TDDC03 Projects, Spring 2004

A Public Key Infrastructure for Peer-to-Peer

Anna Kent
Francisco Urendes

Supervisor: Claudiu Duma

A Public Key Infrastructure for Peer-to-Peer

Anna Kent Francisco Urendes
Information Security
University of Linköping
{annke958, fraur388}@student.liu.se

Abstract

Peer-to-Peer (P2P) has become more and more commonly used over the last few years. As more and more possible applications such as e-commerce are developed, the question of security services such as authentication, non-repudiation, confidentiality, integrity and authorisation increases in importance. Especially authentication becomes even more important. Public Key Infrastructures, (PKI), can be used to provide a basis for adding security mechanisms, especially authentication. A number of PKIs exist that have been in use for some time now. These can be divided into two main groups, hierarchical PKIs such as X.509 and decentralised PKIs such as Pretty Good Privacy (PGP) .In this paper we examine the requirements placed on a PKI by P2P, and give suggestions for how a PKI suitable for P2P should be designed.

Introduction

A new approach to distributed computing has emerged in the last years, P2P. It addresses the problem of organising large-scale computational societies and is an emerging paradigm that today is viewed as a potential technology that could re-formulate well known distributed architectures where all participant computers have equivalent capabilities and responsibilities [10].

As more and more applications for P2P are developed, the concept of security within them, more specifically authentication, becomes increasingly important. These new applications are becoming not only interesting for the normal user, now a lot of companies and institutions could take advantage of these attractive new approaches, for example for e-commerce, exchange of documents among different departments and other things with some value for the peers. They are by nature decentralised, so how should trust in another peer be enabled without the use of a trusted entity in the middle?

PKI (Public Key Infrastructure) provides a method of ensuring confidentiality, authenticity, integrity and non-repudiation. It is based on public key cryptography- public keys are combined with an identity in a digital certificate, which can be verified to allow the mapping between the public key and the given identity to be trusted.

Today, most currently deployed PKIs use centralised hierarchical models for their trust computations, where all participants need to authenticate their communication partner, and to transfer data privately over a public infrastructure. Thus, a central certification authority (CA) or a small hierarchy of those can be employed to provide certificates to all users. This, however, is not particularly well suited to the decentralised nature of P2P. The situation can become even more difficult when several usually competing entities, each owning its own certification hierarchy, want to enter a limited collaboration, and use each other's certificates to secure communication.

In this paper we are going to study how PKI could be applied in this kind of architecture, taking into account the different models of PKI and investigating which one is more suitable for these distributed architectures. We will introduce readers to PKI and the different models that exist. Then we introduce P2P, explaining the requirements it places on a PKI and giving suggestions, with advantages and inconveniences, for how a suitable PKI should be designed in order to cover the security needs of the users of this new approaches.

1. Background

1.1. Defining PKI's terms.

Public Key Cryptography or asymmetric key cryptography is a type of cryptography based on each user having two-associated keys- a public key and a private key. These keys are mathematically related in such a manner that a message encrypted with the public key can only be decrypted with the corresponding private key and vice versa. The mathematical relation between these keys is such that it is computationally infeasible to derive one key given knowledge of the other.
The public key is made available to anyone wishing to communicate securely with the key owner, hence the name "public". In contrast, the key owner must keep the private key secret.

PKI: Public Key Infrastructure. In Nash and Duane's book [1], it is stated, "The primary role of a PKI is establishing digital identities that can be trusted". Public key cryptography can be used to provide confidentiality, integrity, authentication and non-repudiation, digital signatures etc. For this to be viable, however, a method of binding a public key to a trusted identity is necessary. This is exactly what a PKI provides - a method of binding identities to public keys and distributing this information. A PKI usually consists of several parts. These are listed below and the duties described.

Digital Certificate: A digital document giving a trustworthy binding between a user identity and public key.

RA: Registration Authority. One or more Registration Authorities may be present in a PKI. These receive requests from end-users for certificates and attempts to verify the identity given in the request. If the verification is concluded satisfactorily, the RA sends a request to a CA, asking that a certificate with the given identity and public key be issued.

CA: Certification Authority. A trusted authority that issues certificates binding a user identity to a public key. These certificates are digitally signed by the CA to ensure their authenticity. The CA will also take over the duties of the RAs if none are present.

1.2. Defining P2P

P2P: Peer-to-peer: Foster and Iamnitchi [10] define P2P as "a class of applications that takes advantage if resources-storage, cycles, content, human presence-available at the edge of the Internet." In Barkai's paper [5] P2P is defined as "… the sharing of computer resources and services by direct exchange". In P2P computers are linked together via the Internet or another network. Instead of the traditional client/server model, however, where certain computers on the network function as clients and others are servers, each participating computer, now called a peer, functions as both a client and a server. These peers are considered to be equal, and can now share resources directly instead of going through a server. Some examples of resources are files and CPU-cycles.

2. PKI Models

A PKI may be designed after two main models - the hierarchical and decentralised models.
In the figures below the large white circles represent CAs. The small grey circles are end users. The grey and white circles represent end-users that also function as CAs.

1.1. Hierarchical

In a hierarchical PKI, the CA's are, as the name implies, organised in a hierarchical manner. They are classified as belonging to one of two categories - intermediate CA's and leaf CA's. There is one main CA, the root CA, at the top of the hierarchy, which is solely responsible for certifying intermediate CA's. This is known as cross-certification. These CA's in their turn certify other CA's lower down in the hierarchy, which may in turn certify further intermediate CA's, and so on. This continues until a set of CA's is reached that do not certify other CA's. These, instead, certify the end users of the PKI, and are called leaf CAs. In the hierarchical model, certification is not usually mutual. For example, a CA certified by the Root-CA will not certify the Root-CA. Instead, the Root-CA will certify itself.
Hierarchical PKI's will typically be used in organisations that are hierarchically organised, for example businesses and universities.
An example of a simple hierarchical PKI is given below (see figure 1).

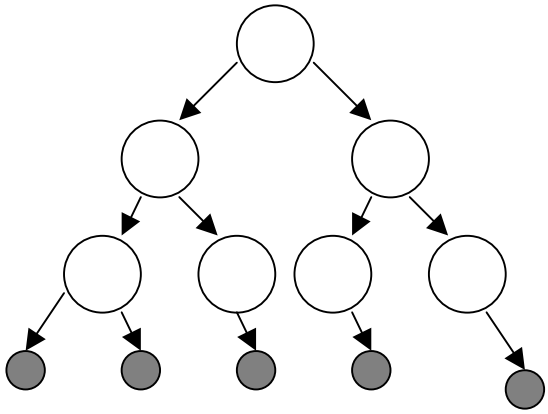

***Figure 1**: A simple hierarchical PKI*

1.2. Decentralised

In Aberer, Hauswirth and Data's paper [4] a decentralised PKI is described as "the public key infrastructure is maintained by the participants itself, without using central controls or specialised structures such as CAs." In this model there are no CA's as such, instead, end-users certify other end-users public keys and thus can be said to function as CAs.
One example of a decentralised PKI is PGP (Pretty Good Privacy.)

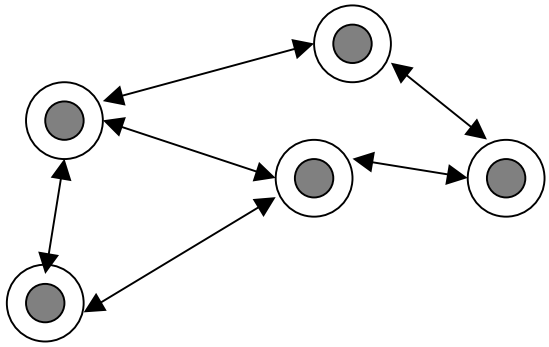

***Figure 2**: A decentralised PKI*

3. PKI for P2P

There have been numerous security threats associated with P2P networks.
If you are using any instant messaging software such as MSN, AOL or Yahoo Messenger, your communication could be eavesdropped upon. Hence you cannot share

secrets over such an unsafe channel, potentially limiting the use of potent applications that are using the emergent P2P technology. Many of these applications are a security risk [6] and threat to privacy as they expose your identity and IP address, by which other users can track you down.

There are many P2P file-sharing applications on the Internet. People generally download content from unknown users and hence can be easily fooled to download an Internet worm or virus, since the content source is not trusted. Also there is a risk of exposing your private files on the Internet. Due to the lack of authentication and security mechanisms in such applications we cannot share files with user based access privileges.

There will be numerous other P2P applications and the existing ones will become more pervasive if we can introduce security by means of authentication and encryption and guarantee the user privacy. In the next section we are going to see how PKI could be used in order to avoid this lack of authentication in P2P.

3.1. Requirements on PKI

P2P is a distributed system where the operations are made directly between the endpoints without a central trustworthy entity**.** Due to the distributed nature of P2P, we should find a PKI model that fits well to this nature. We could say it is the more important requirement, to adapt a PKI model to the distributed nature of P2P, and our work is based in finding a model that follows this requirement.

This requirement can be performed in a pure distributed solution, based in PGP or adding some kind of centralisation in the P2P group, where some endpoints would play the roll of CAs.

Taking into account this requirement and the possible approaches, we have to examine which kind of PKI model could be adapted to such requirements. We will enter in depth into how we could create a suitable PKI for P2P systems. Could PGP be suitable for use in P2P? Could we add some kind of centralisation to P2P in order to use a hierarchical PKI model like X.509? In the next section we are going to study both possibilities seeing advantages, disadvantages and possible improvements of these approaches to P2P systems.

3.2. A semi-centralized approach based on hierarchical PKI.

3.2.1. Adding a PKI X.509. It would certainly be possible, but it would have to be paid for and administrated, which the peers may not agree to. Agreeing among the peers on which CAs to trust may be difficult. This will most likely require that some centralisation be introduced into the peer community, which goes against the decentralised nature of P2P.

3.2.2. Decentralising X.509- dividing up the key into several parts: In this model, several peers will together function as a CA. It would be possible to allow certain peers to function as CAs, but since all peers are not guaranteed to be on-line constantly, this solution is less suitable. Since all peers are not guaranteed to be on-line constantly, The private CA key will be stored in several places. The key will be split into several pieces, with each piece stored by a peer. All of these peers must then sign a certificate for it to be valid. A modification of this model may be used to counteract the fact that all peers will not always be on line. In this model, the key will still be shared among a number of peers. However, all these peers are not required to sign a certificate for it to be valid. Instead, a certain number of them must do so, for example, five out of ten peers.
Some model enabling this form of key-sharing would be needed. There is research being done into this subject but no existing commercial systems as yet.

The peers must have a trusted identity from the beginning. If you don't know the identity of all peers included in a peer-group CA, how can you possibly trust certificates issued by that CA? It would be very difficult to trust a certificate issued by some unknown peers unless you have some certainty about at least their identity. This will probably function in a small peer community where all the peers know each other well and trust each other's given identity, but not at all as well in a wider spread community with many peers. In fact, perhaps a PKI would already have to be in place for this model to work, rendering another one unnecessary.

Also, how is it decided which peers shall be included in a CA? The CAs are supposed to be completely trustworthy and impartial. But, if the different peers that are part of a CA wish, they could conspire to issue false certificates.

Certification requests must be handled. To whom do end-users apply to when they wish to acquire a certificate? And who has the responsibility for verifying the identity? There are probably models that could be used, for example that certain peers may function as Registration Authorities, but this would require that these peers play an active part in the PKI and do verify the identities. It is far from certain that peers will be willing to spend time and effort on this. Naturally, these peers must be trusted. Otherwise they may request that a CA issue a certificate with a false identity.

Another problem that has to be dealt with here is certificate revocation. What happens if, suppose, one of the peer-group CAs is found to have issued false certificates? These CAs are after all, just groups or peers that can issue certificates to anyone they please. All the certificates earlier issued by this CA should of course be

revoked. The CA most likely won't willingly do so itself, so who should have the authority to revoke these certificates, and how is it to be done? This CA should also most likely no longer be allowed to issue certificates, but who will have the authority to revoke this? Perhaps the peers included in this CA should not be allowed to be part of a CA in the future, since they have proved themselves to be untrustworthy, but how can this be handled? The "key compromise" situation may not be a big problem in this model, at least for CA private keys. But if an end-user's private key should be compromised, it must be possible to revoke certificates with the corresponding public key. Perhaps the X.509 revocation lists could be used, but who is then to issue the revocation list? This is a problem that is best handled centrally, but P2P networks are by nature decentralised.

This model goes against the basic idea of P2P- that all peers are considered equal. A hierarchy of CA's is to be established, but how should this be done? What determines that one peer-group CA is more trustworthy than another? Are the peers included in this CA considered to be more trustworthy than the members of the other CAs? And are peers that are members of a CA considered to be more trustworthy than ones that are not? Quite obviously peers cannot now be considered equal in the manner of P2P.

Obviously, there is a lot of work left to be done before this could be feasible. Models for handling certification requests and certificate revocation must be developed. But by far the biggest problems with is model are that it goes against the basic idea behind P2P- that all peers are considered equal, and that some cryptographic method enabling key-splitting is needed.

3.3. A totally distributed approach based on PGP.

We can think in terms of the decentralised architecture of PGP and try to take advantage of its features. The idea would be that once the user has registered in the P2P application he generates his own key pair and becomes an isolated node. Each node in the graph would represent the Public Key (PK) of each user.

The isolated user can then send certification requests to his friends. Each user of the system is a CA and hence can issue certificates, which is in contrast to the traditional PKI model. A certification request is a request made by the user to others to sign a certificate that this particular public key belongs to him. After verifying that the user owns that public key by some offline operation, the process continues arbitrarily with certificates being issued and revoked and is hence called a distributed PKI Model. There is no central control on the issuing of certificates and hence it is totally distributed. We can conceptualise it as a directed graph of public keys with each directed edge being a certificate issued by a public key to the other as shown in figure 3

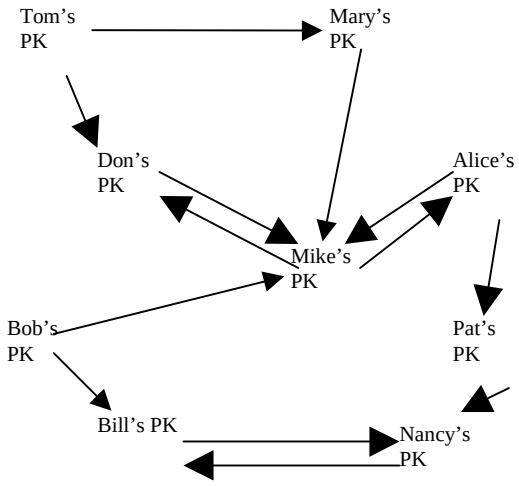

***Figure 3**. Graph of public keys*

Each user can issue new certificates or revoke older ones leading to new edges being introduced or older ones being deleted from the graph. A public key can be verified by using the certification path [7] from the public key of the person who wants to verify to the public key to be verified. These are called certification paths, and will be described in greater detail in the next example.

An example will show that: Antoine gets an electronically signed email from David. To verify the signature of his mail, Antoine needs his public key.

One way to get his key is to send a mail to *pgp-public-keys@keys.pgp.net* with the subject *GET david@tik*. But is this really his public key? Some other person could have created a key with the name David. This person could then send the key onto the key servers.

Antoine has to check the authenticity of this *public key* to see if it is really from David. Antoine could do it searching for another peer who could confirm that the key of David really belongs to David. It is very easy because it can all be done electronically. There is even a tool to do so: the AT&T PathServer. There, Antoine can enter the keyID of his PGP key, which is A. Additionally he enters the keyID of David, which is D. The PathServer gives Antoine the following Figure 4 as a result:

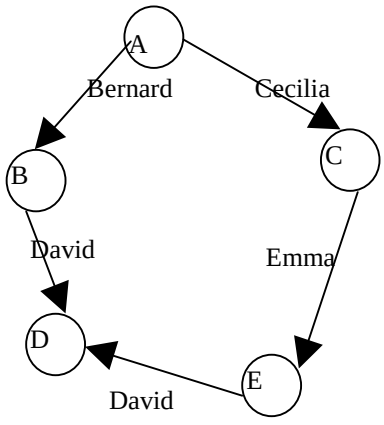

***Figure 4**. Paths from key A to Key D*

The top circle is for Antoine's key, the key at the bottom of the picture is the key of David. An arc from A to B stands for an electronic signature of the key B, done by A.

Antoine can read from the picture: Bernard confirms that the key D really belongs to David, because he has signed David's key. Antoine has signed the key of Bernard and confirms therefore that the key B belongs to him. There is another path from my key to David key via Cecilia and Emma.

To be able to verify such paths it is very important that everyone signs the key of others and submits these signatures to the key servers. This way also others can benefit from such signatures. All these signatures build a kind of a web. That's why this is called the web of trust [8].

The paths between two keys have to be as short as possible. These paths are chains of confirmations. If the path between Antoine's key and David's gets longer Antoine is less sure about the validity authenticity of his key, because the chain of trust is longer. Paths, which do not share a common key between the starting key and the last key, are called disjoint paths. It is important to have as many disjoint paths as possible. The more disjoint paths between two keys the less the probability that someone can fake a confirmation chain by issuing a wrong signature [9]. It is clear that if for example five of my peers assure me that David's key belongs to David, I will trust more than if only one of my peers assures me. It is easy that one of your peers fools you, but it is difficult that five of your peers are fooling you at the same time.

The main problem in these non-centralised security systems is a computational problem, the cost of calculating the certification paths is computationally expensive. Imagine a giant P2P community, to calculate all the certifications paths would be crazy. How many certification paths do we need to establish that the user is who he claims to be? How large is the computational cost? Do we need a quick response about the validity of some key? We have to think of a solution that combine the different responses to these questions. Of course, there will be systems where the more important thing will be to have a high security, and others where the security will be important and also the computational cost and the response time and so on. So according to the needs of a particular system, we have to establish an adequate relation among computational cost, level of security and response time.

On the other hand, we can be optimistic, and thinking about the computational cost will not be a problem due to the exponential growth of processing power and also the increase of the bandwidth in the communications.

6. Conclusion

As we have seen, it is possible to provide a suitable model of PKI in P2P systems, actually we can use the current PKI models but with some restrictions and adding some improvements. The most suitable model for P2P is the decentralized model because it fits perfectly in the distributed nature of P2P. We have studied in this paper how PGP could be a good solution, but of course this solution brings some problems associated with a computational cost as we commented earlier. Also we have seen how a hierarchical model could be used. It would however break in some way the distributed nature of P2P because some kind of centralisation should be introduced in the P2P systems and we would return to the problems of the hierarchical systems: who would be the peer's CAs? Who would control these peer CAs?

A distributed model of PKI like PGP has the complexity associated with its distributed nature, with its advantages and disadvantages. Using it in P2P systems is however possible and in a not too distant future the computational cost problem associated will not be a hinder.

Bibliography

[1] Nash, Andrew, Duane, William et al, *PKI-Implementing and managing e-security*, RSA Press, 2001

[2] Sashi Kiran, Patricia Lareau, Steve Lloyd, "PKI Basics – a technical perspective", http://www.pkiforum.org/pdfs/PKI-Basics-A-technical-perspective.pdf , November 2002

[3] Joel Wise, "Public Key Infrastructure Overview", http://www.sun.com/solutions/blueprints/0801/ publickey.pdf,

[4] Karl Aberer, Anwitaman Datta, and Manfred Hauswirth, "A decentralised public key infrastructure for customer-to-customer e-commerce", *International Journal of Business Process Integration and Management,* http://www.loirpeople.epfl.ch/aberer/GMD-PAPERS/IJBPIM2004.pdf,

[5] David Barkai, "An Introduction to peer-to-peer computing", http://www.intel.com/updates/departments/initech /it02012.pdf

[6] Couch, "Peer-to-Peer File-Sharing Networks: Security Risks", (Sept. 8, 2002).

[7] Elley et al., "Building Certification Paths: Forward vs. Reverse", NDSS'01, http://www.isoc.org/isoc/conferences/ndss/01/2001 /papers/elley.pdf

[8] Germano Caronni, "Walking the web of Trust"

[9] “SPKI/SDSI and the Web of Trust”
http://world.std.com/~cme/html/web.html

[10] Ian Foster, Adriana Iamnithchi, “On Death, Taxes and the Convergence of Peer-to-Peer and Grid Computing.”