and the influence of their inputs. They are treated not as *possibly the identity they claim to be*, but rather as *fully the subset of that identity which they can prove to be.* Approaches which take the former mindset, such as the PGP web of trust - in which an entity may declare an attestor as being partially trusted - are seen as arbitrary and inherently suspect. For instance, with PGP the value of partially trusted attestors is a preference which can be altered at will by the user, and so is not dependent on any formal argument. Naturally this is suspicious, inviting a query as to why go through the trouble of formally signing attestments if their acceptance is going to be based on an amathematical and exploitable method?

However, although the approach of partial authentication may be more comfortable and seemingly effective, its superiority is restricted to theory. Within the confines of a formal argument it allows for mathematical tools and methods to be brought to bear successfully against the task of showing the security of a construction. In practice, however, it runs into difficulty by that what is convenient for the mathematician is not often convenient for the end user. Keeping track - both in formal records and in informal operations - of to what extent an entity has authenticated and managing access to which privileges they may not be denied is often a great difficulty due to the flexibility in architecture required. In contexts in which those privileges have sufficient value to where they must be protected, but not sufficient to where the damage done by their violation is crippling, this difficulty may not be worth its cost. Therefore, in many settings - particularly offline - partial trust is managed by a reasoned judgment as to the likelihood that it could have been gained and the likelihood one would want to exploit it. If that judgment finds that it is unreasonable that the supplicant would be attempting to maliciously authenticate as something they are not, then they are accepted as who they claim to be. The justification is exactly that arbitrary and suspicious one, of viewing a claimant able to substantiate only part of their claim as probably what they say they are, and not just the subset for which they have been able to provide proof.

We will give an example. Take the entrance to the offices of a company. An individual may arrive having forgotten their entrance pass and ask for entry from the guard. In the partial authentication model, they must be denied since they cannot be authenticated. With a less rigid approach however, the guard may be able to build enough of a case for the identity of the claimant as an employee from additional information.

The claimant might be carrying documents or wearing apparel with the insignia of the company, they may be a regular figure passing through the entryway of the firm, they may have made an apparently earnest attempt to enter before realizing they lacked their credentials, they may be vouched for by another individual with valid credentials, and so forth. In such a case, even lacking the formal authentication requirements the guard may consider it probable that they are who they claim to be, and permit them entry despite the lack of credentials. This is not to say that another guard may permit them entry to the confidential human resources records room with the same set of proof for their identity, but especially if additional safeguards internal to the offices limit the exposure of particularly damaging information the guard may consider the risk minimal and the claim of identity likely.

That the guard may do this should not be considered an unreasonable example. That a request to enter a lightly secured facility in this manner would be accepted is perhaps of greater common expectation then denial. However, in an environment in which electronic access management and record keeping is becoming more and more commonplace there are increasing restraints on the capability of entities to manage situations of partial trust in a nuanced and individualized manner. For instance, many offices now use electronic access panels and key fobs for entry, instead of a security officer waving individuals through. Although this lack of flexibility increases security, it does so at the cost of burdening individuals for minor mistakes such as the loss or misplacement of entrance credentials. It is likely for many circumstances this zealousness is unnecessary and the movement to digitally enforced authentication criteria causes more disruption than the security benefit justifies. However, this may not be sufficient to warrant the removal of access control systems, particularly because they provide efficient and automatic record keeping of entries to and exits from the environment. An alternative method, in which a formal digital record can be made but the flexibility of access can be allowed, therefore becomes desirable.

This is particularly the case in situations in which there may be an expectation of honest and careful action on behalf of all entities within the system - contexts in which a cultural trust can be reasonably inferred amongst the members of the community. If substantial additional information must be discerned before a verifier may be willing to grant a claim of identity then this authentication approach is unlikely to have meaningful utility. However, if within a system a high level of trustworthiness may be assumed automatically a lesser amount of such information is necessary. In particular, the attestment by other, properly credentialed entities that a specific entity is eponymous with an identity - even if the attestor is not a known authority - may carry great weight.

All of this provokes our purpose, to construct an entity authentication protocol with maximum flexibility. Where supplicants claiming eponymity with an identity without credentials may be attested for by authenticated entities, with a verifier able to make a reasoned judgment as to the validity of the claim. The effectiveness of our approach will require the assumption to be made that entities will act with integrity and that a sufficient underlying communal trust exists, such that the verifier may be confident in their decision to accept. Our protocol, AHKA, is therefore designed to provide ad hoc keyless authentication - where keyless is on the part of the claimant, not the attestors - and to do so in a manner which provides cryptographic verification of the attestors as well as a cryptographically verified record of the authentication process. To accomplish this, a multisignature scheme, including multiple forms of key distribution providing differing qualities to their resultant keys, will be employed.

The primary use of our protocol is for establishing eponymity for physical access to lightly secured facilities, the archetypal office building or dining hall or library at a university. Its usefulness does not extend to locations which need stricter security, for which purpose more traditional entity authentication methods have greater utility. Our rough guideline for applicability is that if the secure access system is necessary - as opposed to simply useful - then the security demands are too strict for AHKA. Although this is a narrow focus it is also a widespread one, for instance few of our example facilities lack electronic access control in the present era. We do note as well that the scheme uses a common signature form, and so can be extended to incorporate single signatures and the direct entity authentication they may provide. Lastly, as stated the applicability is also dependent on both a philosophical mindset amongst the participants of the scheme and that attestors are easy to find in an ad hoc manner. These are the more dramatically limiting of the requirements for the scheme. The former requirement is demanding, and relatively closed and self-monitoring communities, such as rural towns or colleges, are more likely to satisfy it. Similarly, the latter demands a relatively small community, such that social networks are sufficiently dense so that out of a small random group of entities it is likely at least one of them knows and therefore may attest for the claimant.

1.1 Related Research and Topics

The closest parallel to our track of research are authenticated mobile ad hoc networks (MANETs), which allow entities with no common infrastructure to undertake entity authentication and other processes. Unlike our setting, in such cases a more robust security model is demanded, as the schemes do not assume communal trust which can be drawn upon within the authentication process. Still, some of the contexts for which the schemes are applicable mimic those for which AHKA is, and some of the network behaviors discussed within relevant research lend perspective to the problems of applying authentication to such systems. Examples of papers on the topic include [CBH03, LLZ$^+$07, ZH99]. In addition, although we have used a multisignature approach for our protocol, there are arguments for using a group or ring-based signature scheme instead. Examples of such schemes include [CvH91, BBS04, RST01].

2 Overview

Although the formal construction of the protocol will follow in the next section, we will first give an overview of the approach. Of first concern is the explicit formulation of the principles under whose guidance our construction has been designed. Second is a sketch of the key generation as well as signature and verification components of the protocol. To be dealt with last is the adversarial model under which we will argue our protocol provides for entity authentication.

2.1 Principles

The AHKA protocol is not meant as a general solution for entity authentication. Instead, it is designed with three principles in mind, and its applicability is contained to settings within which those principles are paramount. In *Section 4* we will discuss the size of such systems, but although the number of participating entities must align within bounds, the necessary constitution is not of population but of demographics. It is more important that the entities understand the operation of the scheme and their interest in it, and therefore are committed to its correct operation. The first two principles, *flexibility* and *accessibility*, are the instantiation of the former aspect of this, the knowledge and understanding of the scheme and its purpose. The third and final principle, *communal trust*, does likewise for the commitment to it.

The burden of proof for the eponymity of an identity is not constant throughout all contexts. Rather, the authentication threshold is a judgment made individually or systemically dependent on the specifics of the setting. An important aspect to the establishment of authentication requirements is the culture of the community of entities who will constitute the actors within a scheme. Not only does the culture alter the likelihood that malicious action will be taken within it, it also affects the internal capability to detect and respond to it. If entities are aware and alert to the mechanisms of the system and the behaviors of the community underlying it, then they will be capable of both preemptive and reactionary responses to malicious activity. Due to this dependency it is burdensome for the protocol to restrict the form of attestments or their validity. As such, it should not provide any structure beyond that necessary for the authentication of the attestors as their claimed identities.

Principle 1 (Flexibility)**.** The protocol should place no constraint on the composition of attestments, nor should it place constraints on the discretion of the verifier to adjudge whether an attestment is sufficient.

Note that this principle of flexibility is local and not global. For certain purposes within the system it may be reasonable and necessary to demand strict requirements for attestation. Our principle is only that no mathematical restraints may be placed generally upon the system, and so specific verification processes can define the requirements as they wish, including leaving them arbitrary.

As the effectiveness of the scheme is connected to its ease of understanding by its users, maximizing the latter will maximize the former. It naturally follows that the simpler the scheme the better. We should clarify that by ease of understanding we are not referring to the internal mathematics of its construction, as they may be abstracted away by various means such as smartcards or software with quality user interaction design. Rather, we mean the ease of understanding the protocol-level operation and the roles which various entities play within it. Further, the protocol should not just be easy to understand in the abstract, but easy to use in practice. These symbiotically assist one another, as understanding begets an intuition as to operation, and continued experience raises understanding. Therefore, no demand should be placed on the participating entities which would require more than the most basic of technical knowledge, or at least no demand which cannot be abstracted away by software or hardware design. The use of smartcards in particular is a prime example of this, with a system requiring only the presentation of a physical card a reasonable formulation of the ideal.

Principle 2 (Accessibility)**.** The protocol design should prioritize ease of understanding and its usage should not require more than basic technical capability.

The final guiding principle is also the most fundamental. Although an inflexible and inaccessible protocol design would still theoretically satisfy the basic conceit of our approach, communal trust is strictly necessary. It is the philosophical basis on which the scheme rests, and it will be the justification for the most important assumption we make with regards to our adversarial model. The protocol is designed to allow entities access to services without providing credentials based on the attestment of other entities. Obviously if those attesting entities are dishonest then the scheme provides no security. Although gaining the capability to attest will have a central authority as a gatekeeper, those individual attestors are the gatekeepers to the authentication process itself. The assumption that they will act responsibly and work to uphold the integrity of the system is an absolute requirement of our approach.

Principle 3 (Communal Trust)**.** The protocol design should incorporate the expectation that its users will at least passively work to uphold the integrity of the system.

A comment must be made on the use of the term "passively" in the letter of the principle. Although our principle is just that and is not a strict formal statement, it is still unreasonable to advocate for a conception of communal trust in which all entities work actively to prevent malicious action. A verifying entity may deny an attestment of which they are suspicious, but they are unlikely to follow that denial by calling in the central authority to inspect the matter or taking it upon themselves to distribute information regarding it to other verifiers. A rough guideline for our distinction between active and passive work is that the although we may expect entities to "do the right thing" when presented with a situation which could violate the integrity of the system, we cannot expect them to seek out those choices. In systems where entities are that motivated to protect the system neither accessibility nor flexibility are demanded, and classic, stronger authentication solutions are a better choice.

2.2 Key Generation and Distribution

Generating and distributing keys can be a difficult problem for multisignature schemes. Such schemes often conduct their setup process as a group interaction amongst the entities in the system to which the scheme is being applied. This invites adversarial approaches in which an attacker - either through corruption or impersonation of the other entities participating in the key distribution process - is able to determine sufficient information about the key distributed to an honest party to forge messages in their name. A further discussion of the difficulties encountered by such approaches is discussed in [BN06], which for instance attributes them to both of [MOR01, Bol03]. For this reason our choice of multisignature scheme will be the approach introduced in the former paper itself, based on the Schnorr signature scheme [Sch91]. For this scheme of Bellare and Neven key generation is a random choice of $x \in \mathbb{Z}_p$, and key distribution may be accomplished by any general PKI method, including user-generation with a central registry or certified credentials.

Another reason - beyond its lesser susceptibility to malicious attack of the key distribution process - which encourages our use of the approach of Bellare and Neven is our principle of accessibility. It demands that all aspects of our protocol, including key generation and distribution, be executable by those with little technical skill. Although capable user interaction design could ease the difficulty of entities generating their own keys it is highly unlikely to eliminate it, and so a private key generation (PKG) role for the central authority seems necessary. However, this demands great trust of the central authority by the participating entities in the system, since the authority may pose as any user whose key it has generated. As no restriction on the method of key distribution is cast by the scheme of Bellare and Neven it allows for the inclusion of multiple approaches into our protocol, allowing entities the option to construct their own keys, which reduces the trust necessary in the authority - including further optional privacy protection through the use of pseudonymous credentials.

Three key distribution algorithms are therefore incorporated in the protocol. All three call `keygen` as a subroutine, which algorithm returns a private key for the signature scheme. The first, `std-keydist`, is where the central authority acts as a PKG. After authenticating the entity requesting a key, they generate it with the subroutine call and distribute the key to the entity over a secure channel. The second, `tred-keydist`, is the approach which provides for trust reduced keys, in which the entity runs `keygen` themselves and, after authenticating to the central authority, sends the public key to them over a secure channel. In the final such algorithm, `priv-keydist`, the entity once again uses the subroutine call to generate a key pair. However, instead of authenticating as a known identity to the central authority they choose an anonymity set from amongst the active identities in the system. They then execute an interactive zero-knowledge proof with the authority, which establishes they are eponymous with an identity in the anonymity set without the authority being able to determine which one. Finally, the entity provides the public key to the authority over a secure channel, who may be certain it was generated by a legal member of the system without knowing which entity it represents.

The descriptions in the preceding paragraph omitted an important aspect of each of the `xxx-keydist` algorithms, which is how the public keys are registered within the system. With no restrictions on the PKI mechanisms of the protocol, this can be accomplished either by a central registry with interactive validity checks or by the use of non-interactive certified credentials. Although the latter approach is more common in general, our protocol will use the former - albeit with the caveat that if the protocol is extended to accommodate single signatures then formulating an alternative with certified credentials is straightforward. We have two arguments for this choice. The first is that geographic and communal attitudes mitigate the likelihood of access denial. In the settings where our principles apply there is likely to be little geographical or network dispersement and therefore latency. Further, the central authority is unlikely to deny access to the central registry arbitrarily due to communal trust. As such, the access demands which are of concern for centralized revocation approaches are lessened under our circumstances. Secondly, although we are assuming communal trust we would be remiss to presume it is total in any setting. Instead, the applicability of our protocol requires astute recognition and the capability to respond to malicious action within the system. As such, prompt revocation, which is more attainable with interactivity, is of benefit to our system. In addition, the use of a centralized registry allows for implementing alterations to the validity checking process to be borne primarily by the authority and not the entities, which fits under our principle of accessibility.

One last comment before proceeding is that the lack of geographical dispersion noted in the preceding paragraph invites interesting alternative approaches to key distribution through physical means. For example, `std-keydist` can be accomplished by the central authority issuing a smartcard to entities, such as an employee or student, upon their initial arrival or as needed. A secure and authenticated channel for distribution then becomes of trivial concern. Trust reduced credential generation can also easily make use of such methods, such as with an entity bringing in a public key to be entered into the registry on their behalf. Privacy protecting approaches are more difficult, but not impossible, to imagine developing. For instance, if there is a high-traffic area which only members of the system can access, such as an employee lounge, dropboxes may be used so that an employee can deliver a key to be registered without the authority able to determine which entity it belongs to. Although these approaches may struggle with scale, they provide interesting and useful alternatives to communication system-based mechanisms for key distribution.

Optional Trust Reduction

As noted, `std-keydist` requires entities to place substantial trust in the central authority. Since the latter is acting as a PKG, they have the private key of the entity and may impersonate them for attesting and, if the protocol is extended to incorporate single signatures, signing in general. Our communal trust principle states that this concern may be disregarded, for the central authority would be violating the integrity of the system. Although this is true, the incorporation of an alternative approach is supported by the principle of flexibility, as adding an additional mechanism by which this potential attack by can be mitigated with no effect on communal trust brings flexibility at no cost. Since such a mechanism exists and is of straightforward construction, there is no reason not to incorporate it into the protocol.

Hence the inclusion of `tred-keydist`. Its distinction from `std-keydist` is minimal, with the entity simply generating the key themselves and sending the public key to the authority to be registered, as opposed to the authority generating it and sending the private key to the entity. The only difficulty raised for the trust reduced distribution mechanism is how may the entity be authenticated by the authority? There are a few options. First, they may already possess a mechanism for authentication outside of the keyless approach, such as credentials to an organizational intranet. Second, especially if single signatures are incorporated into the protocol, then if they already hold a key pair generated with `std-keydist` then that may be used for authentication. Lastly, physical methods may be used. In any case, the authentication process, even if burdensome, would only need to be executed once or on rare occasion, as the keys - providing no privacy protection - have no reason to be duplicated.

Optional Privacy Protection

Unlike trust reduction, considering privacy protection does not bring a retort from communal trust. A central authority might very well consider its pledge to uphold the integrity of the system to not include a promise not to track the entities within it. As such, incorporating a method by which entities can acquire pseudonymous or anonymous credentials provides a direct extension of the protocol. Although the latter would be preferable, the increased complexity of anonymous attestment processes - even with simplifications provided by quality hardware and software design - places a burden on those entities without strong technical capability and violates our accessibility principle. Although we are willing to require more technical capacity for entities wishing for privacy protection, as they are likely willing to devote some time to training on and understanding the approach, it is against our principles to pass this onto all entities. As such, any privacy protection must be bolted onto the existing methods of key generation and distribution so as to not disturb our basic approach for standard distribution and the only slightly more complex approach for trust reduced distribution.

As such, a pseudonymous method will be our choice. Although this requires the generation of a new key pair for any action which an entity wishes to be decoupled from all previous actions, the attestment processes for which our protocol is suited lessens this issue. If the protocol was being designed for machine use - which may be making thousands of attestments per second - our approach would not be suitable. However, the use of our protocol is designed around human tempos of interaction and the need for attestment in smaller communities, which means that an entity is unlikely to need more than a handful of pseudonymous credentials in between opportunities to procure more. In addition, our privacy protecting method will require an interactive key distribution process, with the lack of access and communication issues which justified the use of an interactive revocation approach also justifying its use here.

The pseudonymous key distribution will work as follows. After the entity runs `keygen` and gains the key pair they wish to have registered under a pseudonym, they must prove they already possess a private key which is in the central registry without exposing which it is. To accomplish this they will execute a zero knowledge proof of knowledge of secret key (ZKP-KOSK). Multiple methods for this type of proof exist. For instance, if the private key was generated by the central authority the entity can prove inclusion by way of a zero knowledge proof of set inclusion, a topic discussed in great depth in [Pen11], with specific examples being [CL02, CCas08]. However, there are three issues with this class of approaches. First, they require the central authority to know the private key. Second, existent such methods are often, though not always, designed with the party who creates the set acting as prover, which is not the case for our purposes. Lastly and most seriously, since the authority generates the set it is difficult to provide any flexibility with regards to which entities are members of the anonymity set for a pseudonymous key. For instance, an entity may want to prove they are a member of a certain suborganization by showing the private key belongs to an entity in a collection filled only of members of that subgroup. This cannot be comfortably accomplished by such approaches.

Instead, we will use the ZKP-KOSK developed by Chaum and van Heyst in [CvH91] originally for use with group signatures. In this case the verifier need only know the public key, the prover can choose an anonymity set to their liking, and a signature is generated on the message, providing direct message authentication independent to that provided by the channel. The method is not particularly efficient and is interactive, but as before we argue the setting mitigates the difficulties raised. We reiterate as well that physical methods of key distribution are also feasible for this purpose. Although the `std-keydist` and `tred-keydist` are sufficiently straightforward as to where physical distribution seems unnecessary, the additional complexity - both theoretical and practical - of the `priv-keydist` algorithm makes it more appealing.

2.3 Signing and Verification

The surmountable but notable difficulty raised by our use of the multisignature scheme of Bellare and Neven is its multi-round construction. As we shall see its execution requires multiple computations on the part of each signer, with each needing to complete their task for the round and distribute the output before the process can proceed. Various methods for handling the signature process invite concerns. For instance, if smartcards are used throughout the system then having a set number of ports on a card reader limits the number of possible attestors - violating the principle of flexibility - whereas requiring each attestor to swipe in the correct order for each round is time-consuming and confusing - violating accessibility. If every entity is using standard distributed keys then temporarily loading them into memory is an option, but it doesn't extend for trust reducing or privacy protecting keys. Various approaches may be used to mitigate these concerns. One, more adaptable and flexible computational devices can be used as a physical carrier for a key, such as smartphones communicating over a local or general network. Also, although the tradeoff between accessibility and flexibility cannot be eliminated, it might be reasonable within a system for an upper bound to be set on the number of attestments at the point where the diminishing value of an additional one is essentially nil. If eight individuals have attested on behalf of an entity, the ninth attestment is unlikely to move the needle unless on the strength of that individual's character, in which case they may be substituted for a less influential signer. If the physical capabilities of the system can accommodate that upper bound the system may be effectively flexible. Our principles are meant to be just that and not hard and fast rules.

In addition to the attestors the verifier is themselves a party to the signing of the attestment statement, as the purpose of the multisignature is also for record keeping. If the attestment was only relevant for the approval of the verifier, then individual attesting statements could be signed by each attestor and then discarded after access was given. Our construction is designed so that a concise and complete proof that the attestors listed on the attesting document actually did so can be stored and referenced, especially for investigating malicious action. As such, for completeness the verifier is a signer as well. After the multisignature has been computed the verifier runs the verification algorithm on it, and if it accepts then they know it was crafted properly. In the presentation of the `sign` algorithm in the following section, the final step in which the second signature component is calculated therefore only needs to be run by the verifier, although we have left it in the algorithm as a general step in case any other signers wish to confirm the validity of the signature themselves.

An important comment is that the verifier must not compute the `verify` algorithm using the public keys provided by the attestors, or at least not without confirming their validity by comparing key fingerprints against the database. If the verifier checks the identity of the attestor but not the key they provide, an attacker may exploit this by claiming an identity with an active key pair and then providing a different one for which they know a private key. As for efficiency reasons the verifier may wish to use the public key provided locally during the signing process so as not to make a network call to the database, to do this without introducing a security flaw they must verify the signature using the registered key. As a last note, we have not made any comment as to whether the identity whose eponymity is being established must itself be registered in the central database. There is no technical reason why it must be - necessarily, as the access is designed to be keyless. However, in many systems it may be preferred that only currently active identities may be claimed and attested for, since it allows for better organized record keeping since every attestment concerns a known and unique identity. Further, this abets punishment of malicious action - if an entity not known to the system is attested for and then commits illicit behavior there is no additional information to track them down beyond the identities of their attestors.

2.4 Adversarial Model

We are concerned with four different attacks that may be made on the protocol.

1. UNCERT-KEYDIST: An uncertified entity (an entity without any key pair in the active central registry) successfully executing the `priv-keydist` algorithm.

2. UNCERT-FORGE: An uncertified entity successfully executing an attestment process.

3. AUTH-TRED: The authority gaining knowledge of the private key provided to it by the `tred-keydist` algorithm.

4. AUTH-PRIV: The authority gaining knowledge of the private key provided to it by the `priv-keydist` algorithm or knowledge of which member of the anonymity set it belongs to.

We will work with the following assumptions.

Assumption 1 (Integrity)**.** Let $s \in S$ be a possible signer, where S is the set of all entities with active keys in the central registry. Then s will not execute the signature algorithm `sign` on behalf of a claimant entity if they do not know the entity to be eponymous with the claimed identity.

The crucial ommitance in the assumption - which originates in the principle of communal trust providing a guarantee of passive but not active action to uphold the integrity of the system - is that the signer does not demand formal proof of the eponymity. Naturally, if the claimant could always provide it there would be no need for our approach to authentication. Therefore our integrity assumption is weaker, only that they do not knowingly violate the integrity of the system. However, since all entity authentication approaches are broken if attackers can convince attesting parties they are who they are not - for instance in a more traditional certificate-based approach an attacker convincing a trusted third party to issue them an incorrect but valid certificate - we do not consider this to have any practical effect. Our second assumption is the necessary one for the proper mathematical operation of our scheme.

Assumption 2 (Discrete Logarithm Assumption)**.** Let $\mathbb{G}$ be a multiplicative group of large prime order p. An algorithm $\mathtt{A}$ has advantage $\epsilon_{\mathtt{A}}$ in solving the discrete logarithm problem if $\Pr[\mathtt{A}(g, y) = x] \geq \epsilon_{\mathtt{A}}$ where $g \in \mathbb{G}$ is a generator for the group and $g^x = y$. We assume no efficient algorithm $\mathtt{A}$ exists such that $\epsilon_{\mathtt{A}}$ is non-negligible.

Of first concern are the key distribution attacks. We first comment that there is no analogue for UNCERT-KEYDIST pertaining to the `std-keydist` or `tred-keydist` algorithms. Although the exact authentication methods used for their successful execution we have not specified, the lack of privacy protection allows for direct confirmation before acceptance, and so an uncertified attacker has no avenue to attempt to convince the authority they are a member of the system. On the other hand, if that attacker can successfully execute the ZKP-KOSK for some choice of anonymity set, then they may register a key pair with the central authority and gain illicit access. That this attack cannot be successfully executed is a direct consequence of the properties of zero-knowledge proofs. In the proof, the prover uses a one-way permutation to commit the public keys in the anonymity set, and provides the index of the commitment which corresponds to the public key bound to the private key they are proving knowledge of. The verifier is then able to use the committed public key as an input to compute a statement which shows that another value provided by the prover could only have been constructed if they knew the bound private key, thereby establishing the knowledge. For the prover to accomplish this without a private key bound to a public key in the anonymity set would violate the soundness of the ZKP-KOSK.

The second key distribution attack, AUTH-TRED, is quite straightforward to analyze. As Schnorr signatures are being used for the scheme, the authority can only determine the private key bound to the public key they receive in the `tred-keydist` algorithm if they can violate our second assumption - rendering it inoperable. The second authority attack, AUTH-PRIV, is only slightly more interesting. First, the same argument for security against AUTH-TRED protects against the authority gaining knowledge by direct analysis of the public key which they are provided. Second, the authority is unable to do so by exploiting the ZKP-KOSK, since they would need to violate the zero-knowledge property of the proof. The simulator presented by Chaum and van Heyst in [CvH91] shows that the authority can gain no knowledge of either the secret key proven or - up to its inclusion in the anonymity set and therefore registry - its bound public key. We add, as an aside, that there is a philosophical argument under communal trust for interpreting the semi-honest model for the authority as verifier in the ZKP-KOSK as reasonable. If the authority does not wish to allow for privacy protecting keys, they can simply not allow their registration. Therefore it may be reasonable to assume they will execute the proof honestly.

The final attack we must consider is UNCERT-FORGE. The first method the attacker may use is to have an entity from the system with valid keys attest for them. To accomplish this the attestor would need to sign on behalf of a claim of eponymity they do not know to be valid, which would violate our first assumption regarding integrity and so is an inoperable approach. However, the entity may still attempt the attestment process by putting forward themselves - in a networked communications context - or a compatriot as an attestor on their behalf. Importantly, the compatriot must not have a valid key pair themselves to not violate our assumption. For the attacker to be successful, their malicious attestor must be able to successfully execute `sign` such that it will validate, despite the lack of a key in the central registry. The proof of security against this attack is provided by Bellare and Neven in [BN06]. Crucially, their proof assumes that at least a single signer is honest - and since the verifier themselves signs the attestment document, the existence of such a signer is guaranteed. Therefore the attacker is unable to have an attestment statement forged on their behalf, and so cannot successfully execute an attestment process and convince the verifier they are who they are not. We note in conclusion that formally the multisignature scheme of Bellare and Neven has only been proven in the random oracle model.

3 Construction

We define our protocol AHKA = (`pargen`, `keygen`, `std-keydist`, `tred-keydist`, `priv-keydist`, `sign`, `verify`) by the following algorithms. With the exclusion of the `xxx-keydist` algorithms these are all due to [BN06], while the ZKP-KOSK in `priv-keydist` is from [CvH91]. We note as well that the undefined hash functions H_1 and H_2 used in the scheme must be a suitable choice for use in place of the random oracles used in the proof of Bellare and Neven.

3.1 Algorithms and Protocol

Algorithm 3.1 (`pargen`)**.** The central authority chooses a multiplicative group $\mathbb{G}$ of large prime order p, as well as a generator g of $\mathbb{G}$ and two hash functions $H_1 : \{0, 1\}^* \to \{0, 1\}^\alpha$ and $H_2 : \{0, 1\}^* \to \{0, 1\}^\beta$ for $\alpha, \beta \in \mathbb{N}$. The authority then publishes $params = (\mathbb{G}, p, g, H_1, H_2)$.

Algorithm 3.2 (`keygen`)**.** On input $(params, p, g)$, a random secret key $x \in \mathbb{Z}_p$ is generated with companion public key $X = g^x$. Output is the key pair (x, X).

Algorithm 3.3 (`std-keydist`)**.** On request of a signer $s \in S$, the central authority runs `keygen`$(params, p, g)$ and over a secure and authenticated channel gives the output (x_s^{std}, X_s^{std}) to the signer. The central authority then enters X_s^{std} into the central registry.

Algorithm 3.4 (`tred-keydist`)**.** A signer $s \in S$ runs `keygen`$(params, p, g)$ giving output (x_s^{tred}, X_s^{tred}). Over a secure channel they then authenticate as s to the central authority, and give them X_s^{tred}. The central authority then enters X_s^{tred} into the central registry.

Algorithm 3.5 (`priv-keydist`)**.** A signer $s \in S$ runs `keygen`$(params, p, g)$ giving output (x_s^{priv}, X_s^{priv}). They agree on an additional generator g' and a security parameter k with the central authority (both of which may be standardized). The signer then chooses an anonymity set $S^K \subseteq S$ with $s \in S^K$, and constructs the set Γ by choosing a public key $X_{S_i}^*$ from the central registry for every $S_i \in S^K$. Using the secret key x_s^* whose companion public key is in Γ and therefore the central registry, the signer computes $a \equiv (X_s^{priv})^{x_s^*} \pmod p$ and sends (X_s^{priv}, a) to the central authority. The signer and central authority then execute the following zero-knowledge proof.

1. The signer randomly chooses numbers $r_1, \cdots, r_{|\Gamma|}, t_1, t_2, t_3 \in \mathbb{Z}_p \setminus \{0\}$ and a one-way permutation π of Γ. They then send to the central authority

$$x \equiv \left(\frac{g}{X_s^{priv}}\right)^{t_1} (g')^{t_2} \pmod p, \;\; y \equiv (X_s^{priv})^{t_3} \pmod p, \;\text{ and }\; z_{\pi(i)} \equiv \gamma_i (g')^{r_i} \pmod p \;(\text{for all } \gamma_i \in \Gamma).$$

2. The central authority chooses $b \in \{0, 1\}$ and sends it to the signer.

3. If $b = 0$ the prover replies with $(r_1, \cdots, r_{|\Gamma|}, t_1, t_2, t_3, \pi)$, whereas if $b = 1$ they reply with $(u = t_1 + x_s^*, v = t_2 + R, w = t_3 + x_s^*, j)$, where $R = r_i$ where i is the index of X_s^* in Γ and j is its index under π.

4. If $b = 0$, the verifier confirms that x, y, $z_1, \cdots, z_{|\Gamma|}$ are properly constructed, and outputs *accept* if so and *reject* otherwise. If $b = 1$, then the verifier confirms that

$$y \cdot a \equiv (X_s^{priv})^w \pmod{p} \text{ and } x \cdot z_j \equiv a \left(\frac{g}{X_s^{priv}} \right)^u (g')^v \pmod{p}$$

   and if so outputs *accept* or else outputs *reject*.

This process is then repeated k times, giving assurance that the signer is a member of the infrastructure with confidence $1 - 2^{-k}$.

Algorithm 3.6 (`sign`)**.** Input is ($params$, k, $L = \{K, C_2, \cdots, C_n\}$, $\mathcal{L}$, m, H_1, H_2), where $k = x_s^*$ is a private key belonging to the signer, $K = X_s^*$ is its companion public key, C_i for $2 \leq i \leq n$ are public keys in the central registry belonging to the cosigners, and $m \in \{0, 1\}^*$ is the message to be signed. The ordering of L is not global and does not need to be agreed upon by the cosigners, however a specific encoding $\mathcal{L}$ must be. The signing process takes four rounds.

1. The signer chooses a random $r_s \in \mathbb{Z}_p$ and computes $R_s = g^{r_s}$ and $t_s = H_1(R_s)$. They then send t_s to every $C_i \in L$.

2. On receipt of t_i from every C_i, the signer sends R_s to every C_i.

3. On receipt of R_i from every C_i, the signer confirms that for all $2 \leq i \leq n$ the equation $t_i = H_1(R_i)$ holds. The algorithm aborts if this is not the case for some i. If proceeding, the signer computes $R = R_s \cdot \prod_{i=2}^{n} R_i$ and $b_s = H_2(K \,||\, R \,||\, \mathcal{L} \,||\, m)$. They then compute $a_s \equiv x_s b_s + r_s \pmod{p}$ and send it to every C_i.

4. On receipt of a_i from every C_i, the signer computes $a \equiv a_s + \sum_{i=2}^{n} a_i \pmod{p}$.

The output of the algorithm is then the multisignature $\sigma = (R, a)$.

Algorithm 3.7 (`verify`)**.** Input is ($params$, $L = \{C_1, \cdots, C_n\}$, $\mathcal{L}$, m, σ, H_2). For $1 \leq j \leq n$ the verifier computes $b_j = H_2(C_j \,||\, R \,||\, \mathcal{L} \,||\, m)$. If $g^a = R \cdot \prod_{j=1}^{n} C_j^{b_j}$ the algorithm outputs *accept*, else it outputs *reject*.

We may now combine all of these component algorithms to construct the full protocol for AHKA.

Protocol 3.8 (AHKA)**.** An AHKA scheme with a central authority and a collection of entities operates as follows.

1. The central authority runs `pargen` and distributes the output amongst the entities in the system. They also initialize a database of active key pairs.

2. An entity within the system acquires an active key pair by deciding on which key distribution process to undertake. Then

(a) if `std-keydist`, they authenticate to the central authority and receive a key pair according to authority executing the algorithm;

(b) if `tred-keydist`, they authenticate to the authority and generate and register their key pair according to their execution of the algorithm;

(c) if `priv-keydist`, they generate a key pair and register it with the authority according to their execution of the algorithm.

3. Upon request of another entity to attest for a claimed eponymity, if the entity accepts the attestment document constructed by the verifier they then execute `sign` with the verifier and any other attestors.

4. The verifier confirms the claimed identities of the attestors have active key pairs within the central directory and retrieves the relevant public keys from it. They then validate the signature by executing `verify`, and if verified they grant the claimant all access rights and privileges they are afforded by their accepted identity.

3.2 Correctness of Algorithms

We will quickly show the completeness of the ZKP-KOSK as well of the verification process for the multisignature. For the former, we have for the first query in the fourth step

$$y \cdot a \equiv (X_s^{priv})^{t_3}(X_s^{priv})^{x_s^*} \equiv (X_s^{priv})^{t_3+x_s^*} \equiv (X_s^{priv})^{w} \pmod{p}.$$

As for the second query, note that $z_j = X_s^*(g')^{R_1}$. As such, we have

$$\begin{array}{rcl} a\left(\frac{g}{X_s^{priv}}\right)^u (g')^v & \equiv & (X_s^{priv})^{x_s^*}\frac{g^{t_1}g^{x_s^*}}{(X_s^{priv})^{t_1}(X_s^{priv})^{x_s^*}}(g')^{t_2}(g')^R \\ & \equiv & \left(\frac{g}{X_s^{priv}}\right)^{t_1}(g')^{t_2}\cdot g^{x_s^*}(g')^R \\ & \equiv & x\cdot X_s^*(g')^R \\ & \equiv & x \cdot z_j \pmod{p}. \end{array}$$

Combined, these establish that the prover knows x_s^* such that $a \equiv (X_s^{priv})^{x_s^*} \pmod{p}$ and $X_s^* \equiv g^{x_s^*} \pmod{p}$. A simulator which establishes the zero-knowledge of the proof is provided in [CvH91]. As for the correctness of the verification process, if c_i is the corresponding private key for a given C_i we have

$$\begin{array}{rcl} R\cdot\prod_{j=1}^n C_j^{b_j} & = & \prod_{i=1}^n R_j \cdot C_j^{b_j} \\ & = & \prod_{j=1}^n g^{r_j}\cdot(g^{c_j})^{b_j} \\ & = & \prod_{j=1}^n g^{c_j b_j + r_j} \\ & = & \prod_{j=1}^n g^{a_j} \\ & = & g^a. \end{array}$$

Therefore a properly constructed signature will verify.

3.3 Example Use

We conclude this section by giving a brief example usage of the scheme. Suppose that two entities with the veronyms Alice and Bob and another with pseudonym 123ABC are asked by an entity claiming to be Fred to attest to Victor on his behalf. Having each run or requested a run of an `xxx-keydist` algorithm, Alice has key pair (x_A^{std}, X_A^{std}), Bob has (x_B^{tred}, X_B^{tred}), 123ABC has (x_C^{priv}, X_C^{priv}), and Victor has (x_V^{std}, X_V^{std}). First, Victor must decide if he considers one pseudonymous and two veronymous attestments to be sufficient to fulfill the requirements for authentication he has set. If so, all the signers agree on an encoding of their keys, and they agree on an attestment statement. These latter two agreements will likely be standardized within a system, for instance the encoding may be lexicographical on the identities of the signers, and the attestment statement may be along the lines of

"Attestors:Alice,Bob,123ABC;Behalf:Fred;Accepted:Victor;Location:Cafeteria;Time:01-01-2016At1305"

with m a binary representation of it. Either the full attestment statement may be signed or a digest of it may be substituted.

The signers Alice, Bob, 123ABC, and Victor all then execute `sign`. As an example, Alice runs the algorithm with input $(params, x_A^{std}, \{X_A^{std}, X_B^{tred}, X_C^{priv}, X_V^{std}\}, \mathcal{L}, m, H_1, H_2)$. After completion Victor runs `verify` on the resultant signature, executing the process using the public keys listed in the central registry for each identity - not those provided by the signers themselves. If the signature is verified it constitutes a cryptographic proof of attestment. Victor then considers the claimant entity to be Fred and under advisement from the central registry grants them the access rights to which they are entitled, and may archive the attestment statement and the signature so that it may be reviewed in the future.

4 Community Analysis

We have mentioned on multiple occasions the necessary attitude that must exist amongst entities in a system using our protocol, and it is worth expanding on. In addition, a discussion of the scale of systems at which our protocol can be effectively operational is of use as well. For the former, of primary importance is our conception of communal trust. Although the letter of its principle denotes how we expect entities in a system with sufficient communal trust to act, it does not describe the makeup of those systems themselves. In particular, although our protocol - through the use of multisignatures for record keeping and the central registry approach to revocation - has mechanisms for analyzing and punishing behavior within the system, our concept of communal trust requests a base inclination amongst entities away from testing those mechanisms. If the entities in the system can be expected to push and prod at an instantiation of the protocol to find weaknesses then they will likely find them, particularly by exploiting the social dynamics within the community of entities.

As such, punishment and penalty cannot be relied upon to maintain the effective operation of the system. If they must be, then a more traditional entity authentication approach is demanded. What our protocol requests is that the entities participate properly not because they are afraid of being caught and punished, but because they have an invested interest - social, psychological, ideological, financial, etc. - in its correct operation. This is why we have deemed our principle *communal* trust. Our expectation is that the entity will act judiciously because of the expectations and norms of the community of entities within the system, and not due to a local, constrained calculation of risk and reward with regards to undertaking malicious action.

We have also explicitly noted the importance of flexibility for our protocol, which assumes that not only do verifying entities subscribe to the principle of communal trust, but that they also have sufficient capability - socially and not technologically, as the latter may be assisted by interaction design - to adjudge attestments and operate the protocol effectively. Essentially, not only must verifiers wish to protect the integrity of the system by demanding attestation for a claim of eponymity, they must also be capable of judging whether a sufficient burden is met. This capability need not be perfect, in fact, if perfection is a quantifiable and attainable quality for attestments then the flexibility is no longer necessary and attestation requirements may be standardized. Instead, a simple demand of reasonableness - however that may be interpreted - is required. For instance, if a specific subtask within the system requires two identified attestments, then another subtask for which the damage done by malicious use is greater should not have a lesser demand.

Our final comment will be on the scale of the communities. We have little to operate with other then heuristics for this analysis, due to the often limited and inconsistent literature on the subject. Essentially, we are interested in finding some measure for what percentage of different sized social networks the average entity within one would be willing to attest for. This is a difficult task, since it is hard to know exactly what level of relationship is required for attestment without asking directly, and we were unable to find any studies which did so. One study, [KJB+90], asked which members of a local phone book an entity was acquainted with, but they recount drastically different sizes from different experiments, ultimately concluding that the average person has a personal network of around 1500 individuals in larger communities. A classic result in social network theory, Dunbar's Law, meanwhile states that the average individual can have meaningful relationships with around 150 individuals [Dun93, HD03]. The usefulness of this information is to us is suspect, especially since in the former they do not specify how many of those individuals are within certain subcommunities - such as our settings of a school or work organization - and in the latter no geographical bounding is placed. If we take 150 as our number - though not as a lower bound, but instead due to its common reference in discussion of community sizes - as the number of individuals within a community the average entity would attest for, and we demand that the average attestor be willing to attest for 25% of the community for the proper operation of our protocol, then we get a community size of 600 individuals, whereas 10% gives a size of 1500. This range of 600-1500 entities seems reasonable, but the many issues with our reasoning are plain, and some proper analysis, perhaps even just with simulated individuals, would be a useful focus for future investigation.

5 Conclusion

The AHKA protocol is not designed as a generalized entity authentication solution. It relies on behavior from attesting entities that may be a rarity, has constraints on the community size, and is not suitable for authentication in security-critical environments. However, what it lacks in polished and general perfection it makes up for in flexibility and ease of use. In both physical and communication network environments it allows for entities to claim eponymity with an identity even if they are without or have lost their credentials, requiring only other entities willing to attest on their behalf. Although such an authentication approach can be done without cryptographic tools, their addition provides confirmation of the identities of the attestors and the formulation of an incontrovertible authentication record.

Bibliography

[BBS04] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short Group Signatures. In *Advances in Cryptology-CRYPTO 2004*, pages 41–55. Springer, 2004.

[BN06] Mihir Bellare and Gregory Neven. Multi-Signatures in the Plain Public-Key Model and A General Forking Lemma. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 390–399. ACM, 2006.

[Bol03] Alexandra Boldyreva. Threshold Signatures, Multisignatures and Blind Signatures Based on The Gap-Diffie-Hellman-Group Signature Scheme. In *Public Key Cryptography - PKC 2003*, pages 31–46. Springer, 2003.

[CBH03] Srdjan Capkun, Levente Buttya, and Jean-Pierre Hubaux. Self-Organized Public-Key Management for Mobile Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 2(1):52–64, 2003.

[CCas08] Jan Camenisch, Rafik Chaabouni, and abhi shelat. Efficient Protocols for Set Membership and Range Proofs. In *Advances in Cryptology-ASIACRYPT 2008*, pages 234–252. Springer, 2008.

[CL02] Jan Camenisch and Anna Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In *Advances in Cryptology-CRYPTO 2002*, pages 61–76. Springer, 2002.

[CvH91] David Chaum and Eugène van Heyst. Group Signatures. In *Advances in Cryptology-EUROCRYPT91*, pages 257–265. Springer, 1991.

[Dun93] Robin IM Dunbar. Coevolution of Neocortical Size, Group Size and Language in Humans. *Behavioral and Brain Sciences*, 16(04):681–694, 1993.

[HD03] Russell A. Hill and Robin IM Dunbar. Social Network Size in Humans. *Human Nature*, 14(1):53–72, 2003.

[KJB$^+$90] Peter D. Killworth, Eugene C. Johnsen, H. Russell Bernard, Gene Ann Sheller, and Christopher McCarty. Estimating the Size of Personal Networks. *Social Networks*, 12(4):289–312, 1990.

[LLZ$^+$07] Xiaodong Lin, Rongxing Lu, Haojin Zhu, Pin-Han Ho, Xuemin Shen, and Zhenfu Cao. ASRPAKE: An Anonymous Secure Routing Protocol with Authenticated Key Exchange for Wireless Ad Hoc Networks. In *ICC*, pages 1247–1253, 2007.

[MOR01] Silvio Micali, Kazuo Ohta, and Leonid Reyzin. Accountable-Subgroup Multisignatures. In *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pages 245–254. ACM, 2001.

[Pen11] Kun Peng. A General, Flexible and Efficient Proof of Inclusion and Exclusion. In *Topics in Cryptology-CT-RSA 2011*, pages 33–48. Springer, 2011.

[RST01] Ronald L. Rivest, Adi Shamir, and Yael Tauman. How to Leak a Secret. In *Advances in Cryptology-ASIACRYPT 2001*, pages 552–565. Springer, 2001.

[Sch91] Claus-Peter Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[ZH99] Lidong Zhou and Zygmunt J. Haas. Securing Ad Hoc Networks. *IEEE Network*, 13(6):24–30, 1999.

A Framework for Discussion of Entity Authentication Schemes

1 Introduction

As assurance of with whom you are communicating is a security demand of many communications systems, determining the *authorship* of messages within such a system falls within the bounds of cryptography's problem space. However, the mathematical toolkit which the field brings to bear against the problems within its domain is unfit to provide suitable solutions for this task of *entity authentication*. This incapability is inherent to the nature of the problem, as although we can make *proving* rigorous we can not do similarly for the concepts of *entity* and *identity*. Identity is a means by which entities refer to and acknowledge one another - in other words, it is a method of recognition. As such, an identity cannot be proven from first principles, as it becomes valid only when it is accepted by the entities in a system. Without the ability to anchor itself in meaningful axioms, mathematics cannot supply useful results to its study. Therefore, although mathematical techniques allow proving to be placed on firm foundations, identity must remain suspended, carefully balanced on webs of recognition and reference.

It is tempting for cryptographers to attempt to skirt this difficulty altogether. To declare, "we don't care why the entities in a system choose to recognize the identities for each other that they do, nor do we care how that process of recognition occurs. We concern ourselves with the proving of identities and binding those proofs to messages sent under the identity." This restricted purview seeks only to provide a guarantee of *publishership*. Although we will fully discuss the advantages of both the authorship and publishership methodologies when we have established the necessary common ground of terminology, it may be immediately concluded that the latter has substantial benefits. Its narrower suite of assumptions and goals allows for concise consideration of the concerns at hand, and it is sufficient for the purpose of evaluating the soundness, correctness, and security of entity authentication schemes. In many ways, cryptographers who design and evaluate such schemes *de facto* operate in their daily work with publishership - and not authorship - as the security guarantee they wish to establish.

It may be asked why then have we cast our net wider, and leapt immediately to connecting authorship to the task of establishing identity. Our reasoning originates in that whether a scheme provides protection against forgery is only part of its whole - its efficiency and ability to internally replicate the relationships between entities within the system is important to its effectiveness as well. Therefore, without an understanding of the methods of recognition in a system to which it may be deployed, the perspective justifying publishership has little ability to evaluate the usefulness of a scheme. For example, a communication system may demand widespread recognition amongst other entities as to the validity of a certain identity. Within the mechanics of the scheme, that recognition may be expressed directly by each entity or it may act through the conduit of a socially-empowered authority - the latter trusted not to bestow formal recognition unless the connection is widely accepted. Analyzing how identities function in the system is necessary to determine which approach is most useful for that context.

Those adamantly for a full restriction of cryptography's purview to publishership might argue that these are implementation concerns, irrelevant to the goals and domain of questions in the pure field. However, who better to examine and educate how an entity authentication scheme manages these concerns then its designer? Cryptography is too useful to appeal to its purity to justify such a dereliction. That a contextual discussion of applications appears in essentially all literature on entity authentication schemes is a tacit acknowledgment of this role for the cryptographer. A similar argument was made in much greater detail recently by Rogaway [Rog15], although he goes further and appeals for cryptographers to evaluate the moral consequences of potential uses of their work.

The purpose of this paper is to provide a framework for discussion of the usefulness of entity authentication schemes, and perhaps to provide a better vocabulary for discussion of their mathematical faultlessness as well. To provide a collection of terms and their definitions, as well as discourse around key considerations, which may be used to provide a common ground, a not-quite-firm - pliable - foundation, if you will, for those aspects of authorship which cannot be approached through rigorous means. All of the words and phrases which have until now been thrown about haphazardly: identity, entity, authentication, proof, authorship, publishership, entity authentication scheme, authority, will be defined and discussed, as will many other important concepts, such as credential, trust, and reputation. This is very much a position paper. Alternative terminologies and perspectives to the ideas discussed so far, and those to come, are no more or less right than those given herein.

A brief outline of this paper is as follows. Our attention will initially turn in the second section towards the building of the framework. The work we have just required of ourselves - defining terms and phrases and discussing important considerations - will be dealt with in this section. The first step of that process will be to provide a set of core definitions, followed by discussion of a few broader considerations which are important to evaluating the usefulness of entity authentication schemes, such as how they deal with privacy, statefulness, and fraud. The third section of the paper will be a quick overview of some cryptographic building blocks which often underlie the construction of such schemes. Then, the fourth and fifth sections will provide an opportunity to apply the framework we will have constructed to discussion of a selection of entity authentication approaches. The organizational divide between those sections will hinge on whether they require an authority, a crucial attribute as to the systems for which a given scheme is applicable.

2 Framework

Before proceeding, it is worth pausing and clarifying what an entity authentication scheme is. Although we will ultimately provide a concise definition, some of the terms we are yet to define are needed before we may do that. However, a sketch sufficient to frame those definitions is presently attainable and desirable. The goal of an entity authentication scheme is to establish an eponymous connection between a specific entity, known as the *prover*, and an identity, a set of information housed in a document known as a *credential.* The entity for whose inquiry the connection is proven is known as the *verifier.* To establish just the connection without a claim of eponymity, a cryptographically-verified process is employed in which the prover produces information bound to the credential - and therefore the identity - which confirms the connection in question.

At the conclusion of this process, (1) the verifier cannot believe the prover has information they do not have and therefore accept a connection they should reject, (2) the verifier cannot conclude the prover does not have the information if they do and reject a connection they should accept, and (3) the verifier cannot gain enough knowledge of the information to where they may maliciously attempt to prove their own connection to that identity to a third party. The most common cryptographic tool used to provide these three qualities is a *digital signature* supported by a key infrastructure, while the higher level construction known as a *zero-knowledge proof* (ZKP) provides an alternative approach, both of which will be discussed in the third section.

The difficulty which amounts to the *raison d'être* of this work comes in establishing the eponymity between the prover - an entity - and the identity. Within our context, we will use the term eponymous with a derived yet different usage to its common one.

Definition 2.1 (Eponymous)**.** An entity and identity are eponymous if the identity is commonly recognized as representing the entity by the other entities in a system.

The establishment of eponymity is accomplished by a collection of entities in the system attesting to it, staked on their *reputation*. For the verifier to accept a claim of eponymity, they must *trust* in these attestations. This collection may be the prover themselves self-attesting or it may be one or more mutually trusted third parties certifying the connection, the latter actor or actors either in the form of an empowered authority or a network of social contacts. Although the process of attestation itself may be considered mathematically, answering whether and to what extent other entities are required to certify eponymity is beyond its capability. We will not delve into the mathematical specifics of the construction of any schemes, but classic examples include the Feige-Fiat-Shamir Identification Scheme [FFS88] and the Schnorr Identification Scheme [Sch91] - both authority-based - and further discussion of the construction of such schemes may be found in many commonly referenced general cryptographic texts, such as [Sti05, Gol06].

2.1 Core Definitions

We now turn towards a collection of words and phrases which have previously entered into the discussion yet require a more careful treatment than that already given, or if not discussed then the meanings implied by their common usage. The first three of these terms, *entity*, *identity*, and *credential*, all refer to whom the actors in a system are and how the other actors in the system refer to them. As they can best be understood in how they relate to and differ from each other the discussion of their respective definitions will be joined. The next topic will be *authentication*, followed by *trust* and *reputation*. For these latter two our discussion will briefly swing through the field of economics, as the two topics are integral to it and the dismal science provides at least a semblance of formalism to their consideration. We will then properly define *authorship* and *publishership*. The phrase whose definition will then conclude this subsection will be *entity authentication scheme* itself, a simple task when armed with the definitions and discussion preceding our arrival to it.

Entity, Identity, and Credential

The distinction between entity and identity which we will argue may at first glance seem obvious, meaningless, pedantic, or some combination thereof. However, we hold that it is actually an important and in ways complex distinction. This does not, however, prevent a straightforward initial statement. Simply put, *an entity is an actor and an identity is how other entities in a system refer to that entity.* To provide clarity, we introduce the following definitions.

Definition 2.2 (Entity)**.** An entity is a collection - which may be singular - of actors who may reasonably be regarded as operating with united intent and agency.

Definition 2.3 (Identity)**.** An identity is a set of information (identity set) which is not a subset of any other identity set.

Note that our definition for identity provides for a formal approach to its consideration and usage, which may be unexpected from our introductory discussion. However, that the construction of an identity may be made rigorous does not allow the process of its recognition to be made so as well. That is to say that although what an identity *may be* can be formalized, what an identity *may mean* cannot. For the definition of entity the use of the qualified subjective "reasonably be regarded" is necessary, as although in many contexts the composition of entities is well-defined this is not true in the general case - in either direction. As examples a legal corporation which is formally considered united may have internal divisions which operate independently and possibly at odds, and an amorphous grassroots organization with no formal structure or spokespeople may wish to communicate as a united front.

The requirement that an identity set not be a subset of any other identity set necessitates an explanation. Entities are often composed of discrete but united actors. It may be that some of those individuals wish to authenticate as a subset of an identity of the entity - such as a committee member wishing to authenticate as such. If so, the derivative identity proven by the individual must be made distinct from its parent, as else it would be possible for fraudulent authentication as the latter by the holder of the former. Continuing the committee example, an administrative assistant responsible for authenticating documents on behalf of the whole body might be able to prove they actually sit on it. This exposure can occur in systems where only the necessary part of an identity is authenticated or where a poorly designed implementation greedily accepts proofs of identification. As perhaps a clearer example, suppose in a system there are two distinct entities with identities

$$I_A = ('Alice', 'New York,\ NY', 'Cryptographer', 'Professor')$$

and

$$I_B = ('Alice', 'New York,\ NY', 'Cryptographer').$$

In such a case, it may very well be possible for the holder of I_A to successfully authenticate as I_B, since they can prove they are every component of it. Although careful design and implementation of schemes can mitigate this issue, in the general case it is simpler to exclude contained identities from the definition.

Perhaps it remains confusing what an entity and an identity look like - exactly how they differ in practice. We have not yet provided any examples as doing so is non-trivial, for if any name for an entity is an identity, then how may we refer to the entity without viewing it through the lens of an identity? Our solution is that as entities are defined by their agency and intent, we may discuss them independent of an identity by representing them as their roles and actions within a system. For instance, in the phrase “John Smith logged into his computer and sent an email”, the entity is referenced by “logged into his computer and sent an email”, while “John Smith” is its identity. Replacing “John Smith” by a role such as “sender” would provide a more concise reference for the entity. Although one may argue for a stronger sense of self then this provides, it is demanded because of the role of the identity as a means of recognition *by other entities.* From the perspective of the recipient of the email, all they immediately know about the entity who sent it is that they sent it.

This raises an interesting question. In a cryptographic paper which uses Alice and Bob allegories, are “Alice” and “Bob” identities or entities? If we call Alice and Bob the prover and verifier respectively then they are entities, and the use of the names is for the purpose of readability. As in our experience this is the common usage, we believe that in most cases reference to Alice and Bob is intended for them to represent entities. However, as they appear to be identities their use makes establishing eponymity seem vacuous, requiring awkward language such as stating the goal of a scheme is to “prove that Alice is in fact Alice” or similar. Further, conflating the entity and identity reinforces the notion that entities in a system have a “right” identity, which obfuscates the various forms and purposes for which identities are used. It may also beget implementation difficulties, as the information used to prove a connection should be the only accepted link between an entity and identity, but the “self-evidentness” of the connection might seemingly justify the use of imperfect methods for handling circumstances where entities lose or expose their proving information.

Another detail which we must consider is, if identities represent entities, then may those representations be one-to-one, one-to-many, or many-to-many? A superficial consideration of this query yields an apparent paradox. Although there is no reason why an entity may not have multiple identities, for an identity to be of use for establishing eponymity then clearly it must uniquely represent an entity. However, conflicting names is a common occurrence in practice. The solution is hinted at by the terminology from the preceding sentences - although names frequently overlap, identities must be unique. Names are a ubiquitous part of identities, a shorthand that may be used for quick reference. However, they often conflict, and when they do the remaining elements of the identity set must differentiate them. For instance, there might be two individuals both with the name “Jane Anne Smith”, but other information from their identities - such as age, birthdate, address, occupation, etc. - must separate them. Therefore, although names may not be unique, identities must be. As such connections *from entity to identity* may be one-to-one or one-to-many, but connections *from identity to entity* must be one-to-one.

There is one final concern of a definitional nature raised by our requirement for uniqueness which must be noted. As we shall see many entity authentication approaches employ attribute-based authentication - especially for systems which demand privacy protection or granular access control - in which a prover establishes eponymity not with the entirety of their identity but rather only a necessary subset. For instance, if from before the holder of I_A wants to join a mailing list for professors in New York City, the derived identity under which they would construct their message containing a request to join would be

$I'_A = ('New\,York,\, NY',\, 'Professor')$. With these approaches an authority still certifies the entirety of I_A, and the prover executes the guarantee of authorship using their private information relating to I_A to prove their connection to I'_A. However, the identity known to the verifier is only the subset which will often not be unique within the system. So far, all of our definitions and discussion has assumed perfect knowledge of identity information by all of the participants in the scheme. Although it would be a reasonable approach to provide a whole suite of alternative definitions to accommodate this circumstance where components of the identity are hidden from the verifier - especially in such cases where the derived identity is not unique - we will not do so. Ultimately, it is the unique identity I_A which is cryptographically proven. Any techniques used to obfuscate or simplify the exposed elements of it to the verifier do not alter this or lessen the guarantee of authorship provided.

Proceeding, a credential is a document used for asserting a connection to an identity. It contains the identity set, the information needed by the prover to assert their connection to it, and the information needed for other entities to assert the eponymity of that connection. A credential is a defined, practical document, and is the object which instantiates entities and identities within the functional apparatuses of a system.

Definition 2.4 (Credential)**.** A credential is an 4-tuple (Σ, Γ, A, E), where

1. Σ is a j-tuple containing the credential's identification and validity information.
2. Γ is a k-tuple containing the elements of the identity set the credential is asserting.
3. A is an m-tuple containing the necessary information for the prover to cryptographically assert a connection to the credential.
4. E is an n-tuple containing the necessary information for other entities to cryptographically assert the eponymity of the credential.

The validity information of a credential will be discussed to come. In practice, a credential document often contains only a reference to a location - such as a networked database - where Γ may be accessed. In that case, the various components of the lookup system must be considered as part of the implementation of the credential as well, as their proper functioning is required for its proper functioning.

Authentication

Our use of the term authentication will not differ greatly from its common usage. Authentication is a process wherein the authenticator - in our context the verifier - specifies a set of requirements which must be met before they'll consider the authenticatee - the prover - to *be something.* The breadth of the proof may vary, from establishing only what has been immediately shown to authenticating a kernel of a greater whole. Our proofs will be of the latter variety - the usefulness of entity authentication is that an accepted claim of eponymity confers recognition of the prover for all the attributes carried by the identity, so they do not need to individually establish whichever attributes they intend to employ.

Our primary concern then is how formally we intend to use the term authentication, as the requirements in such processes can take one of three dispositions. (1) They may be formal mathematical constructions, such as the requirement that the prover must produce a signature on a message with a specific key, (2) they may be arbitrary yet defined, such as requiring exactly three entities to attest to eponymity, or (3) they may be arbitrary and ill-defined, wherein the verifier sets an at least partially unenumerated collection of factors, some of which may be subjective. An example of this final form might be a bouncer deciding who may enter an exclusive establishment.

An important aspect of authentication which should be explicitly noted is that it is a process of reaching a *threshold*. Many sets of requirements for authentication processes are not straightforward lists of items to be established. Instead, they contain complicated logical structures in which certain subsets of the requirements are sufficient but not others. For instance, one might be required to provide a birth certificate *or* two other forms of government-issued identification. The threshold for authentication is any subset of the items that makes the logical statement defining the requirements true. We may now introduce the following definition.

Definition 2.5 (Authentication)**.** Authentication is a process wherein an entity, the authenticator, establishes a criterion for accepting whether another entity, the authenticatee, possesses an attribute. If the authenticatee can prove they satisfy the criterion by a cryptographically-verified process, then they successfully authenticate as possessing the attribute.

Trust and Reputation

We all have some vague understanding of what trust is, dating to our youth when we were dependent on some guardian figure to provide for our basic needs. Intrinsic to our social selves, we understand trust as putting our faith in other actors to do right by us when they may not be bound by any factor greater than the strength of our relationship to do so. This intuition is not sufficient to serve as a definition itself though, as it raises a number of questions. Does one need to have a predefined expectation of behavior to trust someone, or can the vague "do right by me" suffice? If an expectation is necessary, then is someone trusting anytime another entity is acting towards one of their expectations, or only when the action of the other entity would directly affect them? And if it does need to affect the trustor, does it need to be in a manner that would cause them measurable gain or loss?

Despite the lack of a singular construction, definitions of trust in both the literature and technical documentation have similar underpinnings. These definitions include at least one and up to all three of the concepts of risk, the yielding of agency, and expectation. When risk is included in the definition it requires the trustor to have a meaningful stake in whether the trust is respected. The inclusion of the yielding of agency means the trustor cannot drive the outcome of the trust relationship, requiring trust to be held between two independent parties. Expectation demands the trustor has a specific outcome they wish to gain, instead of a disposition of arbitrary form. Requiring it recognizes that a positive gain can still violate trust, while not requiring it argues - perhaps via ignorance - the trustor hasn't been wronged unless they've been put at a loss. Although the explicit mention of risk and expectation varies, it is rare that a definition does not at least implicitly recognize the yielding of agency. Although in certain contexts excluding risk and expectation from the definition of trust has its uses, authorship is not one of these. For the former, cryptographic approaches are only demanded when risk is present, so its inclusion in the definition brings it inline with the context. As for the latter, the narrow scope of purpose for which we are trusting other entities - just to assert eponymity - allows us to articulate an expectation. It is not necessary for an appeal to vagueness. As such, we may provide a definition of trust for our purposes.

Definition 2.6 (Trust)**.** Trust occurs when an entity, the trustor, engages in a transaction in which they assume risk but yield agency - while retaining an expectation - over the process that will determine the return on that risk to another party, the trustee. If the result of the transaction fulfills that expectation then the trust placed by the trustor in the trustee has been justified, else it has been broken.

For our context, the trust relationship which will be of central focus will be that of the verifier acting as trustor and an attestor acting as trustee, with the expectation being that the trustee will only attest to an eponymity being claimed if it is valid. See [Gam88] for one of the most widely quoted perspectives and definitions of trust. In addition, [BDM95] is an oft-referenced experimental approach to evaluating trust which includes a concise discussion and definition, while [Cab05] provides the latter from within a mathematical economics context. Further definitions and discussion of trust are frequently found throughout economics literature, as well as in network theory papers and technical documentation.

A definition of trust does not provide a full understanding of its operation. In addition to answering what trust is, we must also understand why and whom actors choose to trust. As we shall see, the considerations that go into these decisions determine the effectiveness of systems which rely on trust relationships, and therefore will be influential to our judgment of how entity authentication schemes provide a guarantee of authorship. We will approach these decisions through the lens of reputation. A reputation is an accounting of the previous relevant actions of an entity which provides an indication as to the likelihood trust placed in them will be justified. Often loosely defined as a measure of that likelihood, it is the primary input into any reasoning by the evaluator as to whether to enter into a trust relationship. Unfortunately, although reputation may be something akin to a measure, it is difficult to argue that it may be well-defined. In certain restricted contexts - where the actions of each entity are formalized and risk is a constant - it may be reasonable to view a reputation as a simple historical ratio of *justified trust* : *has been trusted*. Applied to authorship, if the identity whose reputation is being evaluated exists solely for the purpose of its role in the entity authentication scheme then such an approach may be reasonable. However, if it has a purpose - and incentives aligned - outside just the proper execution of the scheme then this approach is too simplistic.

An important comment must be made. Even though entities trust other entities - as it is the entity who has the agency and capability to complete the transaction - reputation is an attribute of identities. Since identities are how entities know each other, they evaluate an entity they are considering trusting from the perspective of that identifier. An entity that has multiple identities can therefore have multiple reputations for the same task under those different identities. Proceeding, the definition we will use for reputation is as follows.

Definition 2.7 (Reputation)**.** The reputation of an identity is an evaluator's perception of the likelihood the entity eponymous with the identity will justify trust placed in it.

For a widely cited general discussion on reputation, see [Das88]. A mathematical approach in the same vein as the discussion of trust is given in [Cab05], while [MMH02] provides a good example - as well as a substantial literature overview - for how network theory deals with the notion.

One distinction - which we will not dwell on as its relevance to our cryptographic context is limited - is that our definitions of trust and reputation have not excluded an actor trusting another to act negatively towards them. Although such a transaction contradicts an intuitive notion of trust, it does not violate the letter of our definitions. As such, we must discern not between two states, but three. An entity may be in a state of trust with another entity - expecting them to fulfill a positive expectation - a state of distrust - of expecting them to fulfill a negative expectation - or in a state of incertitude - of not having sufficient information or having sufficiently contradictory information regarding the reputation of their identity to be able to adjudge whether they are trustworthy. For further discussion of this concept, see [Das88].

While reputation is the primary guideline for whether a rational actor is willing to trust, it is neither singularly responsible nor itself divorced from its employment. Towards this latter point an important consideration which heavily modifies the assessment of reputation is the context of the possible transaction the actor is evaluating. This alteration comes by way of relevancy - if reputation is determined on the basis of past actions, naturally those actions which most closely align with the situation at hand should carry the greatest weight. Although a general sense of trustworthiness is not necessarily a misguided notion - in particular actions which have no direct similarity may require similar traits underpinning them - a general theory of trust would be remiss to reduce a reputation to a single assessment independent of context. However, whether communications systems are sufficiently contained to where it is reasonable to assess a universal reputation within them is not obvious. In fact, it is hard to argue that this is not context-dependent itself, based on the identity in question. Similar to before, if the identity exists solely for the purpose of the entity authentication scheme then an assessed universal reputation might be accurate. However, if it has a purpose beyond the scheme then context-sensitive reputation evaluation is necessary.

In addition to altering the assessment of reputation, the decision by an actor of whether to trust is itself directly affected by context. If the transaction carries low risk and high reward an entity will likely consider accepting a lower reputation for a trustee then if the *risk* : *reward* ratio is reversed. This is discussed under the label *interest* in [Gam88]. If one has little to lose a loss may result in little more than a shrug. If the opposite is true, then it may end in their destruction. A similar concept which affects the decision to trust is *tension*, a dissonance that can occur when an actor thinks they should not trust, yet also feel they must. Taken to its extreme, tension causes *blind trust*, also known as a *trust of passion*. Blind trust can take the form of both trust of necessity as well as trust of attachment, yet in either case it is not a trust of judgment. These two concepts are discussed in-depth in [Gam88], and the latter is the focus of [Dun88]. The existence of tension and trust as a passion poses a pair of questions of subtle importance. Are these rational actions? And if not, then do rational actors ever really trust?

One perspective - espoused in [Wil93] - argues no to both of these, that tension precludes rational trust.

Instead, rational actors act on the basis of *competent calculativeness*, only entering into the relationship we have defined as trust if they are able to completely determine their likely return based on a probabilistic analysis. Although the details of the argument are beyond the scope of this discussion - in particular requiring context of various views on rationality in economics - we consider calculativeness versus trust not as a conflict, but as a continuum. On one end is blind trust, where an actor chooses to trust either irrationally or without any information. On the other is calculativeness, where a rational actor with perfect information enters into a transaction only if their probable return outweighs the risk. In between is where most decisions lie - rational to an extent and entered into with imperfect information. The relevance of interest and tension - and the resultant notion of a calculativeness continuum - to the problem of authorship lies in the goal of entity authentication schemes to prove eponymity to an extent *sufficient for all interested entities.* Where on that continuum an entity is comfortable operating dictates what volume of assertions they will demand, which then dictates the authentication requirements placed on the scheme. Further, to simplify the user experience many entity authentication approaches - such as one we will evaluate, the web browser model - attempt to hide the trust relationships necessary for its proper functioning, and in doing so move the needle on the continuum away from calculativeness.

In addition to particular schemes obfuscating trust relationships amongst users, theoretical cryptographers often attempt to mitigate the direct exposure of their constructions to trust and reputation. They do this by building cryptographic constructions and infrastructure which attempt to *economize on trust*, or treat trust as a scarce resource. This is done by minimizing the trust necessary for the operation of the system and omitting structures which would further its development through incentivization or establishment of reputation. Clearly, more prevalent trust relationships in a system ameliorates much difficulty as entities can more easily find attesting parties. However, the commitment to purity means cryptographic constructions are often designed to operate on a minimum level of systemic trust - beyond which is considered redundant - and do nothing to build it within the system. The general philosophy of this approach is not limited to cryptography, as it "is not just *a* solution: it is possibly *the* standard solution, which, filtered through Machiavelli, Hobbes, Hume, and Smith, has been handed down to the present day as the most realistic, economical, and viable" [Gam88].

A related concept is that of a *lemons market* - wherein bad quality actors drive out good quality actors. Information asymmetry is the mechanism, preventing entities from accurately determining reputations and thereby reducing the likelihood they will enter into trust relationships [Ake70]. As such, unless a system is completely open about the histories of entities for the establishment of reputation - which may be of concern with regards to privacy - entity authentication approaches struggle to function in practice unless mechanisms to punish malicious or negligent entities are built into the system. Certainly some of the difficulties raised by economization of trust and the lemons market are implementation-focused, but many of those difficulties start or could be mitigated at the theoretical level. These difficulties arise either implicitly and explicitly in most of the constructions to which this framework will be applied.

Lastly, it is worth delving slightly deeper into the notion of modeling or quantifying trust and reputation, since such attempts can give a specter of mathematical legitimacy to formalizing them. Much recent output in economics and network theory attempts to do so in one manner or another, with the following references a reasonable tour [MMH02, GKRT04, SYHL06, TB06, ZL05, KMRS09]. Although the term *trust propagation* may be more commonly used, what is actually occurring is *reputation propagation.* An entity stakes their reputation by attesting on the capability of another. But what exactly is being staked? We trust different actors for different things. We trust actors for their skill, their integrity, their commitment, their ability to judge others' skill, others' integrity, others' commitment, the ability of the other to judge the ability of a third party to judge those characteristics, and so forth. Most of the papers referenced above devise a model for reputation propagation which treat it as a single value. Even in situations where we may only be interested in a sole attestment from a network - such as to the eponymity of an identity - at each step in the reputation propagation different quantities are being measured. Attempting to use a mathematical method to determine the output of this process greatly misrepresents the actual mechanisms.

Authorship and Publishership

With all of the preceding discussion and definitions, we may now define our two guarantees which establish the purpose of authentication schemes. We begin with publishership.

Definition 2.8 (Publishership)**.** A cryptographic construction provides a guarantee of publishership for a communication system if

1. an entity can prove to any other entity that it constructed a given message;
2. the proving entity can claim it is eponymous with the identity credited as having authored the message;
3. and entities may formulate attestations that the claim is commonly recognized.

Definition 2.9 (Authorship)**.** A cryptographic construction provides a guarantee of authorship for a communication system if

1. it provides publishership;
2. and a mechanism by which entities may evaluate the reputation of the attestors and determine whether to trust their assertions.

Since the proof of connection between the entity and identity as well as the attestments by other entities may each be treated as mathematical processes, schemes which provide publishership may be well-defined and formalized. General theories of publishership have been of great interest in the cryptographic community, with many constructions of formal logics or grammars for publishership systems, such as in [Mau96, KM00, BFL96, BFK99, BAN89, AT91]. As authorship may not be made rigorous as publishership can, a reasonable question is whether the term "guarantee" should be used for both, as it is a weaker statement for authorship. Our decision to not introduce a new word is due to not wishing to further burden the reader with additional alternative definitions from the common usage, and also due to the difference between the guarantees of authorship and publishership being carried by those words themselves. That said, *assurance* would be a reasonable phrasing for what authorship provides for those wishing to further distinguish between the two.

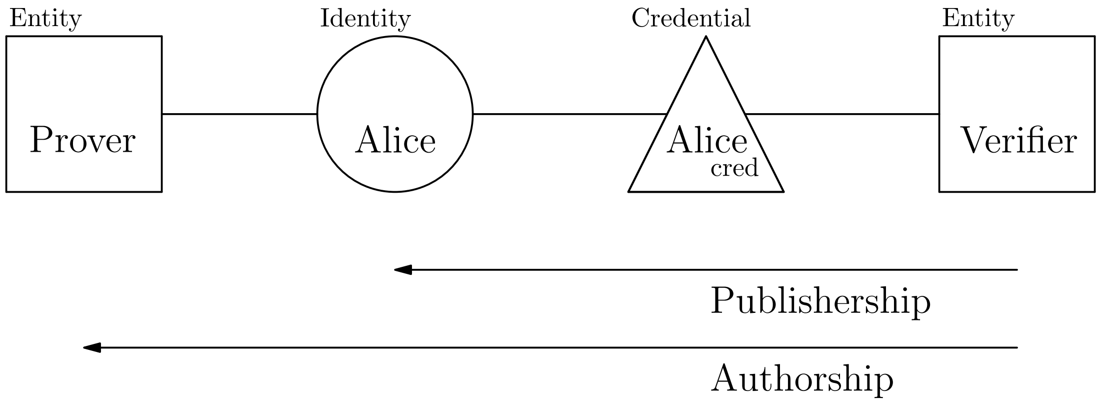


Figure 1: Authorship vs. Publishership.

That schemes which provide publishership may be formalized is the root of its usefulness for the practice of cryptography. It is sufficient to show how entity authentication schemes work, even if it cannot lay claim to showing why they work, and it avoids tricky concepts such as trust and reputation from entering into the discourse. However, its narrowness ultimately renders it incapable of fulfilling the purpose for which we have defined it. Our goal is to establish who the author of a message in a communication system is. Although publishership provides a guarantee that entities in the system may name each other, that is not sufficient. A name or identity may be wrong, misleading, incomplete, or fraudulent, and a guarantee of authorship is demanded to eliminate those possibilities and provide confidence in the eyes of the verifier.

Entity Authentication Scheme

We now conclude this subsection by defining the term entity authentication scheme itself, as well as some variants. First, we will simultaneously define a lesser form of authentication scheme, an *identity authentication scheme*. In an entity authentication scheme it is the actual entity which is authenticated, whereas in an identity authentication scheme only the identity is - an entity authentication scheme provides authorship, an identity authentication scheme only publishership.

Definition 2.10 (Identity Authentication Scheme)**.** An identity authentication scheme (IAS) asserts that the entity holding some private information is connected to an identity in a cryptographically-verifiable manner.

Definition 2.11 (Entity Authentication Scheme)**.** An entity authentication scheme (EAS) asserts that the entity holding some private information is eponymous to an identity in a cryptographically-verifiable manner.

In addition, we will provide a version of this latter definition for use with schemes which rely on trusted third party certifiers to provide authentication. They are the most studied approaches, and a quick way to differentiate them and delineate their attestment requirements will be of use.

Definition 2.12 (Certification-based Entity Authentication Scheme)**.** A k-certification-based entity authentication scheme (k-CBEAS) asserts on the reputation of at minimum k-trusted certifiers for some $k \in \mathbb{N}$ that the entity holding some private information is eponymous to an identity in a cryptographically-verifiable manner.

2.2 Framework Considerations

While the definitions given in the preceding subsection will act as points of reference for our consideration of entity authentication schemes, they in and of themselves do not provide assistance for the analysis beyond classification. Rather, if a scheme aligns with the goals of the system in three primary realms, and if the roles in the scheme properly mirror the roles in the system, then it is a suitable choice for that setting. Of these realms perhaps the greatest importance lies within privacy, to consider which requires background discussion on what forms identities may take. Next is statefulness, as names and attributes frequently change or become obsolete in communication systems, and schemes must have a reasonable approach to accommodating those changes. Lastly, fraud and uncertainty cover how the scheme manages discovery and rectification in situations where malicious action occurs. All three of these discussions will be reasonably terse. Volumes could be filled with their intricacies, yet our purpose is to provide a concise overview that provides a baseline for consideration of how schemes deal with them.

Privacy

As identities are how entities are known their creation and use within a system determines it capability to provide privacy for its users. Although we will not provide a formal definition, a privacy protecting scheme allows entities to undertake an action without any observing entity - from within or without the system - able to determine who was responsible. We will leave "able to determine" as a vague statement, but that the observer cannot prove who undertook the action is insufficient - their observations must provide no evidence as to which entity acted - privacy is a stronger demand then deniability. Similar to reputation, although privacy is the protection of entities it is discussed in terms of identities. Observers are always able to attribute actions to an identity, even if that attribution is self-referential.

One requirement of privacy protecting entity authentication schemes is that they must allow entities the freedom to use multiple identities - including creating them - at will. In the extreme, if an entity is forced to use a single identity then there is no mechanism they may use to hide their behavior, as any observer may link their actions together. Extending this reasoning, it is not difficult to see that any constraint on the use of multiple identities allows at least partial tracking. Although the ability to act under multiple identities is a necessary condition for privacy, it is not sufficient. The broader guarantee of *unlinkability* is required. A scheme provides unlinkability if an entity is able to decouple actions such that an observer cannot determine they were made by the same entity. Considerable discussion of the topic is given in [Bra00], and a formal definition of unlinkability - phrased in terms of indistinguishability - and of many other related and useful terms is given in [PK01].

Although unlinkability is necessary for privacy protection it is not its whole. We cannot ignore perhaps the most effective approach to violating privacy, inspection of information produced by implementations.

For example, an observer may use the geographic location of a terminal to link actions undertaken with decoupled identities. Both of [DW06, ADS03] contain discussion of the issues in successfully using privacy protecting systems in practice. However, these concerns are beyond the scope of this discussion, and our focus will be on the privacy protecting capability of schemes in their abstract formulation.

One concern is how an identity may be constructed within the scheme. For our analysis we will use the classification given in [AL03] - though the specific interpretation of the terms is our own - differentiating between the following three forms.

Definition 2.13 (Anonym)**.** A single use identity.

Definition 2.14 (Pseudonym)**.** An identity internal to the communication system.

Definition 2.15 (Veronym)**.** An identity commonly recognized in multiple communication systems in which the entity participates.

We consider the lack of an identity to be a case of an anonym - with the implied identity "the unknown author of the specific message" - and so we have not introduced a *-nym* for it. For pseudonyms, by an internal identity we mean one developed for use within a system which has no or limited meaning outside it. By contrast, a veronym is commonly recognized across multiple communication systems. The standard example of a veronym is a legal name, yet this archetype may be constricting. The importance of a veronym to privacy is that its use allows tracking across various communication systems, often including the offline individual or collection of individuals. Any name sufficiently widely recognized allows for this cross domain tracking, and so must be considered a veronym. In [PK01] a useful distinction with regards to pseudonyms is made, as they distinguish between pseudonyms which are linked to a veronym publicly from creation, those which become public - either of which reduces their privacy usefulness to that of the veronym - and those which remain unattached.

Most systems have a natural choice for the naming requirements within them. For systems over which sensitive information is communicated and the actions of each entity must be linked back to an offline persona veronyms are usually employed - examples include banking and voting. Pseudonyms are ideal when entities wish to have a reputation - disconnected or not from a veronym - such as for message boards and email. Anonyms are the most privacy conscious choice which - ignoring implementation attacks - guarantee unlinkability. That different use cases demand different requirements underlies one of the central entity authentication philosophies - that entities should have full control and discretion over the identity information they expose. Deemed *selective disclosure*, during the authentication process an entity may disclose exactly what identifying information is needed - and no more - to the verifier. For example, a movie theater may sell discounted tickets to seniors and minors, with at present a photo ID - which includes a name, address, disabilities, and other irrelevant personal information - required to qualify for the discount. Under selective disclosure, an individual may purchase a ticket at a kiosk and prove that they qualify - i.e. are under eighteen or over sixty-five - without specifying which age bracket they fall into, let alone disclose additional information. Stefan Brands is one of the figures behind the development of selective disclosure, and his text [Bra00] contains an excellent discussion of the benefits of the approach.

The principle of selective disclosure was developed as a response to and extension of the approach espoused by David Chaum, casually in [Cha85] and formally in [Cha81]. Although commonly referred to by some derivative of the word "pseudonym", we will use Chaum's name to represent it since its central mechanism makes use of anonyms by our definitions. What Chaum and others extensively developed - and which underlies most all privacy protecting schemes, including selective disclosure methods - is *blinding.* When presenting credentials with certified information both the certifier and verifier are able to track interactions. To provide unlinkability, those entities must not be unable to collude. Blinding accomplishes this by decoupling the action of certifying the attribute and presenting the certification, so that the verifier can determine the authenticity of the certification but the certifier cannot determine whose certification is being evaluated. The verifier cannot track because the certifications are unique, while the certifier cannot track because the certification they are validating is unlinkable with its original construction. Unlike selective disclosure, in which blinding may be used in conjunction with different naming practices to give the prover control over the information being offered, Chaum's proposals adamantly advocate for excluding veronyms from any system in which even partial privacy is a goal.

Whether or not entity authentication schemes should allow for the use of veronyms is an important debate. Chaum and others have constructed privacy protecting approaches for purposes to which we have ascribed a need for veronyms - such as payments [Cha83] and voting [Cha81, CRS05] - while maintaining assurance of eponymity with a hidden veronym. In opposition, Brands argues in [Bra00] that implementation concerns render such approaches impractical. In his view allowing entities to control what information is exposed about them - allowing them to decide what systems should require what identification and giving them the option to disengage from systems which demand too much - is the only reasonable mechanism by which privacy can be protected, particularly for systems which interact with offline structures. The primary counterargument to this position is that allowing veronyms makes the exposure of personal information a negotiation - and verifiers will likely continue to demand unnecessary and invasive information. Many users will either consider the privacy concerns of little importance or be sufficiently inclined to continue interacting with the requesting entity. As such, they will provide the demanded information, maintaining the *status quo* and allowing the verifiers to disregard the small population who advocates for active use of privacy protecting mechanisms. Only by preventing verifiers from demanding veronyms by preventing the capability for their expression within the system can such a situation be avoided. We will not attempt to argue for either perspective since their respective merits have mostly to do with the alteration of social mindsets regarding what information should be privileged. Note as well that the labels "selective disclosure" and "pseudonym" simplify and conflate various allied positions which do not completely agree with either of those we have just presented.

Statefulness

How and for what entities are known changes - names are altered, social hierarchies are reworked, and new attributes are attained while old ones are altered or cease to be. Eponymity is not inherent - an identity does not represent an entity, rather it represents an entity to a certain group at a certain time. Statefulness is therefore demanded of entity authentication schemes. Credentials must expire or be revocable when the identity they enclose no longer accurately represents the entity for whose use they are engaged. As we shall see, efficiency and effectiveness is a trade off for mechanisms of statefulness, and so the system to which it will be applied determines which approach a scheme should take.

Why identifying information becomes invalid is not much in need of discussion. Although in many systems names change rarely this is not universal, and attributes are often changing or expiring. There are two different approaches to handle state for entity authentication. The scheme may be interactive - where the attestor and verifier actively communicate during the process - or non-interactive - where the attestor produces an attesting document which is often part of the credential itself and publishes the information necessary for the verifier to validate the document at will. In the interactive model if a credential is no longer valid due to a change to the identity then the attestor simply stops actively attesting for its validity. The non-interactive model is trickier.

There are two different methods that a non-interactive scheme may use to incorporate state. The first is expiration, where a credential no longer validates when it is used over a certain limit of presentations - in which case it is known as a *limited-show* or *k-show* credential - or past a certain point in time. The second is revocation, where the attestor publishes a document disowning their claim of eponymity. Expiration is effective only in certain contexts. Limited-show credentials work for transportation or cafeteria tickets, but many identities do not expire based on frequency of use - one would not want their passport to be denied because it'd been used on too many occasions. Time expiration works for a non-revocable right that may still be reevaluated and reissued periodically, but it is not sufficient unless there is no chance that the validity of an identity will change within its time frame. In most use cases no such guarantee exists.

Revocation has its own issues. Most importantly it requires the verifier to promptly and correctly receive revocation notices. Further, there is a temporal gap between when an attestor learns they must revoke a claim of eponymity and when the revocation notices can be properly constructed and distributed. As such, revocation-based systems may be exploited, although how effectively is dependent on the specifics of the implementation. Most non-interactive schemes use a combination of expiration and revocation - the former allowing for periodic reevaluation and monitoring, the latter for claims to be rescinded if necessary in between expirations. These difficulties lessen the advantages of a non-interactive approach, which lie in it being more efficient than the interactive model - which requires additional communication between the parties as well as the constant availability of the attestor. Whether interactivity or non-interactivity is superior is dependent on the system in question - whether efficiency or effectiveness is more important - and is also highly dependent on implementation details. As such - like with privacy - we cannot pass judgment on which is universally the superior approach. For reference, [Riv98, MR01] contain a direct argument about the issues and benefits of revocation approaches, as well as comparison to other techniques including interactive models.

Fraud and Uncertainty

That non-interactive revocation may permit malicious action does not set it alone. Protecting against fraud and enabling sanction against its perpetrators is a necessary component of entity authentication schemes.

Another way of expressing the central concern of this discussion - the inability to make entity and identity rigorous - is that we are unable to excise uncertainty from the operation of communication systems because of the social structure that underlies them. Trust is a means of dealing with that uncertainty - it is often phrased as a means of managing exposure to the freedom of others [Gam88]. However trust may be misplaced. Then, opportunities for fraudulent action become number, and the best a scheme may do is minimize those opportunities and provide means by which their exploiters may be punished.

There are two different classes of fraudulent actions we must consider. An attacker may exploit the entity authentication scheme or they may use the entity authentication of the system for malicious action elsewhere in the system. An example of the former is an attacker convincing a certifier that he is Alice when the connection is not commonly recognized or the attacker forging an attestment statement supposedly from a trusted certifier. The latter might be the the attacker using a pseudonym procured properly and unlinked to a veronym for the purpose of purchasing an illegal good. That attackers cannot attack the scheme itself is a necessary assurance. However, the demands of the underlying system will dictate how important mitigation of the latter form of attack is. Specifically, it will be the importance of privacy which dictates its relevance. A system committed to total privacy might have no qualm with its use for malicious purposes, or at least might believe that the moral necessity of protecting privacy requires it to protect even those who use it for illegal or morally questionable actions.

3 Cryptographic Primitives

We will now quickly overview some of the cryptographic primitives used in the construction of entity authentication schemes. First we will look at digital signatures and use them to give perhaps the archetypal example of a k-CBEAS. Then we will briefly discuss the high-level construction of a zero-knowledge proof. For more, we refer to some standard texts in the field [LK15, Sti05, Gol06].

3.1 Digital Signatures

A digital signature scheme is an asymmetric key construction composed of three algorithms, `keygen`, `sign`, and `verify`.

Algorithm 3.1 (`keygen`). A key pair, composed of a public key k_{pub} and private key k_{pri}, are generated by a scheme specific method. The output is $(k_{pub},\ k_{pri})$.

Algorithm 3.2 (`sign`). Input is $(m,\ k_{pri})$, where m is the message to be signed. The signer computes $s = \texttt{comsig}(m,\ k_{pri})$ by calling the subalgorithm `comsig` dependent on the specific scheme in use. They then output s.

Algorithm 3.3 (`verify`). Input is $(m,\ s,\ k_{pub})$. The verifier computes $m' = \texttt{comver}(s,\ k_{pub})$ by calling the subalgorithm `comver` dependent on the specific scheme in use. If $m' = m$, the verifier *accepts*. If $m' \neq m$, they *reject*.

As noted, k_{pub} and k_{pri} are a bound public key and private key respectively, the parameter m is the payload to be signed - in most cases it is a digest of the message, both for efficiency and also because $|m|$ may be bounded by the key length - and s is the signature itself. The operation of signature schemes is straightforward. First, the sender runs `keygen` and gives k_{pub} to the receiver while keeping k_{pri} private. The sender then signs a message M by $\texttt{sign}(m,\, k_{pri})$ where $m = M$ or $m = \mathit{digest}(M)$, and sends $(M,\, s)$ to the receiver. The receiver then verifies the signature by running $\texttt{verify}(m,\, k_{pub})$, including computing the digest if necessary. If `verify` outputs *accept* then the receiver has proof that the holder of k_{pri} sent the message. To provide this assurance, signature schemes require a guarantee of protection against forgeries, that an attacker who does not know k_{pri} cannot construct an s which k_{pub} will accept on input of. Combined with a guarantee the verifier cannot use the information presented to them to determine k_{pri}, all three of our requirements for cryptographically establishing publishership - given back in the introduction to *Section 2* - are provided.

It is easy to see how signature schemes may be extended to entity authentication. If the receiver is confident that the holder of a key pair is a specific entity then they may conclude the message originated with that entity. The archetypal entity authentication scheme is such an approach. A non-interactive 1-CBEAS, it requires a certifier - which we will call a *trusted authority* (TA) - that is trusted by both the prover and the verifier.

Protocol 3.4 (Non-Interactive Authority-based 1-CBEAS)**.** Let the certifier with identity TA be a trusted third party for all members of the system.

1. The certifier uses `keygen` to generate a key pair TA_{pub} and TA_{pri}.

2. The prover uses `keygen` to generate a key pair id_{pub} and id_{pri}.

3. The prover sends id_{pub} to the certifier, the latter using some method to confirm they are communicating with the entity commonly recognized by id.

4. The certifier constructs a certified credential - also known as a certificate - as follows. First, they compute $a = \texttt{sign}(id_{pub},\, TA_{pri})$. Then, referring to *Definition 2.4*, they define $C = (\Sigma,\, \Gamma,\, A,\, E)$ by

   - $\Sigma = (\mathit{serial},\, \mathit{expiration})$,
   - $\Gamma = (id)$,
   - $A = (id_{pub})$,
   - and $E = (a)$.

   In Σ *serial* is an identification number for the credential and *expiration* is information pertaining to the validity of the credential. The certifier then sends the credential to the prover.

5. The prover computes $s = \texttt{sign}(m,\, id_{pri})$ and sends $(C,\, M,\, s)$ to the verifier.

6. The verifier computes $a' = \texttt{verify}(id_{pub},\, TA_{pub})$. If $a' = a$, then the verifier proceeds. If $a' \neq a$, then the verifier rejects the eponymity of the key pair.

7. The verifier computes $s' = \texttt{verify}(m, id_{pub})$. If $s' = s$, then the verifier accepts the message was constructed by the entity known as id, if $s' \neq s$, then the verifier rejects that the message was constructed by that entity.

The need for trust is apparent in the phrase "using some method to confirm they are communicating with the entity commonly recognized as *id*". For the eponymity to be valid, the verifier must trust that the certifier only constructs the certified credential if they properly execute this action.

As for our considerations, the privacy of this scheme is dependent on what form of identity the prover is establishing eponymity with, and therefore what information is necessary for the certifier to accept they are communicating with the proper entity. The certifier may collude with the verifiers to track the actions of the prover, but only up to linking those actions with the dossier they generate during the credentialing process. For statefulness, the scheme uses a non-interactive expiration-based approach, and most such schemes use a *certificate revocation list* (CRL) to revoke certified credentials if necessary before they expire. As for fraud, the certifier may report their knowledge of the prover contained in their dossier - a good example of the connection between privacy and uncertainty. If they know a veronym for the prover then they likely will be able to punish them, whereas if they were asserting a pseudonym they might know substantially less and have fewer options for rectifying the situation beyond revoking the credential.

3.2 Zero-Knowledge Proofs

In addition to digital signature schemes, zero-knowledge proofs provide a general approach to entity authentication. Whereas a signature approach specifically employs public key cryptography, with a ZKP the underlying cryptographic approaches may be left ambiguous. Since various systems may be more or less amenable to various cryptographic methods, this allows for a broader set of systems for which entity authentication schemes may be designed. A zero-knowledge proof is a cryptographic process in which the prover establishes knowledge of some information in a manner which contains the following guarantees.

1. Soundness: The verifier will not believe the prover has the information if they do not.

2. Completeness: The verifier will not reject the prover as not having the information if they do.

3. Zero-Knowledge: The verifier gains no information regarding the information being proven.

In a ZKP, the verifier constructs a set of challenges for which the probability of successfully answering them by random guess is distinguishable from the probability of successfully answering them if given access to the information being proven. If the prover successfully answers them at a rate commensurate with the latter, then their knowledge of the information is proven. Guarantee of the principle of zero-knowledge is provided by showing that the verifier can simulate the role of the prover without the information, and therefore gains nothing from the interaction with the actual prover. Although zero-knowledge proofs of identity are a notable concept in theory, they are rarely used in practice. This is primarily due to the ease of implementation and efficiency of digital signature approaches, in particular non-interactivity.

4 Authority-based Approaches

Most of the schemes which we will analyze make use of at least one *trusted third party* (TTP) as an attestor. In the example in the preceding section we introduced a trusted authority. A TA is a specific type of TTP who holds a central role within the system. Either every entity or a large subset of entities in the system trust the TA for providing authentication attestments, and they are directly incentivized not to violate that trust - for instance as commercial organizations whose revenues depend on their reputation. In the following section we will see examples of schemes which use either TTPs which are not TAs or include no third party attestation whatsoever. In this section, however, we are concerned with a collection of approaches which use the authority model.

We will look first at an "idealized" construction of a *public-key infrastructure* (PKI). In the preceding section, we discussed how extending a signature scheme to an authentication scheme required establishing a mechanism to assert ownership of keys. A PKI is both a social framework and a developed formal construction to provide authorship using asymmetric key cryptography. After we will turn to the web browser model - the public-key infrastructure used on the internet - which deviates from the idealized conception. It will be followed by identity-based approaches - in which identities themselves are used as keys - followed by U-Prove, a construction for providing selective disclosure with an authority-based PKI. We will then conclude with Kerberos, an example of an authority-based scheme which is not a PKI.

4.1 Idealized Public-Key Infrastructure

A public-key infrastructure is any system which employs public-key cryptography to provide third party attestation of the ownership of public keys for their use with entity authentication. PKIs come in a great variety of forms, yet most texts which introduce them begin with an idealized conception containing a few simple models, for example [AL03, Bra97]. Since these constructions act as a platonic ideal for a PKI - and do have some applicability - they are a logical place to begin our survey. As with all of the discussion to come, we will split our analysis into three parts. We will give a brief overview of the construction, discuss it from the perspective of our definitions, and then apply our considerations.

Construction

Since we are looking at a family of approaches this discussion will be somewhat imprecise. Idealized models use networks of one or more authorities connected by trust relationships. Entities in the system directly trust some subset of the TAs, and they will enter into a trust relationship for the purposes of attestation with any authority to which they can trace a trust chain through the network. The arrangement of the network can vary greatly, but usually involves some combination of hierarchical (vertical) relationships and horizontal ones - for more [AL03] contains a large section detailing different configurations. As an example, a verifier trusts an authority known as TA_{A4}, a subauthority to a root authority with identity TA_{A}. The latter has a horizontal trust relationship to another root authority TA_{B}, who has a grandchild subauthority TA_{B3ii} who will attest to a claim of eponymity. Since the verifier can trace a chain up from their local TA to its parent, across to the other root node, and down to the attesting TA, the entity trusts the attestation.

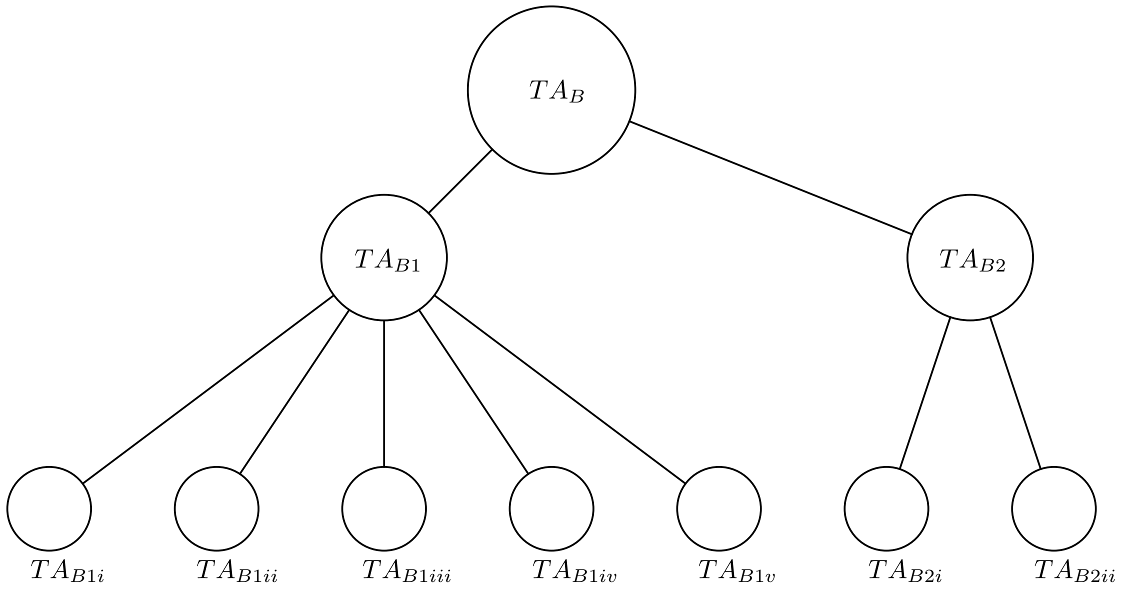


Figure 2: An archetypal example of a hierarchical idealized PKI.

Applying Framework Definitions

Idealized PKIs generally conceive of a TA as a unified body. However authorities commonly have three different roles, and it is simplistic to assume they will be sufficiently united so that the may be considered a single entity under our definition. (1) As the network scales the value of an authority is dependent on how dense their mutual trust networks are, as entities do not gain from entering into relationships with authorities which others do not trust. As such, an authority is incentivized to make its primary concern maintaining those relationships with other authorities. (2) The authority acts as a *registration authority* (RA), whereby they use various mechanisms to validate the eponymity of the claims to which they will attest. (3) The authority acts as a *certification authority* (CA), the body which - when the RA has properly verified a claim - constructs and issues the certified credential. In the idealized model these three components of the TA are often subsumed together, to which [ES00] contains a detailed discussion of the issues with this mentality. From our perspective, this is a prime example of how reputation can often need to be assessed for multiple purposes, even if the authority exists only for its operation within the system. For instance, if state is not particularly important in a scheme - such as for negotiations where the opportunity to verify the credential can occur at multiple times - we may be more likely to trust an authority whose CA role lacks in punctual updating of revocation lists but which is extremely capable as an RA.

Another issue lies in the opposite, not in conflating different entities but in differentiating where no divide may actually exist. We noted that oftentimes a hierarchical model with child TAs is used. Although the specifics vary based on the system and implementation in question, it can be the case that the same entity acts as the root TA and its children and grandchildren. In this case the hierarchy is just an organizational practice designed to simplify who is attesting for whom, both internally and externally. As such, a graph of the TAs in a network might show two hierarchies to be similar, when in fact one has properly independent children who should be considered separate entities, while for another the entire hierarchy should be considered a single unified actor.

Our abstract perspective restricts comment as to how reputation is assessed within such systems. It is dependent on whether an authority exists solely for the purpose of the system, whether they are a public service or a commercial entity, and so forth. However, we can make a comment on how reputation propagates. Most PKIs assume - as we have in our discussion of their construction - that reputation is propagated and trust is chained in a binary manner. Either two authorities are in a state of trust and attesting to the capability of each other or they are not. There is no mechanism by which reputation can propagate for only certain actions, and there is no general mechanism by which the entity using the trust chain may analyze any specific link beyond their own immediate one. Idealized PKIs put the burden of policing the network of authorities on the authorities themselves, with individual entities only able to calculatively approach whether to trust their immediate TAs. As such, these PKI constructions implicitly advocate towards blind trust, with many important relationships hidden within the chain.

Another way in which these constructions take a binary mentality is in the quantity of attestors which are necessary for asserting eponymity. Such schemes almost always take the form of a 1-CBEAS and so view authorities as inherently right, with a single attestment as valuable as arbitrarily many. This approach is common within discussion of authority-based schemes, since the infallibility of authorities is the assumption validating a restriction to publishership. As such, even though there is only benefit gained from allowing multiple attestations when it is reasonable from an efficiency perspective, no mechanism for doing so is usually included in schemes which employ idealized PKIs.

Applying Framework Considerations

Although again constrained by our broad perspective, a few small comments may be made. As authorities knowing and trusting each other is necessary - and as maintaining those relationships is of direct importance to the authorities - idealized PKIs are not designed in a manner which lends itself to privacy protection. It seems reasonable that TAs are likely to communicate frequently, and so the system itself places no barriers to tracking and communication of identifying information. As for statefulness, the longer the chain of trust the less likely it is that the system can be efficient in an interactive model. A hybrid model - where the chain of trust is not interactive but the actual assertion from the authority at its terminus is - is more reasonable, but still suffers the issues with non-interactive approaches, albeit in a more contained manner with authorities who lose substantially from having trust in them revoked. Lastly, the same connectedness of the system which hurts privacy also enables consistent communication regarding fraudulent activity amongst the authorities, though how a scheme handles fraud is specific to the implementation and authority network in question.

4.2 The Web Browser Model

We now turn to a specific PKI, the web browser model, used for verifying the eponymity between a web address and an entity. Although there is a natural predilection to viewing the prover as a supplicant, this intuition is not valid in the general case - with the web browser model an example. In it, the entity navigating to a specific server is the verifier, while the entity in control of the server is claiming eponymity, e.g. the entity in control of *bankofamerica.com* claiming they are known as Bank of America. Like idealized PKI models, a collection of trusted authorities - many of them hierarchically organized - may assert the validity of the claim. However, instead of a network of trust relationships between authorities providing for scale, the verifier trusts each authority directly. Although in the abstract this practice of reputation assessment and trust could be done by the verifier themselves, the model implies that the verifier puts their immediate trust in an intermediary party to track and maintain the list of authorities. Since this is the near universal use of the model in practice and greatly effects our evaluation of it, we will include the intermediary party in our discussion. Before proceeding, we note that much of the consideration of the idealized PKI setting applies to the web browser model as well, and we will not rehash it here.

Construction

The intermediary party is usually the developer of an operating system or browser. Lists of approved trusted authorities are bundled with them, and when an authority asserts on behalf of a server it is accepted if the authority is on that list. The structure of authorities is often hierarchical, although usually for organizational purposes. The root authority is directly included in the list, with subordinate ones tracing their lineage back to the root. Authorities themselves are therefore often larger organizations with sufficient footprint to warrant inclusion, and how they are viewed by the collection of intermediary parties determines whether they can function. Therefore the authorities are beholden first to those intermediary parties and to those who are trusting them to assert on their behalf second. The verifier puts their full faith in those intermediary parties - many entities likely do not even realize an authentication process is occurring. In that process, the entity navigates to the server which produces a certified credential with its public key signed by an authority. The intermediary party - on behalf of the entity - then checks their list to see if the authority is included, and if so attempts to verify the signature on the certificate and accepts the eponymity if it is validated.

Applying Framework Definitions

With regards to entities, identities, and credentials, the web browser model differs from the idealized approaches in that verifiers rarely if ever view the full identity of the server, instead relying on a singleton subset, its web address. Although the certified credential - at present standardized as X.509 in [CSF^{+}08] - may contain substantially more identifying information, entities consider the address to be the identity of the server, as we did with the Bank of America example. This partial approach is unsurprising when considered through the lens of how the web browser model manages trust. Ultimately, the model is designed to place nearly the entire verification burden on the intermediary party, who is expected to manage the process on behalf of the verifier.

The intermediary party decides the trust relationships in the system. They choose who the verifier should trust and therefore the reputation of authorities must be maintained in their eyes. Where the web browser model lies on the calculativeness versus blind trust spectrum is a difficult question. The argument that the web browser model encourages blind trust stems from the intermediary party having no direct risk in the authentication process. It is still the choice of the verifier whether to interact with the server and expose information to them, and the intermediary party can only provide certain technical support which may be ignored or missed. As such, most verifiers blindly trust the authentication process, and numerous attack vectors are enabled if the intermediary party is not prompt in updating their list of trusted authorities or if they are unable to convince the verifier to heed their warnings. The argument for calculativeness is that as the intermediary party is seen as carrying the responsibility for handling the process and protecting the verifier they are strongly incentivized to manage their role as effectively as possible - as opposed to in an idealized model where an authority responsible for an error might be just another link in a chain. As such, entities place their trust in an invested and capable intermediary capable of assisting them which greatly simplifies the process of authentication. We are inclined towards the former argument, albeit with a recognition that there may not be a better choice of approach with sufficient simplicity and capability to scale for common usage.

Applying Framework Considerations

The central role of the intermediary party gives them the opportunity to track the actions of the verifier. If commonly used software they may also facilitate actions external to the roles of the entity authentication scheme, with an extensive capability to track. However, to retain the trust of their users they are incentivized against this, or at least not to share information they gather. The authorities are similarly capable of tracking as in the idealized model, albeit without an already existent framework for communication. As for statefulness, the web browser model does not differ from the idealized PKI, although in practice the X.509 standard is a non-interactive expiration and revocation-based system. As for fraud prevention and uncertainty, that the intermediary party is incentivized to keep whatever information they may ascertain about the actions of the verifier private means that the situation does not greatly differ from the idealized PKI. However, the power an intermediary party has over whether the claim of eponymity will be widely accepted gives them the capacity to strongly respond to fraud by authorities if they are made aware of it.

4.3 Identity-based and Attribute-based Cryptography

The idealized PKI and web browser model are social hierarchies for PKIs, and many explicit schemes may be used in concert with them. The most commonly used are similar to the non-interactive 1-CBEAS we gave in *Section 3*, but we will look closely at some of the alternative approaches whose technical constructions meaningfully alter the mechanics of the PKI as a whole and so effect its applicability. Identity-based cryptography (IBC) and its derivative attribute-based cryptography (ABC) are examples of such. In particular they differ from the spirit - though not the letter - of a credential we have presented, and their mathematical construction initially demanded important concessions to privacy which later methods rectified in ways interesting to statefulness.

Construction

Identity-based cryptography was first developed for signature schemes [Sha85] before being extended to encryption [BF03]. Additional work was required to make it applicable to various forms of PKIs, such as for hierarchical models in [GS02]. An asymmetric key approach, it differs from what we have previously discussed in that the public key is the identity and optional additional information, such as that pertaining to expiration. As such, no formal credential document is required, instead the key itself carries most of the information which would be on it. What is missing is a mechanism by which an authority asserts the eponymity of the keys. In the original constructions, this was provided by the authorities acting as a *private key generator* (PKG). Instead of generating a certified credential for an existent key, the authorities generated the keys themselves. Further, the verification or decryption process required the input of their own public key. As such, if those processes completed successfully the verifier knew that the public key of the signer or encryptor was generated by the PKG, and - up to the trust the verifier had in the PKG - that they were eponymous with the identity that was their key. Although effective, this approach had consequences for privacy. Since the authority generated the keys - including the private key - they could not only track the key issuee if the social hierarchy of the PKI allowed it, they were further capable of impersonating them, signing documents in their name, and accessing their encrypted communications. Various approaches were developed to rectify this issue, including [ARP03] where the PKG and issuee each generate partial components of the private keys, and [Gen03] which reintroduces certificates in a manner which has ramifications for statefulness.

Developed using similar mathematical techniques to identity-based constructions, attribute-based cryptography provides for authentication of attributes in lieu of a full identity by allowing logical formulae of attribute thresholds to be built into the cryptographic processes. This allows a ciphertext to be generated which may be decrypted by only those entities with the correct set of attributes encoded into the identities which are their keys. Although initial constructions for attribute-based encryption such as in [GPSW06] required the authority to specify the logic into the keys themselves - requiring a different set of keys for any different set of authentication requirements - later methods rectified this by allowing attribute-based approaches more extensibility [BSW07].

Applying Framework Definitions

It is the concept of a credential which is most altered by IBC and ABC. In particular, in many technical approaches to a PKI - such as our archetypal example - the credential is explicitly constructed and used. Although a credential still exists in these constructions, it is never explicitly created because it is redundant. The key and the scheme specification contain all of the information necessary for its operation. Formally, the credential has the following construction.

- $\Sigma = (\sigma_1, \sigma_2, \cdots, \sigma_j)$,
- $\Gamma = (\gamma_1, \gamma_2, \cdots, \gamma_k)$,
- $A = (\text{Prover}_{pub} = \sigma_1 \,||\, \sigma_2 \,||\, \cdots \,||\, \sigma_j \,||\, \gamma_1 \,||\, \gamma_2 \,||\, \cdots \,||\, \gamma_k)$,
- and $E = (\text{TA}_{pub})$.

As A makes Σ and Γ redundant and the only element of E is the public key for an authority - which is likely known by the verifier or easily attainable for them - a separate document is superfluous.

The initial IBC constructions also had an effect on trust and reputation within the system, since the PKG role permitting the authority access to the information required much more substantial levels of trust.

Although later approaches helped rectify these concerns, a few interesting concepts may be raised in consideration. First, we are provided with another example of multiple reputations. An authority acting as a PKG in such a scheme would need to be evaluated both on their capabilities as an attestor and their prior known history - if any - of privacy violations. A classification towards this is the hierarchy of trust in authorities - introduced in [Gir91] - which argues there are three different levels of trust that an authority may require. The highest is necessary when - like in the original IBC constructions - the authority can impersonate an entity and access their private information. The middle level occurs when they may only impersonate the entity, and at the least level the authority cannot do even that, or at least cannot without the entity being able to prove disownership of the actions the authority makes in their name. Counterintuitively, in certain systems a higher level of demanded trust may be desired, such as in a work environment where a firm wishes to be able to act in their employees names and view the information they are working with.

Applying Framework Considerations

Not much remains to be spoken regarding privacy, especially as the concerns raised by the PKG role are allayed by the later constructions. Similarly, handling fraud and uncertainty is straightforward in the original approaches, and reduces to much of our previous discussion in the later ones. It is statefulness where one of the more interesting developments to come out of IBC lies. Most IBC and ABC schemes use a non-interactive expiration and revocation approach to statefulness, however Craig Gentry presents an interesting solution in [Gen03]. Its method for dealing with the PKG problem is to have the prover generate its own private key and have the authority generate both an identity-based public key and a signed certificate. Both the key pair and certificate are required for actions within the scheme, so the authority requires the least level of trust - as it does not have the private key - but the prover cannot forge the certificate and so cannot erroneously present their identity as having been certified by an authority. Moreover, since a valid certificate is necessary for the operation of the system it allows strict expiration and revocation rules to prevent malicious action - as opposed to other certified credential systems where an entity may choose to ignore an expired or revoked credential for convenience. Gentry provides a method for constructing a credential which requires common computational capacity - specifically noting that a personal computer can update billions of credentials in seconds - reducing the burden of reissuance to distribution. As such, he advocates for short-lived credentials that are constantly reissued, with the system able to respond interactively for any request for a newer certificate. Although to our knowledge no production system has been developed to test whether this scheme would be suitably efficient in practice, if so it does provide perhaps the best of all worlds when it comes to handling credential statefulness.

4.4 U-Prove

The goal of U-Prove is to provide selective disclosure to its users. Principled on allowing the authenticating entity to control all aspects of the proving transaction, privacy protection is its central focus. Although it employs a PKI, U-Prove includes a collection of interesting techniques which mitigate the tracking risk by authorities which have been of constant concern throughout our discussion of other PKIs. Further, its privacy focused approach requires the inclusion of a creative method to manage fraud and uncertainty. Its principles were developed by Brands in [Bra00] and standardized in [PZ13], while additional anonymous attestation approaches include [CH02, BCC04].

Construction

Although the technical details of the construction greatly differ from those we have previously discussed, from our high-level perspective not much immediately seems notable. Entities receive credentials with eponymous identities from authorities using a public-key infrastructure and prove those identities to verifiers who request them. However, in each step of the process U-Prove takes a decidedly different approach in the name of privacy. Although the initial credential issuing process is similar to our previous discussion - an authority registers the name and attributes the entity wishes to have included on the credential and then issues the document - U-Prove provides a blinding technique by allowing other authorities to reissue the credential without being able to determine the information contained within it. For example, a prover might have a credential issued to them by TA_A, and can then have TA_B reissue it without them being able to learn what information it contains - which they will do provided they trust TA_A to have properly registered the entity. Further, no party - most importantly not TA_A - can link the original credential and its reissued version, and so TA_A cannot link the actions of the entity to the identifying information they registered. As for the proving process, U-Prove provides selective disclosure to the greatest extent possible, with the verifier producing a set of authentication requirements and the prover able to expose only that information from their identity needed to satisfy the requirements and no more.

Applying Framework Definitions

Entities, identities, and credentials are similar to the idealized PKI with U-Prove, except for the capability of a prover to have more granular control over which aspects of their identity are exposed. By our definition the prover can produce derived credentials $\Gamma_i \subseteq \Gamma$ where each subidentity also may include only partial information regarding an element of the identity - such as an age range instead of an age. It is in trust and reputation where U-Prove has its most interesting qualities. In its effort to minimize trust in the proving process the burden for establishing the correctness of a claim of eponymity is shifted to other aspects of the scheme. A prime example of this is the certificate reissuance we have just discussed, as if a credential is misused the reissuing authority bears the burden on their reputation despite not having registered the prover. As such, the relationships between authorities once again become of central importance to the scheme. If insufficient trust is held between authorities in the system then entities may struggle to gain reissued credentials and the privacy protection of U-Prove, even as it provides effective selective disclosure.

That U-Prove is unable to excise trust relationships from its operation - yet does everything possible to mitigate its need for them - makes it an interesting case for much of our discussion on the topic. U-Prove might be said to distrust trust, and is our foremost example of an entity authentication approach which economizes on it. Although it does not directly restrict the content of communications in any manner, the almost paranoid mentality that it brings to exposing information promotes minimizing the amount of trust that entities place in each other. Entities often expose unnecessary information about them when participating in a community as a signal of their trust - U-Prove implicitly counsels against this. As such, what trust is employed in the system - such as that held between authorities for the purpose of credential reissuance - is encouraged to be secured and employed like currency - for specific use and gain - and not cultivated in a manner which would encourage systemic integrity. This exposes U-Prove implementations to a lemons market effect, with entities inclined to utilize information asymmetry to minimize their perceived risk. This confrontational attitude - combined with the privacy protection of the scheme - would be greatly conducive to malicious action and could drive entities incapable or unwilling to compete out of the system.

With regards to the rationality continuum, U-Prove seems at first glance calculative. In particular, the selective disclosure approach allows participants to design authentication processes which lessen the exposure for each party. However, on the converse the privacy-protecting measures increase uncertainty when it comes to evaluating reputation. This difficulty compounds the possible lemons market effect just mentioned, with entities able and incentivized to control their information exposure and the resultant effect on their reputation. As this makes evaluating reputation extremely difficult, U-Prove may incline entities more towards blind trust, but in an insidious manner which provides a gloss of calculativeness. The reissuance model is of particular concern - an authority may frequently misissue credentials, yet many of these mistakes may never come to light because those credentials are hard to trace or never recognized as fraudulent at all.

Applying Framework Considerations

In addition to the privacy protecting processes of blinding reissuance and selective disclosure, U-Prove provides a collection of qualities to confuse tracking attempts by other parties. Of primary use are some techniques that allow entities to fake information or participation in a specific PKI, which although invalid to an in-system verifier appear to be genuine to an outside observer. This allows entities to produce noise which makes tracking even more difficult. Further, U-Prove directly constructs various techniques for self-violation of privacy - such as linking two different transactions together - in case it becomes necessary. As such, an entity is not demanded to find some ad hoc method of doing so.

As for statefulness, U-Prove provides for unlimited-show and limited-show credentials. The former are linkable while the latter are not if used properly. That caveat is how U-Prove manages fraud and uncertainty, as if a limited-show credential is overused then its private key may be recovered. For instance, suppose that a message is determined to have been an order for an illicit good, and the investigators have reason to believe it was sent by a specific entity. If they can force the entity to produce their credentials then they may run them past the threshold, recover the private key, and see if it produces the same signature on the illegal order. If so, then the investigators have a cryptographic proof that the entity was responsible for the message. Although U-Prove seems to provide the best of both worlds on paper, the balance between privacy on one hand and fraud prevention and punishment on the other is still tricky in practice. Privacy is best served by the reissuance of credentials being efficient and painless, however this allows fraudulent actors to constantly re-up credentials and stay one step ahead of the techniques available to investigators.

4.5 Kerberos

We conclude this section with an example of an authority-based entity authentication scheme which does not employ a PKI. Kerberos is an interactive key distribution mechanism where the authority maintains communication with entities and actively issues session keys to them as necessary. Designed hand-in-hand with a software implementation, our discussion will abstract away some of the details of the scheme which are defined due to that implementation - such as how identities are proven to the authority - and instead focus on the aspects of the system which can be generalized.

Construction

Kerberos requires a central authority server which has continual, secured, and authenticated connections to the various entities in the system. It works via a ticket process. When the entity first authenticates it receives a multiuse ticket - often valid till the end of the session or similar - which it may repeatedly redeem for individual tickets which may then be presented to other entities to initiate a direct key exchange. Essentially, the entity proves itself to the authority and receives a document they can then present back to the authority to have specific attestation statements issued to them, which they may then present to the other entities. See [Sti05, SNS88, NT94] for more discussion of Kerberos. It was standardized in [NYHR05], although a number of addendums for various alternative versions of the scheme exist.

Applying Framework Definitions

It is the multiple credentials that set Kerberos apart. Most PKIs are non-interactive and have long-lasting credentials which may last years or indefinitely. Kerberos is designed to be session-oriented and so uses short-lived session keys. However, entities having to reprove their connection to an identity every time they wish to get a new set of attestation keys is time-consuming and often unnecessary. As such, the multiple credential approach is used, one multiuse which lasts for a reasonable time frame for the proving of identity to the authority, the other a short-lived credential for showing the attestation to another entity. Whether reproving an identity is sufficiently burdensome as to warrant the multiuse credential may be a vestige of the era in which Kerberos was developed, but regardless it is an interesting and effective approach.

Applying Framework Considerations

Statefulness is what is of most interest with Kerberos, and although its interactivity allows for interesting authentication techniques like multiple credentials these come with tradeoffs. For instance, the entities in the system - especially the authority - must be able to determine when various documents in the system were issued. This is especially the case with the multiuse identity proof, as the authority must confirm it actually originated in the current session and that a malicious entity isn't trying to reuse an old ticket. Time-synchronization is not an easy problem in computing, and is certainly considered one of the greatest weaknesses of the scheme. The active and central role of the authority also allows for extensive tracking in most formulations of Kerberos. To produce the short-lived ticket the authority learns who it is interacting with and when. As the authority has interactive control over the system, handling detected fraud is easy, as the central authority can simply refuse at will to deal with entities who they have determined are malicious.

5 Authority-less Approaches

The second collection of entity authentication schemes we will consider do not use empowered authorities to assert eponymity. Broadly speaking, two different types of such schemes exist, those that use a less structured trust network for assertions and those that do not use any assertions at all - but rather self-attestment. The example of the former we shall look at is the web of trust, the PKI employed by PGP. For the latter, we will look at user supplied credentials, which includes *username:password* systems, hardware or communications network techniques, biometrics, and others.

5.1 The Web of Trust

Although authority-based offline trust and reputation systems are quite prevalent - consumer review services, government background checks, investigative reporting, and so forth - the most common form of identity attestation is ad hoc and personal. An official certification that a doctor is widely respected at carrying out a procedure often carries less weight then a recommendation from a friend. The web of trust is a PKI model designed with this mindset, that the most meaningful assertion of the eponymity of an identity is from another entity who is personally known and trusted. Initially formulated for the PGP application for asserting the eponymity of email addresses with keys [Zim95], the web of trust approach has been extended to a general technique for a variety of authentication tasks.

Construction

With the web of trust, various entities in the system generate their own key pairs and choose identities to represent them, with an email address the archetype. Entities then digitally sign the keys of others, with the signature also including a specific identity for that key. Usually the decision to sign a key is based on an offline interaction where the attestor can validate the eponymity directly. When a user wishes to authenticate an entity who has a certain key pair they may be given those signatures and validate them, providing a cryptographic proof that the signing entities attested to the eponymity. As those other entities may be controlled by a malicious prover the goal is to create a network of trust relationships such that the proving entity can supply a set of attesting signatures generated by entities who the verifier trusts or may trace a trust chain to. The web of trust uses a bootstrapped model in which individual entities attest on their reputation to the eponymity of entities they know, creating a dense network of social contacts all willing and able to support claims of identity.

Applying Framework Definitions

The active role all parties must play within a web of trust provides a depth of understanding of the system as to the difference between an entity and identity that authority-based approaches lack. Predicated on most every entity confirming eponymity and attesting to it to create the trust network, those entities will be exposed to the difference firsthand. As such, although implementations of webs of trust may use certain methods to simplify reputation determination, the web of trust is the closest a PKI gets to calculativeness. The personal element of the scheme allows for meaningful consideration of the whole of an attesting entity for determining their reputation, and each participating party is aware of how and why attestments are given. This is to not say specific entities are rendered unable to blindly trust other entities, but it at least provides the possibility for calculative assessment to an extent which other methods do not.

Although the mechanisms of the web of trust must be well known by its users, it is far from ideally calculative because of shortcomings in how reputation is assessed. In dense social pockets the web of trust is extremely effective, however it struggles in more general cases - such as the internet - where entities may commonly wish to verify identities from well beyond their immediate circle. That is to say, it struggles to scale. Further, even in relatively well-connected communities it is not reasonable to determine a general rule for acceptance of attestations beyond an immediate step, as per our discussion of reputation propagation. If the most trusted friend of an entity attests to an eponymity they'll most likely accept it, but what if it is a friend of a friend of a friend attesting? PGP includes a default but user-editable model for reputation propagation in which various entities - or rather their representative identities - are assigned either full or partial trust labels, and then an authentication threshold of type and number of required attestors is specified. This approach is not strictly necessary, and a web of trust scheme could use alternative methods or a case by case approach to considering validity. However, with either method some calculativeness is lost as entities are forced to make judgments with incomplete information as to the reputation of entities multiple steps away from them on the web. For more on assessing trust and reputation in webs of trust, see [Car00, AR97].

Applying Framework Considerations

The decentralized nature of the web of trust makes it ideal for privacy. Since issuing and requesting attestments is a local process, no authority is in a position of strength with which to track traffic. However, the web of trust is not a panacea to privacy issues. Not only can implementation tracking occur, but in the dense social networks for which the web is most effective only a small number of informants are required to monitor the vast majority of traffic, an effect discussed in [DW06]. The localized structure also helps with managing fraud and uncertainty, since individual relationships determine reputation. If an entity is mistaken or malicious and they attest improperly - or if an identity is found to misrepresent an entity - then the attesting parties can rectify the situation by renouncing the eponymity and verifiers may reassess reputation directly. However, this won't be effective until the reevaluations are executed, and if state is a difficult issue for authorities with the means and incentive to be as quick as possible, it is orders of magnitude more difficult for the web of trust. Further, an interactive system is not an option as individual entities cannot be universally available for attestment. As such, although rectifying situations of fraud may be straightforward, it cannot be expected that they will be temporally prompt.

5.2 User Supplied Credentials

Despite the wide variety of approaches detailed in our previous examples, many entity authentication processes do not include the participation of attestors. Instead, the proving party provides - in addition to their private information necessary to establish publishership - an authentication factor which only they have access to, such as a government issued identification number or a biometric identifier. Intuitively user supplied credentials are suspect, as often the same identifiers are used by many different verification processes - rendering the factor far from private. However, due to ease of implementation and use systems which employ such credentials are extremely common.

Construction

In the preceding introduction we stated that two different collections of information are used with user supplied credentials, the proving information and the entity authentication information. Although this is common - such as with a website where you sign up with a government identification number but authenticate with a username and password - it is not necessarily the case. Certain schemes - such as those biometric-based - use the same information for both purposes. Authentication factors usually come in one of three forms, something you know - such as an identification number - something you have - such as a smartcard - or something you are - such as a biometric identifier. For added proof of identity some schemes may demand the use of multiple factors to reduce the likelihood of impersonation. However, as by our previous comment not all multifactor authentication approaches are for this strengthening purpose, with some being used for the proof of publishership. For considerable discussion of the strengths and weaknesses of various user-supplied approaches, see [BHOS12, Gor03].

Applying Framework Definitions

User supplied credentials directly conflate entity and identity. The private authentication information constitutes an identity, however it is considered an inherent part of the entity which only they may provide. This is the root issue with such an approach to entity authentication, since it does not recognize that an identity is something others will need to know and therefore is unsuitable as a means of proof. Due to the issues with such credentials, user-supplied schemes are the closest to blind trust out of any method we have yet considered, as the verifier must trust that the entity authentication information is not known by any party which would attempt to impersonate the prover. Although much of the prevalence of such schemes is due to their ease of implementation, it may also be the case that there is comfort in ignorance. A verifier might be more comfortable with suspect authentication - especially if the widespread use of such authentication mechanisms gives cover for their reputation - as if they are responsible for evaluating reputation and choosing which attestors to trust they may bear more responsibility for any fraud which may occur. For certain verifiers who do not prioritize the authentication of its provers but who must do it as a matter of course, such cover may be considered beneficial.

Applying Framework Considerations

Although we espoused the privacy benefits of the web of trust, theoretically the lack of authorities whatsoever should make user-supplied credentials ideal. However, due to small number of attributes usable for entity authentication in such schemes and their common reuse, these approaches are in practice perhaps the least conducive to unlinkability. Since the usefulness of the information is predicated on it being undeniable, it is direct and straightforward to collate information regarding who is authenticating to whom. A tracker would still need the willing or unwilling assistance of verifiers to operate, but they gain extremely useful information by doing so - including that which is needed to impersonate the entity. Therefore the nature of the entity authentication information determines the likelihood a scheme will provide privacy, and conversely that they will provide protection against fraud. There is great uncertainty in systems without attesting entities, and privacy defeating information is often necessary to prevent against it. The concept of statefulness does not generally exist in most user-supplied schemes, as entity authentication information being supposedly inherent to the entity implies permanency. Expiration and revocation are the most common forms of statefulness where it is deployed.

6 Conclusion

We conclude by considering some of the limitations of our framework. Of first note has been our need for brevity. Our discussion of many topics - in particular trust, reputation, privacy, statefulness, and fraud - have been truncated and substantial discussion of alternative definitions and additional aspects to each can and should be considered across the literature. The same difficulty with length applies to our application of the framework to schemes. In all of these cases we have tried to paint a reasonable outline of the major concepts necessary, but works as long as this one could be produced on each of them. As for the framework itself, what struggles it has in application averting the introduction of confusion are mainly the result of various schemes playing components we have made distinct off of one another. For instance, the muddling of entity and identity is often introduced - such as by user supplied approaches and the web browser model - which may cause movement towards blind trust. That we have detached the discussion of entities and identities from that of blind trust within the framework can makes our conception of this effect seem misformulated from the perspective of one or the other.

Overall, we have introduced a framework for analysis of entity authentication schemes. This framework includes definitions and discussion of a number of important topics, most notably authorship and publishership - the security guarantees demanded for the task at hand. More important in many ways then the letter of this work is the mindset of our approach - that entity authentication cannot be formalized - and that considering how schemes handle certain aspects of establishing authorship - even imperfectly so - cannot be excised from cryptography. We have further applied our framework to a collection of commonly used and referenced entity authentication schemes, and hopefully throughout all have provided sufficient justification and consistency to where our framework can reasonably be extended beyond what is directly contained within this short presentation.

Bibliography

[ADS03] Alessandro Acquisti, Roger Dingledine, and Paul Syverson. On the Economics of Anonymity. In *Financial Cryptography: 7th International Conference, FC 2003, Guadeloupe, French West Indies, January 27-30, 2003, Revised Papers*, volume 2742, pages 84–102. Springer, 2003.

[Ake70] George A. Akerlof. The Market for "Lemons": Quality Uncertainty and the Market Mechanism. *The Quarterly Journal of Economics*, pages 488–500, 1970.

[AL03] Carlisle Adams and Steve Lloyd. *Understanding PKI.* Addison-Wesley, 2nd edition, 2003.

[AR97] Alfarez Abdul-Rahman. The PGP Trust Model. In *EDI-Forum: the Journal of Electronic Commerce*, volume 10, pages 27–31, 1997.

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless Public Key Cryptography. In *Advances in Cryptology-ASIACRYPT 2003*, pages 452–473. Springer, 2003.

[AT91] Martin Abadi and Mark R. Tuttle. A Semantics for a Logic of Authentication. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 201–216. ACM, 1991.

[BAN89] Michael Burrows, Martin Abadi, and Roger M. Needham. A Logic of Authentication. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 426, pages 233–271. The Royal Society, 1989.

[BCC04] Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct Anonymous Attestation. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*, pages 132–145. ACM, 2004.

[BDM95] Joyce Berg, John Dickhaut, and Kevin McCabe. Trust, Reciprocity, and Social History. *Games and Economic Behavior*, 10(1):122–142, 1995.

[BF03] Dan Boneh and Matthew Franklin. Identity-Based Encryption from the Weil Pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[BFK99] Matt Blaze, Joan Feigenbaum, and Angelos D. Keromytis. KeyNote: Trust Management for Public-Key Infrastructures. In *Security Protocols*, pages 59–63. Springer, 1999.

[BFL96] Matt Blaze, Joan Feigenbaum, and Jack Lacy. Decentralized Trust Management. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 164–173. IEEE, 1996.

[BHOS12] Joseph Bonneau, Cormac Herley, Paul C Van Oorschot, and Frank Stajano. The Quest To Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes. In *2012 IEEE Symposium on Security and Privacy (SP)*, pages 553–567. IEEE, 2012.

[Bra97] Marc Branchaud. A Survey of Public-Key Infastructues. Master's thesis, McGill University, Montreal, 1997.

[Bra00] Stefan A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy.* MIT Press, 2000.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy, 2007.*, pages 321–334. IEEE, 2007.

[Cab05] Luis MB Cabral. The Economics of Trust and Reputation: A Primer. Accessed at `http://pages.stern.nyu.edu/~lcabral/reputation/Reputation_June05.pdf`, 2005.

[Car00] Germano Caronni. Walking the Web of Trust. In *Proceedings of the IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2000 (WET ICE 2000).*, pages 153–158. IEEE, 2000.

[CH02] Jan Camenisch and Els Van Herreweghen. Design and Implementation of the Idemix Anonymous Credential System. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 21–30. ACM, 2002.

[Cha81] David Chaum. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communications of the ACM*, 24(2):84–90, 1981.

[Cha83] David Chaum. Blind Signatures for Untraceable Payments. In *Advances in Cryptology*, pages 199–203. Springer, 1983.

[Cha85] David Chaum. Security Without Identification: Transaction Systems To Make Big Brother Obsolete. *Communications of the ACM*, 28(10):1030–1044, 1985.

[CRS05] David Chaum, Peter Ryan, and Steve Schneider. A Practical Voter-Verifiable Election Scheme. *Computer Security-ESORICS 2005*, pages 118–139, 2005.

[CSF$^+$08] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), 2008.

[Das88] Parth Dasgupta. Trust as a Commodity. In *Trust: Making and Breaking Cooperative Relations*, pages 49–72. B. Blackwell, 1988.

[Dun88] John Dunn. Trust and Political Agency. In *Trust: Making and Breaking Cooperative Relations*, pages 73–93. B. Blackwell, 1988.

[DW06] George Danezis and Bettina Wittneben. The Economics of Mass Surveillance - and the Questionable Value of Anonymous Communications. In *In Proceedings of the 5th Workshop on The Economics of Information Security (WEIS 2006)*, pages 1–16. Citeseer, 2006.

[ES00] Carl Ellison and Bruce Schneier. Ten Risks of PKI: What You're Not Being Told About Public Key Infrastructure. *Computer Security Journal*, 16(1):1–7, 2000.

[FFS88] Uriel Feige, Amos Fiat, and Adi Shamir. Zero-Knowledge Proofs of Identity. *Journal of Cryptology*, 1(2):77–94, 1988.

[Gam88] Diego Gambetta. Can We Trust Trust? In *Trust: Making and Breaking Cooperative Relations*, pages 213–237. B. Blackwell, 1988.

[Gen03] Craig Gentry. Certificate-Based Encryption and the Certificate Revocation Problem. In *Advances in Cryptology-EUROCRYPT 2003*, pages 272–293. Springer, 2003.

[Gir91] Marc Girault. Self-Certified Public Keys. In *Advances in Cryptology-EUROCRYPT91*, pages 490–497. Springer, 1991.

[GKRT04] Ramanthan Guha, Ravi Kumar, Prabhakar Raghavan, and Andrew Tomkins. Propagation of Trust and Distrust. In *Proceedings of the 13th international conference on World Wide Web*, pages 403–412. ACM, 2004.

[Gol06] Oded Goldreich. *Foundations of Cryptography: Volume 1 Basic Tools.* Cambridge University Press, 3rd edition, 2006.

[Gor03] Lawrence O. Gorman. Comparing Passwords, Tokens, and Biometrics for User Authentication. *Proceedings of the IEEE*, 91(12):2021–2040, 2003.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[GS02] Craig Gentry and Alice Silverberg. Hierarchical ID-based Cryptography. In *Advances in Cryptology-ASIACRYPT 2002*, pages 548–566. Springer, 2002.

[KM00] Reto Kohlas and Ueli Maurer. Confidence Valuation in a Public-Key Infrastructure Based on Uncertain Evidence. In *International Workshop on Practice and Theory in Public-Key Cryptography 2000*, pages 93–112. Springer, 2000.

[KMRS09] Dean Karlan, Markus Mobius, Tanya Rosenblat, and Adam Szeidl. Trust and Social Collateral. *The Quarterly Journal of Economics*, 124(3):1307–1361, 2009.

[LK15] Yehuda Lindell and Jonathan Katz. *Introduction to Modern Cryptography.* Chapman and Hall/CRC Press, 2nd edition, 2015.

[Mau96] Ueli Maurer. Modelling a Public-Key Infrastructure. In *Computer Security-ESORICS 96*, pages 325–350. Springer, 1996.

[MMH02] Lik Mui, Mojdeh Mohtashemi, and Ari Halberstadt. A Computational Model of Trust and Reputation. In *Proceedings of the 35th Annual Hawaii International Conference on System Sciences, 2002.*, pages 2431–2439. IEEE, 2002.

[MR01] Patrick McDaniel and Aviel Rubin. A Response To Can We Eliminate Certificate Revocation Lists?. In *Financial Cryptography*, pages 245–258. Springer, 2001.

[NT94] Clifford Neuman and Theodore Ts'o. Kerberos: An Authentication Service for Computer Networks. *IEEE Communications Magazine*, 32(9):33–38, 1994.

[NYHR05] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. The Kerberos Network Authentication Service (V5). RFC 4120 (Proposed Standard), 2005.

[PK01] Andreas Pfitzmann and Marit Köhntopp. Anonymity, Unobservability, and Pseudonymity – a Proposal for Terminology. In *Designing Privacy Enhancing Technologies*, pages 1–9. Springer, 2001.

[PZ13] Christian Paquin and Greg Zaverucha. U-Prove Cryptographic Specification V1.1 (Revision 3), December 2013.

[Riv98] Ronald L. Rivest. Can we eliminate certificate revocation lists? In *Financial Cryptography*, pages 178–183. Springer, 1998.

[Rog15] Phillip Rogaway. The Moral Character of Cryptographic Work. Cryptology ePrint Archive, Report 2015/1162, 2015.

[Sch91] Claus-Peter Schnorr. Efficient Signature Generation by Smart Cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sha85] Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. In *Advances in Cryptology*, pages 47–53. Springer, 1985.

[SNS88] Jennifer G. Steiner, Clifford Neuman, and Jeffrey I. Schiller. Kerberos: An Authentication Service for Open Network Systems. In *USENIX Winter*, pages 191–202, 1988.

[Sti05] Douglas R. Stinson. *Cryptography: Theory and Practice.* Chapman and Hall/CRC Press, 3nd edition, 2005.

[SYHL06] Yan Lindsay Sun, Wei Yu, Zhu Han, and KJ Liu. Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks. *IEEE Journal on Selected Areas in Communications*, 24(2):305–317, 2006.

[TB06] George Theodorakopoulos and John S. Baras. On Trust Models and Trust Evaluation Metrics for Ad Hoc Networks. *Selected Areas in Communications, IEEE Journal on*, 24(2):318–328, 2006.

[Wil93] Oliver E. Williamson. Calculativeness, Trust, and Economic Organization. *Journal of Law and Economics*, pages 453–486, 1993.

[Zim95] Philip R. Zimmerman. *The Official PGP User's Guide.* MIT Press, 1995.

[ZL05] Cai-Nicolas Ziegler and Georg Lausen. Propagation Models for Trust and Distrust in Social Networks. *Information Systems Frontiers*, 7(4-5):337–358, 2005.

Elliptic Curves and their Cryptographic Applications

1 Introduction

Amongst the many treatments of cryptographic subjects aimed at popular interest, few topics are as widely discussed or engaged with as the employment of elliptic curves. This curiosity is not fanciful, as the utility of elliptic curve cryptography (ECC) is recognized by professional practitioners and considerable research effort is contributed to its further development. However, it is hard to avoid the feeling that it has become the focus of undue fascination amongst enthusiasts. A cursory search reveals numerous “easy to understand” primers and “gentle introductions” into elliptic curve cryptography from research and development organizations such as Microsoft Research, Cloudflare, OpenSSL, and Certicom, popular science and technology journalism organizations such as ArsTechnica and LinuxJournal, and many online writers on computer science and mathematics topics.

One conjecture for the source of this curiosity is the exoticness of elliptic curves, that for those without a formal background the mathematical structure employed for ECC is a fascinating construction, and the cryptographic context provides a “real-world usefulness” which drives interest beyond that which a purely algebraic discussion can provide. A further conjecture - for explaining the interest from those with a stronger mathematical background - is that elliptic curve cryptography functions as a shibboleth for the capability to attain meaningful cryptographic knowledge, as understanding ECC requires a level of mathematical sophistication. One must be comfortable with elementary group and field theory and the basic concepts of complexity theory to be able to approach ECC with any form of rigor.

A reasonable extension of this latter thought may be made. If understanding an introduction to elliptic curve cryptography is an indicator of sufficient mathematical background and interest to develop an amateur understanding of cryptography in general, then authoring such an introduction should very well function similarly for those intending to pursue the field professionally. Necessarily, the author must understand the same underlying mathematical and computer scientific structures as the reader. Further, as the greater demands placed on the author come most notably in the form of context - deciding what to include and what to omit - and clarity - having a clean enough grasp of the subject to be able to convey it successfully - authorship of such an introduction requires a general awareness and capability around broader concepts in cryptography.

The discussion put forward in this paper is meant to act as such a test for the author. A brief overview is as follows. From the reader, it assumes a background in undergraduate algebra, in particular group theory and field theory, as well as some awareness of the core concepts of complexity theory. Also is assumed an at least rough understanding of basic cryptographic terminology which may be expected from common usage or can be easily inferred, e.g., the meanings of encryption and ciphertext. *Sections 2*, *3*, and *4* compose a general introduction to the algebra of elliptic curves, providing sufficient material as needed for the discussion of elliptic curve cryptography itself in *Sections 5* and *6*. The goal of the discussion is to be concise, readable, and to find a balance in terms of its rigor to be accessible and yet, for a relatively strong usage of the term, correct.

2 Weierstrass Equation and Elliptic Curves as Groups

Although elliptic curves are of interest from a wide variety of mathematical perspectives, to build to our cryptographic discussion we will be primarily concerned with them through a relatively narrow scope - the basic behavior of the algebraic group composed of their points and a binary operation known as point addition. To begin we will discuss the general form of elliptic curves as a collection of points defined by solutions to a specific equation, before giving a geometric sketch of the functionality of point addition and an intuitive discussion of why it leads to group-theoretic behavior. The section will conclude with an incomplete proof formalizing that this construction is valid under the group laws.

2.1 Weierstrass Equation

For the purposes of this discussion, an elliptic curve E defined over a field K will be a relation in the form of the *Weierstrass equation*,

$$y^2 = x^3 + ax + b,$$

where $x,\, y \in K$ are variables and $a,\, b \in K$ are constants which satisfy $4a^3 + 27b^2 \neq 0$. That this latter quantity - the discriminant - must be non-zero causes a curve to have no repeated roots, which is necessary for the group-theoretic behavior which will be our central focus. In cryptographic contexts, K will most likely be a finite field $\mathbf{F}_q$, where q is prime or a prime power. Another notable setting is curves over $\mathbb{R}$, as they admit a geometric interpretation which can help provide an intuition for the group-theoretic behavior of elliptic curves in general.

Note that it is possible to generalize the Weierstrass form to include a more expansive class of relations as elliptic curves - a useful capability if the characteristic of K is non-zero - but it is an unnecessary complexity for this discussion. In addition, there exist families of relations which can be shown to be equivalent to curves in Weierstrass form, but whose alternate definitions have certain benefits and/or drawbacks with regards to computations over them. Some of these constructions, most notably Edwards and Montgomery curves, are widely used in cryptographic contexts, and although for simplicity we will focus on Weierstrass curves these alternatives will be briefly discussed in the ECC sections.

To consider the algebraic qualities of a curve defined over a field K, we will be concerned with the set of points $E(K)$ on it,

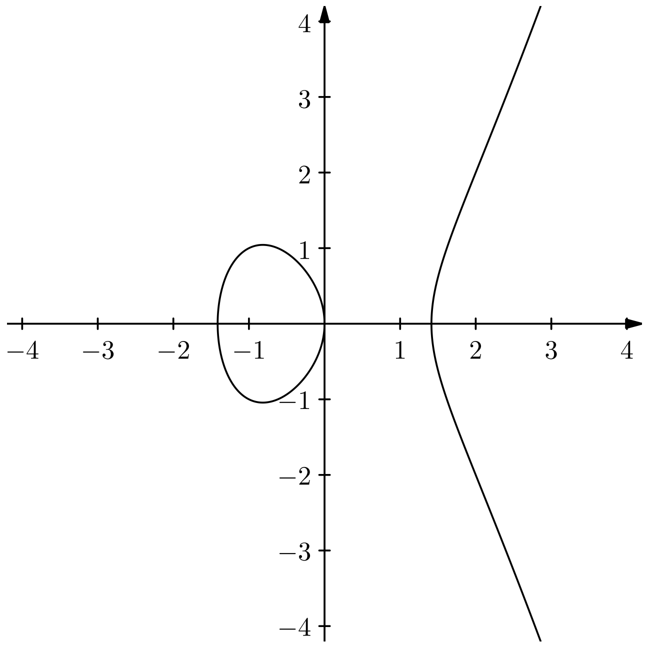
$$E(K) = \{(x,\, y) \in K \times K \mid y^2 = x^3 + ax + b\} \cup \{\infty\}.$$

Going forward, when the term "elliptic curve" is used it may be assumed we are considering this set of points as an algebraic object, as opposed to the equation in Weierstrass form itself. Of immediate curiosity is the inclusion of the point at infinity, or ∞. Although its definition can be made rigorous by use of projective coordinates, for the purposes of this discussion it may be simply considered to sit at top of the y-axis. This point will play an important role in the group functionality which will be built over $E(K)$.

2.2 Intuition for Group Theoretic Behavior

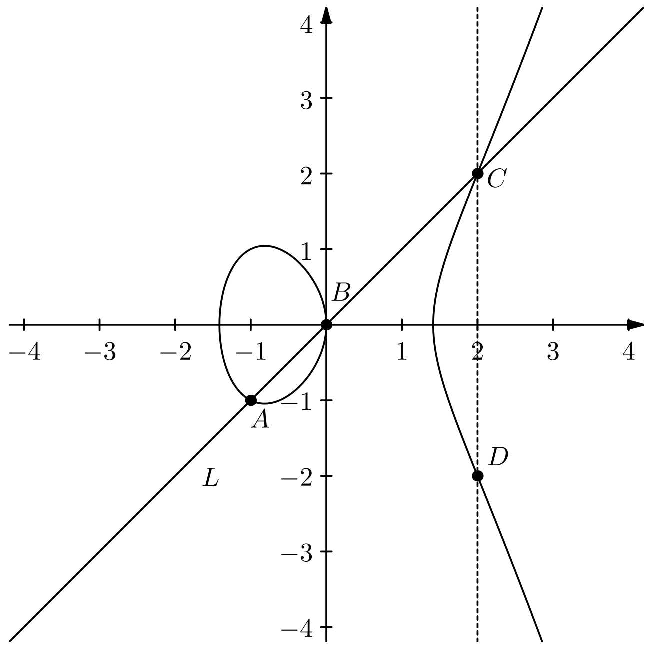
The usefulness of elliptic curves in cryptographic contexts stems from the easy definition of algebraic - or more specifically group-theoretic - behavior over them. Although this quality of easy definition is maintained regardless of the field over which a curve E is defined, the behavior is most easily considered through analysis of a curve over $\mathbb{R}$, as such a curve allows for a geometric discussion which functions as a good heuristic for understanding the behavior in general. The following discussion is not meant to be rigorous, but is simply meant to help provide an intuition for why the group theoretic structure inherent in elliptic curves exists.

Consider the elliptic curve $E(\mathbb{R})$ given by the equation $y^2 = x^3 - 2x$.



Geometrically, this is one of the most common shapes for an elliptic curve to take, an ovoid form covering the negative x values for which $y^2 > 0$, and another form going to both infinities, covering the positive x values for which $y^2 > 0$ as well. The unique quality of this and every other elliptic curve is that if one takes any choice of two points on the curve and draws a line between them - or choosing the same point twice, draws its tangent - that line will touch the curve at one, and only one, additional point. The obvious exception to this, two points positioned vertically inline, are defined to touch the curve a third time at the included point at infinity.

For example, consider the points $A, B \in E(\mathbb{R})$ where $A = (-1, -1)$ and $B = (0, 0)$. Taking the line L between them, with a slope of $\frac{0-(-1)}{0-(-1)} = 1$, gives another point on the curve $C = (2, 2)$. Since the slope of the curve for positive values of y is (by differentiation) $\frac{3x^2-2}{2\sqrt{x(x-2)}}$ and as $x \geq 2$, it is straightforward to see that L will not intersect E again. In addition, $E(\mathbb{R})$ carries a natural symmetry across the x-axis, as if (x, y) is an element of the curve then so is $(x, -y)$, which follows directly from $\pm y$ both being valid solutions to $\sqrt{y^2}$. As such, if C is on the curve, then so is $-C = D = (2, -2)$.



Therefore, there is seemingly a natural mechanism to define a binary operation over $E(\mathbb{R})$, point addition, where the sum is the third point on the line containing the two summands. However, as this operation never leaves the line on which it originates it cannot incorporate the full set of points on the curve, and therefore cannot function as a group operation for all of $E(\mathbb{R})$. Instead, defining point addition by $A + B = D$, the reflection of the third point on the line over the x-axis, provides the necessary structure. For the point at infinity, ∞, we define its reflection over the x-axis to be itself.

For a set S and binary operation $\#$ to constitute a group $(S, \#)$, the structure must have each of four required properties:

1. Closure - For a, $b \in S$, if $a \# b = c$, then $c \in S$.
2. Identity - There exists $e \in S$ such that $a \# e = e \# a = a$ for all $a \in S$.
3. Inverse - For all $a \in S$, there exists $b \in S$ such that $a \# b = b \# a = e$.
4. Associativity - For all a, b, $c \in S$, $(a \# b) \# c = a \# (b \# c)$.

We may consider whether $(E(\mathbb{R}), +)$ has these properties. Closure is a consequence of a line through any two points P and Q contacting the curve at exactly one other point, which has exactly one reflection over the x-axis. Note as well that it does not matter which order P and Q are specified in, and therefore point addition satisfies commutativity, that $P + Q = Q + P = R$ as well. Although there is no natural identity among the points which fall on the curve proper, we define the included point at infinity, ∞, as the identity, such that $P + \infty = \infty + P = P$. The existence of an inverse clearly follows, as for a point $P = (x, y)$ its inverse is $-P = (x, -y)$, where $-y$ is the additive inverse of y in $\mathbb{R}$, since the third point contacted by a line through P and $-P$ is ∞. As for associativity, this is a substantially more complicated property to prove, but it can be shown through tedious calculation using the generalized definition of point addition to follow in the next subsection, as well as by other more difficult but elegant methods.

2.3 Point Addition and the Group Laws

Although the discussion above considered a curve E defined over $\mathbb{R}$, the properties and behaviors described are fundamental to the structure of the elliptic curve itself and exist over whatever field K the curve is defined over. Therefore, we can extend the reasoning to give a formal definition of point addition, and then a proof sketch that $E(K)$ constitutes an additive abelian group.

To formally show the group behavior of elliptic curves, we must first fully define the behavior of the group operation, point addition.

Definition 2.1 (Point Addition)**.** Let $E(K)$ be an elliptic curve defined by $y^2 = x^3 + ax + b$, where K is a field and $x, y, a, b \in K$. Let $P = (r, s)$ and $Q = (t, u)$ be points on the curve (and therefore $P, Q \in E(K)$ and $r, s, t, u \in K$) with $P, Q \neq \infty$. Define $P + Q = R = (v, w)$ as follows.

1. If $r \neq s$, then
$$R = \big(v, w\big) = \big(m^2 - r - t,\, m(r - v) - s\big), \text{where } m = \left(\frac{u - s}{t - r}\right).$$
2. If $r = t$ but $s \neq u$, then $R = \infty$.
3. If $P = Q$ and $s \neq 0$, then
$$R = \big(v, w\big) = \big(m^2 - r - t,\, m(r - v) - s\big), \text{where } m = \left(\frac{3r + a}{2s}\right).$$
4. If $P = Q$ and $s = u = 0$, then $P + Q = \infty$.
5. Lastly, for all $P \in E(K)$, $P + \infty = \infty + P = P$.

For the first and third equation, the value of m corresponds to the growth behavior of an object L whose membership set contains both the points being added. Going back to our preceding discussion for an example, for a curve $E(\mathbb{R})$ the value of m is the slope of the line L containing the summands (for the third equation the value is found by use of implicit differentiation). Although this geometric interpretation breaks down in the general case of fields K we are now considering - necessitating the vague language in the first sentence of this paragraph - viewing L and m as generalizations of “line” and “slope” respectively provides context for why this definition behaves over any field. The equations for $R = (v, w)$ can then be derived by inserting the equation for L into the Weierstrass equation for E, solving the resulting cubic to determine the final point which lies on both L and E, and finally negating the equation for w to reflect over the x-axis. A full derivation of these equations is provided in [Was03]. As for the second and fourth equation, in either case the resultant line is vertical, and therefore P and Q are inverses and the result of their addition is the point at infinity.

To give an example of calculating with these equations, we may use them to give a formal conclusion to the intuitive discussion from the previous subsection by adding two points over $\mathbb{R}$. Take $A = (-1, -1)$ and $B = (0, 0)$ as before. As $-1 \neq 0$, we have
$$m = \frac{0 - (-1)}{0 - (-1)} = 1,$$

$$v = 1^2 - (-1) - 0 = 2,$$

$$w = 1((-1) - 2) - (-1) = 1(-3) + 1 = -2,$$

and therefore $A + B = (2, -2)$, the value of D determined geometrically.

Using *Definition 2.1*, we now have the formal construction needed to show that $(E(K), +)$ is a group.

Theorem 2.2 (Elliptic Curves as Groups)**.** An elliptic curve E defined over a field K constitutes an additive abelian group with the group operation of point addition $(+)$.

Proof Sketch. To constitute an additive abelian group, $(E(K), +)$ must satisfy the following properties:

1. Closure - For $P, Q \in E(K)$, if $P + Q = R$ then $R \in E(K)$.
2. Identity - There exists $e \in E(K)$ such that $P + e = e + P = P$ for all $P \in E(K)$.
3. Inverse - For all $P \in E(K)$, there exists $-P \in E(K)$ such that $P + -P = e$.
4. Associativity - For all $P, Q, R \in E(K)$, $(P + Q) + R = P + (Q + R)$.
5. Commutativity - For all $P, Q \in E(K)$, $P + Q = Q + P$.

We shall show each in turn.

1. Closure - We wish to show that for $P + Q = R$, or $(r, s) + (t, u) = (v, w)$, that $R \in E(K)$. That $v, w \in K$ follows from that $r, s, t, u \in K$ and the closure of K itself. That $R \in E(K)$ comes from the derivation of the equations for point addition, and that the sum of a point addition is a solution to the Weierstrass equation for E.
2. Identity - By the definition of an elliptic curve there is an element $\infty \in E(K)$ for which by the definition of point addition we have $P + \infty = \infty + P = P$ for all $P \in E(K)$. Therefore ∞ constitutes an identity for $(E(K), +)$.
3. Inverse - For any point $P = (r, s)$, we have that $s^2 = r^3 + ar + b$ is a solution to the Weierstrass equation for E, and therefore $Q = (r, -s)$ is also a valid solution. By the definition of point addition, $P + Q = \infty$, and therefore $Q = -P$.
4. Associativity - This is the most complex property to show, see [Was03] for a proof. Note that the associativity can also be shown directly through (a tedious) manipulation of the point addition equations.
5. Commutativity - By the derivation of the point addition equations, R is determined by constructing an object L containing P, Q, and determining the point R where L and E intersect. Since the construction of L is the same regardless of which of P, Q is chosen first, $P + Q$ will determine the same R as $Q + P$. Note that the commutativity can also be shown directly through (a tedious) manipulation of the point addition equations. $\square$

The similarity in language between the informal commentary on whether $(E(\mathbb{R}), +)$ constitutes a group and the proof sketch of *Theorem 2.2* is not coincidental. The generalization of point addition is sufficiently direct to where the two discussions are essentially equivalent, up to formality of language and the generality of the setting. Ultimately, formally showing the adherence to the group law axioms by elliptic curve groups follows very naturally from their intuitive geometric behavior when constructed over $\mathbb{R}$, as only the existence of associativity remains unconvincing - it not being apparent in the geometric discussion either. Although this correspondence with the generalization helps provide grounding for understanding the group law, it is important not to become too reliant on the geometric intuition, as now our discussion will turn towards other fields over which there is no such natural interpretation.

3 Elliptic Curves over Finite Fields

Our attention will now focus upon some specific questions concerning elliptic curve groups when defined over finite fields. In particular, we will be interested in the various methods by which we can determine the exact number of points on such curves, as that information will be important for cryptographic purposes. The section will begin with an overview of the behavior of curves over finite fields, before turning towards discussion of some of these methods and an important theorem underlying them.

3.1 Basic Discussion

As previously noted, when elliptic curves are employed for cryptography the fields they are nearly always defined over are finite fields $\mathbf{F}_q$. Therefore, discussion of the general behavior of such algebraic objects $E(\mathbf{F}_q)$ is a necessary consideration for ECC. The fundamental result with regards to finite fields is their unique (up to isomorphism) existence for all prime powers.

Theorem 3.1 (Galois' Theorem on Finite Fields)**.** The order of any finite field is a prime power. Further, there is a unique finite field of any given prime power order (up to isomorphism).

However, although $E(\mathbf{F}_q)$ is a group for all choices of E and for all finite fields, each defined by a $q = p^n$, there are some limitations on the attributes of such groups. For the following theorem, note that $\#E(\mathbf{F}_q)$ is the number of points in $E(\mathbf{F}_q)$, also known as *the order of the group* $E(\mathbf{F}_q)$.

Theorem 3.2 (Elliptic Curves with Groups over Finite Fields)**.** Let $q = p^n$ be a prime power and let $N = q + 1 - j$, where j is some integer. There is an elliptic curve E defined over $\mathbf{F}_q$ such that $\#E(\mathbf{F}_q) = N$ if and only if $|j| \leq 2\sqrt{q}$ and j satisfies one of the following.

1. $\gcd(j, p) = 1$;
2. n is even and $j = \pm 2\sqrt{q}$;
3. n is even, $p \not\equiv 1 \pmod 3$, and $j = \pm\sqrt{q}$;

4. n is odd, $p = 2$ or $p = 3$, and $j = \pm p^{(n+1)/2}$;

5. n is even, $p \not\equiv 1 \pmod 3$, and $j = 0$;

6. n is odd and $j = 0$.

Exactly determining $\#E(\mathbf{F}_q)$ can be done efficiently, which process will be the focus of the next subsection. For now, this theorem gives a preliminary guideline and provides an initial hint that there is a connection between the order of $E(\mathbf{F}_q)$ and the attributes of the curve.

The membership in an elliptic curve defined over a finite field $E(\mathbf{F}_q)$ is - as per our introductory discussion - the set of points whose coordinates are elements of $\mathbf{F}_q$ and which lie on E. The equations for point addition are applied in the same manner as for any other setting, except that all arithmetic operations are reduced into the finite field. To introduce more clarity we may consider a toy example. Let E be the curve $y^2 = x^3 - 2x$ as in the discussion from the previous section, and define E over $\mathbf{F}_7$. As $\mathbf{F}_7$ is a prime field its arithmetic behavior is equivalent to that of the ring of integers modulo 7. Curves are commonly chosen over prime fields for cryptographic use for this exact reason, as it is straightforward to choose coset representatives and therefore conduct arithmetic with them, simply by considering the operations as being conducted in the ring of integers modulo the same prime. To determine the membership of $E(\mathbf{F}_7)$, we consider which values of x give values for y such that (x, y) is on the curve.

x	$x^3 - 2x$	y	(x, y)	x	$x^3 - 2x$	y	(x, y)
0	0	0	(0, 0)	4	0	0	(4, 0)
1	6	–	–	5	3	–	–
2	4	± 2	(2, 2), (2, 5)	6	1	± 1	(6, 1), (6, 6)
3	0	0	(3, 0)	∞	–	∞	∞

Therefore, we have our group of order 8:

$$E(\mathbf{F}_7) = \{(0, 0), (2, 2), (2, 5), (3, 0), (4, 0), (6, 1), (6, 6), \infty\}.$$

Point addition can then be computed with reduction into the field. For instance, to find $C = (2, 5) + (4, 0)$ we calculate,

$$m \equiv (0 - 5) \cdot (4 - 2)^{-1} \equiv 2 \cdot 2^{-1} \equiv 1 \pmod 7,$$

$$C_x \equiv (1)^2 - 2 - 4 \equiv 2 \pmod 7,$$

$$C_y \equiv 1(2 - 2) - 5 \equiv 2 \pmod 7.$$

And therefore $C = (2, 5) + (4, 0) = (2, 2)$.

3.2 Hasse's Theorem

One important requirement for selection of a curve $E(\mathbf{F}_q)$ for cryptography will be that $\#E(\mathbf{F}_q)$ is sufficiently large as to where, given $P, Q \in E(\mathbf{F}_q)$ such that $Q = kP$ for some large random integer k, one cannot reasonably find k. This concept will be discussed in more depth later, however it necessitates a method for efficiently determining the size of $\#E(\mathbf{F}_q)$ beyond the need for an initial test of possible group behavior as in *Theorem 3.2.* The first guideline will be an important result in the theory of elliptic curves over finite fields, *Hasse's Theorem.*

Theorem 3.3 (Hasse's Theorem)**.** Let E be an elliptic curve over the finite field $\mathbf{F}_q$. Then the order of $E(\mathbf{F}_q)$ satisfies

$$|q + 1 - \#E(\mathbf{F}_q)| \leq 2\sqrt{q}.$$

An accessible proof is available in [Was03], however we are more concerned with using the inequality as a stepping-stone to determining $\#E(\mathbf{F}_q)$ exactly. There will be three methods we will discuss for accomplishing this. The first is specific to a family of curves for which determining $\#E(\mathbf{F}_q)$ can be done directly by algebraic methods.

Theorem 3.4. Let E be an elliptic curve with Weierstrass equation $y^2 = x^3 - kx$ defined over $\mathbf{F}_q$, where p is an odd prime and k is an integer such that $k \not\equiv 0 \pmod{p}$. If $p \equiv 3 \pmod{4}$, then $\#E(\mathbf{F}_q) = p + 1$. If $p \equiv 1 \pmod{4}$, then, if $p = a^2 + b^2$, where a, $b \in \mathbb{Z}$, with b even and $a + b \equiv 1 \pmod{4}$, then

$$\#E(\mathbf{F}_q) = \begin{cases} p + 1 - 2a \text{ if } k \text{ is a fourth power modulo } p \\ p + 1 + 2a \text{ if } k \text{ is a square modulo } p \text{ but not a fourth power} \\ p + 1 \pm 2b \text{ if } k \text{ is not a square modulo } p \end{cases}$$

Note that this theorem applies to our toy example from the previous subsection, and as $7 \equiv 3 \pmod{4}$ it tells us that $\#E(\mathbf{F}_7) = 8$, just as we found by enumeration. The proof of this theorem is not particularly difficult but is quite tedious, and is given in full in [Was03]. However, we can quickly show that the results of *Theorem 3.4* align with the bound given in *Hasse's Theorem.*

Theorem 3.5. *Theorem 3.4* is consistent with *Hasse's Theorem.*

Proof. We wish to show that, for an elliptic curve E over a finite field $\mathbf{F}_q$ where $E = x^3 - kx$, with $k \not\equiv 0 \pmod{p}$, the $\#E(\mathbf{F}_q)$ satisfies

$$|p + 1 - \#E(\mathbf{F}_q)| \leq 2\sqrt{p}.$$

We have four cases. The first case occurs when $p \equiv 3 \pmod{4}$. By *Theorem 4.21*, we have that $\#E(\mathbf{F}_q) = p+1$. Therefore, we have that

$$|p + 1 - \#E(\mathbf{F}_q)| = |p + 1 - (p + 1)| = 0 \leq 2\sqrt{p},$$

and the case is shown.

By *Theorem 4.21*, if $p \equiv 1 \pmod{4}$, then we have one of the final three cases. First we write $p = a^2 + b^2$, where a, $b \in \mathbb{Z}$, b is even, and $a + b \equiv 1 \pmod{4}$. The cases proceed as follows. The second case gives

$\#E(\mathbf{F}_q) = p + 1 - 2a$. We therefore have

$$|p + 1 - \#E(\mathbf{F}_q)| = |p + 1 - (p + 1 - 2a)| = 2a,$$

and therefore it must be shown that $a \leq \sqrt{p}$ for the case to hold. If $b = 0$, then we have that $a^2 = p$ and therefore that $a = \pm\sqrt{p}$. Since if $|b| > 0$ then b^2 increases and therefore a^2 must decrease in magnitude, a can only be equal to or smaller than $\sqrt{p}$, and therefore the case is shown.

The third case gives $\#E(\mathbf{F}_q) = p+1-2a$. Since the argument for the second case bounds the magnitude of a the change of sign makes no difference, and so it holds for this case as well. The final case gives $\#E(\mathbf{F}_q) = p+1 \pm 2b$. The argument is identical as the arguments for the second and third cases, albeit with the roles of a and b swapped. As all four cases have been shown, *Hasse's Theorem* is satisfied. $\square$

Obviously there are many curves for we wish to determine $\#E(\mathbf{F}_q)$ with $E(\mathbf{F}_q)$ not limited to the family for which the previous result applies. A first method of accomplishing this employs *Lagrange's Theorem.*

Theorem 3.6 (Lagrange's Theorem)**.** Let H be a subgroup of the finite group G. Then, the order of H divides that of G.

Further, it can be shown for any $g \in G$, the set $\{\, g^m \mid m \in \mathbb{Z} \,\}$ constitutes a subgroup of order k, where k is the smallest positive integer such that $g^k = e$, where e is the identity element. In the context of elliptic curves, if we take a point $P \in E(\mathbf{F}_q)$ and determine the smallest integer k such that $kP = \infty$ then *Lagrange's Theorem* tells us that $k \mid \#E(\mathbf{F}_q)$. This k is known *as the order of the point P*.

This relationship between the order of the points on a curve and the order of the curve group provides a means by which we can compute $\#E(\mathbf{F}_q)$. Simply, if we can determine the orders of a sufficient number of points in $E(\mathbf{F}_q)$ then only one of the possible values of $\#E(\mathbf{F}_q)$ as given by *Hasse's Theorem* will be a multiple of each of them, uniquely determining the order of the group. An example of such an approach is a modified version of the `Baby Step Giant Step` algorithm. A general algorithm for computing discrete logarithms of elements of groups, in our elliptic curve setting we are interested in using it to find the order of a collection of $P_i \in E(\mathbf{F}_q)$, again the smallest $k_i \in \mathbb{Z}$ such that $k_i P_i = \infty$. From there $\#E(\mathbf{F}_q)$ can be determined as just discussed.

Although this method could be worked out directly, the `Baby Step Giant Step` algorithm uses some techniques to simplify and accelerate the process. Roughly speaking, it works by rewriting a k_i into the equivalence

$$(q + 1)P_i + x(2mP_i) = \pm jP_i,$$

where $j < m$ with $m > q^{1/4}$ randomly chosen, but still reasonably small compared to $q^{1/2}$ itself. After precomputing the possible values of jP_i, the algorithm then iterates through x from $-m$ to m to find a solution to the equivalence. The rewriting naturally excludes a number of invalid choices for k_i, and therefore the algorithm is able to greatly reduce the search space. When used to compute the orders of enough points to determine the order of $E(\mathbf{F}_q)$ the algorithm has running time of approximately $4q^{1/4}$ steps, as opposed to the $4q^{1/2}$ running time of naively trying all possible values for $\#E(\mathbf{F}_q)$ as bounded by *Hasse's Theorem.* For a full description of `Baby Step Giant Step` within the context of computing the order of points on elliptic curves, see [Was03].

However, although `Baby Step Giant Step` may be faster than a naive method of determining $\#E(\mathbf{F}_q)$, it still is inefficient if q becomes too large, especially in modern cryptographic contexts. A more efficient and specialized algorithm is `Schoof's Algorithm`, which has qualities similar to number theoretic sieves used in factoring algorithms. The method has a fair amount of (mathematical) complexity and relies on properties of elliptic curves which we have not discussed, and therefore a full explanation is beyond the scope of this discussion. However, the algorithm is quite efficient, and can be used in practice for determining the order of $E(\mathbf{F}_q)$ even as it reaches cryptographic magnitudes. The algorithm was original presented by Schoof in [Sch85], and he presented a general discussion of determining the order of elliptic curve groups, including `Baby Step Giant Step` as well as his own algorithm, in [Sch95]. In addition, [Was03] contains a discussion of the algorithm.

4 n-torsions and the Weil Pairing

Until now, our discussion has been limited solely to elliptic curve groups as contained algebraic objects. However, we will now turn to the Weil Pairing, a mapping which relates the structure of the elliptic curve group to that of the field underlying the curve, and which has important cryptographic uses. Before discussing it we will need to define a subset of the elliptic curve which will be a component of its domain. We will then give the high-level construction of the pairing, focusing on its qualities which will be relevant for the cryptographic purposes to which we will put it.

4.1 Torsion Points

Let E be an elliptic curve defined over a field K. Consider

$$E[n] = \{P \in E(\overline{K}) \mid nP = \infty\},$$

known as the set of *n-torsion points* of $E(\overline{K})$, where n is a positive integer and $\overline{K}$ is the algebraic closure of K.

Now consider a point $P \in E(\overline{K})$. As before, this point has order k, where k is the smallest positive integer such that $kP = \infty$. Since therefore $(k+1)P = P$, it follows as well that for all multiples of k their multiplication by P is also ∞. As such, a point P is an n-torsion for all $n = jk$, where k is its order and j is a positive integer. It can be shown that the n-torsion points carry a consistent and straightforward structure.

Theorem 4.1. Let E be an elliptic curve over a field K and let n be a positive integer. If the characteristic of K does not divide n (including if it is 0), then

$$E[n] \cong \mathbb{Z}_n \oplus \mathbb{Z}_n.$$

If the characteristic of K, p is non-zero and divides n, write $n = p^r n'$ where $p \nmid n'$. Then

$$E[n] \cong \mathbb{Z}_{n'} \oplus \mathbb{Z}_{n'},$$

or

$$E[n] \cong \mathbb{Z}_n \oplus \mathbb{Z}_{n'}.$$

A heuristic discussion of why this isomorphism holds, as well as formal proof (which requires some additional machinery), are given in [Was03]. As an example, we may consider the sets of n-torsion points for $E(\mathbf{F}_7)$ with E being $y^2 = x^3 - 2x$. The group as before is

$$E(\mathbf{F}_7) = \{(0,\ 0),\ (2,\ 2),\ (2,\ 5),\ (3,\ 0),\ (4,\ 0),\ (6,\ 1),\ (6,\ 6),\ \infty\}.$$

From the definition of point addition, our set of 2-torsion points are

$$E[2] = \{(0,\ 0),\ (3,\ 0),\ (4,\ 0), \infty\}.$$

By determining the order of all the points on the curve, the next few sets are

$$E[3] = \varnothing,$$

$$E[4] = \{\{(0,\ 0),\ (2,\ 2),\ (2,\ 5),\ (3,\ 0),\ (4,\ 0),\ (6,\ 1),\ (6,\ 6),\ \infty\},$$

$$E[5] = \varnothing,$$

and so on. Taking the mappings

$$\begin{array}{ccc} \infty & \mapsto & (0,0) \\ (3,0) & \mapsto & (1,0) \\ (4,0) & \mapsto & (1,1) \\ (0,0) & \mapsto & (0,1) \end{array}$$

it is straightforward to see that $E[2] \simeq \mathbb{Z}_2 \oplus \mathbb{Z}_2$, as in *Theorem 4.1*. Although the n-torsion points have broader uses, for our purposes we are interested in them as they are the elliptic curve analogue of n-th roots of unity. This correspondence will provide the basis for the construction of the Weil Pairing.

Note that the common definition of the n-torsion points as including points with coordinates set in the algebraic closure of K - instead of just those in K itself - is due to uses of the set which primarily fall beyond the scope of this discussion. For the remainder of our discussion, it is reasonable to restrict our view of the n-torsion points instead to the definition

$$E[n] = \{P \in E(K) \mid nP = \infty\},$$

with the necessary caveat that *Theorem 4.1* may no longer hold. In particular, it can be shown that if we restrict as described the definition of the n-torsion points used in the domain of the Weil Pairing, which we will soon meet, then the range of the pairing is similarly restricted from the roots of unity in $\overline{K}$ to those of K, see *Theorem 3.11* in [Was03]. Since this causes us no difficulty, for simplicity going forward we will use this altered definition of the n-torsion points.

4.2 Weil Pairing

The set of n-th roots of unity in K, denoted μ_n, are solutions to the equation

$$x^n = 1.$$

If the characteristic of K does not divide n then μ_n constitutes a cyclic group of order n, as $x^n = 1$ has no repeated roots. There is a natural concurrence between the qualities of n-torsion points in an elliptic curve $E(K)$ and the n-th roots of unity μ_n in K, as for $P \in E[n]$ we have $nP = \infty$ just as for $x \in \mu_n$ we have $x^n = 1$. This relationship underlies a mapping known as the Weil Pairing.

Theorem 4.2 (Weil Pairing)**.** Let $E(K)$ be an elliptic curve defined over a field, and let n be a positive integer. If the characteristic of K does not divide n, then there is a pairing

$$e : E[n] \times E[n] \to \mu_n,$$

with the following properties.

1. For all P, P_1, P_2, Q, Q_1, $Q_2 \in E[n]$,

$$e(P_1 + P_2, Q) = e(P_1, Q) \cdot e(P_2, Q),$$

$$e(P, Q_1 + Q_2) = e(P, Q_1) \cdot e(P, Q_2).$$

   This quality is known as bilinearity.

2. If $e(P, Q) = 1$ for all $P \in E[n]$ then $Q = \infty$, and if $e(P, Q) = 1$ for all $Q \in E[n]$ then $P = \infty$. This quality is known as non-degeneracy.

3. $e(P, P) = 1$ for all $P \in E[n]$.

4. $e(Q, P) = e(P, Q)^{-1}$ for all P, $Q \in E[n]$.

5. $e(\sigma P, \sigma Q) = \sigma(e(P, Q))$ for all automorphisms of $\overline{K}$ such that σ is the identity map on the coefficients of E (i.e., if $E = x^3 + ax + b$, then $\sigma(a) = a$ and $\sigma(b) = b$).

6. $e(\alpha(P), \alpha(Q)) = e(P, Q)^{deg(\alpha)}$ for all endomorphisms α of E.

The proof of the existence of the Weil Pairing - and a method for explicitly computing it - are beyond the scope of this discussion, however they are given in [Was03]. For our purposes, these properties will be sufficient to show the functionality of the pairing in cryptographic contexts.

The Weil Pairing plays a crucial role in many arguments and proofs relating to the structure of elliptic curves. It may be used to prove *Hasse's Theorem*, underlies some of the more effective methods for attacking discrete logarithms over elliptic curves, and may be employed for cryptographic purposes in the context of pairings-based cryptography. The ability of the Weil Pairing to relate some of the structure of a curve $E(K)$ to its underlying field - and therefore allow certain questions regarding the curve to be considered through the substantially more broad scope of K - is what gives the Weil Pairing such strength.

5 Cryptography over Elliptic Curves

Now we turn from an algebraic treatment of elliptic curves to their use in cryptography. This section is intended to take the form of a general introduction to asymmetric-key cryptography, but with a narrow focus towards cryptographic algorithms and protocols which have variants defined over elliptic curves. The section begins with a brief overview of the computational complexity concepts which underlie cryptography in general, before considering the aforementioned examples of specific constructions. The section then concludes with a discussion of the orders of groups considered suitable for implementations of ECC.

5.1 Cryptographic Fundamentals

The foundation of modern cryptography is the one-way function. A truly rigorous definition is beyond the scope of this discussion, but the crucial attribute of such a function is that while the computation of the function is efficient, an attempt to invert the function will fail except for with a negligible probability.

Definition 5.1 (One-Way Function (Rough Definition))**.** A function f is *one-way* if:

1. Easy to compute: There exists a deterministic polynomial-time algorithm `A` which on input x outputs $f(x)$.
2. Hard to invert: For every probabilistic polynomial-time algorithm `B`, the probability it will on input y output $f^{-1}(y)$ is bounded above by a negligible quantity.

The usefulness of one-way functions to cryptography is quite natural. Although the exact definition ascribed to cryptography often varies, the fundamental concern of the field is the securing of communication and informational systems against adversarial attack. Modern cryptographic techniques provide guarantees of security by connecting the ability of an adversary to successfully carry out an attack on a scheme to them needing to successfully invert an one-way function - something they are believed to be unable to do. This allows the conclusion that a successful break of the security is a mathematical contradiction. Although the validity of this conclusion is reliant upon the validity of the assumption of the hardness of inverting the one-way function, this approach provides a sufficiently firm foundation for belief in cryptographic arguments.

An important note on the terminology of the definition is the use of “easy” (or, as will be used synonymously going forward, “efficient”) and “hard” (or “inefficient”) as rough stand-ins for *polynomial-time* and *non-deterministic polynomial time* respectively. Broadly speaking, *polynomial-time* means the time demands for computing the function scales sufficiently well with the size of the input as to likely be feasible to practically compute. On the other hand, *non-deterministic polynomial-time* algorithms will likely be too large or long to compute in practice for relatively small inputs. This sketch of these ideas will be sufficient for our purposes, see [Sip06] or [Gol08] for a general discussion of computational complexity, and [Gol06] or [LK15] for discussion specifically focused on one-way functions.

For the purposes of encryption and digital signatures - two areas within which elliptic curves are employed for cryptographic purposes - an extension of the one-way function is necessary. This is a trapdoor function, which is a one-way function with the additional attribute that there does exist an efficient algorithm for inversion, provided it also receives a specific auxiliary input.

Definition 5.2 (Trapdoor Function (Rough Definition))**.** A function g is deemed *trapdoor* if:

1. Easy to compute: There exists a deterministic polynomial-time algorithm $\texttt{A}$ which on input x outputs $g(x)$.
2. Hard to invert except with trapdoor: For every probabilistic polynomial-time algorithm $\texttt{B}$, the probability it will on input y output $g^{-1}(y)$ is bounded above by a negligible quantity. However, there exists a deterministic polynomial-time algorithm $\texttt{A}_{\texttt{rev}}$ which on input (y, i) for a specific choice of i outputs $g^{-1}(y)$.

An additional requirement, that i cannot be efficiently computed from g but (g, i) pairs can be efficiently computed together, allows i to function as a "secret" or "key" even if g is made public. It should also be noted that there is no general restriction on the number of inputs to the functions or algorithms. The unary form of the rough definitions given is purely for simplicity.

An example use of a trapdoor function is for asymmetric key cryptography. Computational security being equal, modern symmetric encryption algorithms - in which $g^{-1}(y)$ is computed by running the algorithm for $g(x)$ backwards with the same key - are orders of magnitude more efficient than asymmetric key algorithms in which the processes for computing $g(x)$ and $g^{-1}(y)$ are different. However, as the former require both sender and recipient to have the same key, they carry a chicken and an egg problem, as a secure channel is required to transmit the key for the purposes of creating a secure channel. Asymmetric key algorithms have no such key distribution problem. Abstractly, in an asymmetric key cryptosystem Alice determines a (g, i) pair and makes g public. Bob writes his message p_B and computes $g(p_B) = c_B$ using $\texttt{A}$, and sends it to Alice. Eve, intercepting the message, is unable to invert c_B to determine p_B or to determine i from g, however Alice is able to use i as an auxiliary input to $\texttt{A}_{\texttt{rev}}$ to compute $g^{-1}(c_B) = p_B$ and recover Bob's original message.

Although numerous one-way and trapdoor functions form the basis of the many cryptographic primitives, those which underlie the constructions most commonly ported to elliptic curves use a form of the Discrete Logarithm Problem (DLP) as the hard component. The general form of the DLP refers to inverting the repeated application of a group operation. The classic example - a necessary hurdle in most grade school math curricula - is inverting exponentiation over the real numbers, i.e. if one knows that $a^k = b$, then given a and b, determine k. Strictly speaking the DLP is not known to be - and actually is not believed to be - hard in the formal sense, and efficient logarithm algorithms are known in some circumstances. However, in many cases they are not, and in these cases the DLP may be used for cryptographic purposes as though it were the converse of a proper one-way function. One such case is the relevant one to this discussion, inverting repeated point addition on an elliptic curve over a finite field, formally the Elliptic Curve Discrete Logarithm Problem (ECDLP).

Definition 5.3 (Elliptic Curve Discrete Logarithm Problem (ECDLP))**.** Given a pair of points P, Q such that $Q = kP$, determine k.

It is by this mechanism that common modern cryptographic algorithms have been altered to work with elliptic curves. The following algorithms all use the DLP as the basis for their security, and were originally defined over the multiplicative group of some finite field $\mathbf{F}_q^\times$. Their elliptic curve variants are simply the straightforward alterations necessary for the alternate algebraic structure underlying them.

5.2 Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange (DHE) is a key distribution scheme. The key distribution problem for symmetric key algorithms, as briefly noted earlier, must be solved for a cryptographic protocol to utilize their efficiency. One solution is for one party to generate a key for the symmetric algorithm and then send it to the other encrypted by an asymmetric algorithm. Another approach are algorithms such as DHE, in which both parties join together to create a shared secret from which a key can be derived, and do so in a way such that a malicious observer cannot efficiently compute the secret themselves. As such, DHE functions as a cryptographic primitive used for key distribution in a larger cryptographic protocol. It was originally described for a general discrete logarithm setting in [DH76]. Its ECDHE variant was introduced alongside one of the initial treatments of ECC by Miller in [Mil86], however Koblitz's discussion of both ECDHE and ECC in general in [Kob87a] is much more consistent with how they have since become generally discussed, and both authors are usually credited as having co-introduced elliptic curve cryptography.

For Elliptic Curve Diffie-Hellman Key Exchange (ECDHE), two parties compute a shared secret by the following algorithm.

Protocol 5.4 (Elliptic Curve Diffie-Hellman Key Exchange)**.** Two parties agree upon a curve $E(\mathbf{F}_q)$ (where $\mathbf{F}_q$ is a finite field) such that the ECDLP is hard in $E(\mathbf{F}_q)$. They then select a point $P \in E(\mathbf{F}_q)$ such that the subgroup generated by P has large prime order. Then they execute the following algorithm:

1. The first party randomly chooses a secret integer a, and sends aP to the second.
2. The second party randomly chooses a secret integer b, and sends bP to the first.
3. The first party computes $a(bP) = abP$.
4. The second party computes $b(aP) = baP$.
5. As $abP = baP$, both parties have a shared secret.

An eavesdropper can see $E(\mathbf{F}_q)$, P, aP and bP, since they are sent in the clear. Therefore, for ECDHE to be secure, they must not be able to solve what is known as the Diffie-Hellman Problem (DHP). Its elliptic curve formulation is as follows.

Definition 5.5 (Elliptic Curve Diffie-Hellman Problem (ECDHP))**.** Given $E(\mathbf{F}_q)$, P, aP and bP, find abP.

If the eavesdropper can solve the ECDLP then they can compute a from P and aP, and with that compute $a(bP) = abP$. Therefore the hardness of the ECDLP is a required assumption for the security of the protocol. As such, the ECDHP can serve as a alternative, stronger requirement for the security of ECDHE then the ECDLP, as an attacker's inability to solve the ECDHP subsumes into it their inability to solve the former problem. Showing that an attacker in unable to solve the ECDHP is a common technique in proving the security of any discrete logarithm-based algorithm over elliptic curves. Note that this discussion, with the EC-prefix dropped, follows for any cryptographic construction based on the discrete logarithm problem.

In addition to the ECDHP itself, there are a set of associated problems - the Diffie-Hellman problems - which are frequently used in arguments regarding the security guarantees of cryptographic protocols which rely on the ECDLP (or more generally the DLP). This is due to their providing notions of partial compromise which consideration of the ECDHP doesn't confront. For instance, even if an attacker cannot completely determine abP there is nothing in the definition of the ECDHP which prevents them from learning partial information about it - which may allow that attacker to determine some information about the plaintext of a message encrypted using a key derived from it. An analogy, which happens to play an important historical role in the development of this theory, is that if one uses an algorithm based on the ECDHP to encrypt playing cards an attacker may still be able to determine the suit of an encrypted card without being able to fully decrypt it. The aforementioned associated problems each impose stronger requirements, which restrict the extent to which the attacker can determine any useful information. One of these problems is the Decisional Diffie-Hellman Problem (DDH).

Definition 5.6 (Elliptic Curve Decisional Diffie-Hellman Problem (ECDDH))**.** Given P, aP, $bP \in E(\mathbf{F}_q)$, and given $Q \in E(\mathbf{F}_q)$, determine if $Q = abP$.

In other words, if hypothetically Eve has some reason to suspect that $Q = abP$ then can she confirm it? As it turns out, for certain choices of $E(\mathbf{F}_q)$, clever use of the Weil Pairing allows for confirmation of the validity of Q, see [Was03]. If Eve cannot solve the ECDDH, her inability to confirm the validity of any information she might serendipitously obtain about the plaintext provides a strong heuristic as to why her ability to actively determine any information about the plaintext herself is severely limited, as the latter is an intuitively stronger capability than the former - which intuition can be made rigorous. However, even in the opposite case where Eve is able to solve the ECDDH, this weakness in what is deemed the *semantic security* - the ability of an attacker to gain knowledge of the plaintext without directly breaking the encryption - of elliptic curve-based cryptosystems is not of grave concern. This specific weakness can be avoided by careful selection of $E(\mathbf{F}_q)$, and generally many encryption algorithms are provided with imperfect semantic security by their underlying one-way functions, and various methods - usually the introduction of some random component into the plaintext - can be employed to prevent an attacker from using the information gleaned in a useful manner. The ECDDH is a quite strong requirement, and depending on the security guarantee in question the weaker requirements specified by another Diffie-Hellman problem may be sufficient to build a secure scheme even in a setting where the ECDDH is easy. An example is the Boneh-Franklin Cryptosystem, which will be discussed in the next section. For further discussion of the Diffie-Hellman problems, see [Sti05] or [LK15].

5.3 ElGamal Encryption and Digital Signatures

Another example of a modern cryptographic structure which has been altered to work over an elliptic curve $E(\mathbf{F}_q)$ is ElGamal, an asymmetric key algorithm which can be used for both encryption and digital signatures. The original ElGamal algorithm was specified in [ElG85], while its elliptic curve formulation was discussed by Koblitz in both a paper [Kob87b] and book [Kob87a] published roughly concurrently.

Suppose a party wishes to use the encryption variant of ElGamal to allow other parties to communicate with them securely. First, they must generate a key pair, to do which they choose a curve $E(\mathbf{F}_q)$ within which the ECDLP is hard and then select a point P such that the subgroup generated by P has large prime order. They then randomly choose an integer k and compute $Q = kP$. Finally, they publish $(P, Q, E(\mathbf{F}_q))$ as their public key, and keep k secret as their private key.

To encrypt a message to a recipient, the sender then does following.

Algorithm 5.7 (`Elliptic Curve ElGamal - Encryption`)**.** Input is $(P_R, Q_R, E(\mathbf{F}_q)_R)$, which are the components of a public key generated and distributed by the recipient. The sender takes their message, m, and uses an agreed upon method to express m as the plaintext, a point $M \in E(\mathbf{F}_q)_R$. They then select a random $r \in \mathbb{Z}$ and compute $C_1 = rP_R$ and $C_2 = M + rQ_R$. The output is (C_1, C_2).

Receiving the (C_1, C_2) pair, the recipient recovers the plaintext $M = C_2 - k_R C_1$, from which they can determine m. The recovery works as

$$C_2 - k_R C_1 = (M + rQ_R) - k_R(rP_R) = M + r(k_R P_R) - k_R(rP_R) = M.$$

An attacker can recover the plaintext if they can determine either k_R or r, either of which requires the ability to solve the ECDLP. An important note is if the sender reuses a choice of r for two messages m_1, m_2 - something the attacker will be made aware of since C_1 will be the same in both cases - and the attacker learns or is able to guess either plaintext, it allows them to trivially recover the other message, see [Was03]. As such, it is imperative that different choices of r be used for different messages, and as some naive methods for deterministically producing choices of r by simple algorithms omit attacks, the most effective technique is for it to be chosen randomly.

In addition to its use as an encryption algorithm, ElGamal can also be used for the purpose of digital signatures. Much as how a physical signature is intended to authenticate a document based on its uniqueness, a digital signature confirms that a message was created by the holder of a specific private key and was not forged by a malicious party. For instance, if Alice wishes to send Bob a message so that he may confirm it came from her and not from some other party impersonating her, she can sign the message with her private key. Then - provided Bob can be reasonably certain that Alice is the only party with access to the key - he can trust the message originated with her and hasn't been garbled or maliciously altered in transit. Crucially, the former is the extent of the guarantee of a digital signature with regards to authenticating the author, that it is crafted by the holder of a specific private key. If Alice's private key is stolen or accidentally made public then whomever has access to it can sign any message and Bob will believe the message to be authentic. In addition, if an attacker Eve can convince Bob that she is in fact Alice, she can then use her own key to authenticate messages she creates and Bob will believe them to have been created by Alice. Methods that attempt to guarantee the eponymity of an identity with a key holder fall under the realm of entity authentication, whereas digital signatures are only a form of message authentication.

Instead of looking directly at the Elliptic Curve ElGamal Signature Scheme (ECESS), it is easier to consider a slight modification to it which was standardized into the Elliptic Curve Digital Signature Algorithm (ECDSA), put forth by the National Institute of Standards and Technology (NIST) as a common primitive for digital signatures [Lab13]. The ECDSA is more efficient than the ECESS, and the latter requires more maneuvering to successfully sign a message $m \in \mathbb{Z}$ over a curve $E(\mathbf{F}_q)$. To use the ECDSA, the signer first generates a key pair by the following process. They choose a curve $E(\mathbf{F}_q)$ over which the ECDLP is hard and $\#E(\mathbf{F}_q) = jp$, where p is a large prime and j is a small integer (< 10). They then choose a point $P \in E(\mathbf{F}_q)$ such that P has order p, and lastly they choose a private key $k \in \mathbb{Z}$ and compute $Q = kP$. Their public key is then the 4-tuple $(P,\ Q,\ E(\mathbf{F}_q),\ p)$. A signature is then constructed by the following algorithm.

Algorithm 5.8 (`Elliptic Curve Digital Signature Algorithm - Signing`)**.** Input is $(k_S,\ (P_S,\ Q_S,\ E(\mathbf{F}_q)_S,\ p_S))$, which are the private key and public key respectively generated by the signer. They then execute the following steps.

1. The signer composes their message m_L, and then computes $m_H = \text{SHA-1}(m_L)$, where SHA-1 is a specific hashing algorithm (see further discussion).
2. The signer randomly chooses an integer r and computes $S = rP_S = (x,\ y)$.
3. The signer computes $s \equiv r^{-1}(m_H + k_S x) \pmod{p_S}$.

The output is the signature $\sigma = (m_H, S, s)$. Note that $m_H,\ s \in \mathbb{Z}$, whereas $S \in E(\mathbf{F}_q)_S$.

The verifier may then determine the validity of the signature on receipt.

Algorithm 5.9 (`Elliptic Curve Digital Signature Algorithm - Verification`)**.** Input is $(\sigma,\ (P_S,\ Q_S,\ E(\mathbf{F}_q)_S,\ p_S))$. The verifier computes the quantities $t_1 = s^{-1}m_H \pmod{p_S}$ and $t_2 = s^{-1}x \pmod{p_S}$. They then use them to compute $T = t_1P_S + t_2Q_S$. If $T = S$, the algorithm outputs *accept*, else it outputs *reject*.

The verification process works since

$$\begin{aligned}
T &= u_1P_S + u_2Q_S \\
&= s^{-1}m_HP_S + s^{-1}xQ_S \\
&= s^{-1}(m_HP_S + xk_SP_S) \\
&= s^{-1}(m_H + xk_S)P_S \\
&= (r^{-1}(m_H + xk_S))^{-1}(m_H + xk_S)P_S \\
&= r(m_H + xk_S)^{-1}(m_H + xk_S)P_S \\
&= rP_S \\
&= S.
\end{aligned}$$

First of all, note that the use of the asymmetric quality is reversed in a signature scheme. For encryption, the sender encrypts with the public key of the recipient and the recipient decrypts with their private key. In a signature scheme, the sender signs with their own private key and the recipient verifies with the public key of the sender. In both cases, the process relies on the privacy of the private key and the bound nature of a key pair. The former guarantees that only the holder can decrypt an encrypted message or sign a message, while the latter allows for the sender to encrypt the message or verify a signature.

Suppose an attacker Eve wishes to violate the message authentication given by the signature scheme on a message signed by Alice and sent to Bob. The question at hand is whether Eve can successfully forge a message, i.e. can she determine a (m_H, S, s) tuple such that Bob will accept the signature as being created by Alice? Clearly, if Eve can solve the ECDLP she can forge a signature on any message, since she can determine k from P and Q - as both are public - and from there can compose a valid signature on any message she pleases by the same manner as Alice would herself. In addition, Eve could attempt to directly find an m_H, r pairing that verifies. As she is likely looking to forge a signature on a meaningful message, choosing an m_H and determining an r such that s and S have the correct relationship would work, however without knowledge of k no efficient method for accomplishing this is known.

As noted, the ECDSA differs from the ECESS in a few ways. The first is the explicit use of a message digest. A common alteration to signature algorithms and schemes, instead of signing the message m directly, Alice signs a digest $h(m)$, where $h : \{0, 1\}^* \to \{0, 1\}^\ell$ is a cryptographically secure hash function. The ECDSA builds this directly into the algorithm - choosing the SHA-1 hash function which was standardized by the NIST in [Lab12] - as opposed to ECESS, which does not require the use of a message digest or specify an algorithm if one does choose to use one. A cryptographically secure hash function takes input of (ideally) arbitrary length and outputs a digest of length ℓ, with the process being irreversible, i.e., given a digest $h(m)$ it is not possible to determine m. Technically, many such functions have a specific upper bound on input length, but with the bound so large to be irrelevant in practice, for instance, SHA-1 has a maximum input size of 2^{64} [Lab12]. The security of a cryptographic hash function can be reduced to *collision-resistance*, a property such that a hash function h is cryptographically secure if an attacker cannot solve the following problem.

Definition 5.10 (Collision-Resistance Problem)**.** For a hash function h, find $m_1 \neq m_2$ such that $h(m_1) = h(m_2)$.

See [Sti05] or [LK15] for further discussion.

One benefit of the use of hash functions is that $h(m)$ has fixed length output, allowing better determination of the efficiency of the algorithm and no concerns if the length of the message renders it not a member of the field over which the signature can be taken - e.g., in the case of ECDSA, if m was larger than p_S. However, their primary benefit is in confirmation of message integrity. Many software package distribution systems, for instance, require the distributors of packages to upload a digitally signed hash of the package with it. When a client then downloads the package, the distribution software computes the hash of the downloaded data and compares it against the signed hash. If they match, the software - and by extension the user running it -

can be certain the package they have downloaded was the one distributed by the author and not substituted for by a malicious host or altered in transit. Generally speaking, signed hashes provide a concise manner by which the recipient of a message can confirm it was the one that was sent, as well as confirm who sent the message or data - at least up to key ownership as discussed previously. Note that the use of signed hashes for providing message integrity is an asymmetric key approach. As with encryption, symmetric key cryptography provides a more efficient class of algorithms known as message authentication codes, or MACs.

In addition to the use of a message digest, the ECDSA differs from the ECESS in how it signs a message in $\mathbb{Z}$ over an elliptic curve $E(\mathbf{F}_q)$. In the ECESS, a function $f : E(\mathbf{F}_q) \to \mathbb{Z}$ which is not explicitly defined must be used to take the structure of the elliptic curve to $\mathbb{Z}$. This arbitrary choice may cause concerns with regards to the efficiency of the algorithm, the standardization of the algorithm across software suites, and possibly the security of the algorithm - since it cannot be generally evaluated unless assumptions about the choice of f are made. In ECDSA, this is explicitly defined as $f(rP) = x$ where $rP = (x, y)$. In addition, the predefinition of this function allows for a different verification algorithm which is more efficient, see [Was03].

One last note on signature schemes is that in many cases Alice will wish to encrypt the message m in addition to signing it. She can do this using any encryption scheme, not just ElGamal. Of concern is whether it is better to encrypt the message first and then sign, or sign the message and then encrypt. Generally speaking either method can work, but there are some important and subtle considerations, see [LK15] for further discussion.

5.4 Group Order of Curves in Cryptographic Contexts

The cardinality of elliptic curve groups - as was focused on in *Section 3* - has been referred to consistently in the preceding discussion. It has primarily arisen with regards to demands that any curve used for cryptographic purposes must be of sufficient size - meaning it must contain a subgroup large enough for the ECDLP to be difficult within it. Even though there may not be a general solution to the ECDLP there are any number of algorithms - from the naive approach of trying everything to constructions of substantial mathematical complexity - which are able to solve it for groups of insufficient but non-trivial order in a practical timeframe. Although a vague notion that suitable curves exist and can be defined and used has been sufficient for the discussion until now, it is worth considering the rough guidelines surrounding such curves explicitly. Doing so also provides an explanation for why ECC is more efficient than most other forms of public-key cryptography, an important capability for embedded systems and smart card contexts.

First, it is worth considering at what order solving the ECDLP becomes infeasible. The most effective general method for attacking the ECDLP over arbitrary curves is the elliptic curve variant of *Pollard ρ*, a standard method for solving the DLP problem whose elliptic curve formulation is discussed in [Was03]. In 2009 Bos, et. al. used the algorithm and a distributed network of PlayStation 3 hardware to successfully recover a key in *secp112r1* - a 112-bit NIST standardized curve, meaning the length of the binary representation of the order of the group was 112 digits - a process which took over three and a half months [BKK$^+$09]. In their accompanying paper the authors considered it unlikely a serious public attempt could be made at attacking a 160-bit curve before 2020, and although it is likely that state actors are greatly ahead of the public in terms of computational efforts, this result does provide a general guideline for safe curve sizes assuming that mathematical capabilities remain relatively constant. The NIST currently standardizes curves for ECDSA with 192, 224, 256, 384, and 521-bit sizes [Lab13], although the NSA cites 384-bit as the minimum for their Suite B standard [Age]. For a general overview of the current state of active elliptic curve use, see [BHH$^+$14].

The efficiency of ECC compared to other asymmetric key cryptographic options stems from the additional complexity of point addition, as compared to the group operations underlying algorithms set in for instance $\mathbb{Z}_n$. This leads to security equivalences such as RSA-4096 - where the the length of the binary representation of the order of the group $\mathbb{Z}_n$ is 4096 bits - having roughly equal security to an algorithm set over an 133-bit elliptic curve [BSS99]. In situations where memory space is severely limited, the much smaller representations of ECC-related quantities can drastically increase the efficiency of the algorithm in practice as well as make key generation easier.

An interesting addendum to this discussion is that there are alternate formulations of curve equations from which any curve can be easily transformed into Weierstrass form, or vice-versa. The two most notable for cryptographic purposes of these alternate forms are Montgomery curves - where $E(K)$ is defined by the equation $by^2 = x^3 + ax^2 + x$ with $a, b \in K$ - and Edwards curves - for which the equations are of the form $x^2 + y^2 = 1 + dx^2y^2$ with $d \in K \setminus \{0, 1\}$. These formulations have algorithms for computing point addition which have marked differences - and of particular usefulness often drastic accelerations - in comparison to their equivalent Weierstrass algorithms. Implementations therefore gain an expedition which makes them more practical. As such, much of the newest development work in ECC is being done with Montgomery and Edwards curves, and these curves have slightly different comparisons when it comes to efficiency and security as the preceding discussion in this subsection.

One last important caveat to ECC is that not all curves carry the same level of security, even those with a similar number of points on them. This is due to certain curves having structures which are susceptible against certain attacks - of both the mathematical and implementation variety - which may weaken or break them. Since individual curves must be selected and standardized to base cryptographic algorithms and protocols off of, choosing good quality curves to use as the setting for a cryptographic construction is a necessary part of ECC. Choosing weak curves - including those which may have been backdoored by the authorities which standardized them - renders otherwise well-designed constructions insecure. A good example is the MOV Attack, where a certain curve choice allows for use of the Weil Pairing to attack the DLP in an easier setting, see [MOV93], or [Was03] for a high-level overview. For reference, Daniel J. Bernstein and Tanja Lange maintain SafeCurves, a website where they keep information on the security guarantees and current state of curve development [BL15].

6 Pairings-based Cryptography over Elliptic Curves

The final section of our discussion will regard a collection of encryption methods which fall under the banner of pairings-based cryptography. These constructions provide an additional guarantee to encryption - entity authentication - which is a cryptographic assertion as to the ownership of a key pair by an entity. The section begins with a discussion of an initial scheme which provides both encryption and entity authentication, before considering three other schemes with various other attributes.

6.1 Identity-based Encryption (Boneh-Franklin Cryptosystem)

Previously, we noted that digital signatures allow for confirmation of message authorship up to the holding of a private key. The issue then becomes how to match a public key - which is bound to the private key - with some conception of an identity and to the entity represented by that identity? Although a number of entity authentication methods exist, most have an inconvenience and arguable weakness in that they do not directly link the key pair with the entity. Instead, they require a certificate or similar mechanism where an authority - who is trusted by all entities in the system - asserts the connections between the key pair, identity, and entity in a verifiable manner. The issue with this approach is that it requires the sender of the message to actively acquire the certificate and the other information necessary to verify the certificate - a process which can be computationally expensive and in some cases expose implementation vulnerabilities. An example weakness is the certificate revocation problem, that it is difficult for the authority to revoke the validity of certificates in a timely and effective manner, for more on which see [Riv98, MR01].

Identity-based Encryption (IBE) is a method in which the public key is an identity - an explicitly chosen quantity which reflects the entity in some manner, say an email or a driver or commercial license - which is then certified as being eponymous with the entity by a trusted authority. To send a message to an entity, the sender determines the identity of the recipient and uses that publicly available information to encrypt the message and send it to them. Then, only if that identity has been certified as valid by a trusted authority will the recipient be able to decrypt it. Assuming the trust placed by the sender in the authority is honored and that the identity they are sending the message to is unique, the sender can be certain that only the intended recipient can decrypt the message. There are some additional important considerations to IBE. First, since the public key is an arbitrary string of information it is also possible to encode more then just an identity into it, with such extensions including access rights and expiration dates. For instance, one could construct a public key "bob@example.org#exp:01.01.2101#perm:Top_Secret", so that anyone sending Bob a message knows he has top secret clearance through to the beginning of the next century. Also, since the entity authentication process is handled through encryption IBE makes signatures redundant, reducing the computational overhead of the process for messages which need guarantees of both confidentiality and authorship.

One of the first IBE schemes was presented by Boneh and Franklin. Although it may generally be defined for any pairing $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ between two groups $\mathbb{G}_1$, $\mathbb{G}_2$, we will consider it with a slightly modified version of the Weil Pairing to set it over elliptic curves. First, let $E(\mathbf{F}_p)$ be an elliptic curve with Weierstrass equation $y^2 = x^3 + 1$ and p a large prime where $p \equiv 2 \pmod 3$. Let $\omega \in \mathbf{F}_{p^2}$ be a primitive third root of unity, and define $\beta : E(\mathbf{F}_{p^2}) \to E(\mathbf{F}_{p^2})$ by

$$\beta(P) = \beta(x, y) = (\omega x, y),$$

$$\beta(\infty) = \infty.$$

Note that the order of P is the same as the order of $\beta(P)$ due to ω being a third root of unity. We can then construct a modified Weil Pairing for which $e'(P, Q) = e(P, \beta(Q))$. It may be shown that $E(\mathbf{F}_p)$ has order $p+1$, and if we choose p such that $p = 6i - 1$ for some prime i then $6p$ has order i or 1 for each $P \in E(\mathbf{F}_p)$.

The following description of the algorithm is slightly simplified and omits certain attacks, but is a sufficient sketch of the practical construction of the cryptosystem. Strictly speaking, this version of the algorithm has a level of security deemed chosen plaintext indistinguishability (IND-CPA). This requires that the cryptosystem is secure if an attacker can create arbitrarily many plaintexts and view their respective ciphertexts under encryption. However, the standard for public-key cryptography is the stronger chosen ciphertext indistinguishability (IND-CCA), which requires being secure if an attacker can create arbitrary ciphertexts and view them *under decryption*. In this case "secure" roughly means that an attacker cannot distinguish which of two possible inputs corresponds to a specific output of the function - encryption for IND-CPA, decryption for IND-CCA. A cryptographic construction providing this indistinguishability gives it semantic security, as discussed previously. The true Boneh-Franklin scheme uses a transformation to build the IND-CCA secure scheme out of the IND-CPA one given here, although the final difference between the schemes is relatively minor. The reason for giving the simplified version is that the full version of the scheme has some components which seem unnecessary without a deeper understanding of the techniques used in the proof of security for the cryptosystem, which might be misleading. For more, see the original paper at [BF03].

As in the general IBE discussion, the cryptosystem requires an authority who is trusted by the various entities which wish to hold or verify any public key it issues. The authority must be able to securely communicate with any entity to which it intends to issue keys and also must be able to function as a registration authority, i.e. be able to confirm that the entity is eponymous with the claimed identity. This process - of the authority issuing keys and creating a mechanism by which encryption and decryption can happen in an authorized manner - is known as an infrastructure. Its members are the authority and any entities wishing to use it to send or receive messages. Before any keys can be issued, the trusted authority (TA) does as follows.

Algorithm 6.1 (`Boneh-Franklin Cryptosystem - Authority Setup`)**.** An authority initializes the infrastructure as follows.

1. The TA chooses a large prime $p = 6i - 1$ with i prime where $p \equiv 2 \pmod 3$, as well as a curve $E(\mathbf{F}_p)$ with equation $y^2 = x^3 + 1$.

2. The TA chooses a point P of order i in $E(\mathbf{F}_p)$.

3. The TA chooses two hash functions. The first is $H_1 : \{0, 1\}^* \to E(\mathbf{F}_p)$, where $\mathbf{Im}(f)$ is composed of points of order i. The second is $H_2 : \mathbf{F}_{p^2}^{\times} \to \{0, 1\}^n$, where $\mathbf{Dom}(g)$ composed of points of order i, and n is a positive integer, which will need to be equal to or greater than the length of any plaintext.

4. The TA chooses a random $k \in \mathbf{F}_i^{\times}$ and computes $P_{pub} = kP$.

The output is $(k, (p, H_1, H_2, n, P, P_{pub}))$, where k is kept secret and the 6-tuple is made public and distributed.

If an entity with an identity string - for instance $ID = \text{id@example.org}_b$ where the b subscript denotes its binary representation - wishes to join the infrastructure and have a key pair issued to them, the trusted authority executes the following.

Algorithm 6.2 (`Boneh-Franklin Cryptosystem - Key Generation`)**.** Input is $(k,\, H_1,\, ID)$ where ID is the identity string claimed. After confirming they are communicating with the entity eponymous with ID, the TA computes $Q_{ID} = H_1(ID)$ and $D_{ID} = kQ_{ID}$. The output is $(D_{ID},\, Q_{ID})$ which is securely communicated to the entity. The private key is D_{ID} and is kept secret by the entity, while Q_{ID} is the public key and is distributed.

Now another entity, wishing to send a message to the one eponymous with ID, does the following. Note that the symbol $\oplus$, also referred to as XOR, corresponds to the application of the binary *exclusive or* logical operator to input bit streams, or equivalently to their addition in $\mathbb{Z}_2$.

Algorithm 6.3 (`Boneh-Franklin Cryptosystem - Encryption`)**.** Input is $(m,\, ID,\, H_1,\, H_2,\, P,\, P_{pub})$, where m is the message the sender wishes to communicate to the entity known by ID. The sender first computes $Q_{ID} = H_1(ID)$. They then choose a random $r \in \mathbf{F}_i^{\times}$ and compute $x_{ID} = e'(Q_{ID},\, P_{pub})$, where e' is the modified Weil Pairing. The sender then computes $c = (u,\, v) = (rP,\, m \oplus H_2(x_{ID}^r))$, note that $u \in E(\mathbf{F}_p)$ while $v \in \{0,\, 1\}^n$. The output is c, which the sender then sends to the recipient.

Receiving c, the recipient decrypts as follows.

Algorithm 6.4 (`Boneh-Franklin Cryptosystem - Decryption`)**.** Input is $(D_{ID},\, c)$. The recipient computes $y_{ID} = e'(D_{ID},\, u)$ and then $m = v \oplus H_2(y_{ID})$. The output is m.

The decryption works due to two equivalences, first

$$e'(D_{ID},\, u) = e'(kQ_{ID},\, rP) = e'(Q_{ID},\, P)^{kr} = e'(Q_{ID}, kP)^r = e'(Q_{ID}, P_{pub})^r = x_{ID}^r,$$

which follows from the properties of the Weil Pairing, which continue to hold in its modified form. Using this $e'(D_{ID},\, u) = x_{ID}^r$, it follows that

$$v \oplus g(e'(D_{ID},\, u)) = (m \oplus h_2(x_{ID}^r)) \oplus h_2(x_{ID}^r) = m.$$

In effect, the Boneh-Franklin Cryptosystem functions as an asymmetric key version of what is known as a stream cipher, as it creates a stream of bits which are then XOR'd with the ciphertext. The sender - relying on their trust in the TA - can be confident that only the entity eponymous with ID can decrypt their message. In addition, instead of having to use a certificate or trust a database maintainer, they can simply send their message to the holder knowing they won't be able to decrypt it unless they were certified as the proper entity by the TA.

6.2 Attribute-based Encryption

Entity authentication - both in cryptographic contexts and in our broader society - is often based on the concept of a singular identity where we prove our ability to do something by proving the entirety of who we are. However, this approach is a double-edged sword as it allows for a complete record of which entities have access to information or are responsible for a certain action. On the positive this allows for complete consideration as to the worthiness of an uniquely identified entity to have certain roles and privileges, delineates the permissions directly to the entity, and enables a record of events where actions can be traced back to a discrete responsible party. On the negative this allows for indisputable surveillance and tracking, as well as requiring redundant delineation of privileges, e.g. if any of age person can buy alcohol, it is substantially more efficient to certify that "anyone over 21 can purchase alcohol" rather than certifying that "Joe Smith, 42, of Detroit, MI, SSN: 123-45-6789 can purchase wine". These issues beget attribute-based authentication, where instead of asserting an identity which carries certain documentation, rights, and privileges, instead one asserts only whichever of these attributes are necessary for the purpose at hand. A natural question then becomes whether it is possible to extend the IBE model to Attribute-based Encryption (ABE), where the encryption process asserts not that the holder of a public key is a certain entity, but rather that they have a certain set of attributes.

The answer is yes, but first we must consider an important stepping-stone between IBE and ABE, Fuzzy Identity-based Encryption (FIBE). FIBE was developed to extend IBE so that sufficiently close identities - as defined by some measure - could all be paired with the same private key. Sahai and Waters, who first introduced such a scheme in [SW05], gave the example of biometric data. Due to natural noise, any two readings of the same biometric identifier, such as an iris scan, will probably not be exactly alike. However, if we can define an error measure within which range any two iris scans are likely to be from the same individual, then FIBE allows for the data from each of those scans to function as valid public keys for the same private key pair. In more general terms, if we have a private key ω, an FIBE scheme allows for any identity ω'_i to encrypt a message which will successfully be decrypted by ω, provided that $|\omega - \omega'_i| \leq m$, where "less than or equal to" is well-defined under some measure and m is some constant parameter. Although the actual construction of Sahai and Waters' scheme is unnecessary for this discussion, the idea is that each private key is generated as a list of components. If the number of components shared between a private key and a given public key is greater than some specified error-tolerance parameter, then the decryption process for a message encrypted with the public key will successfully execute.

ABE is then a natural extension of FIBE, where each of the different ω'_i represents a given attribute, along with an efficient mechanism to combine these together into a single key that corresponds to a set of attributes. The explicit constructions for ABE are in fact generally more powerful, allowing for an access tree expressing a basic logic to be encoded into the keys as well, allowing for access policies such as $a_1 \vee (a_2 \wedge a_3)$, where the a_i are specific attributes. ABE was originally introduced by Goyal et. al. in [GPSW06], but their construction Key-Policy Attribute-based Encryption (KP-ABE) only allows for the encryptor of some plaintext to specify a set of attributes some subsets of which may be used to decrypt the ciphertext. The onus of properly delineating which specific combinations are sufficient is left to the TA who constructs the keys, which means that the encryptor does not directly control who can successfully decrypt their data. Shortly after KP-ABE was developed Bethencourt, et. al. introduced Ciphertext-Policy Attribute-based Encryption (CP-ABE) in [BSW07]. This incorporates a much more natural access policy construction where the encryptor specifically delineates which attribute combinations can successfully decrypt the ciphertext, and the TA is only concerned with properly certifying and assigning attributes to the various key holders.

Fully describing the construction of the CP-ABE scheme by Bethencourt, et. al. is - although not requiring any additional background to that already given - quite complicated, due to the complex structure necessitated by the access policy construction. Instead, we give a high-level consideration.

Protocol 6.5 (CP-ABE High-Level Construction)**.** A CP-ABE scheme with a trusted authority (TA) and some number of entities operates as follows.

1. The TA generates a set of public parameters *params* and a master key K_{master}.
2. For each entity e in the system, the TA does as follows. First, the TA determines the set of attributes S_e held by the entity. They then use K_{master} to construct a secret key K_e which identifies with S_e, and issue it to the entity.
3. An encrypting entity, wishing to encrypt a plaintext m, determines an access structure T. They then encrypt the plaintext and the access structure with the public parameters to a ciphertext c.
4. A decrypting entity, wishing to decrypt c, attempts the decryption process using their key $K_{decryptor}$. If $S_{decryptor}$ satisfies the access structure T, then Bob recovers the message m.

In addition, CP-ABE allows for any entity to construct a delineated key $K'_{decryptor}$ where the set of attributes which the delineated key identifies with is a subset of $S_{decryptor}$. This allows the decryptor to - for instance - give their assistant a sub-key which allows them to access any information the decryptor can, excluding that which requires a security clearance.

A full discussion of the construction of Bethencourt, et. al. is in [BSW07]. Unlike in Boneh-Franklin, the Weil Pairing is not used to construct a stream of bits to XOR with the plaintext. Instead - and this description is greatly simplified - with choice of suitable curve $E(K)$ the Weil Pairing is used to compute a quantity $e'(P, P)^{\alpha s}$, where $P \in E(K)$ and $\alpha \in \mathbb{Z}_p$ are public and $s \in \mathbb{Z}_p$ is private, with p the order of the elliptic curve group. This quantity is simply multiplied by the plaintext to construct the message component of the ciphertext. Only if an entity attempting to decrypt has the correct set of attributes built into their key are they able to recover the necessary information regarding s to invert the multiplication.

6.3 Certificateless Public-Key Encryption

Although we have discussed the benefits of IBE, FIBE, and ABE over traditional certificate-based entity authentication approaches, there is one substantial drawback to the original construction of them we have so far given. They provide the opportunity for an authority - if the trust placed in them is misguided - to intercept and decrypt messages at will *without the knowledge of the participants in the infrastructure.* For Boneh-Franklin, this can be seen quite simply as the TA generates D_{ID}, the only secret information necessary to decrypt any message sent to the the identity represented by ID. In the paper describing Boneh-Franklin - as well as in as the papers introducing the FIBE and ABE schemes - the term private key generator (PKG) is actually used in place of trusted authority to emphasize this concern. The PKGs generate the private keys they posses them and so may access any information the key holders themselves can. Although this may not necessarily be considered lethal - since assuming total trust in the TA/PKG may be considered a prerequisite for any public-key entity authentication scheme - it is certainly not ideal. Certificate-based systems do not have this issue, since the holders generate their own private keys. If a TA wants to attempt to intercept messages sent to an entity they have to generate their own key pair and a conflicting certificate to bait messages, which as the certificate is a public document creates a record of their malfeasance or negligence.

Girault introduced a basic terminology for this TA/PKG trust hierarchy in [Gir91]. In level 1 schemes, which Boneh-Franklin and the schemes of Goyal et. al. and Bethencourt et. al., fall under, the authority directly possesses secret keys and can impersonate any identity without being detected. In level 2 schemes, the authority doesn't directly posses the keys, but can issue false certificates or some equivalent competing claim of identity which cannot be challenged. In level 3 schemes, any attempt by the authority to impersonate the entity will be detected, and any attempt to intercept messages to the entity will fail. Naturally, after the introduction of IBE focus turned towards solving this problem introduced by trusted authorities taking on the role of key generation. Certificateless Public-Key Encryption (CL-PKE) was one of the first successful attempts at a solution. The scheme - an extension of Boneh-Franklin - is based off of a simple idea, namely that the TA/PKG generates only a partial private key, which is then combined with an additional secret parameter - this time known only to the key holder - to form the full private key. Once again we shall give a simpler formulation of the scheme which is only IND-CPA secure, with the same omitted transformation being used to make it IND-CCA as for Boneh-Franklin. The setting for the scheme is identical as well Boneh-Franklin, including its capacity to be defined for non-elliptic curve settings, and we shall use the same modified Weil Pairing. The last additional note is that this simplified construction actually resides in level 2 of Girault's hierarchy, however a minor modification to make it level 3 is given in the original paper [ARP03].

Algorithm 6.6 (`Certificateless Public-Key Encryption - Authority Setup`)**.** An authority initializes the infrastructure as follows.

1. The TA chooses a large prime $p = 6i - 1$ with i prime where $p \equiv 2 \pmod 3$, as well as a curve $E(\mathbf{F}_p)$ with equation $y^2 = x^3 + 1$.

2. The TA chooses a point P of order i in $E(\mathbf{F}_p)$.

3. The TA chooses two hash functions. The first is $H_1 : \{0, 1\}^* \to E(\mathbf{F}_p)$, where $\mathbf{Im}(f)$ is composed of points of order i. The second is $H_2 : \mathbf{F}_{p^2}^\times \to \{0, 1\}^n$, where $\mathbf{Dom}(g)$ composed of points of order i, and n is a positive integer, which will need to be equal to or greater than the length of any plaintext.

4. The TA chooses a random $k \in \mathbf{F}_i^\times$ and computes $P_{pub} = kP$.

The output is $(k, (p, H_1, H_2, n, P, P_{pub}))$, where k is kept secret and the 6-tuple is made public and distributed.

If an entity with an identity string ID wishes to join the infrastructure and get a key pair, the trusted authority does the following.

Algorithm 6.7 (`Certificateless Public-Key Encryption - Partial Private Key Generation`)**.** Input is $(k,\, H_1,\, ID)$ where ID is the identity string claimed. After confirming they are communicating with the entity eponymous with ID, the TA computes $Q_{ID} = H_1(ID)$ and $D_{ID} = kQ_{ID}$. The output is D_{ID} which is securely communicated to the entity. The partial private key is D_{ID} and is kept secret by the entity.

The entity may then construct their full private key as follows.

Algorithm 6.8 (`Certificateless Public-Key Encryption - Full Private Key Generation`)**.** Input is D_{ID}. The entity chooses a random $j \in \mathbf{F}_i^\times$ and computes their full secret key $S_{ID} = jD_{ID} = jkQ_{ID}$. The output is $(j,\, S_{ID})$, both of which the entity keeps secret.

Unlike in Boneh-Franklin where the public key for an entity is simply their identity, for CL-PKE they must compute a public key which will be used in concert with their identity for the encryption process. However, a sender will be able to confirm the validity of the public key without needing to refer to a certificate, maintaining the IBE principle of the cryptosystem.

Algorithm 6.9 (`Certificateless Public-Key Encryption - Public Key Generation`)**.** Input is $(j,\, P,\, P_{pub})$. The entity computes $A_{ID} = jP$ and $B_{ID} = jP_{pub} = jkP$. The output is $(A_{ID},\, B_{ID})$, which the entity distributes.

As noted, a sender does not need to confirm the veracity of the channel by which they receive the key, as they will perform a check which confirms that it must have been created by the entity which holds the private key bound to the identity they are sending their message to. The sender encrypts their message as follows.

Algorithm 6.10 (`Certificateless Public-Key Encryption - Encryption`)**.** Input is $(m,\, ID,\, A_{ID},\, B_{ID},\, H_1,\, H_2,\, P,\, P_{pub})$, where m is the message the sender wishes to communicate to the entity known by ID. The sender first computes $Q_{ID} = H_1(ID)$. Next, they confirm that $e'(A_{ID},\, P_{pub}) = e'(B_{ID},\, P)$, where e' is the modified Weil Pairing. If not, the sender concludes that the public key is not valid, and aborts encryption. They then choose a random $r \in \mathbf{F}_i^\times$ and compute $x_{ID} = e'(Q_{ID},\, P_{pub})$. The sender then computes $c = (u,\, v) = (rP,\, m \oplus H_2(x_{ID}^r))$, note that $u \in E(\mathbf{F}_p)$ while $v \in \{0,\, 1\}^n$. The output is c, which the sender then sends to the recipient.

The correctness of the validity check is straightforward, as we have

$$e'(A_{ID},\, P_{pub}) = e'(jP,\, kP) = e'(P,\, P)^{jk} = e'(jkP,\, P) = e'(B_{ID},\, P),$$

which confirms that A_{ID} and B_{ID} were constructed by the entity that knows j. Receiving c, the recipient decrypts as follows.

Algorithm 6.11 (`Certificateless Public-Key Encryption - Decryption`)**.** Input is $(D_{ID},\, c)$. The recipient computes $y_{ID} = e'(S_{ID},\, u)$ and then $m = v \oplus H_2(y_{ID})$. The output is m.

The decryption works due to two equivalences, first

$$e'(S_{ID},\, u) = e'(jkQ_{ID},\, rP) = e'(Q_{ID},\, P)^{jkr} = e'(Q_{ID},\, jkP)^r = e'(Q_{ID}, B_{ID})^r = x_{ID}^r,$$

which follows from the properties of the Weil Pairing, which continue to hold in its modified form. Using this $e'(S_{ID},\, u) = x_{ID}^r$, we now have

$$v \oplus g(e'(S_{ID},\, u)) = (m \oplus h_2(x_{ID}^r)) \oplus h_2(x_{ID}^r) = m.$$

The similarity to the description of Boneh-Franklin given previously is quite clear. Essentially, an additional component is introduced to the encryption and decryption process such that two of them can function as secrets, one of which is generated by the TA/PKG to provide certification, the other kept private by the key holder to prevent the TA from intercepting their messages.

That this scheme is only level 2 is apparent from that the trusted authority can simply generate their own full private key and public key for an identity, which it is straightforward to see the sender will accept. The method for extending CL-PKE to level 3 involves generating the public key before the partial private key, and then defining Q_{ID} partially in terms of the public key. The result of this is that the key holder can only generate a single valid key pair. Therefore, the only way by which there can be multiple valid public keys for a given identity is if the TA/PKG generated the others, making their action detectable.

6.4 Certificate-based Encryption

One final extension of IBE worth briefly discussing is Certificate-based Encryption (CBE) - originally introduced by Gentry in [Gen03] - which attempts to combine both IBE and certificate entity authentication approaches in a manner which solves each of their respective concerns without much loss of their respective benefits. Specifically, the construction bounds decryption directly to an identity string - as in IBE - and also actively prevents decryption if there is not a valid certificate for the key pair. As with CP-ABE, we will give a high level consideration of the algorithm. The basic construction given originally by Gentry is built out of Boneh-Franklin, and from a mathematical perspective is of roughly equivalent mathematical complexity to CL-PKE and Boneh-Franklin itself, however we will omit the full discussion of it due to having not presented sufficient background on the structure of certificate-based entity authentication schemes themselves.

Protocol 6.12 (CBE High-Level Construction)**.** A CBE scheme with a trusted authority (TA) and some number of entities operates as follows.

1. The TA generates a set of public parameters *params* and a master key K_{master}.
2. An entity wishing to allow other entities to securely communicate with them generates a secret key/public key pair. They then send the public key and their identity to the TA, who confirms by some mechanism they are interacting with the entity eponymous with the claimed identity. The TA uses K_{master} to create a secret certificate C_R valid for some time period i and sends the certificate to the entity. The entity then distributes their public key.

3. Another entity wishing to send a message m to the aforementioned entity encrypts it using the identity and public key of the recipient, and sends the ciphertext c to them.

4. Receiving c, the recipient decrypts it using their secret key and certificate. The decryption process will be unsuccessful if the sender constructed the message outside of the time period i, in which case the certificate is no longer valid.

Gentry gives two general modes of attack on the scheme, one by an entity attempting to decrypt a message sent to an identity which has not been certified by the TA, the other by the TA attempting to decrypt a message sent to an identity it has certified. For the first, the uncertified entity can generate a key pair for an identity but are unable to decrypt a message sent to them as they do not have a valid certificate. They cannot construct a certificate themselves for their identity, as else they'd be able to break the underlying IBE scheme. As for the TA they are not able to decrypt a message they've intercepted since they are not able to determine the secret key bound to the public key. This solution to the TA attacking using their knowledge of the recipient's private information from their role as PKG is quite similar to that used by CL-PKE, namely that decryption is made to require two secret components only one of which is known to the TA.

There are two primary negatives to CBE in comparison to the previously discussed schemes. The first is a return to a reliance on certificates, however this is slightly mitigated in that they are no longer a single point of failure from a security perspective - being primarily concerned with freshness and no longer with identity. The second negative is that CBE is computationally expensive, due to the singular TA needing to regularly reissue certificates. Exactly how expensive the recertification component of CBE is depends on a trade off, as the usefulness of certificates is heightened by their recency, which incentives shorter choices for the length of the time frames represented by each i, which requires the reissuance of all certificates in the infrastructure with greater frequency. Gentry gives multiple methods by which this computational expense can be reduced by only requiring computations for those identities which have changed or are having their certificates revoked. Combined with a hierarchical TA model in which the responsibility for reissuing certificates can be distributed amongst various sub-TAs, this brings the efficiency of CBE into the realm of practical possibility.

7 Conclusion

We conclude briefly with a note on the limited breadth of topics we have yet covered in the realm of ECC. The discrete logarithm problem is extremely flexible and is used for a wide variety of cryptographic purposes, the vast majority of which have or can be made to have an elliptic curve analogue. Combined with their use for pairings-based constructions - as well as further purposes we have not discussed such as within computational number theory - the usefulness to cryptography of elliptic curves goes far beyond just that which has been discussed herein. However, the topics which have been covered in this paper have been chosen to provide a strong introduction to the most commonly cited and discussed uses of ECC, and through them a general sense of how cryptographic constructions can employ their structure may quite well be glimpsed. For those considering further study, the breadth of cryptographic problems which have solutions within ECC

- and therefore have their study accessible through it - provide weight to its use as a lens through which to approach cryptography more generally.

8 Attributions

The main line of discussion in this paper follows that of Washington [Was03], albeit with some rearranging of the ordering. The following attributions concern explicit components of the text that were referenced directly or indirectly from various sources. Generally speaking any discussion surrounding these components was likely influenced by the sources themselves, although all the exact text in those discussions is my own.

Definition 2.1 is given by Washington [Was03]. In *Theorem 2.2*, the formulation of the group laws comes from an amalgamation of those given by Washington [Was03] and Cameron (who does not explicitly give the group laws for elliptic curves, but rather his formulation of the generalized group laws) [Cam08]. *Theorem 3.1* is a rewording of Theorem 7.43 in Cameron [Cam08]. *Theorem 3.2* is due to Washington, as are *Theorems 3.3* and *3.4* [Was03]. The proof of *Theorem 3.5* is my own, given as an answer to *Exercise 4.8* in Washington [Was03]. *Theorem 3.6* is given by Cameron [Cam08]. *Theorems 4.1* and *4.2* are given by [Was03].

Definitions 5.1 and *5.2* originate, with changes of wording, with Goldreich [Gol06]. The remainder of the definitions in the *Section 5* are given by Washington [Was03], albeit with some of the wordings changed, in some cases to align them more closely with the definitions given by Stinson [Sti05] or Katz and Lindell [LK15]. *Algorithms 6.1-4* are Washington's formulations given in [Was03] of the BasicIdent scheme given originally by Boneh and Franklin [BF03]. *Protocol 6.5* is a simplified concatenation of the relevant definitions in [BSW07]. *Algorithms 6.6-11*, are my own, albeit inspired by Washington's formulations used for *Definitions 6.1-4*, of the BasicCL-PKE scheme given originally by Al-Riyami and Paterson [ARP03]. *Protocol 6.12* is a simplified concatenation of the relevant definitions in [Gen03].

Bibliography

[Age] National Security Agency. NSA Suite B Cryptography. `https://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml`. Accessed: November 22nd 2015.

[ARP03] Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless Public Key Cryptography. In *Advances in Cryptology-ASIACRYPT 2003*, pages 452–473. Springer, 2003.

[BF03] Dan Boneh and Matthew Franklin. Identity-based Encryption from the Weil Pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.

[BHH$^+$14] Joppe W. Bos, J. Alex Halderman, Nadia Heninger, Jonathan Moore, Michael Naehrig, and Eric Wustrow. Elliptic Curve Cryptography in Practice. In *Financial Cryptography and Data Security*, pages 157–175. Springer, 2014.

[BKK$^+$09] Joppe W. Bos, Marcelo E. Kaihara, Thorsten Kleinjung, Arjen K. Lenstra, and Peter L. Montgomery. On the Security of 1024-bit RSA and 160-bit Elliptic Curve Cryptography, 2009.

[BL15] Daniel J. Bernstein and Tanja Lange. SafeCurves: choosing safe curves for elliptic-curve cryptography. `safecurves.cr.yp.to`, August 2015.

[BSS99] Ian F Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic Curves in Cryptography*, volume 265. Cambridge University Press, 1999.

[BSW07] John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy, 2007. SP'07*, pages 321–334. IEEE, 2007.

[Cam08] Peter J. Cameron. *Introduction to Algebra.* Oxford University Press, 2nd edition, 2008.

[DH76] Whitfield Diffie and Martin E. Hellman. New Directions In Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[ElG85] Taher ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. In *Advances in Cryptology*, pages 10–18. Springer, 1985.

[Gen03] Craig Gentry. Certificate-Based Encryption and the Certificate Revocation Problem. In *Advances in Cryptology-EUROCRYPT 2003*, pages 272–293. Springer, 2003.

[Gir91] Marc Girault. Self-Certified Public Keys. In *Advances in Cryptology-EUROCRYPT91*, pages 490–497. Springer, 1991.

[Gol06] Oded Goldreich. *Foundations of Cryptography: Volume 1 Basic Tools.* Cambridge University Press, 3rd edition, 2006.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective.* Cambridge University Press, 1st edition, 2008.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In *Proceedings of the 13th ACM conference on Computer and Communications Security*, pages 89–98. ACM, 2006.

[Kob87a] Neal Koblitz. *A Course in Number Theory and Cryptography.* Springer-Verlag New York, 1987.

[Kob87b] Neal Koblitz. Elliptic Curve Cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[Lab12] Information Technology Laboratory. Secure Hash Standard (SHS). Technical Report FIPS 180-4, National Institute of Standards and Technology, March 2012.

[Lab13] Information Technology Laboratory. Digital Signature Standard (DSS). Technical Report FIPS 186-4, National Institute of Standards and Technology, July 2013.

[LK15] Yehuda Lindell and Jonathan Katz. *Introduction to Modern Cryptography.* Chapman and Hall/CRC Press, 2nd edition, 2015.

[Mil86] Victor Miller. Use of Elliptic Curves in Cryptography. In *Advances in Cryptology-CRYPTO85 Proceedings*, pages 417–426. Springer, 1986.

[MOV93] Alfred J. Menezes, Tatsuaki Okamoto, and Scott Vanstone. Reducing Elliptic Curve Logarithms to Logarithms in a Finite Field. *IEEE Transactions on Information Theory*, 39(5):1639–1646, 1993.

[MR01] Patrick McDaniel and Aviel Rubin. A Response To Can We Eliminate Certificate Revocation Lists?. In *Financial Cryptography*, pages 245–258. Springer, 2001.

[Riv98] Ronald L. Rivest. Can We Eliminate Certificate Revocation Lists? In *Financial Cryptography*, pages 178–183. Springer, 1998.

[Sch85] René Schoof. Elliptic Curves over Finite Fields and the Computation of Square Roots mod p. *Mathematics of Computation*, 44(170):483–494, 1985.

[Sch95] René Schoof. Counting Points on Elliptic Curves over Finite Fields. *Journal de Théorie des Nombres de Bordeaux*, 7(1):219–254, 1995.

[Sip06] Michael Sipser. *Introduction to the Theory of Computation.* Course Technology, 2nd edition, 2006.

[Sti05] Douglas R. Stinson. *Cryptography: Theory and Practice.* Chapman and Hall/CRC Press, 3nd edition, 2005.

[SW05] Amit Sahai and Brent Waters. Fuzzy Identity-Based Encryption. In *Advances in Cryptology–EUROCRYPT 2005*, pages 457–473. Springer, 2005.

[Was03] Lawrence C. Washington. *Elliptic Curves: Number Theory and Cryptography.* Chapman and Hall/CRC Press, 1st edition, 2003.