

Toward Acceptable Metrics of Authentication

Michael K. Reiter

Stuart G. Stubblebine

AT&T Labs—Research, Murray Hill, New Jersey, USA
{reiter, stubblebine}@research.att.com

Abstract

Authentication using a path of trusted intermediaries, each able to authenticate the next in the path, is a well-known technique for authenticating entities in a large-scale system. Recent work has extended this technique to include multiple paths in an effort to bolster authentication, but the success of this approach may be unclear in the face of intersecting paths, ambiguities in the meaning of certificates, and interdependencies in the use of different keys. Several authors have thus proposed metrics to evaluate the confidence afforded by a set of paths. In this paper we develop a set of guiding principles for the design of such metrics. We motivate our principles by showing how previous approaches fail with respect to them and what the consequences to authentication might be. We then propose a direction for constructing metrics that come closer to meeting our principles and thus, we believe, to being satisfactory metrics for authentication.

1 Introduction

Determining the owner of a public key, or conversely determining the public key for a user, appears to be a basic ingredient for executing transactions securely in any large-scale open system. Due to the lack of a single authority for providing this information in a system having many different administrative domains, many systems (e.g., DSSA [7], SPX [19], PEM [10], PGP [24]) resort to authentication by a *path* (or *chain*) of authorities. In this model, the user locates a path (sequence) of authorities such that (i) the user can authenticate the first authority in the path, (ii) each authority in the path can authenticate the next authority in the path, and (iii) the last “authority” in the path is in fact the targeted person or key of interest. If the user trusts every authority on the path, then perhaps it can believe that a proper name-to-key

binding has been obtained. To our knowledge, using such paths for authentication was first proposed in [5] (for authentication based on shared keys) and, in addition to being used in the aforementioned systems, has been supported in [21, 8, 12, 23].

A path of authorities is weak because it relies on the correctness of every authority in the path; if any authority in a path incorrectly authenticates the next authority, then the user can be misled regarding the authentication of subsequent authorities in the path, including the target. A natural approach to increasing assurance in the authentication of the target is to use multiple paths. However, the assurance provided by these multiple paths may be unclear, especially if they have authorities in common or authorities that act in a correlated way. When combined with ambiguities in the assertions that authorities make and ambiguities regarding who is actually making the assertions, it may be difficult to complete the authentication with any confidence.

Several researchers have therefore proposed *metrics* for measuring the assurance provided by a collection of paths (e.g., [20, 4, 15, 14, 16]). For example, a metric might take as input several such paths of authorities and return a numeric value, where a higher value indicates greater confidence in the name-to-public-key binding (for the target name or public key) that those paths support. Extensions for supporting metrics in X.509 certificates have been proposed (e.g., [15]), and we ourselves have deployed a web service called PathServer that, as we will argue here, can be viewed as computing a metric to support the authentication of PGP keys [16] (see <http://www.research.att.com/~reiter/PathServer>). Based on our evaluation of several metrics, we believe that the design criteria for these metrics are not widely agreed upon. Indeed, most metrics—including our own—seem to have been put forth with attention to a few specific goals, at the expense of other, arguably important, properties.

The goal of this paper is to elucidate some of these

properties that we believe to be important. Specifically, in this paper we offer a set of design principles for metrics of authentication, and we illustrate each using (usually shortcomings of) metrics already proposed. We consider only metrics whose goal is to measure the confidence in a name-to-public-key binding that a collection of information (e.g., certificates) supports. Our principles focus on three main areas, namely the meaning of values output by the metric, the extent to which metric outputs can be manipulated by malicious behavior (e.g., the compromise of cryptographic keys), and the prospects for effectively making use of the metrics in practice. While we demonstrate many of these principles by exhibiting limitations in existing metrics, we emphasize that the principles, not the limitations, are the main point of this paper. We also propose a direction for constructing metrics that, we believe, come closer to meeting our principles and thus to being acceptable as metrics of authentication.

For clarity it is also worthwhile to comment on what we are *not* trying to do in this paper. First, the reader should not confuse our work with recent efforts to capture principles for the design of cryptographic protocols [3, 1, 18]. The present work has little to do with protocols; we care about how to evaluate the confidence that authentication paths afford, and not the protocols by which certificates (or any other structures) are communicated. Second, user policy that maps metric output values to a “yes/no” decision as to whether the confidence in authentication is “good enough” is also beyond the scope of this paper. Third, we do not claim to have identified a complete set of principles for the design of metrics or, learning from [18], that there are no exceptions to our principles. We are, however, unaware of compelling counterarguments or counterexamples to our principles, and we believe them to be sound advice for the design of metrics.

As mentioned above, we use prior work on metrics to illustrate our principles, and so we outline this work in Section 2. We put forth our design principles in Section 3. Based on these principles, we then explore a new direction for constructing metrics in Section 4. We conclude in Section 5.

2 Overview of proposed metrics

The metrics that we use for illustration are due to Beth, Borcharding, and Klein [4], Zimmermann [24, 17], Maurer [14], and Reiter and Stubblebine [16]. The Zimmermann and Reiter-Stubblebine procedures were not presented as metrics per se in their publications,

but we take the liberty of interpreting them as metrics for our purposes. In addition, the Beth-Borcharding-Klein and Reiter-Stubblebine metrics are not limited to public key infrastructures, but for simplicity we describe them only in this context. We describe each metric below only to the extent necessary to set the stage for the rest of the paper; some will be described in less detail than others in this section, but will be expanded upon later in the paper. Other work on metrics is described in [20, 15], but since these metrics evaluate only a single path of authorities (notably using path length as a metric), we do not use them for comparison here.

Each of the metrics described below operates in the context of a model that consists of a directed graph whose nodes and edges are labeled in various ways. However, no two metrics share the same model (i.e., the same graph), and it is important for the rest of the paper to understand the differences in the models that the different metrics use. Indeed, a contribution of this section is distinguishing the various models that have been proposed to capture a “certification graph”, and subsequent sections yield insight into the relative advantages and disadvantages of each.

Except where we explicitly state otherwise, we consider only how each metric performs on a model containing only consistent information, i.e., where there are no conflicting reports regarding the owner (or other attributes) of a key. While how a metric behaves on conflicting information is important, we typically omit this issue to simplify discussion. Also, in the rest of the paper we will use the following terminology. An *entity* is something that possesses and makes use of a private/public key pair, e.g., a person, authentication server, or certification authority. The *user* is the person applying the metric for the purpose of gaining assurance in a name-to-key binding.

Beth-Borcharding-Klein The Beth-Borcharding-Klein metric takes as input a set of trust relationships that can be represented by a directed graph. The nodes of the graph are entities. There are two types of edges in this graph. The first type is a “direct edge”; the direct edge $A \rightarrow B$ means that A believes it can authenticate (i.e., has the public key for) B . The second type of edge is a “recommendation edge”; the recommendation edge $A \rightsquigarrow B$ represents that A trusts B to authenticate other entities or to recommend other entities to authenticate or further recommend. Associated with each recommendation and direct edge is a value in the range $[0, 1]$. In the case of a direct edge $A \rightarrow B$, this value is A ’s estima-

tion of the probability that A really holds the correct public key for B . The value on a recommendation edge $A \rightsquigarrow B$ represents the degree of A 's trust in B as a recommender, where higher values indicate stronger trust. The authors present a formal model to justify these values [4].

Given a specific query, say user A wanting the public key for entity B , the metric computes a value in the range $[0, 1]$, using all paths from A to B whose last edge is direct and whose other edges are recommendation edges, such as $A \rightsquigarrow C \rightsquigarrow D \rightarrow B$. The exact rules used and an example of such a computation are given in Section 3.2.

Maurer The Maurer metric takes a directed graph as input, as well. As in Beth-Borcherding-Klein, the nodes of this graph are entities and there are two types of edges, which we will again call “direct” and “recommendation”. However, the semantics of these edges are subtly different in the Maurer model, in that these edges represent syntactic constructs, e.g., certificates. A direct edge $A \rightarrow B$ means that the user evaluating the metric “holds a certificate for B 's public key (allegedly) issued and signed by entity A ”. Similarly, a recommendation edge $A \rightsquigarrow B$ denotes that the user is in possession of a recommendation (for recommending or authenticating other entities) for B allegedly signed by entity A . Associated with each recommendation and direct edge is a value in the range $[0, 1]$, called a *confidence parameter*, that is assigned by the entity that created (the construct represented by) the edge. Given a specific query, e.g., user A wanting the public key for B , the metric computes a confidence value in the range $[0, 1]$ for the key that the model suggests is B 's, using the confidence parameters specified for the edges as probabilities.

Reiter-Stubblebine The Reiter-Stubblebine metric takes a directed graph as input, but again this graph differs from those for the Beth-Borcherding-Klein and Maurer metrics. In this case, the nodes of the graph are public keys (actual keys, with no references to any entities), and an edge $K_1 \rightarrow K_2$ means that the user evaluating the metric has a certificate signed by the private key corresponding to K_1 (i.e., K_1 can be used to verify the signature) and that assigns attributes to K_2 . The attributes bound to K_2 in this certificate, which are assumed to assert K_2 's owner (among other things, perhaps), are included as a label on the edge $K_1 \rightarrow K_2$. There are no other values associated with edges or nodes.

Reiter and Stubblebine developed two related met-

rics in [16]. Each metric takes as input the above graph, a key that the user wishes to authenticate (the target key), a key that the user trusts (the source key, e.g., her own), and a bound b on the length of paths to consider. The first metric returns a maximum set of node-disjoint paths of length at most b from the source key to the target key. The second metric returns an integer k and set of paths of length at most b from the source to the target such that k nodes have to be removed (compromised) to break all the paths; the value of k returned is the maximum k for which such a set of paths exists, and is called the *connectivity* from the source key to the target key. If we insist that these metrics produce a numeric output, then in the case of disjoint paths it would be the number of disjoint paths that it returns, and in the other case it would be the connectivity. When convenient, we will use the disjoint paths metric only in our discussion.

Zimmermann The metric that we attribute to Zimmermann is that used in PGP [24], one of the most popular civilian public key management systems in the world today. Zimmermann's graph resembles (but also preceded) the Reiter-Stubblebine graph. Its nodes are keys, and the edge $K_1 \rightarrow K_2$, labeled with attributes, represents a certificate that binds these attributes to K_2 and that can be verified with K_1 . It differs from the Reiter-Stubblebine graph, however, in that the user augments each node with a *trust value*, which is one of unknown, untrusted, marginally trusted, or fully trusted.

PGP computes the *legitimacy* [17] of each node as follows.¹ PGP first declares to be legitimate the node K_0 representing the user's key and any node K such that $K_0 \rightarrow K$ is an edge in the graph. PGP then repeats the following until no more keys can be determined to be legitimate: if for some node K , either (i) there is an edge to K from a legitimate fully trusted node or (ii) there are edges to K with identical labels from two legitimate marginally trusted nodes, then K is declared legitimate. The numbers of edges required from fully trusted or marginally trusted nodes can be adjusted, but one and two are the defaults, respectively. In practice, determinations of node legitimacy are interwoven with assigning trust values to nodes. That is, a trust value is assigned to a node only after it has been determined to be legitimate and thus its owner is assumed to be known (i.e., named on the

¹This description is derived from [24, 17] and our own experiments with PGP 2.6.2. It is also simplistic in some regards. In particular, it omits discussion of the `CERT_DEPTH` parameter for limiting path length.

one edge to it from the fully trusted node or the two edges to it from the marginally trusted nodes). For the purposes of modeling, however, the end result is the same.

Intuitively, PGP might not be considered to implement a metric, but rather to simply determine whether a key is legitimate (authenticated) according to the policy described above. Alternatively, one might construct a metric from PGP by issuing multiple queries to PGP with different parameters to determine, e.g., the actual number of edges from legitimate marginally trusted nodes to a target node.

3 Design principles

Even with as brief an overview as that in Section 2, it is clear that these metrics differ significantly. Rather than giving a point-by-point comparison, however, we think it more beneficial to attempt to draw principles from these metrics that are desirable in general. We divide our principles into three general categories: meaning of the metric results, sensitivity of the metric to entity misbehavior, and the practical effectiveness of the metric. For each principle, we illustrate why it is desirable by demonstrating how one or more metrics fall short of it, and what the consequences might be to authentication decisions.

Some of the principles that we propose will seem obvious and in fact are not new, at least in spirit. Notably, Maurer proposes high-level desiderata for models of public key infrastructures [14, Section 2.7]. Our list of principles shares certain ideals with Maurer’s, but here we also strive for more specific principles. In addition, to our knowledge many of our principles and our demonstrations of how proposed metrics fall short with respect to them are new. Again, some principles may be obvious, but since we can demonstrate metrics that do not comply, we believe that even the obvious ones bear repeating.

3.1 Meaning

We begin with the most basic desideratum of a metric, namely that its output be meaningful. Clearly all of the metrics we consider have strived for this, though we will argue that some achieve it better than others. One of the primary factors that determines the degree to which a metric is meaningful is the precision of its model. Principle 1 is an important consideration, we believe, in constructing a model.

Principle 1: Creating the model to which a metric is applied should not require the user to infer bindings between keys and their owners. In particular, when representing certificates in a model: *Entities don’t sign certificates, keys do.*

One motivation for this principle is that establishing name-to-key bindings is arguably a difficult and error-prone process; otherwise we would not need metrics. A metric that falls short of this principle is Maurer’s. To repeat from the previous section, the edge $A \rightarrow B$ exists in the Maurer model if the user evaluating the metric “holds a certificate for B ’s public key (allegedly) issued and signed by entity A ” [14, Definition 3.1]. Maurer uses the word “allegedly” because “without verification, there exists no evidence that the certificate was indeed issued by the claimed entity.” Put another way, when the entity that allegedly signed the certificate is claimed with the certificate, this claim is at best a hint and at worst an opportunity to be misled. It is presumably for this reason, however, that in some systems a certificate includes *no claim at all* of the *entity* that signed it, but rather only a claim of the *key* that signed it. For example, a PGP certificate indicates only an identifier for the public key that can be used to verify the signature on the certificate [17, Chapter 6]. In such cases, how a certificate should be represented in the Maurer model is ambiguous, and presumably the user must infer the certificate’s signer from other certificates for the key that verifies the certificate’s signature. One interpretation even allows a certificate to be represented by multiple edges if different certificates indicate different owners for its verification key.

A similar concern arises in the Beth-Borcherding-Klein model, the other model in which nodes are entities. Evaluating this metric requires the user to collect values from other entities for the various direct and recommendation edges. However, before the user can safely assign a value to the edge $A \rightarrow B$ or $A \rightsquigarrow B$, the user must authenticate this value as having come from A . Assuming that this authentication is performed cryptographically (e.g., via a certificate), again the user is asked to determine a key that can be used to authenticate A in order to form the model for the query she wants answered.

A second motivation for this principle is that modeling a certificate as being signed by only an entity hides the key used to sign the certificate. This can result in ambiguities when representing certificates if, e.g., there are multiple certificates signed by different private keys owned by the same entity.

Principle 2: The meaning of the model’s parameters should be unambiguous. This especially applies to the meaning of probabilities and trust values in models that use them.

As one would expect, ambiguous semantics of a model’s parameters can generally lead to different metric values depending on one’s interpretation of the parameters. Such discrepancies in interpretations must be resolved before the output of a metric can be meaningful, especially if one entity relies on numbers from another entity with a different interpretation.

This issue arises in Maurer’s metric, for example, because the policy that dictates how confidence parameters are assigned to certificates and recommendations is left unspecified. This raises two concerns. First, Maurer’s interpretation of these confidence parameters as probabilities is not sufficiently justified. Indeed, the suggested means for determining confidence parameters (e.g., “speaker identification over a telephone line should be assigned a confidence parameter of at most 0.95”) seem to bear no relationship to random experiments. Second, since the user evaluating the metric presumably must adopt the confidence parameters for certificates and recommendations determined by their creators, any ambiguities in the semantics of these parameters can be compounded by misinterpretation by the user.

Beth-Borcherding-Klein is more complete in this regard, prescribing a detailed formula for each entity to compute the label on each of its edges (trust relationships) based on the numbers of *positive and negative experiences* that the entity has with that trust relationship. PGP leaves the specification of trust designations outside the model and provides little guidance for determining them, but since these trust designations are treated as confidential data, they are not propagated from one user to another.

Principle 3: A metric should take into account as much information as possible that is relevant to the authentication decision that the user is trying to make.

This principle is subject to some caveats; see Section 3.3.

Principle 3 is desired because if a metric produces output based on limited information, there generally will be a greater effort required to interpret whether the authentication is “good enough”. This is demonstrated, for example, with the Reiter-Stubblebine metric. As described in Section 2, the Reiter-Stubblebine model consists solely of a graph whose nodes represent keys and whose edges represents certificates available to the user evaluating the metric. In contrast to

the other three metrics, the Reiter-Stubblebine metric makes no effort to take into account trust relationships or recommendations among entities; indeed, entities appear nowhere in its model (except named within the labels attached to edges, but the metric does not consider these). As such, when the metric returns a set of disjoint (or connective) paths, the user is left to determine whether the paths are “good enough” based upon who she trusts and the labels on the various edges in the paths. Even worse, since the metric does not take into account trust information, it may actually *inhibit* the user’s decision on whether to adopt the recommended name-to-key binding, by including in its returned paths some nodes and edges that the user is unfamiliar with, at the expense of others that the user would have preferred.

PGP is somewhat better in this regard. Its model does allow the user to specify what keys it trusts for certification, but on the other hand it provides no help to the user in making this decision, i.e., it has no way to account for recommendations. The means by which the user determines who to trust for certification is outside the model. The Maurer and Beth-Borcherding-Klein metrics do provide a way to accommodate recommendations from other entities.

Principle 4: The output of a metric should be intuitive. It should be possible to write down a straightforward natural language sentence describing what the output means.

The motivation for this principle is clear: in order for a user to determine what metric value is “good enough” for her application, she must know what the metric output means. The Maurer metric is one, we believe, that is suspect in this regard. This metric computes a confidence value for a name-to-key binding as the probability that the binding can be derived from the initial view of the user, where the random event is the selection of the initial view, i.e., the selection of a random subset of the certificates and recommendations available to the user, using a distribution defined by the confidence parameters assigned to edges. As this experiment does not correspond to a familiar procedure in the real world, it remains to be seen whether the average user is willing to understand and believe a metric computed in this way.

3.2 Sensitivity

In this section we discuss the sensitivity of metrics to misbehavior of entities. The type of “misbehavior” that we focus on is deceit by one or more entities represented in the model (or that supply input

to the model) in which the metric is applied, in an effort to manipulate the output of the metric to increase the user’s confidence in the authentication. If an attacker is able to inflate the metric output to the point that the application accepts the authentication, then the metric is not serving its purpose. To illustrate our point, we demonstrate that the Beth-Borcherding-Klein metric is overly sensitive to misbehavior. In fact, this metric has the property that a single misbehaving entity can increase or decrease the result of the metric arbitrarily.

To show this, it is necessary to review the specific rules used to compute the Beth-Borcherding-Klein metric. Recall from Section 2 that each edge in the Beth-Borcherding-Klein model is labeled with a value in the range $[0, 1]$. Suppose that A wants to authenticate (determine the public key for) B . Beth, et.al., propose and justify the following rules for computing an aggregate “score” for A ’s authentication of B based upon the values on the edges of the paths (of the form described in Section 2) connecting A to B .

1. If there is a path $A \rightsquigarrow \dots \rightsquigarrow C$ with recommendation value v_1 and a recommendation edge $C \rightsquigarrow D$ with value v_2 , then the path $A \rightsquigarrow \dots \rightsquigarrow C \rightsquigarrow D$ has recommendation value $v_1 \cdot v_2$.
2. If there is a path $A \rightsquigarrow \dots \rightsquigarrow C$ with recommendation value v_1 and a direct edge $C \rightarrow B$ with value v_2 , then the path $A \rightsquigarrow \dots \rightsquigarrow C \rightarrow B$ has direct trust value $1 - (1 - v_2)^{v_1}$.
3. If for each $1 \leq i \leq m$, there are n_i distinct paths from A to B ending with the edge $C_i \rightarrow B$, with direct trust values $v_{i,1}, \dots, v_{i,n_i}$, then the combined direct trust value is

$$v_{com}(A, B) = 1 - \prod_{i=1}^m \sqrt[n_i]{\prod_{j=1}^{n_i} (1 - v_{i,j})}$$

To see an example of applying these rules, consider the graph of Figure 1(a), which is based on an example from [4]. By rules 1 and 2 above, paths $A \rightsquigarrow C \rightsquigarrow F \rightarrow B$, $A \rightsquigarrow D \rightsquigarrow G \rightarrow B$, and $A \rightsquigarrow E \rightsquigarrow G \rightarrow B$ yield direct trust values of .204, .173, and .765, respectively. Combining these with rule 3, we get $v_{com}(A, B) = .649$.

Now consider the graph of Figure 1(b), which is a manipulation of the graph in Figure 1(a) caused by D ’s misbehavior. Here, D has created additional artificial paths from A to B through other nodes H, I, J that D “invented” for the purposes of altering the metric output. The trust value assigned to

path $A \rightsquigarrow D \rightsquigarrow H \rightarrow B$ by rules 1 and 2 above is .821, and similarly for $A \rightsquigarrow D \rightsquigarrow I \rightarrow B$ and $A \rightsquigarrow D \rightsquigarrow J \rightarrow B$. Rule 3 then yields a combined trust value of $v_{com}(A, B) = .998$. What this example shows is that a single misbehaving node, by manipulating the graph used in the computation of the Beth-Borcherding-Klein metric, can drive the metric arbitrarily close to any value it chooses, and in particular to a high value that inflates the confidence expressed by the metric. Thus, in the face of malicious entities, it is unclear that Beth-Borcherding-Klein is a useful metric.

We should note that Beth-Borcherding-Klein allows for the exclusion of paths based on “constraint sets”, and thus a user that is familiar with the graph structure could, e.g., explicitly exclude paths through H, I , and J . If the user is not familiar with what the graph structure should be, however, then the user might have no basis to exclude such paths.

The above example leads us to the following principle.

Principle 5: A metric should be designed to be resilient to manipulations of its model by misbehaving entities, and its sensitivity to various forms of misbehavior should be made explicit.

An example of a metric that follows this principle is the Reiter-Stubblebine metric. The disjoint paths (or the connectivity) from the source key to the target key degrades gracefully in the face of misbehaving nodes, in the sense that a misbehaving node can inflate the number of disjoint paths from the source key to the target key by at most one. Indeed, given the origins of disjoint paths and connectivity in the network reliability literature, these metrics can be seen as primarily a measure of fault-tolerance.

Another example of a metric that falls short of this principle is Maurer’s. Maurer’s is generally not as sensitive to misbehavior as Beth-Borcherding-Klein, but it still fails to be explicit about how sensitive a score is that it returns. The score returned by the Maurer metric can range from being very sensitive (e.g., if it is computed using only a single path from the source to the target) to very tolerant (e.g., if many disjoint paths are involved).

3.3 Effectiveness

In this section we focus on the practical effectiveness of the metric, or in other words, the characteristics of a metric that make it simple or difficult to utilize in a large-scale system. Since the Reiter-Stubblebine and Zimmermann metrics are presently

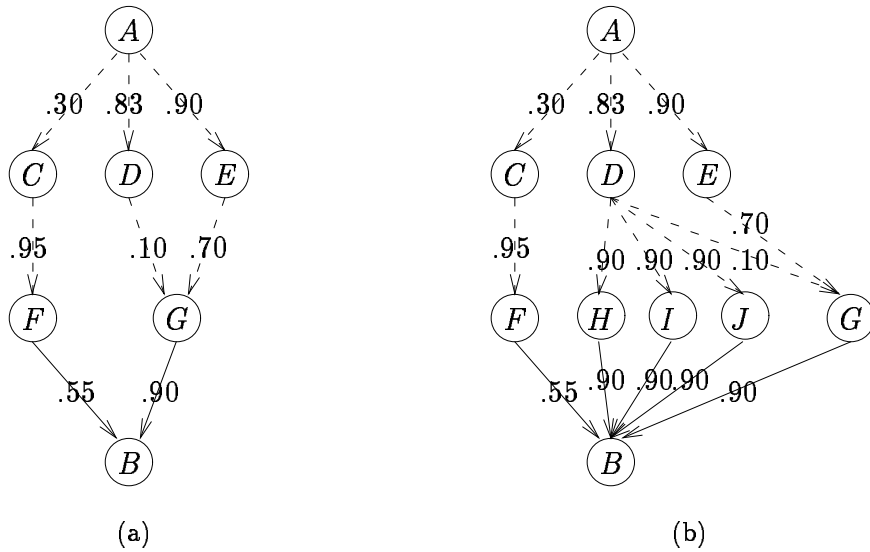


Figure 1: Effect of misbehaving node on BBK (recommendation edges are dashed)

in use (the former as a web service and the latter in a standalone program), one might presume that these have certain advantages in this arena. This is true to some extent, though one can argue that this ease-of-use has been achieved at the cost of hiding certain decisions from users that should not be hidden.

Principle 6: A metric’s usability should not rely on it hiding authentication-relevant decisions from the user that cannot be accurately automated. A decision that could affect authentication should be hidden from the user only if it can be reached using unambiguous, well-documented, and intuitive rules.

An example of a metric that does not adhere to this principle is PGP’s. As discussed in Section 2, in PGP a user assigns a level of trust to each node (key), which is one of unknown, untrusted, marginally trusted, and fully trusted, based upon its apparent owner. By default, PGP will declare a key legitimate if it is certified by one fully trusted key or two marginally trusted keys.

It is in this mechanism that PGP strays from Principle 6 above. It is often the case that a single user has two or more keys and uses each of these keys to certify another. An actual example is shown in Figure 2. The first line describes a key-to-name binding, namely the binding between the key with identifier C7A966DD and the name Philip R. Zimmermann <prz@acm.org>. The second and third lines show that Jeffrey I. Schiller

<jis@mit.edu> has signed this binding with two different keys, namely those identified with ODBF906D and 4DOC4EE1.

Now, if the user tells PGP that Jeffrey I. Schiller <jis@mit.edu> is a marginally trusted introducer, then PGP will mark keys ODBF906D and 4DOC4EE1 as marginally trusted, and therefore key C7A966DD will be declared legitimate (i.e., bound to Philip R. Zimmermann <prz@acm.org>). This occurs despite the fact that only one marginally trusted *person* was involved in certifying the key, whereas such a conclusion should supposedly require two.

This example points to a deeper problem than just an oversight in the PGP implementation. Rather, it points to the difficulty of determining if two keys are adequately independent for the purposes of independent certification, and we are aware of no foolproof way to automate this decision. For example, simply verifying that the names bound to ODBF906D and 4DOC4EE1 are different does not suffice, since they may have different email addresses but still indicate the same person, or since the key owners may not be the same but still act in a correlated way (e.g., two close friends). We thus believe that this decision should not be hidden from the user. However, appealing to the user impacts the ease-of-use of the metric, as this decision can be unclear even for the user.

This problem is not unique to PGP. The Reiter-Stubblebine metric, if interpreted as returning simply a number of disjoint paths, would share this prob-

Type	bits/keyID	Date	User ID
pub	1024/C7A966DD	1993/05/21	Philip R. Zimmermann <prz@acm.org>
sig	0DBF906D		Jeffrey I. Schiller <jis@mit.edu>
sig	4DOC4EE1		Jeffrey I. Schiller <jis@mit.edu>

Figure 2: PGP output showing signatures on key C7A966DD

lem. In reality, the implementation of the Reiter-Stubblebine metric in PathServer returns the actual paths, leaving this problem to the user to figure out (see the discussion after Principle 3 in Section 3.1).

Principle 7: A metric should be able to be computed efficiently.

This principle is obvious, but surprisingly it plagues three of the four metrics used for illustration in this paper. Much of [16] is devoted to finding ways to *approximate* the metrics it proposes, since one of the Reiter-Stubblebine metrics is NP-hard and the other is coNP-hard. The given procedures for evaluating the Beth-Borcherding-Klein and Maurer metrics are exponential in the size of the model in the worst case [4, 14], though Maurer also discusses directions to approximate his metric. The only metric of the four for which it is known how to compute the metric efficiently in all cases is Zimmermann’s.

We conclude this section with a principle concerning the output of the metric in the face of partial information.

Principle 8: A metric’s output on partial information should be meaningful.

The motivation for this principle is as follows: in a large-scale system, it may be difficult or even impossible to gather all certificates that have been created. As a result, metrics will almost certainly be applied only to a subset of the certificates that actually exist at that time, and indeed some of these certificates may have been revoked. Principle 8 simply says that the metric’s output should have some relevance even when computed on this partial information.

One metric that does not obey this principle is Beth-Borcherding-Klein. It is easy to verify (and it is noted in [4]) that additional edges added to its model can increase or decrease the metric output by an arbitrary amount. Therefore, the metric’s output on partial information may give the user little insight into the “actual” quality of the name-to-key binding. If, on the other hand, PGP determines a key to be legitimate, then the key will remain legitimate no matter

what additional certificates are obtained. Similarly, the disjoint paths returned by the Reiter-Stubblebine metric survive the addition of new certificates to the graph, and the user can be assured that a path exists even if, unbeknownst to the user, certificates on all but one of the paths have been revoked. This offers the user some basis to decide whether the authentication is good enough for its application.

An interesting refinement of Principle 8 is to express it precisely in terms of *monotonicity*, e.g., that adding more certificates to a metric’s input should never decrease the level of assurance indicated by the metric’s output (cf., [4, 14]). To be complete, such a definition must carefully treat certificate revocations and certificates that indicate conflicting information. We leave such a definition as an exercise to the reader.

4 Toward better metrics

As indicated in the previous section, we believe that none of the metrics we have used in our discussion—or for that matter no metric that has been proposed—fully meets our principles for the design of metrics. In this section we outline a metric that, we believe, can come close to meeting our principles. The metric is based on the concept of insurance for name-to-key bindings, which we expect would be appropriate for many business applications.

The model on which our metric will operate will again be a directed graph. As in the Zimmermann and Reiter-Stubblebine metrics, the nodes of this graph are public keys, and the edge $K_1 \rightarrow K_2$ exists in the model if the user is in possession of a certificate that assigns attributes (including an owner) to K_2 and whose signature can be verified using K_1 . Each edge is labeled with the attributes included in the certificate that the edge represents. As we have throughout this paper, we restrict our attention to graphs containing only consistent assertions about the attributes for each key.

Each edge $K_1 \rightarrow K_2$ also has a numeric label that represents the amount of money for which the owner of K_1 insures the attributes and behavior of K_2 , i.e., the value for which the owner of K_1 will be liable to

the user if the attributes bound to K_2 in the certificate are incorrect, or if (the private key corresponding to) K_2 is used to mislead the user, intentionally or otherwise. In particular, if the private key corresponding to K_2 is compromised and used maliciously, then the owner of K_1 is liable for the stated amount. In effect, the owner of K_1 is indemnifying the user against losses incurred by a false authentication of K_2 based on a certificate it verified with K_1 , or by the misbehavior of K_2 . This form of insurance is called *surety bonding*, as described in [13]. It is also reminiscent of (but different from) insurance represented in some draft banking certificate standards (e.g., [22]) and characteristic of VeriSign certificates (see <http://www.verisign.com/>).

The insurance label of the edge $K_1 \rightarrow K_2$ must be obtained from the owner of K_1 in some reliable way, and so it is natural for this value to be stored in the certificate that $K_1 \rightarrow K_2$ represents. Note that we are not asking the user to determine the true owner (or other attributes) of K_1 , in accordance with Principle 1, or to determine that K_1 has not been compromised. Indeed, K_1 could have been compromised and used to forge the certificate $K_1 \rightarrow K_2$, including all its attributes and the insured value it contains. In this case, however, whoever certified K_1 would be liable, and this can regress along a path arbitrarily far (cf. [13]). That is, on any path from a trusted source key to a target key (both specified by the user), if $K_1 \rightarrow K_2$ is the first *liable edge* in that path, i.e., the first edge on which the attributes are inaccurate or the certified key (K_2) misbehaves, then the owner of K_1 (specified by the key that certified K_1) is liable. In practice, rules for determining which edge is the first liable edge on a path would need to be established. We henceforth assume that such determinations can be made.

Obtaining a false name-to-key binding for the target key implies that every path from the source key to the target key must have some first liable edge. Once these edges are identified, the owners of the keys that created those edges can be held liable, each for the insured amount on the liable edge(s) that it created. It follows that a natural and prudent metric to compute would be the *minimum insured amount* of the name-to-key binding for the target key. That is, over all possible ways of choosing liable edges that intersect every path from the trusted source key to the target key, what is the minimum amount of money that the user can expect to recover?

This amount can be captured precisely using a well-known tool from graph theory called a *minimum capacity cut* [6]. Let K_s denote the trusted source key,

K_t denote the target key, and for each edge $K \rightarrow K'$, let the *capacity* of the edge, denoted $c(K, K')$, be the insured amount of the edge. For nonexistent edges $K \rightarrow K'$, let $c(K, K') = 0$. A K_s, K_t -cut (or just a *cut*, when K_s and K_t are understood) is a partition of the nodes of the graph into two sets A and B such that $K_s \in A$ and $K_t \in B$. The *capacity* of a K_s, K_t -cut (A, B) , denoted $c(A, B)$, is simply the total capacity of the edges from A to B , i.e.,

$$c(A, B) = \sum_{K \in A, K' \in B} c(K, K').$$

A minimum capacity K_s, K_t -cut is then a cut with minimum capacity over all possible K_s, K_t -cuts. An example of a minimum capacity cut is shown in Figure 3. Note that any set of liable edges that intersects every path from K_s to K_t naturally induces a K_s, K_t -cut: remove these liable edges from the graph, insert any nodes reachable from K_s into A , and define B to be the complement of A . It follows that the capacity of a minimum capacity cut is a minimal amount for which the name-to- K_t binding is insured.

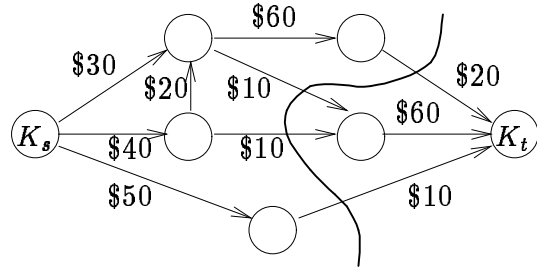


Figure 3: Minimum cut yields \$50 insurance for name-to- K_t binding

To summarize, our metric takes the graph, a trusted source key K_s , and a target key K_t as input, and returns an amount for which the name-to-key binding is insured by computing the capacity of a minimum capacity K_s, K_t -cut. Extensions of this metric could refine this computation based on trust. For example, the model could allow the user to limit the nodes that the metric includes in its computation, based on her trust in their apparent owners (specified by the edges that certify them) to pay if held liable. This is similar to PGP’s trusted designations.

There are numerous “real world” issues that this metric does not address, such as payment of insurance premiums, identifying liable parties, and recovering funds from them, if necessary. However, the

need for real world support for this metric is, we believe, due to the fact that it returns *meaningful* results. Rather than returning abstract probabilities, the metric reduces the problem to something we understand: money. We also believe that the metric can satisfy our other principles, for the following reasons.

Principle 1: The user is not required to ascertain name-to-key bindings to construct the model for this metric, as described above.

Principle 2: The notion of insurance is well-defined in business and legal culture and, we expect, can be extended naturally for this application. The extensions described above to allow the user to specify trust in entities to pay is also grounded in well-established business practice: for example, Dun & Bradstreet Corporation (see <http://www.dbsna.com>) provides industry-standard reports that rate the solvency and payment history of organizations using a well-defined rating system. (Note that if an organization such as Dun & Bradstreet is used to assign trust designations, then it is acting as a trusted recommender.)

Principle 3: This metric enables a user to weigh her financial risk associated with each transaction against the amount she can expect to recover if a name-to-key binding relied upon for the transaction is false. We expect that this information is adequate for a user to determine, for most business applications, whether the assurance in the name-to-key binding is sufficient.

Principle 4: The output of this metric is intuitive and natural: it is simply an amount for which the name-to-target key binding is insured.

Principle 5: This metric computes an insured value for the name-to-target key binding that the user can safely expect to recover if misled, regardless of what entities misbehave or what keys are compromised (other than the trusted source key). In particular, the metric's output is always bounded from above by the capacity of the cut $(\{K_s\}, V \setminus \{K_s\})$, where V is the set of all nodes. So, the level of insurance offered by the trusted source node prevents malicious entities from increasing the metric output above that level.

Principle 6: We are unable to identify any decisions within this metric that could affect authentication and that are wrongfully hidden from the user. In particular, the metric incorporates no determinations of key or entity independence for the purposes of authenti-

cation, which is the point on which the PGP metric stumbled. The primary decision (or more accurately, assumption) made within this metric is that the creator of the first liable edge on any path from the trusted source key to the target key will pay the amount for which it insured that edge. However, as we have already described, this decision can be left to the user, by allowing the user to designate her trust in nodes to pay using standard business reports.

Principle 7: The capacity of a minimum capacity cut can be computed using any *maximum flow* algorithm [6], of which there are many efficient examples (see [9, 2, 11] and the references therein).

Principle 8: On partial information the metric returns a meaningful result: an amount for which the name-to-key binding is insured (though it might be insured for more). If an entity's responsibility for a certificate it creates extends beyond any premature revocation of that certificate, then even unknown certificate revocations pose no threat to the insured value of a name-to-key binding. Otherwise, computing a minimum capacity cut with every edge capacity set to one outputs the number of certificates that would need to be revoked to leave the binding uninsured.

The metric in this section seems to overcome some limitations of other metrics, in part because its outputs can have direct relevance for a range of business transactions, and in part because it places responsibility on each certifier to assess and assume risk for the certificates it creates. Ultimately what will determine the economic viability of such a metric are market forces, and in particular, how much users are willing to pay for insurance for name-to-key bindings. Given the minimal amount of commerce presently transacted using public key technology, it may be too soon to tell. We nevertheless believe it useful to anticipate the range of economic metrics of authentication, of which the aforementioned is but one example.

5 Conclusion

In this paper we put forth a collection of principles that we believe to be useful for designing metrics for authentication. We argued their utility by demonstrating potentially negative effects that proposed metrics suffer by not following them. Finally, we offered one direction for constructing metrics that, we believe, come closer to satisfying our principles.

It is our hope that this work will initiate a broader discussion in the scientific community regarding metrics of authentication. Since only a handful of metrics have been proposed, it is likely that other design principles will arise as this area grows in visibility. Further attention to design principles must be paid before metrics are standardized or adopted for wide-scale use.

This work leaves a number of open problems. Of course, identifying limitations of our principles, or new principles, is a direction that deserves attention. An intriguing challenge is to fully develop and deploy a metric that meets our principles, perhaps one along the lines proposed in Section 4.

Acknowledgements

We are very grateful to Malte Borcherting, Raph Levien, and Ueli Maurer for insightful comments and clarifications. We also thank the anonymous referees for their suggestions.

References

- [1] M. Abadi and R. Needham. Prudent engineering practice for cryptographic protocols. *IEEE Transactions on Software Engineering* 22(1):6–15, January 1996.
- [2] R. K. Ahuja, J. B. Orlin, and R. E. Tarjan. Improved time bounds for the maximum flow problem. *SIAM Journal of Computing* 18:939–954, 1989.
- [3] R. Anderson and R. Needham. Robustness principles for public key protocols. In D. Coppersmith, ed., *Advances in Cryptology — CRYPTO'95* (Lecture Notes in Computer Science 963), pages 236–247, Springer-Verlag, 1995.
- [4] T. Beth, M. Borcherting, and B. Klein. Valuation of trust in open networks. In D. Gollman, ed., *Computer Security — ESORICS '94* (Lecture Notes in Computer Science 875), pages 3–18, Springer-Verlag, 1994.
- [5] A. D. Birrell, B. W. Lampson, R. M. Needham and M. D. Schroeder. A global authentication service without global trust. In *Proceedings of the 1986 IEEE Symposium on Security and Privacy*, pages 223–230, April 1986.
- [6] L. R. Ford, Jr. and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics* 8:399–404, 1956.
- [7] M. Gasser, A. Goldstein, C. Kaufman and B. Lampson. The Digital distributed system security architecture. In *Proceedings of the 12th NIST/NCSC National Computer Security Conference*, pages 305–319, October 1989.
- [8] V. D. Gligor, S. Luan, and J. N. Pato. On inter-realm authentication in large distributed systems. In *Proceedings of the 1992 IEEE Symposium on Research in Security and Privacy*, pages 2–17, May 1992.
- [9] A. V. Goldberg and R. E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM* 35:921–940, 1988.
- [10] S. Kent. Internet privacy enhanced mail. *Communications of the ACM* 36(8):48–60, August 1993.
- [11] V. King, S. Rao, and R. Tarjan. A faster deterministic maximum flow algorithm. In *Proceedings of the 3rd ACM Symposium on Discrete Algorithms*, pages 157–164, 1992.
- [12] B. Lampson, M. Abadi, M. Burrows and E. Wobber. Authentication in distributed systems: Theory and practice. *ACM Transactions on Computer Systems* 10(4):265–310, November 1992.
- [13] C. Lai, G. Medvinsky, and B. C. Neuman. Endorsements, licensing, and insurance for distributed system services. In *Proceedings of the 2nd ACM Conference on Computer and Communications Security*, pages 170–175, November 1994.
- [14] U. Maurer. Modeling a public-key infrastructure. In E. Bertino, H. Kurth, G. Martella, and E. Montolivo, ed., *Computer Security — ESORICS '96* (Lecture Notes in Computer Science 1146), Springer-Verlag, 1996.
- [15] S. Mendes and C. Huitema. A new approach to the X.509 framework: Allowing a global authentication infrastructure without a global trust model. In *Proceedings of the 1995 Internet Society Symposium on Network and Distributed System Security*, February 1995.
- [16] M. K. Reiter and S. G. Stubblebine. Path independence for authentication in large-scale systems. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, April 1997.
- [17] W. Stallings. *Protect Your Privacy, A Guide for PGP Users*. Prentice-Hall, 1995.
- [18] P. Syverson. Limitations on design principles for public key protocols. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy*, pages 62–72, May 1996.
- [19] J. J. Tardo and K. Alagappan. SPX: Global authentication using public key certificates. In *Proceedings of the 1991 IEEE Symposium on Research in Security and Privacy*, pages 232–244, May 1991.
- [20] A. Tarah and C. Huitema. Associating metrics to certification paths. In *Computer Security — ESORICS '92* (Lecture Notes in Computer Science 648), pages 175–189, Springer Verlag, 1992.
- [21] International Telegraph and Telephone Consultative Committee (CCITT). The Directory – Authentication Framework, Recommendation X.509, 1988.
- [22] ANSI X9F1, ANSI X9.45 Enhanced Management Controls Using Attribute Certificates, 1994 (draft).
- [23] R. Yahalom, B. Klein and T. Beth. Trust-based navigation in distributed systems. *Computing Systems* 7(1):45–73, 1994.
- [24] P. Zimmermann. *PGP User's Guide*, Volumes I and II, October 1994. Included in the PGP 2.6.2 distribution.