Trust Management for E-Commerce[*]

Dr. Audun Jøsang
Co-operative Research Centre for Enterprise Distributed Systems Technology
Faculty of Information Technology
Queensland University of Technology
GPO Box 2434
Brisbane, Qld 4001, Australia
Email: ajosang@dstc.edu.au

Nam Tran
School of Computer Science and Software Engineering
Monash University
900 Dandenong Rd
Caulfield East, Vic 3145, Australia
Email: nam@dstc.monash.edu.au

The work reported in this paper has been funded in part by the Co-operative Research Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Industry, Science & Resources).

ABSTRACT All human interaction is based on trust, meaning that we choose interaction partners and make commitment decisions based on how much we trust the other party. This applies to commerce as well as to e-commerce. In normal commerce, established frameworks, legal and other, provide protection and assurance upon which trust is built. Because e-commerce is largely based on information technology, IT security becomes a crucial trust factor. In fact we claim that a condition for e-commerce to be generally accepted is that the public trusts that appropriate security measures have been taken to protect businesses and consumers from misuse and fraud. This paper discusses management of trust related to IT security in the e-commerce environment.

Security of E-Commerce

E-commerce in open computer networks such as the Internet requires a set of security services in order to counter threats of misuse and fraud. The ***authentication*** security service shall provide proof of identity and thereby prevent an attacker from masquerading as a legitimate user. ***Non-repudiation*** provides proof of expedition or receipt, so that it shall be impossible to falsely claim not having sent or received a digital message. Message ***confidentiality*** shall ensure that only legitimate users can read a message, and message ***integrity*** shall ensure that illegitimate modification, deletion, creation or replay of digital messages does not pass undetected. ***Availability*** shall ensure that an application is not disrupted by illegitimate actions. The security policy defines what is legitimate in every case. In practice security services are usually implemented by means of cryptographic mechanisms and one type of mechanism can often provide several security services. Encryption will for example provide both confidentiality and integrity.

Public-key cryptography is the basis of several important security services such as non-repudiation and authentication and is an essential building block in SSL (Secure Sockets Layer) that is used for securing Web communication. A public/private key pair is used for encryption and digital signature and it is expected that every user and e-commerce player will have its own public/private key pair which will form the basis for the user's or organisation's digital identity in the e-commerce

[*] Appears at Virtual Banking 2000, a virtual conference located at: `http://virtualbanking2000.com`

environment. This requires the secure generation and distribution of potentially hundreds of millions of public/private key pairs, which poses a formidable key management challenge.

Public-key infrastructures (PKI) simplify key management and distribution but create trust management problems. A PKI refers to an infrastructure for distributing public keys where the authenticity of public keys is certified by ***Certification Authorities*** (CA). A ***certificate*** basically consists of the CA's digital signature on the public key together with the owner identity, thereby linking the two together in an unambiguous way. The structure of digital certificates is standardised by the ITU X.509 standard [X509]. In order to verify a certificate the CA's public key is needed, thereby creating an identical authentication problem. The CA's public key can be certified by another CA etc., but in the end you need to receive the public key of some CA, usually called the root CA, out-of-band in a secure way, an various solutions can be imagined for that purpose.

However, there is a problem in this design. What happens if a CA issues a certificate but does not properly check the identity of the owner, or worse, what happens if a CA deliberately issues a certificate to someone with a false owner identity? Furthermore, what happens if a private key with a corresponding public-key certificate is leaked to the public domain by accident, or worse, by intent? Such events could lead to systems and users making totally wrong assumptions about identities in computer networks. Clearly CAs must be trusted to be honest and to do their job properly and users must be trusted to protect their private keys. Trust management includes methods for assessing policies regarding issuance and handling of public-key certificates and for determining whether these policies are adhered to by CAs and users.

Digital certificates and PKIs represent an attempt to mimic real-world human assessment of identity and trustworthiness in an automated and mechanical fashion, but present implementations are based on a very limited trust model making them inadequate as a general tool for trust assessment and decision making. We will first explain how present PKIs are managed.

Web PKIs and Managed PKIs

The cryptographic aspects of PKIs are relatively well understood. The practical deployment of PKIs on the other hand requires management, and so far we have seen the emergence of two types of PKI management.

The Internet uses a particular type of PKI that we will simply call the ***Web PKI***. All root CA public keys are delivered with the Web browsers as ***self signed*** X.509 certificates, i.e. the public key has been certified by the corresponding private key. The only purpose of self certification is to simplify the certificate handling; the application always deals with public keys in the form of digital certificates. Self certification provides no additional trust in the public key, and as such the term "self certification" can be misleading.

Since root certificates are distributed with the browsers they can not easily be upgraded. Root key management must in fact follow the pace of browser releases and distribution. Not only must changes be shipped with the next release of the most popular browsers, the users must continuously upgrade their computers with the newest release. If for example public key revocation shall be useful it must be possible to enforce it relatively rapidly. Because this is not possible for root certificates it is in practice not possible to revoke them.

The most popular application of digital certificates is presently to establish secure Web connections using the SSL protocol. SSL provides confidentiality and server authentication, with client authentication as an option. Another popular application is email encryption and signing. Digitally encrypting the body of email messages with the recipient’s public key provides confidentiality. Digitally signing the message with the sender’s private key provides sender authentication. A third application is to digitally sign SW components in order to authenticate SW component manufacturers. The security problem users are facing regarding active components such as Java applets and Microsoft's ActiveX components is whether such imported programs can be safely executed. One way this can be solved in Web browsers is to have the components digitally signed by the manufacturer's public key, but this only indicates the SW manufacturer's identity and does in principle not say whether it is safe to let the SW component be executed.

Indeed, safe execution of software components requires much more than authentication of the software’s manufacturers and origin. This is a crucial problem with software engineering in general. But it is particularly relevant in e-commerce as software components facilitating electronic transactions on the network are more and more dynamic in nature. In traditional systems, system testing increase the confidence that the system will execute safely. In e-commerce systems, software components come and go dynamically and the opportunity for thorough integration testing lessens if not diminishes.

In contrast to Web PKIs, a so-called ***managed PKI*** does not distribute root public keys piggy-backed with Web browsers, but is based on separate out-of-band procedures managed by the organisation that operates the PKI. This organisation usually operates CA servers from which user certificates can be downloaded. Managed PKIs are operated by an organisation to meet specific needs within the organisation or as a business activity. The organisation will have full control over the trust structure in the PKI hierarchy, but without being Web-born managed PKIs do not easily get global coverage. Managed PKIs can provide high trust and thus be suitable for high value transactions.

Organisations operating managed PKIs can decide to, or be enforced by law in a particular country, to establish cross certification to other managed PKIs and in that way create a PKI consisting of several interlinked certification hierarchies. Secure distribution of the root public key is essential for managed PKIs, and a typical solution is to equip each user with a smart card containing the users private key in addition to the root public key.

Trust Management

There are many ways of describing trust. In the context of e-commerce and IT security we will define trust in principals as the expectation or belief that they will behave according to a given policy and without malicious intent, and trust in systems as the expectation or belief that they are secure and will resist malicious attack. Trust is thus a belief and we assume trust to be based on evidence, experience and perception. In the physical world trust in things and in other people is based on our experience with them, information we have received about them and how they appear to us. All this makes trust a very subjective phenomenon, meaning that I don't necessarily trust the same things or the same people as you and vice versa. The number of people we can potentially relate to within a physical world is also limited by distance and physical constraints. On the Internet on the other hand the number of people we can potentially relate to is only limited by the number of people that are on-line.

The challenge is to be able to make trust assessments in the e-commerce environment, and this boils down to finding methods for receiving reliable evidence about systems and remote transaction partners in computer networks. A transaction partner can be someone you already know but it can also be someone who is totally unknown and with whom you have never interacted before and never will again.

Trust assessment for e-commerce must be based on all the evidence that can be practically collected and should provide a basis for decision making. Trust management can be defined as the activity of collecting, codifying, analysing and presenting security relevant evidence with the purpose of making assessments and decisions regarding e-commerce transactions.

Policy Based Trust Management

The degree to which a relying party trusts the binding between the public key and the owner identity stored in a certificate depends on several factors, including the practices followed by the CA in verifying the identity of the owner, the CA's operating policy, procedures and security controls, the owners obligations e.g. regarding secure storage of the private key, and legal obligations of the CA such as e.g. warranties and obligation limitations.

According to X.509 a certification policy is "a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements" [X509]. To the degree that a certification policy exists or is applicable to a particular application it provides users with evidence for assessing the correctness of the information contained in certificates issued by CAs that adhere to the policy.

Certification policies in the form described by X.509 are not widely used. Instead, a detailed description of the practices followed by a CA is usually found in its Certification Practice Statement (CPS). According to the American Bar Association's Digital Signature Guidelines *"a CPS is a statement of the practices which a certification authority employs in issuing certificates"* [ABA95]. The CPS defines under what conditions certificates are issued and which liabilities the CA takes on itself and which are put on the user. Usually a CA offers different certification classes with varying degree of confidence in the key-to-name binding with the purpose of being suitable for different applications. The general principle is: The higher the confidence, the more thorough the identity verification before issuing a certificate. For commercial CAs it can be added that the price you pay for a certificate increases with the confidence level.

The concepts of certificate policy and CPS were developed by different bodies for different reasons. A CPS is very specific and usually applies to a single CA whereas a certificate policy is more general and is intended to be applicable to larger domains. It is interesting to notice that the Web PKI consists of isolated hierarchies with distinct root CAs at the top. The reason for that is partly that CPSs are hard to compare and make compatible and thereby making cross certification between hierarchies difficult.

It is a common misunderstanding that X.509 certificates express trust in users and organisations. Present certificates only certify the binding between a public key and a name. When issuing certificates a CA is in fact not even expressing trust in the key-to-name binding, the CA merely states that a set of objective pieces of evidence have been collected and verified according to the CPS. In the end it is the relying party who based on evidence that he or she receives who determines a level of trust and decides whether to transact with the supposed owner of that public key. Evidence about the root CA itself is crucial for this purpose, because anybody can issue certificates with syntactically correct contents, but only CAs that are well managed and operated can issue certificates that are semantically correct.

Policy based trust models are useful because they specify which objective evidence has been collected and verified by the CA. However, for this to be meaningful the relying party should really read the CPS, or if applicable the certification policy, every time a secure Web site is accessed. This would require the relying party to read a document of at least 10 pages each time a secure Web site is visited and is therefore hardly ever done. The trust model of the Web PKI is more the result of the need for a simple and objective trust model rather than the need for good trust management. Although present policies only apply to the key-to-name binding they could be extended to include for example credit rating which would provide a basis for trusting principals. This approach is described in e.g. [JY98]. For such policies to be practical they must describe a set of objective criteria regarding credit rating to be verified by the certificate issuer. As far as we know no CA has yet tried to go down this path, but it would provide evidence for assessing the principals themselves and not only the key-to-name binding.

Subjective Trust Management

A purely policy based approach to trust management poses several problems. For example a certificate policy or a CPS is a piece of rather lengthy and complex prose text that can only be read by humans. Although attempts are being made to standardise certification policy classes and components it is difficult to imagine automatic policy assessment by computers. Certificate chaining, either within a hierarchy, or by cross certification between hierarchies, requires equal policies. If it shall be meaningful to trust the root CA in the own hierarchy as a representative of other CAs either within the own hierarchy or in a cross certified hierarchy then the policy strength must equal for all CAs. However, "policy strength" is a multi-dimensional measure that is difficult to compare, and this is one reason why cross certification is difficult to achieve. Real-world trust is intuitively diluted in a chain of recommendations, but present PKI implementations only provide a binary trust model. Also, real world trust is often based on multiple recommendations but present PKI implementations only handle single certification chains. Finally present certification policies do not attempt to say anything about the reliability or credit rating of the owner of a public key. For example PKIs give little support for answering questions such as e.g.: *"For a given certification path and a given user what is the upper transaction value you are willing to risk?"* and *"For a given transaction how can you select the transaction partner that will minimise your risk?"*.

One possible solution to the above mentioned problems is to include subjective measures of trust within a certificate and combine these parameters in order to deduce trust in remote principals and systems. Such a trust measure could represent the policy in addition to the reliability or credit rating of principals. It would be understood by humans and be suitable for automatic processing by computers. In addition, trust dilution could be handled by combining trust measures in series, and trust from multiple certification paths can be combined in parallel. This would for example allow cross certification to be established between arbitrary pairs of CAs by simply specifying within the certificate how much one CA trusts the other. Finally trust measures can be combined with transaction utility functions in order to support decision making.

This type of trust model is for example described in [Jøs99]. A problem related to this model is how users can consistently determine subjective trust measures. Another problem is that the trust measures are communicated to other users and thereby can not be kept confidential whereas a CA might not want to disclose that it distrusts users or other CAs. Finally CAs might not want to make any explicit statement about trust in other CAs as it might imply liabilities. However, these problems can be overcome. Determining trust measures is similar to the problem of determining risk and reliability which is commonly done in many situations. Disclosing subjective trust measures to others is similar to publishing company credit rating which is common practice in the financial world. Finally expressing subjective trust does not need to involve liabilities. If a trust assessment can be objectively judged wrong it simply means that the principal issuing that trust assessment looses credibility, and although it is not a liability issue it can be a serious negative impact. Introducing liabilities as an incentive to issue reliable assessments is therefore not needed.

Trusts in Intentions versus Competence

As mentioned earlier, authentication of manufacturers and origins does not guarantee safe execution of a software component. In conventional trust management, trust usually refers to the expectations and beliefs that a person acts without malicious intent or a system can resist malicious attacks. Certainly, "malicious" is the keyword and that definition is reasonable for the domain of security. However, considering the complimentary aspect of competence or correctness is useful.

A person with perfectly good intention or a software component without malicious behaviours can still break things horribly. This is because the person might not be competent to do the job or the software is not "fit for purpose", i.e. it does not behave maliciously but still not correct. The software could have been written by perfectly honourable and trustworthy people, too.

To provide better security for e-commerce, it is desirable that there are mechanisms to support trusting people's competence and trusting software's correct behaviours. These types of trust do exist in normal commerce. High standards of requirements for professional bodies memberships provide the beliefs and expectations that the members are competent to do their job. In addition, track record of performance also builds up confidence. For example, you might trust some stock broker's advice more than another's because the former has a better track record in making profitable investments. There could be similar mechanisms in e-commerce by associating with people's or organisation's identities more properties describing their competence in doing certain things.

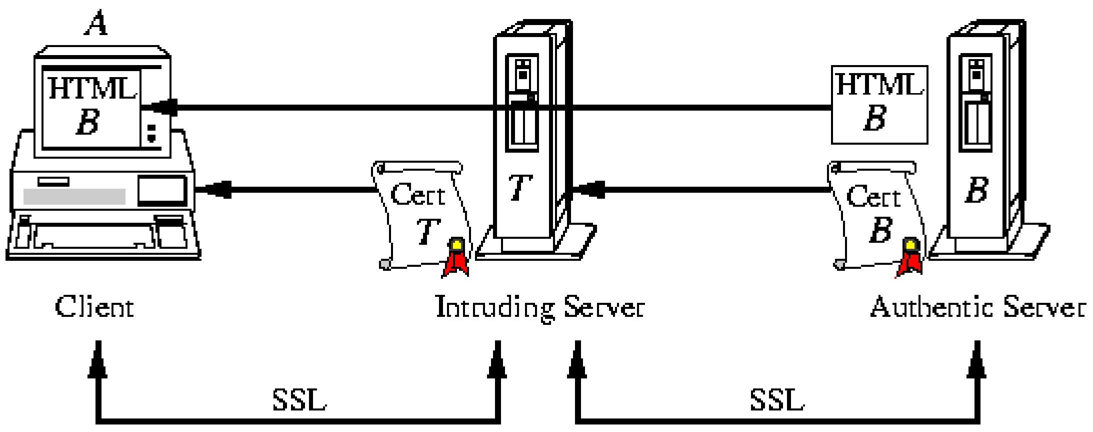
For software, things might be a bit more difficult. Guaranteeing correct behaviours of software has been some kind of an elusive goal for software theorists and practitioners. Full rigor of formalism can provide absolute guarantee of correct behaviours but mostly for more trivial cases and is still too expensive to be useful in general software engineering. However, progress is being made and if software can be guaranteed to behave exactly as specified, then it is possible decide weather to allow or prohibit execution of some piece of software without relying solely on trusting the intention of the manufacturers.

Trusting the System Interface

The special thing about security is that it does not add any functionality to applications, so that users don't necessarily notice when it is broken. It is therefore crucial that the interface provides enough evidence to show that the security is working, and this is very much part of trust management. At present the Web interface provides poor trust management facilities to the user, i.e. it is not able to reliably present the necessary evidence to the relying party in an appropriate form. An example will illustrate this.

Assume a user *A* who wants to access Web services from secure server *B* which for example can be a bank providing on-line financial transaction services to its clients. In the normal scenario, user *A* points his Web browser (client) to bank *B*'s Web site. The Web server returns *B*'s certificate Cert-*B* to *A*'s browser which verifies the certificate using the pre-stored public key of the root CA that generated Cert-*B*. After successful certificate validation *A*'s browser continues the communication with *B* in secure SSL mode.

The figure below shows user *A*'s client machine on the left and bank *B*'s server on the right side. We will show that the intruder *T* in the middle is able to make both *A* and *B* think they are communicating with each other although they in fact communicate with *T*.



In the attack, the intruding Web server *T* acts as a relay between *A* and *B* passing the HTML pages from *B* to *A* and the requests from *A* to *B*. For the attack to work user *A* must initially point her browser to *T* instead of to *B*, and this can happen in various ways. A false URL can for example be placed on a portal until somebody accesses *T* from from the portal in the belief that he or she accesses *B*. After a successful attack, the false URL can be removed in order not to leave any evidence of where the attack came from. Another more serious threat is that false certificates can be inserted into somebody's browser by means of active contents such as Java applets and ActiveX components. Active contents could also be used to modify a user's personal bookmarks and place false URLs on the person's browser.

As soon as client *A* has established a SSL connection to the intruding server *T* using *T*'s certificate Cert-*T*, the intruding server sets up a SSL connection to bank *B* using the bank's certificate Cert-*B* and simply relays the data sent by *A* and *B* to the opposite sides via two different SSL connections, including possible user passwords, so that *A* and *B* think that they are communicating with each other. When *A* sends a request to transfer money from her own account for paying a bill, *T* is able to modify the destination account number and the amount.

When a secure SSL connection is established the server is supposed to be authenticated by the client, as indicated by the key or the padlock icon on the browser window. However, this only indicates that something is authenticated and not what in particular, which for all practical purposes means that nothing at all has been authenticated.

The browser does allow viewing a certificate by clicking on the padlock icon, but users hardly ever do this, and even security aware users who view the certificate when accessing a secure Web site can have difficulty in judging whether the information on the HTML page has been sent by the owner of the certificate. The browser does also check that the domain name in the certificate is the same as the domain name pointed to by the browser, and aware users can notice that the intruders domain name is different from the expected domain name of the bank. However, users do not usually inspect the URL for the domain name when browsing the Web. Distinct domain names can appear very similar, for example differing only by a single letter so that a false domain name may pass undetected.

This example shows that on-line banking applications based on Web interface alone are vulnerable to this type of man-in-the-middle attack. Such applications should always use additional security features in the form of special SW on the client side, mutual authentication between server and client and user dynamic password generators etc.

Active contents provides a great potential for applications but poses a serious security threat because it makes the integrity if the Web browser vulnerable to attack. Digitally signing applets is not a good solution because users are required to accept or reject received active contents purely based on the identity of the company that designed it. These companies will usually be unknown so that users are unable to make an informed decision. Most users will accept active contents anyway because they want the functionality. The only good solution is to restrict what active contents can do by mechanism, such as the Java Sandbox which has been used in the Netscape browser for years. In present releases however, the Netscape browser allows digitally signed applets to exit the Sandbox and for example access system files, in the same way that Microsoft's ActiveX components have always done in the Internet Explorer.

The system interface should make it easy for users to view, understand and believe in security related evidence. This becomes even more difficult to achieve when new devices such as WAP terminals are being introduced due to their present primitive interface. A suggestion for improvement is to include the server company logo in digital certificates and let the interface always display the logo when a secure connection is established. This will allow a quick visual check anytime without extra mouse clicks in order to verify the server identity. Alternatively an audible message can be included in the certificate and played to the user. However, users might not pay any attention to these features after a while, so that the problem of user awareness will not go away. All we can do is to create a system interface that helps users to be more aware.

Conclusion

Trust management can be defined as the activity of making trust assessments by collecting, codifying, analysing and presenting security relevant evidence, with the purpose of making assessments and decisions. Presently there is no systematic and reliable way of obtaining evidence about systems and potential transaction partners in the e-commerce environment, and present PKI implementations need to provide better trust management facilities. Better still, the evidence should include, in addition to belief and expectation of intent, proof of competence and correct behaviours. The integrity of the system interface is particularly important because it is through the interface that evidence ultimately is presented and trust is created.

References

[X509] ITU. *Recommendation X.509, The Directory: Authentication Framework*. International Telecommunications Union, Telecommunication Standardization Sector(ITU-T), 1996.

[ABA95] ABA. *Digital Signature Guidelines: Legal Infrastructure for Certification Authorities.* American Bar Association, 1995.

[Jøs99] A.Jøsang. *Trust-based decision making for electronic transactions.* In the Proceedings of the Fourth Nordic Workshop on Secure Computer Systems (NORDSEC'99). Stockholm University, Sweden, 1999.

[JY98] M.Jakobsson and M.Yung. *On Assurance Structures for WWW commerce*. In Proceedings of Financial Cryptography 98, Springer, 1998.