A Coloured Petri Net Trust Model

Peter Lory
Universität Regensburg
Institut für Wirtschaftsinformatik
D-93040 Regensburg, Germany
Peter.Lory@wiwi.uni-regensburg.de

Abstract

Public-key infrastructures are a prerequisite for security in distributed systems and for reliable electronic commerce. It is their goal to provide the authenticity of public keys. Formal models for public-key infrastructures (trust models) contribute decisively to a deeper understanding of the desirable design principles of these infrastructures. The trust model of the present paper is based on the modelling technique of coloured Petri nets. These are a special class of high-level Petri nets with an intuitively appealing graphical representation and a few, but powerful primitives. Elaborate and well tested software is available.

1. Introduction

Public-key cryptography is a prerequisite for electronic commerce and electronic government. Although public keys do not need to be kept secret, and in fact wide knowledge of an entity's public key is desirable, the security problem is that Alice must know for certain that a particular public key really does belong to Bob. If Alice can be tricked into thinking that Mallory's public key is Bob's, Mallory can impersonate Bob to Alice. The protection of public keys against attacks is the vulnerable spot of public-key cryptography. Please note that here, and in the following entities are often called Alice and Bob (following tradition in cryptography). However, the reader should keep in mind that they could be a human, a server, a client machine or a personal token like a chipcard or something else.

It is the goal of a *public-key infrastructure* (PKI) to solve the above mentioned problem. Public-key infrastructures rest on the concept of a *public-key certificate*. A certificate binds the entity's identity to the specified public key. If Alice has several certificates, she can build a *chain of certificates* where each public key is certified by the previous entity in the chain, and where she has specified the first public key as authentic and all intermediate entities as trustworthy. Section 2 shows how trust can be propagated in a similar way by the concept of a *recommendation*. Usually, these pieces of information are stored at different places. So, a public key infrastructure can be seen as a distributed database of public-key certificates, recommendations and further information. Thus, it forms a web of certificates and recommendations. Trust plays a prominent role in this web. Consequently, models for public-key infrastructures are often called *trust models*.

Usually, a user of a public-key infrastructure has a set of statements about the authenticity of certain public keys and on the trustworthiness of certain entities. Together with the available collection of certificates and recommendations this makes up the user's (Alice's) view to the public-key infrastructure. It is not the aim of this paper to discuss the problem, how Alice can find the necessary set of information that enables her to prove the authenticity of a certain public key. Rather, this paper gives a formal method that finds all the statements about the authenticity of public keys that can be derived from Alice's view. For this purpose it uses the modelling technique of coloured Petri nets (see [3]). The model in [7] achieves the same goal by a logical calculus, from which the present model is derived to a high extent. However, the modelling technique of coloured Petri nets is more easily accessible for unexperienced users. Additionally, the present model can be embedded into Petri nets for cryptographic protocols in a straightforward manner. The model focuses on the main aspects of public-key infrastructures and does not yet include certificate revocation.

2. Alice's view

Certificates propagate authenticity of public keys. However, this goal is achieved only if the user of the certificate trusts its issuer. Since the former cannot know personally all the entities he/she has to rely on, there is also a need for propagation of trust. This task is done by recommendations. A recommendation can be considered as a signed statement about the trustworthiness of another entity.

Proceedings of the 14th International Workshop on Database and Expert Systems Applications (DEXA'03)
1529-4188/03 $17.00 © 2003 IEEE

Let it be Alice's aim to establish the authenticity of another person's, for instance Bob's, public key. For that purpose she builds her initial view, which includes all the certificates and recommendations that can be relevant for authenticating Bob's public key and can be retrieved from the public-key infrastructure. Additionally, Alice's view includes statements as a part of her belief, such as authenticity of certain public keys and trust in certain entities. Following [5] and [7] trust is modelled with respect to entities and not with respect to keys. Formally speaking, Alice's view is a set of statements of the type given in the following definition (cf. [7]).

Definition 1 (Statements and Alice's view) Alice's view is a set of statements of the following type: *Aut(X,P)* says that Alice is convinced that the public key P belongs to entity X (authenticity). *Cert(X,P,Y,Q)* says that Alice holds a certificate, which asserts that Q is a public key for entity Y. This certificate is allegedly issued and signed by entity X. The signature passes verification by the public key P. *Trust(X,1)* says that Alice is convinced that entity X is trustworthy of level 1, i.e. this entity can be trusted for issuing certificates. *Rec(X,P,Y,i)* says that Alice holds a recommendation of level i for entity Y, i.e. it asserts that entity Y is trustworthy of level i. This recommendation is allegedly issued and signed by entity X. The signature passes verification by the public key P. *Trust(X,i)* with $i > 1$ says that Alice is convinced that entity X is trustworthy of level i, i.e this entity can be trusted for issuing recommendations of level $i-1$.

A remark about the word "alleged" in the definitions for certificates and recommendations seems in place: Without verification, it is not clear that entity X has issued the certificate or the recommendation, respectively. However, if Alice can gain evidence that the public key P belongs to entity X, she can verify that entity X is indeed the issuer. Alice's view allows a graphic representation. Figure 1 gives the graphic elements for the statements of Definition 1.

$$
\begin{aligned}
Aut(X,P): &\quad A \longrightarrow (X,P)\\
Trust(X,i): &\quad A \overset{i}{\dashrightarrow} X\\
Cert(X,P,Y,Q): &\quad (X,P) \longrightarrow (Y,Q)\\
Rec(X,P,Y,i): &\quad (X,P) \overset{i}{\dashrightarrow} Y
\end{aligned}
$$

Figure 1. Graphic elements illustrating Alice's view ("A" refers to Alice)

Example 1 The public-key infrastructure of this example is a global hierarchy model as suggested in the concept

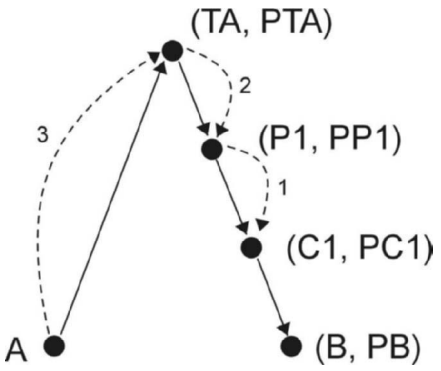

Figure 2. Alice's view (Example 1)

of *Privacy Enhanced Mail* (PEM), which has been presented in [4]. The hierarchy consists of the single *trust anchor* TA (the Internet Policy Registration Authority). The TA directly issues certificates to a second tier of entities designated *policy certification authorities* ($P1, P2, \ldots$), which, in turn, issue certificates to certification authorities ($C1, C2, \ldots$). These certification authorities issue certificates to (subordinate) certification authorities or directly to users (individuals, organizations). Let the user B with his public key PB be registered at certification authority $C1$, which is registered at $P1$. Alice wants to prove the authenticity of user B's public key PB. Within the frame of PEM it is reasonable to assume that she is able to collect those statements in her view that are given in Figure 1. There is a chain of certificates from TA to PB. In this chain, $PP1$ and $PC1$ denote public keys for $P1$ and $C1$, respectively. The accompanying recommendations may be established by off line means. For instance, in the case of a "high assurance" policy certification authority $P1$ the recommendation $Rec(P1, PP1, C1, 1)$ might mean, that $P1$ executes a legal contract with the certification authority $C1$, which forces $C1$ to use a high level of authentication when it grants certificates to its users; e.g. $C1$ might be obliged to strictly use as a policy the same level of authentication it would employ in issuing ID cards. This piece of information, digitally signed by $P1$'s private key (the companion of the public key $PP1$), is the recommendation $Rec(P1, PP1, C1, 1)$.

Example 2 The ICE-TEL project has grown out of the Privacy Enhanced Mail (PEM) concept and organises a public-key infrastructure as a web of hierarchies (see [1]). Each separate hierarchy is referred to as a *security domain*. Each security domain has at its apex a single certification authority, called the *trusted point*. This trusted point may certify both users and subordinate certification authorities within the domain confirming that they all abide by the same overall security policy. In the language of Definition 1 the latter

can be interpreted as a recommendation for the subordinate certification authority issued and signed by the trusted point. The name and public key of the trusted point is known to all the objects in the domain. This public key will initially be distributed by some out of band proprietary means. This public key of the trusted point and its associated policy id are stored in the user's *personal security environment*, which can be modelled as the user's (Alice's) view. By construction this view contains both an *Aut*-statement and a *Trust*-statement with respect to the trusted point (cf. Definition 1). Let B be another user in the same security domain and let Alice want to prove the authenticity of this user's public key PB. Then the scenario is similar to Figure 2 with three levels only and the trust point at the apex. So, the comments made in Example 1 apply here, too.

However, ICE-TEL is more flexible than PEM. It allows cross certification to remote security domains. The security administrator of Alice's local trusted point vets the remote domain on behalf of the users in the local domain who trust him to do so, and issues a cross certificate for the remote trusted point, which does not only certify the public key of this entity but also the policy in the remote domain. The latter is accomplished by an appropriate entry in the *policy mappings field* in the X.509 Version 3 certificate. In the language of Definition 1 this is a recommendation for the remote trusted point issued by Alice's trusted point. The administrator of Alice's trusted point can actually limit the number of steps in the chain of certificates he is willing to accept by a proper choice of the level in his recommendation. Indeed, in the X.509 Version 3 certificate the local administrator can actually limit the number of users in the remote domain that are to be trusted by specifying (via the *name constraints* field) a subset of the user names from the remote domain whose certificates are to be trusted. Let Alice be a user in the security domain of the trusted point $TP1$ with public key $PTP1$, whereas user B belongs to the security domain of the trusted point $TP2$. If Alice wants to prove the authenticity of user B's public key PB without authentic knowledge of the public key $PTP2$ of $TP2$, she will try to built a chain of certificates and accompanying recommendations first from $(TP1, PTP1)$ to $(TP2, PTP2)$ and then from $(TP2, PTP2)$ to (B, PB). Alice has an *Aut*-statement for the public key of her trusted point $TP1$, and it is assumed that her trust in $TP1$ is sufficiently high. The graph of the resulting view is given in Figure 3. It is structurally equivalent to the graph in Figure 2.

With the exception of node "A" (for Alice) the nodes in the graph of Figures 2 and 3 are pairs of entities and public keys. Each pair represents a binding between an entity and its alleged public key. Whether this binding is authentic under Alice's view or not can be decided by the formal method given in the next section.

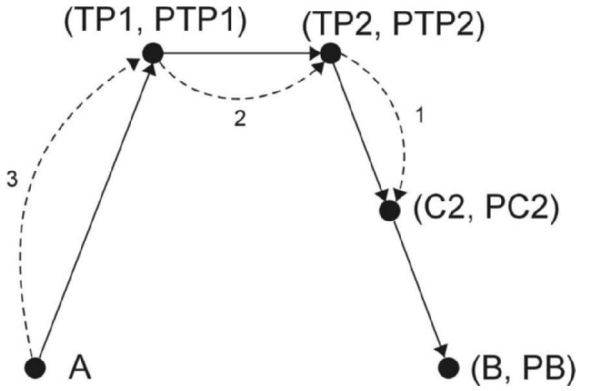

Figure 3. Alice's view (Example 2)

3. The coloured Petri net model

Coloured Petri nets are special high-level Petri nets. A thorough description is given in [3]. These nets have a graphic representation which makes it easy even for non-experts to grasp systems of high complexity. Coloured Petri nets have a well-defined semantics which unambiguously defines the behaviour of the net.

Petri nets have been invented to model processes. The process Alice has to deal with is a result of Alice's interest in *Aut*-statements. Each of these statements proves the authenticity of a binding between a public key and an entity. The other statements, *Trust*, *Cert* and *Rec*, are of no direct value for Alice. Their purpose is to support the derivation of new *Aut*-statements. These derivations have to satisfy certain rules. These rules can be formalized and made precise as transitions in a coloured Petri net. This net (see Figure 4) models the process of deriving all the *Aut*-statements that are consistent with Alice's view. It will be explained in detail below.

Figure 4 has been drawn by the Design/CPN-software. This is a graphic computer tool which supports the practical use of coloured Petri nets. Resources and technical support on Design/CPN are available via the web site [2]. All simulations in the present paper have been performed with this software.

The coloured Petri net of Figure 4 uses seven colour sets (types). They are defined in the global declaration node. The net has four places: `Alice`, `Certificates`, `TrustPool` and `Recommendations`. The place `Alice` acts as a pool of *Aut*-statements. Consequently, its colour set is `Aut`, which is defined as the cartesian product of the colour sets `Entity` and `Keystr`. Thus, a token of this type is a pair of strings. The first string identifies an entity; the second string represents a key (cf. Definition 1). The place `Certificates` collects the *Cert*-statements.

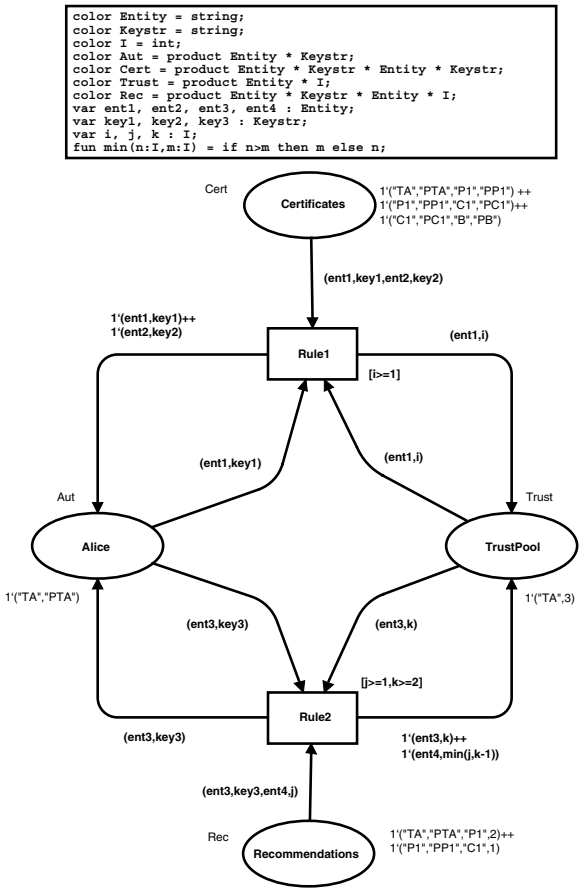

Figure 4. The coloured Petri net trust model

These statements can be modelled as quadruples of strings with identifiers for entities in the first and third components and keystrings in the second and fourth components. The colour set (`Cert`) of this place is defined accordingly in the global declaration node. The places `TrustPool` and `Recommendations` act as a pools of *Trust*-statements and *Rec*-statements, respectively. The corresponding colour sets, `Trust` and `Rec`, are again defined in the global declaration node according to Definition 1. The initial marking in Figure 4 corresponds to Alice's view of Example 1. Please note, that at each place the marking is a multi-set (see [3]) over the colour set attached to the place. For example, the initial marking at the place `Recommendations` means that this multi-set contains one appearance of the token `("TA","PTA","P1",2)` and one appearance of the token `("P1","PP1","C1",1)`. The transitions `Rule1` and `Rule2` are the core of the model.

Rule1: This transition has three incoming and two outgoing arcs and a guard. The variables of this transition are: `ent1` and `ent2` of colour `Entity`, `key1` and `key2` of colour `Keystr`, `i` of colour `I`. Let now the data values `"TA"`, `"PTA"` , `"P1"`, `"PP1"`, and 3 be assigned to the variables `ent1`, `key1`, `ent2`, `key2`, and `i`, respectively. This creates a *binding* (which should not be confused with the concept of a binding between an entity and a public key). The pair consisting of a transition and a binding of its variables forms a *binding element*. In order for a binding element to be *enabled* in a certain marking of the places, it must be possible to bind data values to the variables appearing on the surrounding arc expressions and in the guard of this transition such that each of the arc expressions evaluate to tokens which are present in the corresponding input place. Additionally, the guard must be satisfied.

For the above binding element these requirements are fulfilled (in the case of the initial marking of Figure 4). If a binding element is enabled, it is ready to *occur*. An occurrence of the above binding element removes a token with the values `("TA","PTA","P1","PP1")` from the place `Certificates`, it removes a token with the values `("TA","PTA")` from the place `Alice` and it removes a token with the values `("TA",3)` from the place `TrustPool`. Further, it adds the tokens `("TA","PTA")` and `("P1","PP1")` to the place `Alice` and the token `("TA",3)` to the place `TrustPool`. Hence, the occurrence of the above binding element has the effect, that the token `("TA","PTA","P1","PP1")` (representing the corresponding *Cert*-statement) is removed from the place `Certificates` (the pool of certificates) and the token `("P1","PP1")` (a new *Aut*-statement) is added to the place `Alice`, which acts as the pool of *Aut*-statements. The tokens `("PA","PTA")` and `("TA",3)` return to their places. This is essential, because they may be needed in further steps. It is not necessary to return the token `("TA","PTA","P1","PP1")`. This token acts as a certificate with the only purpose to establish the statement *Aut(P1,PP1)* (represented by the token `("P1","PP1")`). Once this is done successfully, the certificate cannot be of any further value.

The generalization of this example is straightforward and shows that the transition `Rule1` acts as a producer of statements about the authenticity of public keys. It states that Alice can derive the authenticity of the binding between the entity Y and the public key Q (denoted by *Aut(Y,Q)* and represented in the model by the token `("Y","Q")`), if the following three conditions are satisfied:

1. Alice holds a certificate, which says that Q is a public key for entity Y. The alleged issuer and signer of this certificate is entity X and the signature passes verification by the public key P. This is denoted by *Cert(X,P,Y,Q)* and represented in the model by the token `("X","P","Y","Q")`.

2. Alice has or can derive the authenticity of the binding between entity X and public key P. This is denoted by *Aut(X,P)* and represented in the model by the token `("X","P")`.

3. Alice has or can derive trust of level i with $i \geq 1$ for entity X. This is denoted by *Trust(X,i)* and represented in the model by the token `("X",i)`. It is tacitly assumed that trust of level i implies trust of lower levels.

Rule2: This transition has also three incoming and two outgoing arcs and a guard. The variables of this transition are: `ent3` and `ent4` of colour `Entity`, `key3` of colour `Keystr`, `j` and `k` of colour `I`.

The transition `Rule2` acts as a producer of statements about the trustworthiness of entities. It states that Alice can derive trust in entity Y of level m (denoted by *Trust(Y,m)* and represented in the model by the token `("Y",m)`), if the following three conditions are satisfied:

1. Alice holds a recommendation, which says that Y is trustworthy of level j with $j \geq m$. The alleged issuer and signer of this recommendation is entity X and the signature passes verification by the public key P. This is denoted by *Rec(X,P,Y,j)* and represented in the model by the token `("X","P","Y",j)`. It is tacitly assumed that a recommendation of level j implies recommendations of lower levels.

2. Alice has or can derive the authenticity of the binding between entity X and public key P. This is denoted by *Aut(X,P)* and represented in the model by the token `("X","P")`.

3. Alice has or can derive trust of level k with $k - 1 \geq m$ for entity X. This is denoted by *Trust(X,k)* and represented in the model by the token `("X",k)`. It is again tacitly assumed that trust of level k implies trust of lower levels.

Please note, that the tokens that enter the transition `Rule2` from the places `Alice` and `TrustPool` return to their places. This is essential, because they may be needed in further steps. However, it is not necessary to return the token that enters the transition from the place `Recommendations` (see [6]).

4. Role of the occurrence graph

The prime interest in the application of the coloured Petri net of Figure 4 is to find all the reachable markings of the place `Alice`, because these markings correspond directly to those *Aut*-statements that can be derived from Alice's view, if the marking corresponding to this view is chosen as initial marking. This is closely related to the concept of the occurrence graph. This graph contains a node for each reachable marking and an arc for each occurring binding element (see [3]). Several dynamic properties of the Petri net including its boundedness properties can be investigated using the occurrence graph.

Particularly useful is the *best upper multi-set bound*, which is delivered in the *standard report* of the Design/CPN occurrence graph tool. The best upper multi-set bound for the place p is defined as the multi-set $\max_{M \in V} M(p)$, where V is the set of nodes (reachable markings) of the occurrence graph and $M(p)$ denotes the marking for place p (see [3]). This definition is sound because all occurrence sequences of the coloured Petri net of Figure 4 are finite (see [6]). The best upper multi-set bound for the place `Alice` in this coloured Petri net contains exactly those tokens that belong to reachable markings of this place.

Example 3 In Example 1 Design/CPN calculates the best upper multi-set bound for the place `Alice` to

```
1`("B","PB") ++ 1`("C1","PC1") ++
1`("P1","PP1") ++ 1`("TA","PTA").
```

This notation for a multi-set follows the output of the Design/CPN software and has been explained in Section 3. Thus, the result proves the authenticity of the public key PB for user B.

References

[1] D.W. Chadwick, A.J. Young, and N. Kapidzic Cicovic. Merging and extending the PGP and PEM trust models: The ICE-TEL trust model. *IEEE Network*, 11:16–24, 1997.

[2] Design/CPN online. `http://www.daimi.au.dk/designCPN/`.

[3] K. Jensen. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*, Volume I, II, III. Springer, Berlin, 1997.

[4] S.T. Kent. Internet privacy enhanced mail. *Communications of the ACM*, 36:48–60, 1993.

[5] R. Kohlas and U. Maurer. Reasoning about public-key certification: On bindings between entities and public keys. *IEEE Journal on Selected Areas in Communication*, 18:591–600, 2000.

[6] P. Lory. A process-oriented model for authentication on the basis of a coloured Petri net. In W.M.P. van der Aalst, A. ter Hofstede, and M. Weske (eds.), *Proceedings of the International Conference on Business Process Management 2003 (BPM'2003)*, Lecture Notes in Computer Science, Springer, Berlin, to appear.

[7] U. Maurer. Modelling a public-key infrastructure. In E. Bertino, H. Kurth, G. Martella, and E. Montolivo (eds.), *Proceedings of the 1996 European Symposium on Research in Computer Security (ESORICS'96)*, Lecture Notes in Computer Science, 1146:325–350, Springer, Berlin, 1996.