A Probabilistic Trust Model

Gareth Karl

Department of Computer Science, University of Auckland

Abstract: An identity certificate is a digitally signed certificate that binds a public key to information about the alleged holder of that key. Guaranteeing the authenticity of identity certificates is an open problem in Public Key Cryptography. Public Key Infrastructures (PKIs) are designed to distribute identity certificates and provide users with some degree of trust in their authenticity. All PKIs are based on one or more of three trust models: the Web of Trust, hierarchical trust and trust lists. I focus on the Web of Trust. Probabilistic reasoning is an approach to reasoning with uncertain information. Probabilistic reasoning's ability to handle uncertain information makes it an obvious candidate for inclusion in PKIs. I consider the benefits and disadvantages of a probabilistic version of the Web of Trust, with particular reference to a scheme described by Maurer. Finally, I assert that Pretty Good Privacy (PGP) is an example of a PKI that is well suited to a probabilistic Web of Trust.

1 Cryptography and Public Key Infrastructures

1.1 Conventional Cryptography and Public Key Cryptography

Cryptography is the study of how to communicate secretly when an adversary can monitor your communication. Prior to the 1970s all cryptographic techniques were reliant on the sender and the receiver already sharing some secret information. These techniques are known as symmetric encryption. The secret information is called the *key*, while an unencrypted message is called *cleartext*, and an encrypted message is called *ciphertext*. When using any symmetric encryption algorithm, the cleartext is encrypted with a function (let this be *f*) and unencrypted with its inverse, so *f(cleartext, key) = ciphertext* and f^{-1}*(ciphertext, key) = cleartext*.

Public key cryptography was first proposed by Diffie and Hellman in 1976 [1]. It does not require that two parties share secret information in advance. This is because it includes two keys: the public key that anyone may know and the private key that one party must keep secret. Public key cryptography uses asymmetric encryption algorithms that are based on a type of *one-way function* known as a *trapdoor function*. A one-way function is a function that is easy to compute but whose inverse is hard to compute. A trapdoor function is a one-way function whose inverse is easy to compute if some additional information is known. This additional information is the private key. Let *f* again be the encryption function and let *g* be the decryption function. Then, *f(cleartext, public key) = ciphertext* and *g(ciphertext, private key) = cleartext*. A private key can also be used to digitally sign a document so that anyone with the corresponding public key can verify that it was signed by that private key. The entity that knows the private key is referred to as the *key holder*.

1.2 Identity Certificates and Public Key Infrastructures

If a message is encrypted with a public key, complexity theory gives us reason to believe (but not proof) that only someone who has the corresponding private key can decrypt the message using a realistic amount of resources. Also, anyone with access to a public key can check whether a message has been signed by someone who holds the corresponding private key. However, this leaves two problems. Firstly, it is possible for more than one person to have knowledge of any particular private key. The private key must be stored somewhere, and this leaves open the possibility that it may be stolen. Secondly, you must know who the holder of the private key is in order to know who is capable of decrypting or signing with it. A key holder may claim to be someone other than who they are. Diffie and Hellman suggested a directory that listed all key holders and their public keys [1, p.648]. Thus, knowledge of the problem of distributing public keys has existed for as long as knowledge of public key cryptosystems themselves. An *identity certificate* is a public key that is bound by digital signature to some information that helps identify the key holder. An entity trusts a certificate if they believe that the entity identified by the certificate really is the key holder. An entity can state that they trust a certificate by digitally signing that certificate.

This is one *form* of trust that is relevant to this study. The *degree* to which this trust is held will depend on many factors including how the public keys are distributed. A *Public Key Infrastructure (PKI)* is a means of distributing identity certificates that is intended to provide trust in these certificates. A *Certificate Authority (CA)* is a widely trusted entity that is responsible for verifying and signing identity certificates. They are often responsible for generating key pairs as well. An *anchor point* is the initial trust in a public key that an entity has. This trust may have been gained from personal transferal of a key or it may be based on widespread trust of a certificate. Anchor points are linked to the identity certificates that the entity wishes to trust through *certificate chains*. When discussing relationships in a PKI, the names of Alice (or A), Bob (or B) and Charlie (or C) are often used to simplify explanations. Alice, Bob and Charlie are entities in the PKI and may or may not be people.

1.3 A Taxonomy of Trust Models

All PKIs use a *trust model* to provide trust in the certificates they distribute. Linn has identified five trust models so that PKIs can be classified according to the trust model upon which they rely [3]. These are subordinated hierarchy, cross-certified mesh, hybrid, bridge CA and trust lists models. Linn makes an important point when he states that a PKI may be what he calls ‘multi-headed’, meaning that it applies the arguments of more than one trust model [3, p.2]. A glaring omission from Linn’s taxonomy is the Web of Trust. PKIs that are reliant upon the Web of Trust do not include Certificate Authorities and instead take a distributed approach to verifying identity certificates. In its simplest form, the Web of Trust (incorrectly) states that if Alice knows Charlie’s public key and

Charlie has signed Bob's identity certificate, then Alice can trust that certificate. In other words, Charlie's key is the trusted anchor point and the certificate chain can consist of any identity certificates that have been signed by already trusted keys.

Linn's subordinated hierarchy, cross-certified mesh, hybrid and bridge CA models can be grouped together because they all rely on Certificate Authorities as the anchor point and on the hierarchical nature of the model to provide trust in the certificate chains that are constructed. The subordinated hierarchy model is hierarchical because it consists of a tree with a universally trusted CA at the root, non-CA entities as the leaves and subordinate CAs in between. The other three models are hierarchical because they are variants upon the subordinated hierarchy model with extra certification between subordinate CAs to provide more flexibility. Some of these do not have universally trusted root CAs. The trust lists model is quite different to all the other models because it relies on having lots of anchor points to provide trust in lots of certificates instead of using long chains or dense trees. The best example of a trust lists model is a modern web browser because it has a large set of keys built into it. This leaves three types of trust models: the Web of Trust, hierarchical trust and trust lists. I will focus on variants of the Web of Trust.

2 A More Precise View of Trust

2.1 Trust in Entities

Another form of trust (besides trust in a certificate) that is relevant in a PKI is trust in an entity. This is trust that a particular entity has a particular characteristic, such as the ability to make a payment or a responsible approach to signing identity certificates. The second of these characteristics is particularly important to the analysis of PKIs and trust that an entity has this characteristic is the only form of trust in an entity that is discussed in this paper. The closest Alice can come to being absolutely sure that a certificate binding Bob to a public key is correct is to have Bob verify the certificate in person by providing the public key (or a unique hash of the key). This is the personal transferal that I mentioned in section 1.2. However, this approach is impractical when Alice and Bob are geographically separated or do not know each other personally. It is also impractical if Alice wishes to know the public keys of many entities or many entities wish to know Bob's public key. This is why a PKI must rely on some variant of the trust models that I have outlined. There is no convenient means of generating trust in public keys without trust models. All of the trust models make use of trust in entities, although the trust lists model does not use it as much.

2.2 Improving the Web of Trust with Recommendations

The simple Web of Trust model is flawed because trust in a certificate that binds an entity to a public key should not imply any trust in that entity. Although it may sometimes be acceptable to trust all

entities in a certificate chain from a hierarchical PKI, this is not true of the Web of Trust. The Web of Trust can be improved by requiring trust in entities to be stated explicitly. The improved Web of Trust is useful if Alice knows from previous experience that Charlie is very cautious about signing certificates and therefore trusts certificates signed by Charlie (for simplicity, Alice trusts Charlie). If:

1) Alice trusts Charlie; and
2) Alice trusts a certificate that binds Charlie to a public key (call this K_C); and
3) there is a certificate signed by K_C that binds Bob to another public key (call this K_B);

then:

Alice can trust the certificate binding Bob to K_B.

In essence, Charlie has recommended the certificate that binds K_B to Bob. Since Alice trusts Charlie's recommendations, she is willing to encrypt a message intended only for Bob using K_B and send it over an insecure channel. Also, she is confident that anything signed by the private key corresponding to K_B was signed by Bob. It is important to note that this does not imply that Alice trusts Bob. Therefore, Alice does not trust a certificate that has been signed by this private key since Bob may sign inaccurate certificates. If Alice wishes to trust such a certificate, Alice must trust Charlie not only to recommend the certificate that binds K_B to Bob, but also to recommend Bob himself. For Alice's part, this is quite a leap of faith. However, it could potentially result in a large increase in the number of certificates that Alice trusts. If:

1) Alice trusts Charlie to recommend other entities; and
2) Alice trusts a certificate that binds Charlie to a public key (call this K_C); and
3) there is a certificate signed by K_C that binds Bob to another public key (call this K_B); and
4) there is a recommendation signed by K_C that Bob should be trusted;

then:

Alice trusts Bob.

This idea can be built up recursively, with the next step being that Alice trusts Charlie to recommend Bob's recommendations. Each step is less likely to be justified than the previous and results in Alice trusting more certificates.

2.3 A Meaningful Measure of Trust?

Trust is rarely absolute since people trust many things, but often only to some extent. I alluded to this in section 1.2, when I mentioned degrees of trust. Even though trust clearly comes in differing degrees, it is not easy to describe these degrees in a meaningful and non-arbitrary way. The first step towards defining meaningful degrees of trust is to decide what it means to trust one entity more than another. There are two possible meanings of 'Alice trusts Charlie more than Bob' that stand out:

(a) Alice trusts Charlie with everything that she trusts Bob with and more.
(b) Alice thinks her trust in Charlie is less likely to be mistaken than her trust in Bob.

I have already touched upon (a) by discussing recommendations that are built up recursively. It seems intuitive that if you trust Charlie to recommend other entities as trustworthy certificate signers then you would also trust Charlie to sign certificates himself. After all, he could always just

recommend himself. In order to avoid confusion, this will be referred to as *levels* of trust. This idea only works because we are considering a trust that is recursive. Otherwise, it would be possible for Alice to trust both Charlie and Bob with things that she does not trust the other with. Then the two trusts would be incomparable.

However, this paper is primarily about (b) and how to measure it. It is far from obvious that a *meaningful* scale that is based on (b) could be devised. For example, the four levels of trust that are used by Pretty Good Privacy (described in section 4.2) could be given the values 0, 1, 2 and 3, however this is a totally arbitrary scale since we don't know how much more 'trustworthy' 0 is than 1 (or 1 than 2, et cetera). Suppose that a meaningful scale could be devised, then the next step would be to find a method of calculating a trust value on that scale from some other trust values on the scale. This would be a calculus of trust values. This calculus would be useful to anyone wanting to understand or manage the risk of wrongly trusting certificates that is inherent in the use of PKIs. Different amounts of trust measured by such as scale will be referred to as *degrees* of trust. Maurer assumes the existence of a meaningful scale by treating degrees of trust as being probability values from a 'well-defined random experiment' [4, p.3]. This allows him to calculate the degree of a trust that resulted from reasoning of the type described in section 2.2. Since Maurer's degrees are probability values, they are on a scale from 0 to 1.

3 Maurer's Probabilistic Trust Model

3.1 A Deterministic Model of Reasoning with Trust

Before Maurer can describe the nature of the calculus that he performs on measurements of trust, he has to formalize a method of reasoning about trust relationships similar to that which I described. Initially, he describes a deterministic model (one that considers all statements to be either definitely valid or definitely invalid). He does this by defining four types of statements and two inference rules. Maurer defines his statements as follows [4, p.9]:

> **Definition 3.1.** *Statements* are of one of the following forms:
>
> - *Authenticity of public keys.* $Aut_{A,X}$ denotes Alice's belief that a particular public key P_X is authentic (i.e., belongs to entity X) and is represented graphically as an edge from A to X: $A \longrightarrow X$.
> - *Trust.* $Trust_{A,X,1}$ denotes Alice's belief that a particular entity X is trustworthy for issuing certificates. Similarly, her belief that X is trustworthy for issuing recommendations of level $i-1$ is denoted by $Trust_{A,X,i}$. The symbol is a dashed edge from A to X labelled with the trust level: $A \overset{i}{\dashrightarrow} X$.
> - *Certificates.* $Cert_{X,Y}$ denotes the fact that Alice holds a certificate for Y's public key (allegedly)[9] issued and signed by entity X. The symbol is an edge from X to Y: $X \longrightarrow Y$.
> - *Recommendations.* $Rec_{X,Y,i}$ denotes the fact that Alice holds a recommendation of level i for entity Y (allegedly) issued and signed by entity X. The symbol is a dashed edge from X to Y labelled with i: $X \overset{i}{\dashrightarrow} Y$.

...

[9] We use the word "alleged" because without verification, there exists no evidence that the certificate was indeed issued by the claimed entity.

He defines his inference rules as follows [4, p.10]:

Definition 3.2. A statement is *valid* if and only if it is either contained in $View_A$ or if it can be derived from $View_A$ by applications of the following two inference rules:

$$\forall X, Y: \quad Aut_{A,X},\ Trust_{A,X,1},\ Cert_{X,Y} \vdash Aut_{A,Y} \qquad (1)$$

and

$$\forall X, Y, i \geq 1: \quad Aut_{A,X},\ Trust_{A,X,i+1},\ Rec_{X,Y,i} \vdash Trust_{A,Y,i}. \qquad (2)$$

$View_A$ is the set of all statements that Alice initially considers valid. $\overline{View_A}$ is the closure of $View_A$ under rules (1) and (2), so Alice should consider all statements in $\overline{View_A}$ to be valid. There are two more inference rules that state that a trust in an entity to particular level implies trust in it to all lower levels. This is not necessary since any consistent $View_A$ would include these lower levels anyway. Maurer includes them to make his examples less complicated.

3.2 A Probabilistic Approach to Trust Measurement

Maurer extends his deterministic model into a probabilistic model in order to use the probability of a statement as a measure of trust. Let S be a statement for which Alice requires a trust measurement. This is likely to be a statement of the form $Aut_{A,B}$, but there is no reason why the model is less suitable for measuring the other three types of statement. Let $\mathrm{P}(X)$ be the probability that X is true. In the probabilistic model, the measurement of the degree of trust of S is $\mathrm{P}(S \in \overline{View_A})$. This is referred to as the *confidence value* of S. This is to differentiate it from $\mathrm{P}(S \in View_A)$, which is the result of the aforementioned 'well-defined random experiment' [4, p.3]. The probability that a derived statement is in $\overline{View_A}$ is calculated using probabilistic logic.

The calculation of $\mathrm{P}(S \in \overline{View_A})$ is reasonably straightforward and is described on pages 15 to 18 of [4]. S_A is defined as the set of all statements that Alice considers could be in $View_A$. Initially all that is $\mathrm{P}(s \in View_A)$, for each $s \in S_A$. Let M be the set of minimal subsets of S_A that could be used to derive S. For each $X \in M$, $\mathrm{P}(X \subseteq View_A)$ can be determined from $\{\mathrm{P}(x \in View_A) : x \in X\}$. Similarly, $\mathrm{P}(\exists X \in M : X \subseteq View_A)$ can be calculated from $\{\mathrm{P}(X \in View_A) : X \in M\}$. This method of calculating the confidence value of S (namely, $\mathrm{P}(S \in \overline{View_A})$) guarantees that it will be greater than or equal to $\mathrm{P}(S \in View_A)$. For an explanation of probabilistic reasoning, see Appendix A.

3.3 Assumptions and Weaknesses in Maurer's Scheme

Superficially, Maurer's paper appears to produce the far-reaching result of a meaningful trust measure that can be applied to all PKIs. However, his approach has a number of weaknesses that mostly result from assumptions he makes in his model. Only one of these assumptions was implicit. His model could still be useful in any particular PKI if it tends to conform to the assumptions made.

3.3.1 Measuring Initial Trust Statements

Probabilistic logic allows the probability values of certain statements to be calculated from the probability values of certain other statements. Therefore, the probability values of some statements must already be known. Furthermore, the calculated values will be inconsistent if the values used to calculate them are inconsistent. In the case of Maurer's trust model, the values that must be known initially are $\mathrm{P}(S \in View_A)$, for each $s \in S_A$. However, Maurer does not provide a means of measuring these initial values. Therefore, the entire scheme is reliant upon Alice's ability to accurately measure or estimate the probability that her trust is well founded for each statement in her initial view. These probabilities must have the characteristics of a well-defined random experiment. The scheme is equally reliant upon the ability of other entities that provide Alice with recommendations to make similar measurements or estimations.

Maurer's scheme does make progress towards meaningful trust measurement. It provides a means of deriving trust values for statements that have been inferred (possibly from statements made by more than one entity) using only the trust values of statements that were directly asserted. Sometimes it is easy to provide reasonable estimates of the trust values of statements in the initial view. For example, if Bob gives Alice a disk with a public key on it and tells her that it is his public key (all of this happens face to face) then Alice can confidently give $Aut_{A,B}$ a trust value very close to 1. However, it is often far more difficult to work out trust values. Consider the statement $Trust_{A,B,1}$. The only way to work out a trust value for this that is appropriate for a probabilistic scheme such as Maurer's would be to consider Bob's previous recommendations. If Bob has not made many previous recommendations then a reasonable trust value would not be attainable. This is probably not a major drawback since Alice is unlikely to want to use recommendations from someone without a track record.

Suppose that Bob has made many recommendations. It still may not be easy to tell how accurate these recommendations were. Even supposing that Alice knows how many of Bob's recommendations are accurate, $\frac{\text{Number of Accurate Recommendations}}{\text{Total Number of Recommendations}}$ may not be an appropriate trust value. Other factors may affect the trustworthiness of Bob's recommendations (such as how well he knows the supposed key holder). This problem is not specific to Maurer's scheme but is

inherent in probabilistic reasoning and therefore will effect any scheme for measuring trust that is based on probabilistic reasoning.

3.3.2 Independence of Trust Statements

Maurer also explicitly assumes that the trust values of the statements in Alice's initial view are independent. Of course, in reality this is unlikely to be the case. For example, if Bob and Charlie work for the same company, then the trust value of $Rec_{B,X,1}$ is likely to be lower if $Rec_{C,X,1}$ is known to be invalid than if it is known to be valid. However, future probabilistic schemes may be able to include such dependencies. A model that includes n entities could have at most $n \cdot (n-1)/2$ pairs of entities. It may be possible to develop a rule for calculating the probability of one statement given that another is statement is true. These rules would use the independence values associated with the pairs of entities that are included in the two statements. A different rule for each combination of statement types would probably be necessary. If these rules were effective, it would be possible to model the interdependence of the statements in Alice's initial view while maintaining tractability. Then Bayes' Rule could be applied to calculate trust values (see Appendix A for an explanation of Bayes' Rule). It should be noted that Maurer's scheme successfully handles interdependent certification paths (as opposed to interdependent entities) [4, p.].

3.3.3 Holders of Multiple Key Pairs

Maurer's statements and inference rules implicitly assume that any particular entity has only one key pair. In fact, it even assumes that every entity is (correctly or incorrectly) bound to at most one public key. This simplifies Maurer's notation but makes his model less useful since entities may have multiple private keys that they use for different purposes or treat with differing caution. Rather than stating that 'a certificate was allegedly signed by Bob', it is both more accurate and allows more flexibility to say that 'a certificate was signed by K_B, which is allegedly Bob's private key'. This is well demonstrated by El Bakkali and Kaitouni's [2] approach to trust reasoning in PKIs. Their scheme is deterministic and based upon Predicate Calculus. Their inference rules are far more complicated. This is partly a result of the inclusion of policy constraints in their reasoning but it is also partly a result of their inclusion of keys (not just entities) as parameters in all of the appropriate relations. The pay off is that their model better approximates an actual infrastructure.

3.3.4 Comparing Trust Measurements with Policy Constraints

El Bakkali and Kaitouni intend that Alice's choice of policy constraints govern whether or not she should have confidence in a certificate. Taking such an approach, instead of trying to measure or calculate trust values (based on probabilistic logic or otherwise), has both benefits and disadvantages. An obvious disadvantage is that it does not take into account the uncertainty that is inherent within all PKIs because entities are being expected to trust second hand information. As with all predicate calculus, it treats statements as either definitely true or definitely false. The

compensation is that the policy constraints approach is more nuanced than the probabilistic approach. Maurer cannot model some things that El Bakkali and Kaitouni can. For example, El Bakkali and Kaitouni allow Alice to specify that Bob's recommendations are only trusted if the recommended entity is within a particular domain (perhaps because Bob is more familiar with entities within that domain), but Maurer could not do this. On the other hand, Maurer can represent some of the restrictions that are modelled by El Bakkali and Kaitouni. For example, a maximum path length of n could be enforced by giving all initial statements of the form $Trust_{X,Y,i}$ a trust value of 0 if $i > n-1$. Future probabilistic approaches could be made more nuanced by complicating the statements and axioms upon which they are based.

El Bakkali and Kaitouni claim that their use of policy constraints instead of trust values makes it easier to understand statements made in their scheme [2, p.368]. This is true for the trust value of a single statement on a non-meaningful scale since one value on such a scale cannot be understood to mean anything in particular. Values for at least two statements made by the same entity are required and even then the only understanding is relative (for example, Alice trusts Bob more than Charlie) if the scale is not meaningful. However, I dispute this for trust values on a meaningful scale. In particular, a scale that accurately represents probabilities would be easy to understand and a great aid to decision-making. On such a scale, Alice should trust a value of n (where $0 \leq n \leq 1$) for a particular statement if she is willing to accept that her trust in that statement will be misplaced about $n \cdot x$ out of every x times. Since Maurer makes assumptions that are not always true, his scheme does not reach this goal.

4 Pretty Good Privacy and Maurer's Trust Model

4.1 An Introduction to PGP

This brief explanation of Pretty Good Privacy (PGP) summarizes part of the documentation for PGP version 6.5.1 [5]. PGP is a system developed by Phil Zimmerman that incorporates a flexible PKI along with the software required for the practicalities of Public Key Cryptography (PKC). It includes both secure communication and digital signature. PGP recognizes two formats of certificate: X.509 certificates and PGP certificates.

X.509 certificates are supposed to conform to an international standard but in practice they have been extended differently to add functionality for specific purposes and an X.509 certificate created for one situation may not be applicable to another. Since X.509 certificates were specifically designed for a hierarchical PKI, they may only be signed by the issuer, a Certificate Authority. An X.509 certificate consists of an X.509 version number, a public key, the unique distinguished name of the key holder, the issuer's unique name, a validity period and the algorithm used to sign the certificate [5, pp.25-26].

In contrast to X.509, more than one person may sign a PGP certificate. It may also contain more than one means of identifying the key holder. Instead of a signature applying to the key as a whole, it applies to the binding of a specific form of identification and the key. People are identified in different ways by different associates and this approach allows an entity to sign only those of the users identities with which it is familiar. For example, a friend may sign the key holder's nickname and personal email address while a business partner may sign the key holder's full name and business email address [5, p.24]. A PGP certificate consists of a PGP version number, a public key, one or more forms of identification of the key holder, a self-signature, a validity period and the key holder's preferred symmetric encryption algorithm [5, pp.23-24].

4.2 The Web of Trust in PGP

In PGP, users have the ability to specify whether or not a key is a trusted introducer (the PGP name for a key whose signature is sufficient to provide trust in a certificate). This combines with the flexibility of PGP Certificates to allow PGP to be used with almost any trust model. However, PGP has traditionally been used with the Web of Trust. In fact, in the PGP 6.5.1 documentation it states that the Web of Trust is the PGP view of trust [5, p.32]. The version of the Web of Trust that is endorsed in the official PGP documentation [5] and that PGP certificates were designed for is not as naïve as the version discussed in section 1.3, but it is not as sophisticated as the version in section 2.2.

When discussing the hierarchical trust model, the PGP documentation calls the root CA a meta-introducer (the PGP name for a key that can be used to nominate other keys as introducers). However, it does not include meta-introducers in its discussion of the Web of Trust. In this respect, it is better than the original version, since it does not rely upon all key holders being responsible. However, it lacks any notion of levels of trust. Therefore, it cannot create trustworthy chains containing more than three certificates (the entity who wants to trust a certificate, the entity specified by the certificate and the trusted introducer who signed the certificate). This results in a model that cannot provide trust in a large number of certificates and makes the use of the 'six degrees of separation' metaphor [5, p.32] inaccurate.

The degrees of trust in PGP's version of the Web of Trust are not very helpful. PGP has four of what I have called degrees of trust (it calls them levels of trust). These are implicit trust, complete trust, marginal trust and no trust. Implicit trust is the trust an entity has in itself. There is no clear idea of how much trust complete trust or marginal trust are. It is only clear that a certificate must be signed by one completely trusted introducer or two marginally trusted introducers to be considered trusted. Presumably, 'Alice completely trusts Bob' does not mean that Alice knows with absolute certainty that Bob will never sign an incorrect certificate, since Alice could never know this. It is more likely

to mean that Alice thinks the likelihood that Bob will sign an incorrect certificate is at least as low as the likelihood she will. This would imply that 'Alice marginally trusts Bob' means that Alice believes that the likelihood that Bob will sign an incorrect certificate is such that two entities with this likelihood are as unlikely to independently sign an incorrect certificate as Alice alone is. If this is the case, then there should be a countable infinity (the cardinality of the set of natural numbers) of degrees of trust such that *n* signatures of degree *n* are required for Alice to trust the certificate. If this is what is meant by marginal and complete trust, then the scheme is far less powerful than it could be. If not, then it is not clear what is meant.

4.3 Finding Maurer's Assumptions in the PGP Trust Model

Section 3.3 showed that Maurer's scheme is far from perfect. However, I believe that it is a better Web of Trust model than that which is currently endorsed by PGP. This is because the PGP version of the Web of Trust includes all the inaccurate assumptions and other problems in Maurer's scheme as well being unable to create trust in as many certificates and being less clear about the meaning of its trust values.

As with Maurer's scheme, PGP users must work out how much trust they have in a key. While the imprecise nature of the three options in the current PGP scheme may make it easier to decide, this imprecision is mirrored in the outcome that a statement is valid, marginally valid or invalid instead of the continuous range of possible outcomes in the Maurer scheme.

There is an implicit assumption of the independence of statements in the current PGP trust model, unlike Maurer's explicit assumption. If Charlie has two keys, both of which are marginally trusted by Alice, he can use both of them to sign a certificate and make Bob's key valid from Alice's perspective. Or, more likely, two employees of the same company, both marginally trusted by Alice, may sign a certificate. This would appear no different to two totally unrelated and marginally trusted entities signing that certificate.

Both schemes overlook the possibility of one entity having multiple key pairs but this causes different problems because Maurer defines trust with respect to entities but PGP defines it with respect to keys. It is impossible to represent an entity with multiple key pairs in Maurer's scheme. As seen in the previous paragraph, the current PGP scheme can represent this but it produces undesired results.

Neither scheme uses policy constraints. However, the unclear meaning and imprecision of the current PGP trust statements make this scheme more vulnerable to El Bakkali and Kaitouni's criticism [2, p.368] as it was outlined in section 3.3.4.

4.4 Applying Maurer's Model to PGP

Maurer's model should be adjusted to handle a single entity that holds multiple key pairs. The statements $Aut_{A,X}$, $Trust_{A,X,i}$, $Cert_{X,Y}$ and $Rec_{X,Y,i}$ should be replaced with Aut_{A,X,K_X}, $Trust_{A,X,K_X,i}$, $Cert_{X,K_X,Y,K_Y}$ and $Rec_{X,K_X,Y,K_Y,i}$ respectively, where K_X is one of X's keys and K_Y is one of Y's keys. Similar adjustments should be made to the inference rules. This idea is based on El Bakkali and Kaitouni's approach [2]. Note that this change to the deterministic model would result in counter intuitive results from the probabilistic logic. If Alice's initial view included the statements Aut_{A,X,K_1}, Aut_{A,X,K_2}, $Trust_{A,X,K_1,i}$, $Trust_{A,X,K_2,i}$, $Cert_{X,K_1,Y,K_Y}$ and $Cert_{X,K_2,Y,K_Y}$ then the trust value for Aut_{A,X,K_Y} would be higher than if the initial view did not contain the statements Aut_{A,X,K_2}, $Trust_{A,X,K_2,i}$ and $Cert_{X,K_2,Y,K_Y}$. Therefore, a change from standard probabilistic logic would be necessary when the two certification paths used different keys held by the same entity. For example, let $S_1 = \{ Aut_{A,X,K_1}, Trust_{A,X,K_1,i}, Cert_{X,K_1,Y,K_Y} \}$ and $S_2 = \{ Aut_{A,X,K_2}, Trust_{A,X,K_2,i}, Cert_{X,K_2,Y,K_Y} \}$. The confidence value of Aut_{A,X,K_Y} should be $\max(P(S_1), P(S_2))$ instead of $P(S_1 \cup S_2) - P(S_1 \cap S_2)$.

Having made this change, PGP would only have to be altered slightly to make it compatible with the adjusted version of Maurer's trust model. Replacing the PGP trust levels (implicit, complete, marginal and no trust) with a degree of trust (a rational number between 0 and 1) and a level of trust (a natural number) is all that would be necessary. These could be implemented as a 16 bit positive integer (treated as a multiple of $\frac{1}{2^{16}}$) and an 8 bit positive integer (treated as a natural number between 0 and 255) respectively. This would allow sufficient precision for the degree of trust and far more levels of trust than anyone would realistically be willing to use.

5 Conclusions

All public key infrastructures must be based on one or more of three kinds of trust model: the Web of Trust, hierarchical trust and trust lists. The simplistic version of the Web of Trust assumes that all entities in a public key infrastructure can be trusted, which is unlikely to be true in any non-hierarchical public key infrastructure. The Web of Trust is only applied to non-hierarchical infrastructures and therefore its simplistic version is not useful. More complicated Web of Trust models overcome this by requiring trust in entities to be explicitly stated.

Trust in others is rarely absolute and so a measure of trust is helpful. A useful measure of trust will have values with obvious meanings and a great many values to increase precision. It is possible to design variants of the Web of Trust that calculate trust values on a continuous scale by using probability as the trust value. Since the values are interpreted as probabilities they are meaningful.

It is possible to develop a calculus for probabilistic trust values using probabilistic reasoning. A weakness of any probabilistic trust model is that it requires accurate initial trust measurements to be made, and probably by more than one entity. Although this is possible for some trust statements, a general approach to making initial probabilistic trust measurements has not been put forward. This is the main obstacle in the way of probabilistic trust models.

Maurer has outlined a probabilistic Web of Trust model. His model assumes that all statements about trust are independent and that entities are only bound to one public key. Neither of these assumptions is true in general, but future probabilistic Web of Trust models may be able to avoid making them. Maurer's model also suffers from an inability to handle many specific restrictions that users might want to put on certificates, such as stating that they are only valid in a particular domain. This could be handled on an ad hoc basis but it would complicate the trust statements and inference rules upon which the model is based.

Pretty Good Privacy combines a flexible public key infrastructure with the software that is needed for public key cryptography. Maurer's trust model, with one improvement, would be well suited to PGP. A minor change would have to be made to PGP keys. This would be preferable to the current Web of Trust model that is recommended for use with PGP. That model cannot provide trust in many identity certificates and it is unclear what its types of trust are supposed to mean.

References

[1] Diffie, W., & Hellman, M. E. (1976). New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22 (6), pp.644-654.

[2] El Bakkali, H., & Kaitouni, B. I. (2001). A Predicate Calculus Logic for the PKI Trust Model Analysis. In *Proceedings of the IEEE International Symposium on Network Computing and Applications* (pp. 368-371). Los Alamitos, California: IEEE.

[3] Linn, J. (November 6, 2000). *Trust Models and Management in Public-Key Infrastructures*. Retrieved April 11, 2003, from ftp://ftp.rsasecurity.com/pub/pdfs/PKIPaper.pdf

[4] Maurer, U. (1996). Modelling a Public-Key Infrastructure. In E. Bertino, H. Kurth, G. Martella & E. Montolivo (Eds.), *European Symposium on Research in Computer Security* (pp.33-43). Rome: Springer-Verlag.

[5] Network Associates, Inc. (1999). The Basics of Cryptography. In *An Introduction to Cryptography (the PGP 6.5.1 documentation)* (pp.11-36). Santa Clara, California: Author.

[6] Tanimoto, S. L. (1995). Probabilistic Reasoning. In *The Elements of Artificial Intelligence Using Common Lisp* (pp.329-374). New York: Computer Science Press.

Appendix A: Probabilistic Reasoning

Sometimes it is useful to be able to reason with inexact or uncertain information. One approach to drawing conclusions from this sort of information is called probabilistic reasoning. Anyone can offer an opinion on the likelihood that a particular statement is true. The scale used in probabilistic logic is from 0 to 1, where 0 is definitely false and 1 is definitely true. Although an individual is able to offer any opinion on the truth of a statement (and therefore any certainty value for that statement), some statements are true and some are false. Assigning some other value is simply incorrect. Sometimes it is clear what the probability of a statement is. For example, the probability that a fair coin will land heads up when tossed is ½. When it is not so easy to assign correct probability values, any calculations made using them are, at best, only as accurate as the initial values. The truth values provided by Maurer's scheme are treated as probability values.

There are two main rules that are used to calculate a probability value from other probability values. They are used by Maurer and have been defined as follows:

> "Let A and B be events having probabilities $\mathrm{P}(A)$ and $\mathrm{P}(B)$, respectively.
> 1. Additive Law: $\mathrm{P}(A \cup B) = \mathrm{P}(A) + \mathrm{P}(B) - \mathrm{P}(A \cap B)$.
> …
> 2. Multiplicative Law: $\mathrm{P}(A \cap B) = \mathrm{P}(A) \times \mathrm{P}(B \mid A) = \mathrm{P}(B) \times \mathrm{P}(A \mid B)$.
> Here $\mathrm{P}(B \mid A)$ is $\mathrm{P}(B$ given $A)$ …" [6, p.332]

Since Maurer does not assume that an entity knows the probability of $\mathrm{P}(B \mid A)$, he is forced to assume that all statements are independent so that $\mathrm{P}(A \cap B) = \mathrm{P}(A) \times \mathrm{P}(B)$. In section 3.3.2, I briefly discuss the possibility that the independence assumption could be replaced by a method of calculating the interdependence of any two statements. If this was done, Bayes' rule could be used to calculate the trust values in an interdependent variation of Maurer's trust model. Bayes' rule is:

> "$\mathrm{P}(H \mid E) = \dfrac{\mathrm{P}(E \mid H)\,\mathrm{P}(H)}{\mathrm{P}(E)}$
> where
> $\mathrm{P}(E) = \mathrm{P}(E \mid H)\,\mathrm{P}(H) + \mathrm{P}(E \mid \neg H)\,\mathrm{P}(\neg H)$" [6, p.333]

The additive law, the multiplicative law and Bayes' rules all work with probabilities that are assumed to sum to one. Symbolically, $\mathrm{P}(H) + \mathrm{P}(\neg H) = 1$. There are other scales of probability that need not sum to one [6, p.330] but Maurer does not consider these.