The Importance of Trust in Computer Security

Christian Damsgaard Jensen

Department of Applied Mathematics & Computer Science
Technical University of Denmark
DK-2800 Kgs. Lyngby, Denmark
Christian.Jensen@imm.dtu.dk

Abstract. The computer security community has traditionally regarded security as a "hard" property that can be modelled and formally proven under certain simplifying assumptions. Traditional security technologies assume that computer users are either malicious, e.g. hackers or spies, or benevolent, competent and well informed about the security policies. Over the past two decades, however, computing has proliferated into all aspects of modern society and the spread of malicious software (malware) like worms, viruses and botnets have become an increasing threat. This development indicates a failure in some of the fundamental assumptions that underpin existing computer security technologies and that a new view of computer security is long overdue.

In this paper, we examine traditional models, policies and mechanisms of computer security in order to identify areas where the fundamental assumptions may fail. In particular, we identify areas where the "hard" security properties are based on trust in the different agents in the system and certain external agents who enforce the legislative and contractual frameworks.

Trust is generally considered a "soft" security property, so building a "hard" security mechanism on trust will at most give a spongy result, unless the underlying trust assumptions are made first class citizens of the security model. In most of the work in computer security, trust assumptions are implicit and they will surely fail when the environment of the systems change, e.g. when systems are used on a global scale on the Internet. We argue that making such assumptions about trust explicit is an essential requirement for the future of system security and argue why the formalisation of computational trust is necessary when we wish to reason about system security.

1 Introduction

Most security models and policies, and the technologies needed to support these models and enforce these policies, are based on security abstractions that have emerged in the context of military security or centralized (corporate) computing environments. Common for these environments is that there is a single authority to define and enforce security policies and to punish transgressions. During the past 20 – 30 years, these abstractions have been extended to cover local area networks, corporate intranets, virtual private networks and virtual organisations, but most of these extensions only work in environments where there is a single explicit root of authority for security enforcement. This authority is rooted in criminal-, civil- or military law, which requires appropriate

J. Zhou et al. (Eds.): IFIPTM 2014, IFIP AICT 430, pp. 1–12, 2014.

and effective enforcement agencies or in the local security organisation within a corporation or a joint entity, as part of a virtual organisation, where the scope and responsibilities of this entity is governed by a contract and where conflicts will be resolved through an agreed legal framework.

In general, the root of authority is responsible for defining security policies that are interpreted by security mechanisms according to the underlying security model. These security policies can be interpreted as both explicit and implicit statements of trust in the different entities in the system. Moreover, the security mechanisms that enforce these security policies implemented in software and hardware and frequently rely on information from other subsystems, which raises further trust issues. In order to understand the importance of trust in computer security, we need to examine the relationships between the root of authority and all other entities in the system as they are enforced by security mechanisms that interpret security policies according to the underlying security model. We therefore need to understand the most common security models, policies and mechanisms and the way that they are implemented and enforced in practice.

The notions of security models and policies are used inconsistently in the literature, so, in order to facilitate our discussion, we start by presenting our definition of a security model, -policy, and -mechanism. A *security model* is an abstract specification of the desired security properties of a system, which defines abstract representations of all security relevant entities and specifies the rules that govern their relationships. A *security policy* defines a mapping between the (abstract) security model and the (concrete) entities in a given system, either directly through permissions, e.g. specifying allowed operations for user *u* on file *f*, or indirectly through inference rules, e.g. defining dynamic rules that will resolve to *allow* or *deny*, when instantiated with the specific values of user *u* and file *f*. Finally, a *security mechanism* is the set of functions in the underlying system that is responsible for interpreting and enforcing the security policy according to the security model.

Security models are commonly divided into two main classes: information flow models and access control models. The information flow models aim to control the flow of information inside a system, so that protected information will only be able to flow to authorised locations (a location may be a variable, a subsystem, or a specific device, depending on the granularity of the information flow analysis). Information flow models are commonly used with formal methods to prove that the system consisting of software and protocols conform to the formally specified security policy [9]. These formal proofs generally require complete knowledge about all possible information flows (variable assignments, method invocations, message transmissions, etc.) in the system, which requires access to the source code of all the software and all the tools that have ever been used to build and verify the system.[1] The access control model aims to control the way that passive entities, such as information and other resources (e.g. input from external devices) managed by the system, can be accessed by active entities, such as running programs, subsystems or external processes. The access control model needs the ability to enumerate all active and passive entities in the system and to associate policies with all requests by the former to perform operations on the latter. The first

[1] In his 1984 Turing Award lecture, Ken Thompson illustrates how tools, such as a compiler, can be abused to generate malicious code from well behaved source code [20].

requirement is trivial because the system needs identifiers to correctly manage all system entities anyway. The second requirement can be met in a number of different ways, but the conceptually simplest way to do this is through an access control matrix [15] that defines a row r_i for each active entity i and a column c_j for each entity j, which includes both active and passive entities in the system, and where the authorised operations, such as read, write, append, call, etc., for active entity i on entity j is encoded in the access control matrix in position (i,j) – if there are no authorised operations for entity i on entity j the position is simply left empty. This access matrix model has some limitations, [14] and it is cumbersome and error prone to specify security policies, so higher level access control models, such as the Bell & LaPadula confidentiality model [5], the Biba integrity model [6] and Role-Based Access Control [11,18], have emerged to facilitate the specification and evolution of access control policies. Access control policies are normally enforced by a special component, called a *guard* or a *reference monitor*, implemented in each of the subsystems that manage the entities, e.g. in each of the file servers that manage the files in a distributed file system. It is important to note that the access control model specify policies that control the interaction between entities, which means that the security properties are profoundly different from those obtained through the information flow model. On the one hand, the access control model requires no knowledge about the implementation or the internal state of any of the components in the system, which means that it can be much more generally applied, but on the other hand, it provides no guarantees about the confidentiality or integrity of data once access has been granted, e.g. any entity that has been authorised to perform certain operations on another entity according to the access control policy may, in principle, provide a proxy service for this operation to all other entities (both authorised and unauthorised). Security properties achieved through the access control model are therefore contingent on the behaviour of all authorised entities, i.e. that all active entities can be trusted to help enforce the security policy.

In practice, the necessary access to source code or formal specifications for all software and hardware used to implement a system as well as formal proofs that the running system conforms to specifications, means that information flow models are primarily used in high security systems, such as military information systems, or some parts of avionics and automotive control systems. The common need to dynamically install and update software without access to source code or complete specification, means that most commercial computer systems are protected by policies and mechanisms based on the access control model.

In the rest of this paper, we examine the access control model in more detail and discuss some of the trust issues that arise when this model is used. Section 2 examines the access control model and identifies some of the implicit trust assumptions that emerge as a consequence of the way that we specify and enforce access control policies in computer systems. In Section 3, we examine some of the fundamental trust issues that relate to the technical components in the system. This is followed, in Section 4, by an examination of the implicit trust assumptions relating to societal factors in the environment where the system operates. Finally, we present a summary of our discussions, present our conclusions and outline interesting directions for future work in Section 5.

2 Access Control

Access control mechanisms are designed to prevent unauthorised users from accessing resources managed by the system and to limit the access of authorised users to exactly those operations allowed by the access control policy. Access control policies are normally divided into two main classes, mandatory- and discretionary access control policies. The mandatory access control policies define system wide rules that cover all entities in the system and are normally centrally defined, i.e. it is not possible for an active entity to make any modification to the system that violates these policies. Discretionary access control policies allow active entities to define or change certain aspects of the access control policies, e.g. a user u_i may grant access rights to a file f that u_i has created to another user u_j. The objective of most mandatory access control policies are to provide strong security guarantees, similar to the ones that can be obtained through the information flow model, and they are often quite restrictive and cumbersome to use. In practice, this means that most current computer systems used outside high security environments are protected by discretionary access control policies. As the specification of discretionary access control policies are, at least in part, left as an exercise to the users, there is an obvious element of trust vested in those users.

The enforcement of an access control policy relies on a *reference monitor*, which is a central component that mediates all access to entities managed by the system – the active entities are normally called *subjects* and the passive entities are normally called *objects*.[2] When objects reside on several nodes in a distributed system, then the access control mechanism must have a reference monitor on each of these nodes. A simplified view of the access control mechanism is shown in Figure 1.

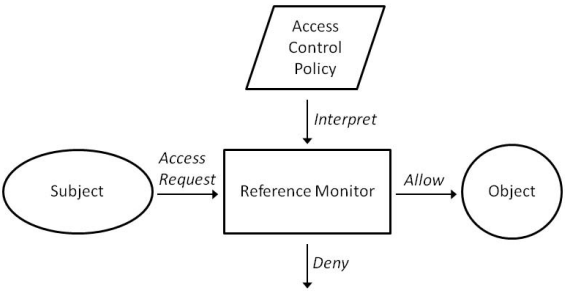

Fig. 1. Access Control Model

The figure shows a subjects which requests to access an object in a particular way. The reference monitor verifies that the requested operation conforms to the access control policy and allows the operation to proceed if it does; otherwise the request is denied. This presentation of the access control model is fairly simple and easy to understand,

[2] Subjects may also be considered objects in operations by other subjects, e.g. when one process starts or stops another process.

but it omits several important details. First of all, there are a number of technical issues that must be considered; we refer to these as technical trust issues. The reference monitor is a system entity, which is only able to interact with other system entities, such as processes running on the same computer. Access control policies, however, normally employ a higher level of abstraction, e.g. access rights are granted to human users or roles within an organisation, so the system needs an unambiguous way to represent this information and associate it with specific system entities. This is typically achieved by requiring users to "log in" to systems which associates their unique identifier (UID) with all processes that they run. The access request and the user credentials, e.g. the UID, must be communicated to the reference monitor using a secure channel that ensures the integrity and authenticity of the request. Similarly, the reference monitor must have a secure channel to read the access control policy, which must be correctly managed as a protected object in the system. We examine these technical issues in greater detail in Section 3. Second, the access control mechanism has little power to constrain the authorised subject's use of the object resources once access has been granted. Such constraints are generally imposed by elements outside the system, which we, for lack of a better word, refer to as social trust issues. These issues are examined in greater detail in Section 4.

3 Technical Trust in Security

Traditional access control policies rely on the authenticated identity of the user requesting to access a particular resource. This means that the system has to establish the identity of the user (this is known as identification) and verify that the user really is who he claims to be (this is known as authentication). The identification and authentication is normally done through the "log in" process mentioned above, where the user provides a username (identification) and a password (authentication). When the user has logged in, the UID will be associated with all processes that the user runs and can be transmitted to other systems that manage objects that the user wishes to access. Before the user can log in to the system, he has to be enrolled, i.e. an account has to be created for the user. The enrolment serves two purposes, it allocates a unique UID for the user and records the information (password or biometrics) that will be used to authenticate the user, and it defines the initial access control policy for the user; this second step is known as authorisation.

When the user has been enrolled and and has logged in to the system, he can start programs that may eventually request to access protected resources in the system. This requires the subject to forward its credentials along with the access request to the reference monitor. If the object resides on the same host as the subject, access requests are normally made through the system call interface, but when the object resides on a different computer, the subject normally submit credentials that have been protected by some cryptographic mechanism to the reference monitor. In the latter case, the verification of the cryptographic credentials normally require the help of a trusted third party, such as a key distribution center (KDC), like the authentication- and ticket granting service employed in Kerberos [16], or a Public Key Infrastructure (PKI), e.g. based on X.509 [17] or SPKI [10] certificates or even anonymous credentials [8].

These technical issues are illustrated in Figure 2 and will be discussed in greater details in the following.

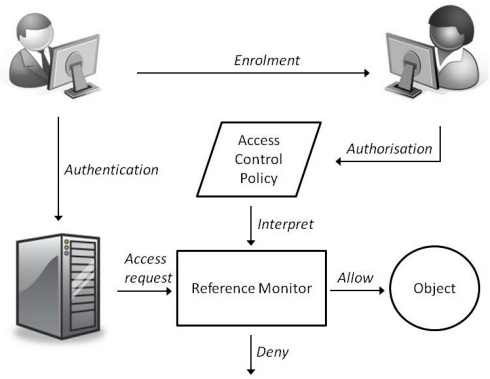

Fig. 2. Access Control in Practice

3.1 Enrolment

There are two types of enrolment, enrolment in the system, i.e. the account creation mentioned above, and enrolment in the organisation. Enrolment in the system is normally the only type of enrolment necessary for personal or stand-alone systems, while some form of enrolment in the organisation is necessary when users register on a website, subscribe to services or get hired by an organisation – in most cases, successful enrolment in the organisation is followed by enrolment into the systems managed by the organisation.

Part of the enrolment process is to verify the identity of the user and to establish whether the user is sufficiently trusted to be enrolled at the desired level in the organisation. If we consider the case of a new employment, the human resources (HR) department will examine the user's degree diplomas and diplomas from other courses along with the user's employment history in order to determine whether the user is competent. The HR department will also perform a background check, which may include the user's financial situation, affiliation with contentious organisations, a possible criminal record and frequently also a psychological assessment, in order to determine the trustworthiness of the potentially new employee. This illustrates an interesting point about the enrolment in organisations. In most cases, a comprehensive and meticulous trust evaluation is performed by the HR department, which help select the most suitable applicant. The final decision, to hire a person is binary, i.e. the trust evaluation is only used to decide whether a person should be allowed to pass the security perimeter and become part of the insiders.

3.2 Authorisation

Once the user has been enrolled in the organisation, the HR department will initiate the enrolment into the system and define the authorisations of the user, i.e. specify the

initial access control policy for the user. This policy is primarily based on the functions that the person is hired to perform and will grant necessary access to all objects that the user may need to function in his new job. This illustrates an interesting point about the enrolment process and the authorisation of users. Despite the comprehensive and meticulous trust evaluation is performed by the HR department, the authorisation is primarily based on the functions that the person is hired to perform.

3.3 Authentication

Authentication is a binary process that, if it succeeds associates the user's UID with the subject. While the authentication community accepts that there may be some uncertainty associated with the authentication, e.g. passwords may be easy to guess [7,1] and biometric authentication operates with explicit notions of false acceptances and false rejections, the access control community generally assumes that the authentication mechanism is perfect, i.e. the access control mechanism has blind trust in the authentication mechanism.

In a distributed system, where users are authenticated on remote computers, the user credentials must be transported securely from the host where the user has authenticated to the host where the object resides and where the reference monitor will be invoked to grant or deny access to the object. As mentioned above. this is typically done using a trusted third party (TTP), such as a KDC or a PKI. In the case of a KDC, the party that authenticates the user will typically also manage the keys required to secure the credentials, i.e. the reference monitor only need to consider a single TTP. In the case of a PKI, the reference monitor has to trust the local authentication server and the hierarchy of certificate authorities (CAs) that are necessary to validate the signature of the local authentication server. Moreover, the subject and the object need to share a common trusted root CA, which raises interesting problems in large-scale open dynamic systems. In particular, the failure of the Dutch national CA DigiNotar [19] has demonstrated the brittleness of PKI based authentication infrastructures.

3.4 Access Request Verification

Access requests from the local node are normally passed to the reference monitor in system calls, which can be considered a secure mechanism. If subject and object reside on different hosts, it is not only necessary to consider the security of the credential transport mechanism, as discussed above, it is also possible that the host where the subject runs has been compromised while the host where the object resides remains secure. We therefore need to examine the different elements that determine the security status of the subject.

First of all, the computer hardware must comply exactly to specification. While incorrect implementation or missing functionality will impact all process and therefore quickly be discovered, but there is a rising concern about extra functionality, such as back doors, being built into computer and network equipment hardware [22]. Second, the operating system must not be compromised, which requires both the protection of the BIOS, the boot loader and the entire operating system code and the boot process to have executed correctly. While there have been some efforts to guarantee the integrity

of the bootstrap process [21,2,4], these efforts have had little impact on common computer systems. Moreover, recent leaks by Edward Snowden suggest that the U.S. National Security Agency (NSA) is able to corrupt parts of the computing industry [12], which raises important issues about the trustworthiness of the underlying computing platforms. Finally, the user program must be correct in the sense that they do only request the necessary resources and preserve the confidentiality and integrity properties defined in the security policy.

This means that, when the reference monitor grants a subject access to an object, it has to trust the hardware of the subject host, the integrity of the BIOS, boot loader and operating system on that host, the system administration procedures, including the boot sequence, software updates etc., of the subject host and the software that the user executes and which requests the access on behalf of the user. Moreover, the reference monitor must trust all the same elements of all the hosts in the transitive closure of the chain of trusted third parties used to attest and certify security properties in the systems. These trust aspects are not adequately covered by current security models, where the security of external components are generally implicitly assumed. Research has been carried out to address some of these issues, but this work is fragmented and there is no comprehensive model that captures all relevant trust aspects.

Finally, the behaviour of users is normally considered external to the system security models, which makes it difficult to reason about the practical security of a computer systems. The importance of considering human behaviour as part of the system security model is probably best illustrated by the recent leaks of classified information by Chelsea (Bradley) Manning and Edward Snowden. When the U.S. Army and the NSA, considered two of the most security conscious organisations in the world, fail to accurately assess the trustworthiness of enrolled users, what can smaller organisations with fewer resources and less attention to security hope to achieve.

4 Social Trust in Security

In the previous sections, we examined many of the current security abstractions and identified a number of areas, where the security of the system depends on trust vested in different system components and, more importantly, in the users. In the following, we examine some of the factors that really constrains users and prevent them from violating the security policy and compromise the security of the system. The factors that we believe are among the most important constrains on human users are: Management, influences from colleagues and social norms in the society, the prevalent morals, ethics and sometimes religion in a society, legislation and the effective enforcement of laws, these factors are shown in Figure 3.

A more intricate understanding of the effects of these factors must draw a broad set of disciplines including psychology, sociology, philosophy, religion, economics and law. In the following, we present a brief outline of some of these effects as we understand them.

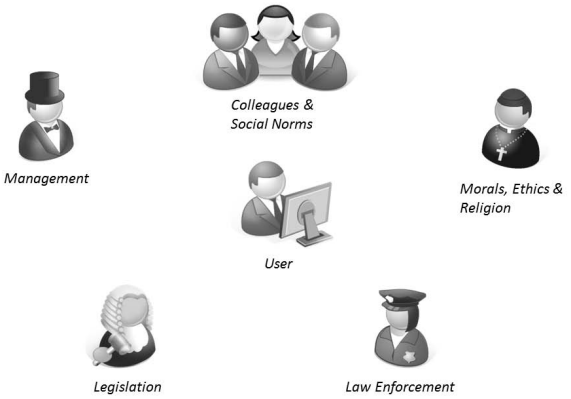

Fig. 3. Real Constraints on Human Behaviour

4.1 Management

Management is the root of authority for the work on security in the organisation, i.e. it defines the overall security policies, determines the consequences for policy violations and allocates resources for the enforcement of the security policies. By specifying the security policies and communicating them to all users of the system, management defines the boundaries for acceptable behaviour, which most users will respect. In many cases, the authority of management is derived from some form of contract, such as an employment or service provision contract, between the management and the user. If the user violates the terms of this contract, different forms of sanctions may be put in place, ranging from an admonition to termination of the contract. Fear of possible sanctions will constrain users and help promote desired behaviour in the user population only when the enforcement is seen as effective, i.e. the risk of discovery is high and the sanctions are carried out indiscriminately.

4.2 Colleagues and Social Norms

Humans are social animals, so most users will try to adapt their behaviour to the social norms in the environment. When other users set good examples, most users will do the same from a desire to "fit in". If, on the other hand, there is a common disregard for management and disrespect for the rules and policies, many people will pay less attention to those rules and policies. The pressures exercised through social norms are often situational, which means that changes in the environment may result in changes of what is seen as accepted behaviour. This means that organisations with a high churn rate can rely less on organisational culture and positive reinforcement from the behaviour of other users. The contextual effects of social norms are in many ways common to the more long term constraints imposed by morals, ethics and in many cases religion.

4.3 Morals, Ethics and Religion

A person's morals and values are predominantly shaped by the ethical and religious values of the surrounding society in which that person grows up. They are normally long lived and in many cases independent of the context, e.g. immigrants will often retain many of the core values of the countries that they come from. In largely homogeneous societies, computer users tend to share the same fundamental values, which often makes it easier to define and explain security policies, because management can harness commonly accepted notions of right and wrong. Such notions are often, to some degree, encoded into the security policies and the specification of the access control policies. Such tacit assumptions about the reasonable behaviour of computer users, however, mean that access control policies defined in one environment may work poorly in another environment, because some fundamental values are different, so computer users may behave in different ways, e.g. in Sweden, tax returns are considered public information and are therefore readable by everyone, but in most other countries such information would be considered private and therefore kept confidential.

In increasingly open and global societies, security policies will be defined in one environment, but the objects may be accessed by subjects that execute in a completely different environment. This makes it challenging to define security policies, because it becomes necessary to reason about the context in which the policy is interpreted as well as the context in which it is defined.

4.4 Legislation

Similar to the effects of morals and ethics, different societies have developed different laws to deal with many of the same issues. Security policies are normally extensions of the legal framework, i.e. they specify rules that are not explicitly covered by legislation, but this relationship between legislation and security policy is often implicit.

Tacit notions of what is legal and illegal are often important when security policies are defined, but particular legislation, such as copyright protection, data protection and criminal law, may be very different in different countries, e.g. charges had to be dropped against the Philippine authors of the "I Love You" virus in 2000, because there were no laws in the Philippines against writing malware at the time [3].

The differences in legislation makes it necessary to thoroughly understand the legal framework in which the security policy is to be enacted and the provisions in law that facilitate the enforcement of security policies.

Legislation and security policies defines the limits of acceptable behaviour, so well-behaved users will know when they behave well, but the full effect of the law is only achieved through credible enforcement and effective sanctions.

4.5 Enforcement

Laws and policies that are not effectively enforced become declaration of intent that people observe when it is convenient. Effective enforcement of policies requires that the enforcement agency is equipped with sufficient resources and powers to carry out this task. It is, however difficult to quantify security and to demonstrate the direct benefits

of spending on security, so many security agencies operate with fewer resources than they feel that they need.

It is generally simple for a well resourced security agency to enforce policies inside an organisation, but all security agencies are limited by jurisdiction, which often prevents them from directly pursuing outside threats, e.g. trace hackers across multiple networks in different countries.

Unless the agencies that are put in place to enforce the security policies offer a credible threat of detection and unless violations are rigorously sanctioned, users tend to ignore rules and policies that they do not understand and which get in the way of the immediate task.

5 Conclusions

In his thought provoking keynote at the First ERCIM Workshop on Security and Trust Management, Dieter Gollmann explained "why trust is bad for security".[3] The main thrust of his argument is that the concept of trust is extremely broad and ambiguous. In particular, he pointed out that the word "trust" has often been misappropriated by the computer security community to describe technologies that provide neither trust nor security in any great way, e.g. trusted computing base, trusted third parties, trusted computing, etc. As such, his arguments are very similar to the critique of technical trust technologies that we presented in Section 3, but we argue that most existing "hard" security abstractions are really founded in trust. This may be trust in the implementations of hardware and software, the technical staff that maintain the systems, the users who run programs, the staff that enrol and provision these users or external factors, such as societal norms, legislation or "the system's" ability to enforce rules and sanction violations.

In our view, we need to make trust a first class citizen of our security models if we are to successfully reason about the security of global computing systems. This requires development of formal models of trust, both qualitative and quantitative, and new ways of reasoning about security which take these models into account.

References

1. Password recovery speeds, `http://www.lockdown.co.uk/?pg=combi` (visited April 15, 2014)
2. Arbaugh, W.A., Farber, D.J., Smith, J.M.: A Secure and Reliable Bootstrap Architecture. Tech. Rep. MS-CIS-96-35, University of Pennsylvania,, School of Engineering and Applied Science, Computer and Information Science Department, Philadelphia, Pennsylvania, U.S.A (1996)
3. Arnold, W.: Technology; Philippines to Drop Charges on E-Mail Virus. The New York Times (August 22, 2000), `http://www.nytimes.com/2000/08/22/business/technology-philippines-to-drop-charges-on-e-mail-virus.html` (visited April 15, 2014)

[3] A paper summarizing these thoughts are published in ENTCS [13].

4. Balacheff, B., Chen, L., Pearson, S., Plaquin, D., Proudler, G.: Trusted Computing Platforms - TCPA Technology in Context. Prentice Hall (2003)
5. Bell, D.E., LaPadula, L.J.: Secure Computer Systems, Vol. I. Mathematical Foundations and Vol. II. A Mathematical Model. Tech. Rep. MTR-2547, The MITRE Corporation (1973)
6. Biba, K.J.: Integrity Considerations for Secure Computer Systems. Tech. Rep. MTR-3153, The MITRE Corporation (1977)
7. Bonneau, J.: The science of guessing: analyzing an anonymized corpus of 70 million passwords. In: IEEE Symposium on Security and Privacy, San Francisco, CA, USA (May 2012)
8. Brands, S.A.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge (2000)
9. Denning, D.E.: A lattice model of secure information flow. Commun. ACM 19(5), 236–243 (1976)
10. Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T.: SPKI Certificate Theory. Tech. Rep. RFC 2693, Internet Engineering Task Force (IETF) (September 1999)
11. Ferraiolo, D., Kuhn, R.: Role-based access control. In: In 15th NIST-NCSC National Computer Security Conference, pp. 554–563 (1992)
12. Ferranti, M.: Report on NSA 'secret' payments to RSA fuels encryption controversy. PC World (December 23, 2013), http://www.pcworld.com/article/2082720/report-on-nsa-secret-payments-to-rsa-fuels-encryption-controversy.html (visited April 15, 2014)
13. Gollmann, D.: Why trust is bad for security. Electron. Notes Theor. Comput. Sci. 157(3), 3–9 (2006)
14. Harrison, M.A., Ruzzo, W.L., Ullman, J.D.: Protection in operating systems. Commun. ACM 19(8), 461–471 (1976)
15. Lampson, B.W.: Protection. In: Proceedings of the 5th Princeton Conference on Information Sciences and Systems (1971)
16. Neuman, C., Yu, T., Hartman, S., Raeburn, K.: The Kerberos Network Authentication Service (V5), RFC 4120 (July 2005)
17. Neuman, C., Yu, T., Hartman, S., Raeburn, K.: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks, Recommendation X.509 (October 2012)
18. Sandhu, R.S., Coyne, E.J., Feinstein, H.L., Youman, C.E.: Role-based access control models. Computer 29(2), 38–47 (1996)
19. The Dutch Ministry of the Interior and Kingdom Relations: DigiNotar CA certificates will be revoked on September 28 2011 (September 2011), http://www.logius.nl/english/news-message/titel/diginotar-ca-certificates-will-be-revoked-on-september-28-2011/, (visited April 15, 2014)
20. Thompson, K.: Reflections on trusting trust. Commun. ACM 27(8), 761–763 (1984)
21. Wobber, E., Abadi, M., Burrows, M., Lampson, B.: Authentication in the taos operating system. ACM Trans. Comput. Syst. 12(1), 3–32 (1994)
22. Wolf, J.: U.S. lawmakers seek to block China Huawei, ZTE U.S. inroads. Reuters (October 8, 2012), http://www.reuters.com/article/2012/10/08/us-usa-china-huawei-zte-idUSBRE8960NH20121008 (visited April 15, 2014)