Using probabilistic argumentation for key validation in public-key cryptography

Rolf Haenni *

Center for Junior Research Fellows, University of Konstanz, Konstanz D-78457, Germany

Received 1 February 2004; received in revised form 1 April 2004; accepted 1 May 2004
Available online 17 September 2004

Abstract

The purpose of this paper is to show how probabilistic argumentation is applicable to modern public-key cryptography as an appropriate tool to evaluate webs of trust. This is an interesting application of uncertain reasoning that has not yet received much attention in the corresponding literature.

© 2004 Elsevier Inc. All rights reserved.

Keywords: Probabilistic argumentation; Web of trust; Public-key infrastructure; Cryptography

1. Introduction

In large open networks like the internet an increasing demand for security is observed. In order to establish a confidential channel between two users of the network, classical single-key cryptography requires them to exchange a common secret key over a secure channel. This may work if the network is small and local, but it is infeasible in non-local or large networks. To simplify the key exchange problem, modern public-key cryptography provides a mechanism in which the keys to be exchanged are not secret. In such a framework, every user possesses a key pair consisting of...
a (non-secret) public key and a (secret) private key. Only public keys are exchanged. They are used to encrypt messages to be sent to the owner of the key or to verify digital signatures issued by the owner of the key.

Before using someone else’s public key to encrypt a message or verify a signature, one should make sure that the key really belongs to the intended recipient or the indicated issuer of the signature. Achieving authenticity of public keys can be done in several ways. The most popular approach is based on the concept of digital certificates. The idea is that different users or entities of a network certify public keys of other network users. This leads to a certificate graph. Of course, certificates should only be issued if the key’s authenticity is verified. On the basis of a certificate graph, one can then evaluate the authenticity of the keys on the basis of how much trust one assigns to the different issuers of the certificates. Because such an evaluation depends on trust, it is common to call such a certificate graph web of trust. Section 2 gives a short introduction to public-key cryptography and webs of trust. For more information we refer to the literature [14,23].

PGP (Pretty Good Privacy) is a popular and widely used implementation of public-key cryptography for email security [28]. It organizes public keys on the basis of a web of trust [22]. PGP’s way of evaluating the web of trust is a simple mechanism based on three pragmatic rules. Some authors have tried to formalize the concepts of trust and confidence more properly [1–3,17,24,25], but approaches to look at the problem from the perspective of the uncertain reasoning or AI community are rare. One exception is the idea of applying Dempster–Shafer theory in the context of a distributed reputation management [26]. Another exception is Maurer’s approach [13] on the basis of probabilistic logic [6,15]. An analysis and overview of different authentication metrics is given in [18].

Among the various formalisms for uncertain reasoning, probabilistic argumentation [10] seems to be one of the most promising candidates. Ordinary Bayesian networks, for example, fail as a possible candidate because they require the underlying graphs to be acyclic [16] (whereas general certificate graphs are cyclic). The basic concepts of probabilistic argumentation are summarized in Section 3. As Section 4 demonstrates, by modeling trust as the probability of somebody’s reliability, translating a web of trust into a corresponding probabilistic argumentation system is straightforward. And it leads to a one-to-one correspondence between the concepts of certificate chains and arguments. We will present an efficient algorithm to compute such arguments. Degree of support, which is the probability that at least one argument holds, or the probability of provability that the authenticity of a key is logically inferable from the given web of trust, can then be used to measure quantitatively the overall reliability of all possible certificate chains and thereby to rate the validity of the corresponding public key.

The approach we propose in this paper is in many ways analogue to Maurer’s trust model described in [13], but in addition we will present a concise algorithm and show how to properly include key revocations. We also think that the framework of probabilistic argumentation, in comparison with probabilistic logic, is a more intuitive approach to uncertain reasoning and, in terms of available computational techniques, more advanced.
In addition to contributing a new method to the field of distributed key management, the goal of this paper is twofold. First, it is supposed to increase the awareness of people interested in reasoning under uncertainty for this interesting application in public-key cryptography. Second, the paper intends to demonstrate how to use probabilistic argumentation in real world applications and to underline the value of this elegant formalism.

2. Public-key cryptography and the web of trust

Modern cryptography consists of two major tasks: encryption and signing. To transmit a message $m$ securely from sender $A$ to recipient $B$, both sender and recipient have to be equipped with a corresponding pair of public and private keys. Private keys are kept secret, whereas public keys are widely available for any recipient. From $A$’s perspective, sending $m$ over an insecure channel (e.g. the internet) to recipient $B$ requires $A$ to encrypt the message with $B$’s public key and to digitally sign it with $A$’s own private key. On the side of recipient $B$, the message is decrypted with $B$’s private key and the digital signature is verified with $A$’s public key. Provided that $A$ and $B$ have properly exchanged their public keys, this simple scheme realizes the main security goals (secrecy, message integrity, authenticity, non-repudiation) for such a two-party communication (see Fig. 1).

Public keys are usually distributed with the aid of key servers. Before sending encrypted messages to recipients $B$, $A$ may copy $B$’s public key from a key server. On the other side, $B$ may copy $A$’s public key from a key server in order to verify $A$’s digital signatures. The question is whether the keys copied from the key servers are really owned by $B$ and $A$, respectively. A possible attacker or opponent $O$ could easily generate key pairs and post the corresponding public keys in $A$’s or $B$’s name onto the server. Encrypting a message falsely to $O$’s public key enables $O$ to decrypt and read the message (and at the same time disables $B$ to decrypt the message). Similarly, verifying a digital signature with false public key enables the attacker $O$ to sign messages in the name of $A$. An important issue is thus the verification of public keys before using them. One way to verify a public key is to compare its nearly unique fingerprint (a hash code of fixed length) over a secure channel (e.g. the telephone line). This method may work in small or local networks with few users, but is impractical in large networks like the internet.

![Fig. 1. Encrypted and signed two-party communication over an insecure channel.](image-url)
The most practical way to solve the public key exchange problem is to use digital certificates. A digital certificate can be seen as a digitally signed public key. For example, to issue a certificate for $A$, the issuer $C$ digitally signs $A$'s public key with $C$'s own private key. By doing this, $C$ certifies that $A$ is the true owner of the key. Of course, certificates should only be issued when the public key was either obtained or successfully verified over a secure channel. Note that a digital certificate may consist of signatures from different issuers.

If $A$ receives $B$'s certificate issued by $C$, then $A$ has good reasons to accept the corresponding public key as $B$'s true public key, whenever the following three conditions are satisfied:

1. $A$ fully trusts $C$ to always carefully verify public keys before issuing certificates,
2. $A$ has received or verified $C$'s public key over a secure channel,
3. $A$ has successfully verified the certificate using $C$'s public key.

A collection of digital certificates is called public-key infrastructure (PKI). In practice, there are two approaches to build PKIs.

### 2.1. Certificate authorities

The first approach requires the certificates to be issued by trustworthy certificate authorities (CA). For example, if $C$ is a trustworthy CA (i.e. before issuing a certificate, $C$ carefully checks if the applicant is the true owner of the public key), then the users of a large network may exchange their public keys by exchanging respective certificates issued by $C$. Certificates issued by $C$ can be verified using $C$'s public key. From the successful verification follows then the authenticity of the corresponding public key.

If more than one CA issues certificates, it is possible that the different CAs mutually issue certificates to each other. This leads to certificate trees which are usually organized hierarchically. Fig. 2 shows such a tree in which network users are represented by circles and CAs by squares. An arrow from entity $X$ to entity $Y$ (users or CAs) represents $X$'s certificate issued by $Y$. The formal notation for such a certificate will be $X \Rightarrow Y$.

If $A$ has an authentic copy of $Aut_1$'s public key, then the authenticity of $M$'s certificate $M \Rightarrow Aut_3$ can be verified using $Aut_3$'s certificate $Aut_3 \Rightarrow Aut_2$ and $Aut_2$'s cer-

![Fig. 2. Example of an undirected certificate tree with certificate authorities.](image-url)
certificate $Aut_2 \Rightarrow Aut_1$. Such a certificate chain $M \Rightarrow Aut_3 \Rightarrow Aut_2 \Rightarrow Aut_1 \Rightarrow A$ requires $A$ to fully and unconditionally trust all CAs along the path between $M$ and $A$. If any CA in the path has incorrectly issued the certificate of the next CA, then $A$ can be misled regarding the authenticity of $M$'s certificate. Note that there is a unique certificate chain between any two users attached to such a certificate tree.

The major advantage of a centralized PKI is that every user is required to employ only one secure channel in order to get an authentic copy of its own CA’s public key. The major disadvantage is the requirement of unconditional trust in all CAs involved.

### 2.2. Web of trust

The second approach does not require certificate authorities. The idea is that every user in the network can issue certificates. This leads to certificate graphs rather than trees. In such a decentralized context, one usually speaks about signing public keys rather than issuing certificates. Thus, every user collects signed public keys from different keys servers or other sources. A personal collection of signed public keys is called key ring. Note that every individual key ring defines a corresponding certificate graph (which is a sub-graph of the complete certificate graph of all signed public keys).

Fig. 3 shows the certificate graph that corresponds to $A$’s key ring. An arrow from user $X$ to user $Y$ means that $Y$ has signed $X$’s public key. Question marks represent users whose public keys are unknown to $A$. In the example shown in Fig. 3, $A$ has directly signed the public keys of $B$, $C$, $D$, $E$, and $F$. This means that $A$ has received or verified these keys over a secure channel and accepts them as the authentic keys of $B$, $C$, $D$, $E$, and $F$, respectively. In other words, the public keys of $B$, $C$, $D$, $E$, and $F$ are valid for $A$. Many other keys in the graph are signed by users different from $A$. User $G$, for example, has signed the keys of $L$ and $M$. From $A$’s perspective, $G$ is called introducer of $L$’s and $M$’s certificate.

Fig. 3. Example of a decentralized certificate graph.
In order to indirectly validate someone else's public key, \( A \) must have full confidence in all introducers along the path of at least one certificate chain. This means that \( A \) must consider the corresponding introducers to be fully trustworthy in the sense that they only issue certificates for public keys received or verified over secure channels. In the certificate graph depicted in Fig. 3, there is only one certificate chain \( L \Rightarrow G \Rightarrow B \Rightarrow A \) from \( L \) to \( A \). Thus, in order to validate \( L \)'s public key, \( A \) has to trust both \( G \) and \( B \). On the other hand, there are two certificate chains \( M \Rightarrow G \Rightarrow B \Rightarrow A \) and \( M \Rightarrow H \Rightarrow C \Rightarrow A \) from \( M \) to \( A \). In order to validate \( M \)'s public key, \( A \) must trust either \( G \) and \( B \) or \( H \) and \( C \).

A certificate graph in which the validity of the public keys is evaluated on the basis of trust is called web of trust. Note that hierarchical certificate trees with fully trustworthy CAs are particular webs of trust in which all introducers receive maximal trust.

A general web of trust allows the owner of the key ring to specify gradual levels of trust for all individuals involved in the web. Completely trustworthy and untrustworthy are the two extreme cases of maximal and minimal trust, respectively. Evaluating the validity of the public keys should then lead to gradual levels of validity. Full validity, for example, only results from full trust along the path of at least one certificate chain (such as in the hierarchical case with CAs). The evaluation of such a general web of trust on the basis of probabilistic argumentation systems is the contribution of this paper.

### 2.3. PGP's web of trust

PGP is one of the most popular tools for public-key cryptography. The software can be used to encrypt and digitally sign electronic mail. It is based on a web of trust with some particular characteristics. First of all, PGP allows (only) three levels of trust: completely trustworthy, marginally trustworthy, and untrustworthy. Note that the owner of the key ring automatically receives full trust. In order to rate a public key as “valid”, PGP either requires

(a) the key to belong to the owner of the key ring,
(b) a signature from at least one \(^1\) completely trustworthy introducer with a “valid” public key,
(c) signatures from at least two \(^2\) marginally trustworthy introducers with “valid” public keys.

Otherwise, the key is rated as “invalid”. \(^3\) Note that all public keys directly signed by the owner of the key ring are “valid”. An example to illustrate PGP’s trust model is

---

\(^1\) One is the default value, but a different (higher) value may be chosen by the user.
\(^2\) Two is the default value, but a different (higher) value may be chosen by the user.
\(^3\) PGP also defines “marginally valid” public keys, but they are considered as “invalid” by default.
shown in Fig. 4. Gray circles stand for completely trustworthy, gray semicircles for marginally trustworthy, and white circles for untrustworthy introducers. “Valid” public keys are indicated by nested circles.

A’s public key is “valid” because it is owned by A. The public keys of B, C, D, E, and F are all “valid” because they are directly signed by A. H and I are “valid” because C is “valid” and completely trustworthy. J is “valid” because E is “valid” and completely trustworthy. K is “valid” because J is “valid” and completely trustworthy. N is “valid” because both H and I are “valid” and marginally trustworthy. Finally, P is “valid” because J is “valid” and completely trustworthy. All other keys are “invalid”.

The PGP trust model is unsatisfactory in many ways. First of all, although trust is a gradual quantity that reflects someone’s confidence in someone else’s reliability, PGP provides only three levels of trust. Similarly, by simply distinguishing between “valid” and “invalid” public keys, PGP is not able to gradually rate the authenticity of the keys. Another problem is the rule that keys signed by at least two marginally
trustworthy introducers are rated as “valid”. This rule seems to be the product of a pragmatic way of evaluating webs of trust, but it is certainly not the result of a proper and well-founded trust model. In fact, one can easily construct counterintuitive examples such as the ones shown in Fig. 5.

On the left-hand side of Fig. 5, PGP’s trust model rates B’s public key as “valid”, whereas on the right-hand side, it is rated as “invalid”. However, because there is any desired number of possible certificate chains in the web of trust on the right-hand side, each chain including one marginally trustworthy and one completely trustworthy introducer, one would expect to rate the validity of B’s key with a much higher degree than in the web of trust shown on the left-hand side with only two possible certificate chains.

3. Probabilistic argumentation

Probabilistic argumentation is a relatively new formal theory of automated reasoning [10]. The goal is to judge hypotheses in the light of the given uncertain and partial knowledge. Hypotheses represent open questions about the unknown or future world. The ingredients of probabilistic argumentation are a formal language $\mathcal{L}_V$ over a set of variables $V$ and a fully specified probability distribution $P(A)$ over a subset of variables $A \subseteq V$. It turns out that the two classical approaches of automated reasoning, that is logical reasoning on one hand and probabilistic reasoning on the other hand, are nothing but opposite extreme cases of this unifying theory, namely for $A = \emptyset$ and $A = V$, respectively. And in fact, the concepts of provability in logic and probability in probability theory are replaced by a more general concept of probability of provability (see Section 3.2). In this paper, we will only consider the simplest case of a possible formal language, namely the language of propositional logic.

From a qualitative point of view, the problem is to derive arguments in favor and counter-arguments against the hypothesis $h$ of interest. An argument is a defeasible proof built on uncertain assumptions. In other words, arguments are combinations of true or false assumptions that permit to infer logically the truth of the hypothesis $h$ from the given knowledge base. Every argument provides thus a sufficient reason that proves the hypothesis in the light of the available knowledge. And it finally contributes to the possibility of believing or accepting the hypothesis. In other words, arguments support and counter-arguments defeat the hypothesis $h$. Notice that counter-arguments can be regarded as arguments in favor of the negated hypothesis $\neg h$ and vice versa.

A quantitative judgement of the situation is obtained by considering the probabilities that the arguments and counter-arguments are valid. The credibility of a hypothesis is measured by the total probabilities that it is supported or defeated by arguments. Conflicts are handled through conditioning. The resulting sub-additive degree of support and super-additive degree of possibility are probabilities of provability and correspond to belief and plausibility, respectively, in the Dempster–Shafer theory of evidence [19,21]. This connection will be further commented in Section 3.2.
For the construction of probabilistic argumentation systems in the context of a propositional language, consider two disjoint sets $A$ and $P$ of propositions. The elements of $A$ are called assumptions and represent uncertain events, unknown circumstances, or possible states or outcomes. $\mathcal{L}_{A\cup P}$ denotes the propositional language over the total set $V = A \cup P$ of available propositions. The given uncertain knowledge base is then encoded by a set of sentences $\Sigma \in \mathcal{L}_{A\cup P}$. This set is conjunctively interpreted. If all sentences of $\Sigma$ are clauses, then it is possible to interpret $\Sigma$ as a conjunctive normal form $\xi_1 \land \cdots \land \xi_r$.

3.1. Arguments, counter-arguments, conflicts

Consider the case where another propositional sentence $h \in \mathcal{L}_{A\cup P}$ represents a hypothesis about some of the propositions in $A \cup P$. What can be inferred from $\Sigma$ about the possible truth of $h$ with respect to the given set of uncertain assumptions? Possibly, if some of the assumptions are set to true and others to false, then $h$ may be a logical consequence of $\Sigma$.

More formally, if $T_A$ denotes the set of all consistent conjunctions of non-repeating literals over $A$, then such a term $a \in T_A$ is an argument for $h$, if and only if

$$f(a, \Sigma) \models h.$$  

Similarly, if $\{a\} \cup \Sigma \models \neg h$, then $a$ is a counter-argument against $h$. As stated above, counter-arguments are arguments for $\neg h$. An argument $a$ for $h$ is called minimal, if there is no shorter argument $a' \subset a$ for $h$ which subsumes $a$. The sets of all minimal arguments and minimal counter-argument with respect to $h$ and $\Sigma$ are denoted by $\text{Args}(h, \Sigma)$ and $\text{Args}(\neg h, \Sigma)$, respectively. Note that every $a \in \text{Args}(h, \Sigma)$ increases the support for $h$, whereas every $a \in \text{Args}(\neg h, \Sigma)$ decreases the possibility of $h$.

If a term $a \in T_A$ is both argument and counter-argument of $h$, then it is called conflict. Conflicts are inconsistent with the knowledge base $\Sigma$. They represent impossible states of the world which have be excluded. Note that conflicts are arguments for $\bot$. The set of all minimal conflicts is denoted by $\text{Args}(\bot, \Sigma)$.

As an example, consider two sets of propositions $A = \{a_1, a_2, a_3\}$ and $P = \{X, Y\}$ and a knowledge base $\Sigma$ given as a set

$$\Sigma = \{a_1 \rightarrow X, \neg a_2 \rightarrow Y, a_3 \land Y \rightarrow X, a_2 \rightarrow \neg X\}$$

of material implications. If $X$ is the hypothesis of interest, then there are two minimal arguments for $X$, one minimal counter-argument against $X$, and one minimal conflict:

$$\text{Args}(X, \Sigma) = \{a_1, \neg a_2 \land a_3\},$$
$$\text{Args}(\neg X, \Sigma) = \{a_2\},$$
$$\text{Args}(\bot, \Sigma) = \{a_1 \land a_2\}.$$  

Computing the sets $\text{Args}(h, \Sigma)$, $\text{Args}(\neg h, \Sigma)$, and $\text{Args}(\bot, \Sigma)$ is the main computational problem of probabilistic argumentation [10]. Efficient approximation algorithms are obtained by focusing the search on the most relevant arguments [7]. It is also possible to define convenient anytime algorithms which, upon interruption,
return the solution found so far \[8,9\]. The quality of the approximation increases monotonically when more computational resources are available. The method is based on cost functions and returns lower and upper bounds. In Section 4.3, we will present a simple special purpose algorithm designed for the specific problem addressed in this paper.

3.2. Degrees of support and possibility

In order to judge \( h \) quantitatively, let all assumptions \( a \in A \) be linked to corresponding prior probabilities \( p_a = P(a) \). We suppose them to be mutually independent. By doing so, we get a fully specified joint probability distribution \( P(A) \) over the space \( \{0, 1\}^{|A|} \) of all possible configurations of true and false assumptions. Notice that, from a conceptual point of view, any other way of specifying the joint distribution \( P(A) \), for example with the aid of a Bayesian network or by an explicit list of values, would be fine too. This means that the specification of arbitrary dependencies between the elements of \( A \) is possible. For example, the case of two perfectly correlated assumptions \( a_1, a_2 \in A \) means that \( P(x) = 0 \) for all configurations \( x \in \{0, 1\}^{|A|} \) for which \( a_1 \) is true and \( a_2 \) is false, or vice versa.

In the case of a given set of independent probabilities \( p_a \), the probability of a term \( x \in \mathcal{F}_A \) is determined by

\[
P(x) = \prod \{p_a : a \in x\} \cdot \prod \{1 - p_a : \neg a \in x\}.
\]

If \( T \subseteq \mathcal{F}_A \) is an arbitrary set of terms, then \( P(T) \) denotes the overall probability of all terms included in \( T \). It corresponds to the probability that at least one term of \( T \) is true. Note that any such set \( T = \{x_1, \ldots, x_n\} \) can be interpreted as a disjunctive normal form \( x_1 \lor \cdots \lor x_n \) (DNF for short). The problem of computing \( P(T) \) is thus equivalent to the general problem of computing probabilities of DNFs, for which the so-called inclusion–exclusion formula provides a mathematically sound but very inefficient solution. More sophisticated methods have been developed in the domains of reliability theory (under the name of sum-of-disjoint-products algorithms, e.g. see [12]) and knowledge compilation [5]. For further information on this we refer to the corresponding literature, in particular to Darwiche’s d-DNNF compiler [4], which we consider the state-of-the-art in the field.

Consider now the conditional probability that at least one argument for \( h \) is true under the condition that none of the conflicts of \( \Sigma \) is true. This is a quantitative measure of how much \( h \) is supported by arguments in the light of the given knowledge. It depends on the two sets \( \text{Args}(h, \Sigma) \) and \( \text{Args}(\perp, \Sigma) \). If the set \( \text{Args}(\perp, \Sigma) \) is considered as DNF, then \( \neg \text{Args}(\perp, \Sigma) \) represents the condition that conflicts are impossible. This allows us to define degree of support of \( h \) as

\[
dsp(h, \Sigma) = \frac{P(\text{Args}(h, \Sigma) \mid \neg \text{Args}(\perp, \Sigma))}{1 - P(\text{Args}(\perp, \Sigma))}.
\]
For a more detailed derivation of the above formula we refer to [10]. Degrees of support are probabilities of provability [16, 20]. They form a non-monotone and non-additive measure that allows a proper distinction between uncertainty and ignorance. Note that this concept is equivalent to the notion of (normalized) belief in the Dempster–Shafer theory of evidence [19, 21]. In other words, if all propositional sentences in the knowledge base \( \Sigma \) as well as all prior probabilities \( p_a \) are transformed into corresponding Dempster–Shafer mass functions over local domains (see [11] for details on this), and these are all combined by Dempster’s rule, then the resulting joint belief function \( Bel(H) \) is identical to \( dsp(h) \) as defined above.

A second way of judging the hypothesis \( h \) is to look at the conditional probability that no counter-argument is true under the condition that none of the conflicts of \( \Sigma \) is true. This is a quantitative measure of how possible \( h \) is in the light of the given knowledge. Thus, degree of possibility of \( h \) is defined as

\[
dps(h, \Sigma) = P(\neg \text{Args}(\neg h, \Sigma) \mid \neg \text{Args}(\bot, \Sigma)) = 1 - dsp(\neg h, \Sigma).
\]  

Degree of possibility is equivalent to the notion of plausibility in the Dempster–Shafer theory. Note that \( dsp(h, \Sigma) \leq dps(h, \Sigma) \) for all \( h \in \mathcal{L}_{A,P} \) and \( \Sigma \subseteq \mathcal{L}_{A,P} \). In general, it is possible to interpret \( dps(h, \Sigma) - dsp(h, \Sigma) \) as the degree of ignorance involved in the judgment of \( h \) with respect to \( \Sigma \). The particular case of \( dps(h, \Sigma) = 0 \) and \( dps(h, \Sigma) = 1 \) represents total ignorance over \( h \). On the other hand, \( dsp(h, \Sigma) = dps(h, \Sigma) \) means that we have a case of classical probabilistic reasoning.

Consider the example at the end of the previous subsection and suppose that \( p_{a_1} = 0.2 \), \( p_{a_2} = 0.4 \), and \( p_{a_3} = 0.1 \) are the probabilities of the assumptions. The probabilities of the DNFs formed by the respective sets of arguments, counter-arguments, and conflicts are then as follows:

\[
P(\text{Args}(X, \Sigma)) = P(a_1 \lor \neg a_2 \land a_3) = P(a_1 \lor \neg a_1 \land \neg a_2 \land a_3) = 0.2 + 0.8 \cdot 0.6 \cdot 0.1 = 0.248,
\]

\[
P(\text{Args}(\neg X, \Sigma)) = P(a_2) = 0.4,
\]

\[
P(\text{Args}(\bot, \Sigma)) = P(a_1 \land a_2) = 0.2 \cdot 0.4 = 0.08.
\]

Finally, according to (3) and (4), degree of support and degree of possibility are computed as follows:

\[
dsp(X, \Sigma) = \frac{0.248 - 0.08}{1 - 0.08} = 0.183,
\]

\[
dsp(X, \Sigma) = 1 - \frac{0.4 - 0.08}{1 - 0.08} = 0.652.
\]

Although there is only a weak support, the hypothesis \( X \) remains quite possible. This is an example where gathering more information should precede any rash decision for or against \( X \).
4. Trust evaluation based on probabilistic argumentation

We will now see how to encode a web of trust as a probabilistic argumentation system. Because trust can be seen as someone’s confidence in someone else’s reliability, we denote the reliability of an introducer $X$ by the proposition $\text{rel}(X)$. Gradual confidence in $X$ can then be quantified by the subjective prior probability $P(\text{rel}(X))$ of $X$ being a reliable introducer. The special case where $X$ is a fully trustworthy CA is encoded by $P(\text{rel}(X)) = 1$. On the other hand, $P(\text{rel}(X)) = 0$ stands for a case in which $X$ deserves no trust at all.

In a similar way, we use the proposition $\text{Val}(X)$ to represent the case where $X$’s public key is valid. Note that there is usually no prior knowledge about how certain $\text{Val}(X)$ is. It is therefore not possible to specify corresponding prior probabilities.

If $\mathcal{U} = \{X_0, X_1, \ldots, X_n\}$ is the set of all users included in the key ring owned by $X_0$, then

$$A = \{\text{rel}(X_0), \text{rel}(X_1), \ldots, \text{rel}(X_n)\},$$

$$P = \{\text{Val}(X_0), \text{Val}(X_1), \ldots, \text{Val}(X_n)\},$$

are the two sets of propositions needed to build a corresponding probabilistic argumentation system. Note that the probabilities $P(\text{rel}(X_1))$ to $P(\text{rel}(X_n))$ are specified by $X_0$, whereas $X_0$ is implicitly assumed to be fully reliable by default and thus $P(\text{rel}(X_0)) = 1$. Similarly, because $X_0$’s own public key is implicitly valid, $\text{Val}(X_0)$ is true by default. In Section 4.4, we will discuss the case in which trust is not directly specified by $X_0$, but indirectly with the aid of so-called recommendations.

In order to formulate $X_0$’s certificate graph as an assumption-based knowledge base $\Sigma$, consider the set $\mathcal{C}$ of all certificates contained in $X_0$’s key ring, except those that are issued by unknown users. A single certificate $c \in \mathcal{C}$ of the form $X_i \Rightarrow X_j$ translates then into the following propositional sentence:

$$\gamma(c) = \gamma(X_i \Rightarrow X_j) = \text{rel}(X_j) \land \text{Val}(X_j) \rightarrow \text{Val}(X_i).$$

The idea of this translation is to consider $X_i$’s public key as valid whenever $X_j$ is a reliable introducer with a valid public key. Note that this corresponds to Rule (b) in PGP’s trust model. In Maurer’s model [13, Definition 3.2], it is equivalent to the first inference rule, except that here the certificate itself is not explicitly linked to a corresponding proposition. The complete knowledge base $\Sigma$ is then the following set $\Sigma$ of propositional sentences:

$$\Sigma = \{\text{Val}(X_0)\} \cup \{\gamma(c) : c \in \mathcal{C}\}.$$
responding computational problems are restricted to sets of minimal arguments
responding hypotheses
Val
Xi
for all
interest. Every minimal argument
mal) certificate chain from
literals, there will be no counter-arguments against
Val
Args
implicitly true and therefore a part of the knowledge base, we have
4.1. Qualitative evaluation
Section 4.3).
This will considerably reduce the complexity of corresponding computations (see
Horn clauses
This is the complete knowledge base on which the evaluation of the public keys
will be based. Note that such a knowledge base consists of Horn clauses only.
This will considerably reduce the complexity of corresponding computations (see
Section 4.3).
4.1. Qualitative evaluation
How can such a knowledge base be used to evaluate the validity of the involved
public keys? First of all, if it is
Xi
’s key to be rated, then
Val(Xi)
is the hypothesis of
interest. Every minimal argument
Arg
(Val(Xi), \Sigma)
corresponds then to a (minimal) certificate chain from
Xi
to
X0.
Furthermore, because \Sigma contains only positive literals, there will be no counter-arguments against
Val(Xi). This implies
\text{Args}(\neg \text{Val}(X_i), \Sigma) = \emptyset, \quad (9)
\text{Args}(\bot, \Sigma) = \emptyset \quad (10)
for all
Xi \in \mathcal{H}.
Finally, this means that the evaluation of the public keys and the corresponding computational problems are restricted to sets of minimal arguments
Arg
(Val(X_i), \Sigma)
for some
Xi \in \mathcal{H}.
Note that, because the proposition
Val(X_0)
is implicitly true and therefore a part of the knowledge base, we have
Arg
(Val(X_0), \Sigma) = \{ \top \}, where \top denotes the argument that always holds.
Consider the example from above and suppose we are interested in the public keys
of
J
and
P.
In both cases, there are three minimal arguments supporting the corresponding hypotheses
Val(J)
and
Val(P), respectively:
The first minimal argument \( \text{rel}(A) \land \text{rel}(D) \land \text{rel}(J) \) for \( \text{Val}(P) \), for example, corresponds to the certificate chain \( P \Rightarrow J \Rightarrow D \Rightarrow A \). Non-minimal certificate chains such as \( P \Rightarrow J \Rightarrow E \Rightarrow F \Rightarrow A \) correspond to non-minimal arguments and are thus not included in the above sets. Note that every minimal argument for \( \text{Val}(P) \) is obtained from a corresponding minimal argument for \( \text{Val}(J) \) by conjoining it with \( \text{rel}(J) \). This will be the key observation for a procedure that computes the minimal arguments for all public keys included in the key ring (see Section 4.3).

The following table shows the minimal arguments for all public keys. To get a more compact representation of these sets, we write \( A \land B \), for example, instead of \( \text{rel}(A) \land \text{rel}(B) \).

<table>
<thead>
<tr>
<th>( X_i )</th>
<th>( \text{Args}(\text{Val}(X_i), \Sigma) )</th>
<th>( X_i )</th>
<th>( \text{Args}(\text{Val}(X_i), \Sigma) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( A \ptions_2 )</td>
<td>( \top )</td>
<td>( J )</td>
<td>( A \land D, A \land E, A \land F \land K )</td>
</tr>
<tr>
<td>( B )</td>
<td>( A )</td>
<td>( K )</td>
<td>( A \land D \land J, A \land E \land J, A \land F )</td>
</tr>
<tr>
<td>( C )</td>
<td>( A )</td>
<td>( L )</td>
<td>( A \land B \land G )</td>
</tr>
<tr>
<td>( D )</td>
<td>( A )</td>
<td>( M )</td>
<td>( A \land B \land G, A \land C \land H )</td>
</tr>
<tr>
<td>( E )</td>
<td>( A )</td>
<td>( N )</td>
<td>( A \land C \land H, A \land C \land I, A \land D \land I )</td>
</tr>
<tr>
<td>( F )</td>
<td>( A )</td>
<td>( O )</td>
<td>( A \land C \land I, A \land D \land I )</td>
</tr>
<tr>
<td>( G )</td>
<td>( A \land B )</td>
<td>( P )</td>
<td>( A \land D \land J, A \land E \land J, A \land F \land K \land J )</td>
</tr>
<tr>
<td>( H )</td>
<td>( A \land C )</td>
<td>( Q )</td>
<td>( A \land D \land J \land K, A \land E \land J \land K, A \land F \land K )</td>
</tr>
<tr>
<td>( I )</td>
<td>( A \land C, A \land D )</td>
<td>( R )</td>
<td>( – )</td>
</tr>
</tbody>
</table>

Note that there is a one-to-one correspondence between the concept of minimal arguments and Maurer’s minimal subsets \( \forall \subseteq \text{View}_{X_i} \) in his model based on probabilistic logic [13].

### 4.2. Quantitative evaluation

From (9) follows that \( \text{dps}(\text{Val}(X_i), \Sigma) = 1 \) for all users \( X_i \in \mathcal{U} \). This means that degree of support is the only relevant quantity to rate the validity of \( X_i \)'s public key. It corresponds to the probability that the introducers of at least one certificate chain are all reliable. Note that (10) implies \( P(\text{Args}(\bot, \Sigma)) = 0 \). As a consequence, we get

\[
\text{dsp}(\text{Val}(X_i), \Sigma) = P(\text{Args}(\text{Val}(X_i), \Sigma))
\]

for all \( X_i \in \mathcal{U} \). Suppose now that \( A \) has specified the reliability of the introducers \( B \) to \( R \) according to the following table (second row). By default, we have

\[
\text{Args}(\text{Val}(P), \Sigma) = \begin{cases} \text{rel}(A) \land \text{rel}(D), \\ \text{rel}(A) \land \text{rel}(E), \\ \text{rel}(A) \land \text{rel}(F) \land \text{rel}(K) \end{cases}
\]

\[
\text{Args}(\text{Val}(P), \Sigma) = \begin{cases} \text{rel}(A) \land \text{rel}(D) \land \text{rel}(J), \\ \text{rel}(A) \land \text{rel}(E) \land \text{rel}(J), \\ \text{rel}(A) \land \text{rel}(F) \land \text{rel}(K) \land \text{rel}(J) \end{cases}
\]
\(P(\text{rel}(A)) = 1\). The corresponding degrees of supports are shown in the third row of the table.

\[
\begin{array}{cccccccccccccc}
X_i & A & B & C & D & E & F & G & H & I & J & K & L & M & N & O & P & Q & R \\
\hline
P(\text{rel}(X_i)) & 1 & .5 & .9 & .1 & .8 & .6 & .9 & .4 & .5 & .8 & .2 & .5 & .0 & .1 & 0 & 3 & .1 & .6 \\
dsp(\text{Val}(X_i), \Sigma) & 1 & 1 & 1 & 1 & 1 & 1 & 5 & 9 & .91 & .842 & .862 & .45 & .648 & .635 & .465 & .673 & .172 & 0 \\
\end{array}
\]

Of course, \(A\)’s own public key automatically receives maximal support. The keys of \(B, C, D, E,\) and \(F\) receive maximal support because they are directly signed by \(A\). \(R\)’s public key receives no support at all, because no certificate has been issued for \(R\) which leads to \(\text{Args}(\text{Val}(R), \Sigma) = \emptyset\). All other keys are rated with values between 0 and 1. For example, consider the case of \(P\)’s public key with a degree of support of

\[\text{dsp}(\text{Val}(P), \Sigma) = P(\text{Args}(\text{Val}(P), \Sigma)) = 0.673.\]

This means that there is a good amount of support for \(P\)’s public key, but it may not be sufficient to accept the key as a valid public key and use it to encrypt messages or validate signatures. In practice, it is convenient to specify a threshold \(0 \leq \beta \leq 1\) for which \(X_i\)’s key is accepted if and only if \(\beta \leq \text{dsp}(\text{Val}(X_i), \Sigma)\). Maximal security means then to work with \(\beta = 1\), but in such a case, most keys will rejected. On the other hand, all keys are accepted for \(\beta = 0\).

### 4.3. Computing arguments

General algorithms to compute minimal arguments for hypotheses are well documented in the literature [7–10]. Here, we are interested in a special purpose algorithm which allows to compute minimal arguments in the context of the particular type of knowledge base obtained from a web of trust. Compared to the general case, three properties of such a particular \(\Sigma\) make the computation considerably less complex:

- all sentences of \(\Sigma\) are Horn clauses (material implication with positive literals only),
- every sentence of \(\Sigma\) (except \(\text{Val}(X_0)\)) contains exactly one assumption \(\text{rel}(X_i)\),
- only simple hypotheses \(\text{Val}(X_i)\) are of interest.

As we have already observed earlier, this means that there is a one-to-one correspondence between (minimal) arguments for \(\text{Val}(X_i)\) and (minimal) paths between \(X_i\) and \(X_0\) in the certificate graph. In this particular situation, the problem of computing arguments is therefore equivalent to computing paths in a directed graph. For this, we may take any classical search algorithm such as \(\text{depth-first}\) (backtracking) or \(\text{breadth-first}\) search and adapt it such that all solutions (instead of just one) are found. Particular attention has to be paid to the problem of infinite loops which may be caused by the cycles of the certificate graph.

In the following, we present a recursive procedure that computes the set of minimal arguments for all users \(X_i \in \mathcal{U}\). This means that the algorithm computes all minimal paths from all nodes \(X_i\) to \(X_0\). Let \(\Gamma(X_i) \subseteq \mathcal{U}\) denote the set of all users
in \(X_0\)'s key ring for which \(X_i\) has issued a certificate. In the graph of Fig. 3, for example, we have \(\Gamma(A) = \{B, C, D, E, F\}, \Gamma(B) = \{G\}, \Gamma(C) = \{H, I\}\), etc. The idea of the algorithm is based on the observation made in Section 4.1: if \(x\) is a minimal argument for \(Val(X_i)\), then \(rel(X_i) \land x\) is a (possibly non-minimal) argument for all propositions \(Val(X_j)\) with \(X_j \in \Gamma(X_i)\).

According to this remark, the core of the algorithm is a recursive function \(add\_argument(x, X)\), which adds a new argument \(x\) to the current set of arguments \(Args(X)\) for user \(X\). Initially, these sets are all empty. Adding an argument means checking for minimality: if there is an argument \(x' \in Args(X)\) with \(x' \subseteq x\), then \(x\) is non-minimal and the recursion stops. Otherwise, \(x\) is added to \(Args(X)\) and all non-minimal arguments \(x' \subseteq x\) are deleted from \(Args(X)\). Finally, the function recursively calls itself for all \(Y \in \Gamma(X)\) and with the extended argument \(rel(X) \land x\).

\[
\begin{align*}
&\text{function } add\_argument(x, X) \\
&\text{begin} \\
&\text{unless } \exists x' \in Args(X): x' \subseteq x \text{ then} \\
&\text{begin} \\
&\text{Args}(X) \leftarrow \{x\} \cup \{x' \in Args(X) : x' \not\subseteq x\}; \\
&\text{loop for } Y \in \Gamma(X) \text{ do } add\_argument(rel(X) \land x, Y); \\
&\text{end}; \\
&\text{end.}
\end{align*}
\]

If \(X_0 \in \mathcal{U}\) is the owner of the key ring, then the computation of the arguments is initiated by calling the above function by \(add\_argument(\top, X_0)\). The argument \(\top\) initially assigned to \(X_0\) is unconditionally true and reflects the fact that \(X_0\)'s own public key is implicitly valid. Note that \(\top \land x\) and \(x\) are equivalent terms. Once the recursion comes to an end, we obtain the complete sets of minimal arguments \(Args(X_j) = Args(Val(X_j), \Sigma)\) for all users \(X_j \in \mathcal{U}\) included in the key ring.

The procedure induced by the above recursive function is similar to a recursive depth-first backtracking search. As a consequence of this, it is possible that several non-minimal arguments are found during the computation. They are temporarily stored in the corresponding set \(Args(X)\), from which they are deleted when a shorter argument is found. To avoid such non-minimal solutions, one can define a similar procedure in the form of a corresponding breadth-first search. The length of the arguments determines then the order in which they are generated, and non-minimal arguments are thus impossible. This means that the time complexity of such an algorithm depends linearly on the total number
\[
N = \sum_{X \in \mathcal{U}} |Args(Val(X), \Sigma)|
\]  

of all minimal arguments. With respect to the number \(n\) of all users, there may be exponentially many such minimal arguments. But this is the worst case scenario which seems to be atypical in practice. Note that if the complexity of the computation is problematical, it may still be possible to use the above method as an interruptible anytime algorithm [27]. The resulting confidence values are then lower bounds that are guaranteed to be on the "safe side".
4.4. Key revocations

Normally, an introducer who issues a digital certificate confirms the validity of the corresponding public key. A certificate can thus be regarded as positive evidence which broadens the basis for accepting a key. In this sense, evaluating a web of trust is monotone, because adding new certificates increases the corresponding values \( \text{dsp}(\text{Val}(X_i), \Sigma) \) monotonically.

In this subsection, we will discuss the case of an extended web of trust that contains negative evidence in the form of key revocations. Intuitively, a key revocation is a certificate the confirms the falsity of a key. They can be important in at least two different situations:

- Suppose \( X \) is trying to verify \( Y \)'s public key over a secure channel, but the verification fails. \( X \) concludes that the key does not belong to \( Y \) and revokes it.
- Suppose \( X \)'s public key will need to be removed from service. This may be because the corresponding private key has been compromised or irretrievably lost. “Role keys” provide good examples of innocently compromised keys. For example, when a member of staff leaves an institution, it is undesirable that they should still be able to read mail sent to the official institution address. In such a case, the institutions CA may revoke \( X \)'s public key.

Consider the web of trust depicted in Fig. 3 and suppose \( C \) has revoked \( G \)'s public key and \( I \) has revoked \( J \)'s public key. Formally, we write \( G \not\approx C \) and \( J \not\approx I \). How does this affect the evaluation of the public keys?

Let \( \mathcal{R} \) be the set of all key revocations included in the key ring. A single key revocation \( r \in \mathcal{R} \) of the form \( X_i \not\approx X_j \) can then be expressed by the following propositional sentence:

\[
\rho(r) = \rho(X_i \not\approx X_j) = \text{rel}(X_j) \land \text{Val}(X_j) \rightarrow \neg \text{Val}(X_i). \tag{13}
\]

The idea of this translation is analogue to (7): if \( X_j \)'s public key is valid and if \( X_j \) is a reliable revoker, then \( X_i \)'s public key is invalid. In the following, we will thus consider the evaluation of public keys based on an extended knowledge base

\[
\Sigma = \{\text{Val}(X_0)\} \cup \{\gamma(c) : c \in \mathcal{G}\} \cup \{\rho(r) : r \in \mathcal{R}\}. \tag{14}
\]

First of all, note that \( \Sigma \) still consists of Horn clauses only. This means that we can still expect simple computations. In fact, it is easy to see that every certificate chain \( Y \not\approx Y_1 \Rightarrow \cdots \Rightarrow Y_i \Rightarrow X_0 \) corresponds to a counter-argument against \( \text{Val}(Y) \) and vice versa. Counter-arguments can therefore be derived from the sets \( \text{Args}(\text{Val}(X_i), \Sigma) \). For example, if we extend the graph of Fig. 3 with \( G \not\approx C \) and \( J \not\approx I \), then we get

\[
\text{Args}(\neg \text{Val}(G), \Sigma) = \{\text{rel}(A) \land \text{rel}(C)\},
\]

\[
\text{Args}(\neg \text{Val}(J), \Sigma) = \left\{ \begin{array}{l}
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(I)
\end{array} \right\},
\]

\[
\text{Args}(\neg \text{Val}(X_i), \Sigma) = \emptyset \quad \text{for all } X_i \neq G, J.
\]
For both $G$ and $J$ we have thus arguments as well as counter-arguments. This means that the set of conflicts $\text{Args}(\perp, \Sigma)$ is no longer empty. In general, if we have non-empty sets $\text{Args}(\text{Val}(X_i), \Sigma) \neq \emptyset$ and $\text{Args}(\neg \text{Val}(X_i), \Sigma) \neq \emptyset$ for some $X_i \in \mathcal{U}$, then minimal conflicts are constructed by conjoining the corresponding minimal arguments and counter-arguments. For example, because $\text{rel}(A) \land \text{rel}(B)$ is an minimal argument for and $\text{rel}(A) \land \text{rel}(C)$ a minimal counter-argument against $\text{Val}(G)$, it follows that

$$\text{rel}(A) \land \text{rel}(B) \land \text{rel}(A) \land \text{rel}(C) \equiv \text{rel}(A) \land \text{rel}(B) \land \text{rel}(C)$$

is a conflict with respect to $\Sigma$. In fact, if we suppose that $A$, $B$, and $C$ are all reliable ($A$ is totally reliable by default), then $B$'s certificate and $C$'s key revocation for $G$ are in conflict to each other, which is impossible. More conflicts are obtained from $I$'s key revocation for $J$. There are three minimal arguments and two minimal counter-arguments for $\text{Val}(J)$, from which we can construct six (possibly non-minimal) conflicts:

$$\begin{align*}
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(D) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(E) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(E) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(F) \land \text{rel}(I) \land \text{rel}(K), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(F) \land \text{rel}(I) \land \text{rel}(K).
\end{align*}$$

Together with the conflict $\text{rel}(A) \land \text{rel}(B) \land \text{rel}(C)$ obtained for $G$ and by dropping non-minimal ones, four minimal conflicts remain:

$$\text{Args}(\perp, \Sigma) = \left\{ \begin{array}{l}
\text{rel}(A) \land \text{rel}(B) \land \text{rel}(C), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(E) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(F) \land \text{rel}(I) \land \text{rel}(K) \end{array} \right\}.$$

Of course, the presence of counter-arguments and conflicts has a great impact on the quantitative evaluation of the public keys. In fact, every additional key revocation decreases the degree of support and also the degree of possibility of some $X_i \in \mathcal{U}$. Recall that the absence of key revocations implies $\text{dps}(\text{Val}(X_i), \Sigma) = 1$ for all $X_i \in \mathcal{U}$. If we reapply (3) and (4) for the current example and the same prior probabilities $P(\text{rel}(X_i))$ as in Section 4.2, we get the following new values:

<table>
<thead>
<tr>
<th>$X_i$</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
<th>O</th>
<th>P</th>
<th>Q</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(\text{rel}(X_i))$</td>
<td>.3</td>
<td>.9</td>
<td>.6</td>
<td>.8</td>
<td>.8</td>
<td>.9</td>
<td>.5</td>
<td>.2</td>
<td>.5</td>
<td>.0</td>
<td>.1</td>
<td>.0</td>
<td>.6</td>
<td>.1</td>
<td>.3</td>
<td>.1</td>
<td>.6</td>
</tr>
<tr>
<td>$\text{dps}(\text{Val}(X_i), \Sigma)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>.733</td>
<td>.747</td>
<td>.755</td>
<td>.83</td>
<td>.41</td>
<td>.353</td>
<td>.1</td>
<td>.604</td>
<td>.155</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$\text{dps}(\text{Val}(X_i), \Sigma)$</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>.267</td>
<td>.896</td>
<td>.991</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Note that the key revocations for $G$'s and $J$'s public keys do not only influence the values of $G$, $J$, and their dependents, but also of all other users except of those who have a direct certificate from $A$ (including the revokers $C$ and $I$). This is due to the fact that $C$'s key revocation for $G$, for example, negatively influences $A$'s initial trust in $C$, because it is in conflict with the certificate chain $G \Rightarrow B \Rightarrow A$. As consequence, we can also expect lower values for all users who depend on a certificate issued by $C$, which is confirmed by the values of the above table.

In the presence of key revocations, the decision between valid and invalid keys is more complicated. It depends not just on $\text{dps}(\text{Val}(X), \Sigma)$, but also on $\text{dps}(\text{Val}(X), \Sigma)$. A simple approach is specify two thresholds $\beta$ and $\beta'$ with $0 \leq \beta' < \beta \leq 1$. A key is accepted whenever $\beta \leq \text{dps}(\text{Val}(X), \Sigma)$ (as before), and it is rejected for $\beta' \geq \text{dps}(\text{Val}(X), \Sigma)$. Otherwise, no definite decision is taken. For $\beta = 0.75$ and $\beta' = 0.5$, for example, $A$ would thus accept the keys of $B$, $C$, $D$, $E$, $F$, $J$, and $K$, but reject the key of $G$. All other keys are considered as “unclassified” until additional certificates or key revocations allow a better judgment.

4.5. Recommendations

Another possible extension of a web of trust is the inclusion of recommendations. Because it is impossible to personally know all entities of a large network, it may be difficult to assign adequate trust values $P(\text{rel}(X))$ to all users $X \in \mathcal{U}$. In a classical web of trust (one without recommendations), this problem is usually solved by assigning minimal trust $P(\text{rel}(X)) = 0$ to all unknown users $X$. This implies that the resulting confidence values $\text{dps}(\text{Val}(X), \Sigma)$ are guaranteed to be pessimistic, that is on the “safe side”.

A more sophisticated possibility to judge the trustworthiness of unknown users is to consider so-called explicit recommendations from other users. Such recommendations can be thought as signed statements about the trustworthiness of other users and are similar to certificates. A distributed public-key management that includes explicit recommendations has first been introduced in [25,1]. A more complete view on this is given in [13]. Because recommendations are often sensitive information to be treated confidentially, which demands additional cryptographical protocols, they have not yet been used in systems such as PGP.

The model introduced by Maurer in [13] allows several levels of recommendation. A recommendation of the first level is for someone to be a trustworthy issuer of certificates. A recommendation of the second level is for someone to be trustworthy in giving recommendations of the first level, and so on. In general, a recommendation of level $i$ is for someone to be trustworthy in giving recommendations of level $i - 1$. In a certain sense, certificates can be regarded as recommendations of level 0.

In principle, it is possible to distinguish individual trust values $P(\text{rel}(X))$ for all possible levels $i = 1, 2, \ldots$, but here we will only consider one value $P(\text{rel}(X))$ that is applicable to all levels. At first sight, this is similar to Maurer’s assumption that trust of level $i$ implies trust of all levels $j < i$, but it means also, that only one level of recommendations will be considered. Note that this is not a conceptual restriction, but it will help to keep the discussion reasonably simple.
Consider again the web of trust depicted in Fig. 3. Suppose \( H \) is unknown to \( A \), but consider the case in which \( H \) has been recommended by \( G \) and \( I \). Formally, we write \( H \leadsto G \) and \( H \leadsto I \). How does this determine the trustworthiness of \( H \) from the point of view of \( A \)? Let \( \mathcal{M} \) be the set of all available recommendations. A single recommendation \( m \in \mathcal{M} \) of the form \( X_i \leadsto X_j \) can then be translated into the following propositional sentence:

\[
\mu(m) = \mu(X_i \leadsto X_j) = \text{rel}(X_j) \land \text{Val}(X_j) \rightarrow \text{rel}(X_i).
\]  

(15)

Essentially, this is equivalent to the second inference rule in Maurer’s model [13, Definition 3.2], except that here the recommendation itself is not explicitly linked to a corresponding proposition. The idea of this translation is the following: if \( X_j \) is considered to be trustworthy (reliable in giving recommendations and issuing certificates), and if \( X_j \)'s public key is valid, then \( X_i \) is also considered to be trustworthy. In the following, we will thus consider the evaluation of public keys based on an extended knowledge base

\[
\Sigma = \{\text{Val}(X_0)\} \cup \{\gamma(c) : c \in \mathcal{C}\} \cup \{\rho(r) : r \in \mathcal{R}\} \cup \{\mu(r) : r \in \mathcal{M}\}.
\]

First of all, note that the new sentences \( \mu(m) \) are also Horn clauses. As a consequence, we can still expect the necessary computations to be simple. In fact, given \( G \)'s and \( I \)'s recommendations for \( H \), if we consider the hypothesis \( \text{rel}(H) \) with respect to user \( H \), then it is easy to see that \( \text{rel}(H) \) is true whenever either \( \text{Val}(G) \) together with \( \text{rel}(G) \) or \( \text{Val}(I) \) together with \( \text{rel}(I) \) is true. This means that the set of minimal arguments for \( \text{rel}(H) \) can be derived from the sets \( \text{Args}(\text{rel}(G), \Sigma) \) and \( \text{Args}(\text{rel}(I), \Sigma) \). The leads then to

\[
\text{Args}(\text{rel}(H), \Sigma) = \left\{ \begin{array}{l}
\text{rel}(A) \land \text{rel}(B) \land \text{rel}(G), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(I)
\end{array} \right\},
\]

and the corresponding numerical value \( \text{dsp}(\text{rel}(H), \Sigma) \) measures the trustworthiness of user \( H \).

Now, let the goal be to evaluate the validity of \( N \)'s public key. Because one of the three original minimal arguments for \( \text{Val}(N) \) depends on \( \text{rel}(H) \) (see Section 4.1), we can determine the new set \( \text{Args}(\text{Val}(N), \Sigma) \) by substituting \( \text{rel}(H) \) with the three elements of \( \text{Args}(\text{rel}(H), \Sigma) \). After dropping non-minimal arguments, this leads to

\[
\text{Args}(\text{Val}(N), \Sigma) = \left\{ \begin{array}{l}
\text{rel}(A) \land \text{rel}(B) \land \text{rel}(C) \land \text{rel}(G), \\
\text{rel}(A) \land \text{rel}(C) \land \text{rel}(I), \\
\text{rel}(A) \land \text{rel}(D) \land \text{rel}(I)
\end{array} \right\},
\]

from which the new value for \( \text{dsp}(\text{Val}(N), \Sigma) \) can be derived. By doing so, note that \( \text{rel}(H) \) must be considered as an element of the set \( P \) instead of the set \( A \), which means that user \( A \) does not need to specify a prior probability \( P(\text{rel}(H)) \). This remark applies to all users whose trustworthiness depends on recommendations from others.

The problem of computing minimal arguments in an extended web of trust with recommendations can be solved as illustrated above. We do not give an explicit algo-
rithm here, but one can think of a recursive procedure similar to the one of Section 4.3, that is executed subsequent to the initial call of \textit{add\_argument}($T$, $X_0$).

5. Conclusion

This paper investigates trust evaluation in public-key infrastructures based on probabilistic argumentation. It is remarkable how straightforwardly certificate graphs, key revocations, and recommendations are expressible as probabilistic argumentations systems. Degree of support seems then to be an appropriate quantity to rate the validity of public keys. We propose the results of this paper to be taken as the basis for a more sophisticated trust model in cryptographic applications like PGP.

Future work will focus on how to include dependencies between the users. Due to the clear conflict management of probabilistic argumentation and the expressive power of the underlying logical language, we already see a number of possible ways to extend the basic model accordingly. Right now, these are preliminary results that need to be worked out.

Acknowledgment

I would like to thank the anonymous referees for their valuable comments. This research is supported by (1) Alexander von Humboldt Foundation, (2) German Federal Ministry of Education and Research, (3) German Program for the Investment in the Future.

References


