# Secure Information Sharing Enabled by Trusted Computing and PEI Models

Ravi Sandhu
*George Mason University and TriCipher Inc., USA*
sandhu@gmu.edu

Kumar Ranganathan
*Intel System Research Center Bangalore, India*
kumar.ranganathan@intel.com

Xinwen Zhang
*George Mason University, USA*
xzhang6@gmu.edu

## ABSTRACT

The central goal of secure information sharing is to "share but protect" where the motivation to "protect" is to safeguard the sensitive content from unauthorized disclosure (in contrast to protecting the content to avoid loss of revenue as in retail Digital Rights Management). This elusive goal has been a major driver for information security for over three decades. Recently, the need for secure information sharing has dramatically increased with the explosion of the Internet and the convergence of outsourcing, offshoring and B2B collaboration in the commercial arena and the real-world demonstration of the tragic consequences of lack of information sharing in the national security arena. As technology has made the "share" aspect ever easier so has it increased the difficulty of enforcing the "protect" aspect. The central contribution of this paper is to show that the emergence of industrial strength Trusted Computing (TC) technology offers a range of novel solutions to the long-standing problem of secure information sharing. To this end we introduce a new framework of three layered models to analyze requirements and develop solutions, and demonstrate the application of this framework in context of TC and secure information sharing. The three layers are policy models (topmost), enforcement models (middle), and implementation models (bottom). Hence the name PEI models. At the policy model layer the secure information sharing space is divided into three categories called password based, device based, and credential based. For each of these policy categories various enforcement and implementation models can be developed. While we believe the PEI framework is relevant to security problems beyond secure information sharing, our goal in this paper is to demonstrate its application in this particular arena and identify questions for future research in this context. An essential benefit of PEI is that the three layers allow us to focus on the more important issues at a higher level of abstraction at the policy and enforcement layers, while leaving deep detail to the implementation layer. This paper focusses on the policy and enforcement layers with only passing mention of the implementation layer.

## Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection—*Access controls*; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Unauthorized access*

## General Terms

Security

## Keywords

secure information sharing, trusted computing, access control, authorization, security framework, PEI models


*ASIACCS* '06 March 21-24, 2006, Taipei, Taiwan.


## 1. INTRODUCTION AND BACKGROUND

The quest for secure information sharing has been a central but elusive goal for information security for over three decades. The stumbling block is simple to understand but difficult to solve. Digital information is easy to copy and transport, and read access to any copy is as good as read access to the original.[1]

There are fundamentally two overall approaches to information sharing. (For simplicity, we will henceforth understand information sharing to mean secure information sharing.) The first and more traditional approach allows information to be copied from one object to another but only if the copies are protected at least as strongly as the original. This is the approach of mandatory access control and some forms of originator control as discussed below. The second and more recent approach focusses on preventing the ability to copy information. The content of protected objects is rendered for an authorized user in a carefully controlled and confined environment where the user or other software on the same computer is unable to make copies of the rendered information. In this approach it is assumed that the protected object is encrypted in such a manner that access to copies of the protected object does not allow access to the cleartext (or more generally clearmedia) content except to authorized users. This is the approach of trusted and confined viewers, and is the focus of this paper.

[1]We distinguish secure information sharing from retail Digital Rights Management (DRM). Secure information sharing seeks to protect unauthorized disclosure of sensitive information. Retail DRM is concerned with protecting content (typically entertainment-oriented content) for purpose of protecting loss of revenue. In retail DRM it is often acceptable to relax controls on appropriately degraded versions of the content, such as analog recordings versus digital recordings or low resolution versus high resolution images. In secure information sharing loss of quality or resolution is by itself typically insufficient to allow controls to be removed or relaxed because the essential sensitive content still comes through. A thorough discussion of the difference between secure information sharing and retail DRM is beyond the scope of this paper but we believe the central distinction is captured in this footnote. Some discussion on this issue is available in [24, 39].

## DAC, MAC and ORCON

Historically there have been four specific approaches to information sharing. The first of these is classical discretionary access control (DAC) wherein it is clear that the problem has not been solved at all because DAC does not correlate controls on copies of information with controls on the original.[2] This is, of course, well known but we include DAC in our list since it has historically been proposed for this purpose and much literature on the topic of controlling read access has been published. Also DAC attempts to enforce controls on sharing information at the discretion of the "owner" of the object. This appears to be a desirable goal for secure information sharing, except that classical DAC simply fails to achieve it. We sometimes like to think of secure information sharing as DAC done "correctly."

The second approach is generally called mandatory access control (MAC) which "solves"[3] the secure information sharing problem in a very specific and rigid framework.[4] MAC allows information to flow in one direction in a lattice of security labels. Copies of information made from one or more objects inherit the least upper bound of the labels from the individual objects. In this manner MAC is able to propagate information flow controls from the original objects to various copies (in whole or in pieces) that might be made. So long as the information flow requirements align with this policy MAC is a reasonable approach to secure information sharing. However, we can state empirically that this alignment has not turned out to be very common over the past three and a half decades of experience. In particular MAC does not provide any controls for the owner of an object to "share but protect" amongst other users at the same or higher security levels. Traditional formulations of MAC propose to introduce this additional capability by means of adding a DAC component [8] but since DAC does not solve secure information sharing its addition does not really provide this desired capability of secure information sharing within the same security level. We can characterize MAC as coarse-grained one-directional secure information sharing across security levels wherein the need for fine-grained information sharing both within and across security levels is left unsolved.

The third approach called originator control or ORCON was discussed by several authors in the late 80's and early 90's including [2, 13, 19] and more recently by Park and Sandhu [22]. In contrast to MAC where the coarse-grained policy for information sharing is driven by security labels, ORCON shares with DAC the notion that the owner of the information decides who should have access to it. Like DAC, ORCON is fine-grained in that each object and each user can be treated differently and is discretionary in that the owner is the principal source of the policy to be enforced. Graubart [13] suggests the notion of propagated ACLs (access control lists) whereby the ACLs on each object are propagated to other objects whenever information from the original object is copied (in whole or in part). The ACLs accumulate by taking their intersection. Park and Sandhu [22] discuss an approach to ORCON based on trusted viewers that prohibit the copying of information from ORCON-protected objects. This brings us to the topic of trusted computing and its applications to secure information sharing.

## Trusted Computing

The fourth approach to information sharing is based on trusted hardware and software on the client side. The concept is that protected content can only be rendered on the client (and obtained in cleartext or clearmedia for the purpose of rendering) by a trusted viewer. There have been numerous attempts to build trusted viewers in software alone using commodity computers and the Microsoft Windows™ [5] operating systems. Such viewers have been built for purpose of retail DRM (e.g., Microsoft Windows Media Player™ and Apple iTunes™) as well as enterprise DRM (e.g., Authentica™ and Microsoft™ products).[6] It has become widely accepted that software-only trusted viewers cannot provide a high degree of protection since there are too many avenues for software-only attacks to extract the protected content in clear form. Thus a trusted viewer must have a foundation of trust in hardware. Over the years many forms of trusted hardware have been discussed. Smith [38] provides a comprehensive discussion of various approaches. A brief perspective is also provided in [34].

In recent years there have been significant efforts by mainstream vendors to build industrial-strength trusted hardware. The Trusted Computing Group (which replaced the earlier Trusted Computing Platform Alliance) has developed specifications for a Trusted Platform Module (TPM) to provide a hardware root of trust for various purposes on a client computer. A tutorial-style exposition of TPM technology is given by Pearson et al [25]. Intel's LaGrande Technology™ (LT) [1] uses the TPM as an external trusted processor for its basic root of trust but provides additional capabilities in the LT chip set to protect memory, protect input and output (by providing trusted paths), and an additional ring privileged beyond the existing ring 0. The application of this technology to build trusted operating systems is discussed by England et al [9]. Some applications to P2P systems are discussed by Zhang et al [34, 40]. Applications in other domains are discussed by several authors including [4, 14, 26, 27, 35, 36, 37].

The combination of LT (or LT-like) processors with TPM (or TPM-like) support along with suitable software for utilizing these capabilities has come to be called Trusted Computing (TC). While the term Trusted Computing has been used in the past for other technologies, perhaps most notably for Trusted Computing Base [8], it has become commonplace to use the term as indicated above and we adopt this meaning in this paper. There is, of course, considerable difference between the modern approach to TC compared to the past. First of all modern TC is designed for a distributed and dynamic, open environment wherein "trusted" application software

[2] There is considerable confusion in the early and current literature as to the exact definition of DAC. A thorough analysis of DAC definitions is beyond the scope of this paper and is not authoritatively available in the existing literature. For our purpose here we understand DAC to mean owner-based DAC as motivated and discussed in the early papers of Lampson, Graham and Denning [12, 17], embodied in the so-called Orange Book [8] and elaborated in the more recent work of Osborn, Sandhu and Munawer [21, 33].

[3] The reason for the quotation marks is that MAC does not directly address the covert channel problem [8] without which it is not clear to what extent MAC controls information flow. Likewise in its simplest form MAC does not address inference and aggregation issues, as well as downgrading and other related issues.

[4] There is ample confusion about what is precisely meant by MAC. For our purpose we will take MAC to mean the simple-security and star-properties first proposed by Bell and Lapadula [5], subsequently axiomatized by Denning [7] and embodied in [8]. Also see Sandhu [29] for a more recent and accessible discussion.

[5] Microsoft Windows, Microsoft Windows Media Player, Apple iTunes, Authentica, Microsoft and LaGrande Technology are either registered trademarks or trademarks of their respective companies in the United States of America and/or in other countries. Other product names mentioned in this article may be trademarks or registered trademarks of their respective companies and are the sole property of their respective manufacturers.

[6] Enterprise DRM is a form of secure information sharing where the enterprise is the principal repository of the information being shared and the enterprise is also the principal source of policy for access to this information.

can be executed and protected from interference from other software on the same platform. Thus the trust mechanisms provide greater security for software execution within a single platform. Secondly there is direct support for platform-to-platform propagation of trust. TC technologies seek to protect data in creation, processing, storage, and transfer primarily by exposing the cryptographic secrets required to access the data only to software which has a verifiable chain of trust, be it on a single computer or across multiple computers. Reliance on appropriate application software to actually enforce the security policy is an integral part of this approach. TC primitives include cryptographic operations and trusted storage of root keys as a foundation for security. This is a sharp departure from previous approaches to trusted computing. Notably, modern TC technology is a product of industry initiatives with little direct input from the academic and research communities. As a consequence there is so far limited exposure of this important technology in the research literature. Likewise there is little guidance to industry regarding the use of this technology in support of traditional and new access control objectives.

Our primary interest in this paper in TC is its enabling of trusted viewers. Using TC primitives protected content can be cryptographically sealed so that it can only be obtained in clear form on suitably trusted client computers with approved hardware and software configuration. The degree of assurance to which this is achieved will vary and depends on a number of factors, including the degree of hardware tamper resistance, the degree of confidence that the trusted software actually works properly and the degree to which information may leak due to side-channels. For the purpose of this paper we simply assume that the overall assurance of the system is sufficient for handling the sensitivity of the information that is being shared. For simplicity we also assume a homogeneous environment wherein the identical hardware and software configuration is available on all clients. The general outline of how a trusted viewer is implemented on a modern TC platform is well known and we refer the reader to papers such as [40, 34] for additional details.

In this paper we simply assume that a trusted viewer is available with a suitable degree of assurance. The main question is what exactly we would want the trusted viewers to do, including the policies for access, and how these goals would be achieved. We will propose three layers of models for this purpose in Section 3 and discuss how to apply this layered framework to secure information sharing in the remainder of the paper. Before doing so we first give a high-level description of the information sharing space and define the subset that will be considered in this paper.

## 2. SECURE INFORMATION SHARING

The highest level distinction in our opinion is between secure information sharing and retail Digital Rights Management (DRM), discussed earlier in the paper. Information sharing pursues the goal of "share but protect" because of the sensitivity of the content, be it for business, personal or national security reasons. Retail DRM is concerned with protecting revenue. In retail DRM the "information" itself is readily disclosable but its use needs to be protected so as not to lose revenue. A thorough analysis of the similarities and differences between these two arenas is beyond the scope of this paper and remains an open topic in the literature. A partial analysis is available in [24, 39].

Trusted Computing is clearly a dual-use technology that can be used for both secure information sharing and retail DRM. There are many similarities between these two arenas, including trade-offs between usability and security and the support of a legal system to mitigate assurance weaknesses in the deployed security. Nonetheless fundamental motivation is different in the two cases and this inevitably leads to differences in the threat models and security, system and usability trade-offs that are feasible. A truly fundamental distinction between the two is that the business model for generating revenue is much more relevant to determine these trade-offs in the case of retail DRM whereas this is much less so for information sharing. In the latter case the sensitivity of the information typically directly ties to mission objectives. In this paper our focus is exclusively on information sharing.

The next level of distinction we make is between read-only and read-write secure information sharing. A trusted viewer is a feasible approach in a read-only scenario where content cannot be changed. Clearly there are many situations where the content needs to be changed and not just viewed by an authorized user. A typical example of a read-write situation is a collaborative project where team members may work on a set of documents collectively. In such cases the trusted viewer must be extended to become a trusted authoring tool also. This brings in many policy and technical complications.[7] Hence our desire to separate the two cases. The read-only case presents a sufficiently rich domain for investigation and is clearly one of considerable practical interest. This paper is limited to the read-only case.

The final distinction we make in this paper is based on the mechanism for authorizing access. For concreteness let us assume a model where information is stored in containers called objects. We say that authorization for information access is content-independent whenever the access control mechanism can decide whether or not a specific user has access without rendering the encrypted object content in clear. In contrast content-dependent access requires that the clear content be examined before this decision can be made. A typical example of a content-dependent authorization is one that allows user Alice to view salary records where the salary is less than a specified amount such as $100K. Determination of whether or not Alice has access to a particular salary record requires access to the cleartext content of the record before this determination can be made.[8] In this paper we limit ourselves to the content-independent case.

To summarize we have the following three high-level distinctions with respect to information sharing.

1. Secure information sharing versus retail Digital Rights Management.

2. Read-only versus read-write secure information sharing.

3. Content-independent authorization for information access versus content-dependent authorization.

This paper is limited to read-only secure information sharing with content-independent authorization.

[7] For instance who gets to control access to the modified document? Is it the creator of the original document or the person who modified the document? Do we need agreement of both? Another example of policy complications is consideration of what kinds of changes are allowed. Is it only possible to add comments and various notes without actually deleting existing material or can existing material be removed? Can comments be modified and removed by subsequent reviewers even if the original content cannot be changed? Yet another example is composite documents that contain multiple other documents in part or whole. How does policy for the composite document relate to policy for the parts?

[8] Approaches such as storing salary records with salaries above $100K in a separate file from those with salaries below $100K can deal with situations planned for up-front but do not provide the true flexibility of content-dependent authorization.

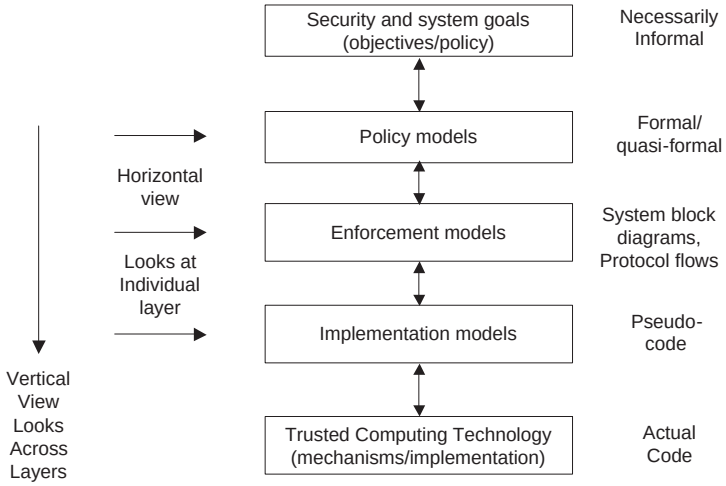


**Figure 1: The PEI Models Framework**

# 3. THE PEI MODELS FRAMEWORK

Designers of security systems have traditionally made a distinction between policy and mechanism (see, for example, [18]). The general goal has been to build flexible and robust mechanisms that can conveniently support a wide range of policies. Policy is concerned with "what" security needs to be enforced while mechanism is concerned with "how" the security is being enforced. In early days this distinction was applied to a single operating system running a small number of applications. The early literature thereby took an approach of a binary two-layer distinction between policy at the top and mechanism at the bottom. Modern systems are distributed and have multiple trust and service dependencies. Trying to close the policy-mechanism gap in a single step is hardly viable in such a complex environment. The early literature also treats the policy layer as being rigorous and well defined, whereas in reality the highest level of policy is often informal and fuzzy. One of the most difficult steps is to take informal policy requirements and provide sufficient rigor, structure, and detail in the next level of policy requirements that can be effectively handed off to security engineers to enforce. Moreover, at the highest level, policy is fundamentally driven by business concerns and needs and as such is likely to remain informal and fuzzy. Rather than trying to force formality and rigor where it is inappropriate, a realistic framework should explicitly accommodate an informal policy layer at the top. For variety we use the terms objective and policy interchangeably. While our focus is on security we also recognize that there are important system objectives that need to be articulated at this layer and will impact other layers.

In this paper we propose to introduce three layers of models in order to close the overall gap from informal policy to concrete code in multiple steps.[9] This framework is illustrated in Figure 1. At the top we have a layer of informal security and system policy requirements. This layer is necessarily informal. It is the layer at which business issues dominate. We feel that it is a mistake to try and formalize this layer. Rather we should recognize it explicitly for what it is. The bottom layer is actual running code. For the purpose of this paper this bottom layer is assumed to be structured around trusted computing technology.[10]

The fundamental problem of security engineering is to close the enormous gap between the top and bottom layers. Inevitably this requires compromises, decisions about trade-offs, and adjustments as the system gets built and deployed. We introduce three layers of models to close this gap in multiple steps. We emphasize that this framework is absolutely not intended to be a top-down waterfall-style software engineering methodology.[11] The main purpose of the framework is to enable work to proceed concurrently at all layers while keeping clear as to which decisions are being made at which layer. In our experience so far too much security research and practice focusses on just one layer or confuses issues which cut across multiple layers.

The name PEI for the framework comes from the three middle layers of policy, enforcement and implementation models. The purpose of a policy model is to take informal high-level objectives and flesh out rigor and detail using a formal or quasi-formal notation. A well-known example of a successful policy model is the lattice model for mandatory access control [5] which captured the informal concept of enforcing information flow using the security labels that had been in use in the military and national security arenas in the paper world. This model has a rigorous mathematical founda-

[9] In previous work Sandhu introduced the OM-AM framework [30] which uses two layers (called models and architectures) to close this gap. The PEI framework defined here can be viewed as a generalization of the OM-AM framework. Of course, the notion of layers is commonplace in computer systems research and practice. Nonetheless PEI, and earlier OM-AM, represent novel approaches to security engineering.

[10] More generally, the framework allows other kinds of security technology to be used at this layer so the tie to trusted computing is not inherent to PEI. We conjecture that having too much freedom at the bottom-most layer makes the framework more difficult to apply. We also conjecture that decisions about the technology to be used at the bottom-most layer are not entirely technical but in practice will often be driven by various top-layer business concerns.

[11] Clearly modern software engineering itself has abandoned the highly idealized top-down view with which this discipline started. Nonetheless there is, in our experience, a cultural bias in the community to interpreting layered diagrams such as Figure 1 as indicating a top-down process. Moreover, deviations from top-down are only grudgingly accepted as departures from the ideal. In our view a strict top-down process is so totally unworkable in trying to close this enormous gap between informal desire and concrete code in context of modern distributed systems, that we need to completely drive it out of our culture as a vestigial notion.

tion [7, 11] and has been shown to apply to information-flow policies that are driven by integrity and separation concerns [6, 29] in addition to the original confidentiality concerns [5]. Another well-known and more recent policy model is the RBAC96 model [32] for role-based access control, and its derivatives [10, 31], which accommodate a wide range of business relevant policies beyond the purview of traditional MAC and DAC while also encompassing MAC and DAC [21]. While aspects of RBAC96 have been formalized (for example [3]), role-based models tend to have a more quasi-formal character as compared to lattice-based models. In the early days the access matrix model [12, 17] served as a useful policy model and its formal aspects have been elaborated [15, 28]. More recently the UCON model for usage control has been proposed [23] as a new foundation for access control which combines traditional authorization with obligations and conditions as well as continuity of decisions and mutability of attributes to accommodate a wide range of access policies beyond the purview of traditional access control models. Aspects of UCON have been recently formalized [41]. Thus the literature contains many examples of policy models. They generally use mathematical notation and are more or less formal in nature. In some cases formal analysis of the models is possible. In other cases formal analysis may be undecidable or intractable. Sometimes formal analysis may not be of interest or relevance. So a wide range of possibilities exist at this layer.

While the policy models focus on the "what" aspect, the enforcement and implementation models address the "how" aspect of enforcing the desired policy. The enforcement models address the big picture of the "how" question, at the level of system block diagrams and protocol flows. The protocol flows can be formalized and analyzed to establish various security properties of the system. At the same time they leave a certain level of detail unspecified which is then elaborated in the implementation models. The implementation models are focussed on specific issues identified in the enforcement models layer. These issues need to be resolved in sufficient detail to require descriptions at a pseudo-code level of detail and precision. These abstract characterizations of enforcement and implementation models will be made more concrete by the examples given later in this paper.

The relationship between models at adjacent layers is many-to-many. A single policy model can be enforced by multiple enforcement models (with possibly different trade-offs between security, trust, performance, cost, convenience, etc.). Likewise a single enforcement model may support multiple policy models. There is similarly a many-to-many relationship between enforcement models and implementation models.[12]

In the rest of this paper we apply this PEI framework to the problem of secure information sharing, as scoped out earlier, in the context of TC. We begin with a horizontal view at the policy models layer to identify three basic alternatives. For each of these policy alternatives we then drill down vertically to consider alternate enforcement and implementation models.

# 4. POLICY MODELS

In Section 2 we had scoped the secure information sharing problem for the purpose of this paper along three major dimensions. Our focus is on protection of sensitive content rather than revenue protection, on read-only access rather than read-write access and on content-independent authorization rather than content-dependent authorization. The motivation for this scoping was discussed earlier. This level of informal analysis is appropriate at the topmost security and system policy layer of the PEI framework.

Given this scope there is obviously still an enormous gap between agreement on the high-level policy goals and actual code to enforce security. At the policy models layer we seek to identify major approaches to achieve these high-level goals. A complete analysis is out of scope of this paper but we will outline the most major decision that needs to be made and also identify some of the issues that a more detailed analysis would need to investigate. Our approach is driven by a mix of consideration of use cases and technical feasibility.

The essential paradigm of TC is to protect content by sealing it to trusted viewers. The TC mechanisms will ensure that the only means to access the trusted content is through the trusted viewer. The trusted viewer, in turn, is responsible for ensuring that access to cleartext content is provided only to authorized users.[13] We believe that the fundamental question at the policy models layer is how to authorize legitimate access to an object or document (for convenience, we will use these terms interchangeably). The answer to this question will drive numerous other details, and is by itself the single most important component of policy. Given that the enforcement of access is grounded in TC and Trusted Viewers we can identify three distinct methods for authenticating authorized users.[14] These three methods are respectively grounded in "what the user knows" (password-based sharing), "what the user has" (device-based sharing) and "what the user is" (credential-based sharing).[15]

In password-based information sharing, authorized users are required to demonstrate knowledge in order to access the protected content. Of course, the password needs to be communicated to legitimate recipients. We assume that there is some out-of-band mechanism for doing this. Password-based authorization to a protected document brings with it certain important implications. A basic characteristic of TC is that protected content is typically stored persistently on disk and thereby can be easily propagated from one user to another by transporting the encrypted bits. Thus the protection of the content is only as good as the protection of the password. Since the password can be communicated quite easily there is an implicit assumption in this model that legitimate users will not abuse this trust.[16] This is an intrinsic property of any password-based sharing policy. The main advantage of the

[12]This many-to-many relationship between adjacent layers further indicates the futility of a top-down approach in this context.

[13]As we will see in Sections 5, 6 and 7 there are numerous ways to achieve this goal. In some cases the Trusted Viewer can access the clear content on its own and then independently determine whether or not the user attempting to access the content is properly authorized. In other cases the content itself cannot be decrypted without the presentation of the required authorization credentials since the encryption keys are cryptographically coupled with the secrets that underly the authorization. TC and Trusted Viewers thus give us two basic means of enforcement of access to authorized users.

[14]There are other authentication methods that could be considered, such as use of hardware tokens, smart-cards, biometrics, etc. The particular three methods we have selected do not require any additional hardware beyond that included in TC-enabled computers.

[15]It is possible to consider combinations of these approaches, such as password + device + credential based. For simplicity we do not consider such combinations explicitly in the paper. Combinations can of course be constructed from the individual cases if desired.

[16]Note that a user can always show the sensitive content to another user by rendering it in a protected environment where the latter user is physically present (if possible). Further a user can always disclose information by memorizing aspects of the content and communicating these to unauthorized recipients. Rendered content can also be captured by a variety of recording devices (unless these are somehow physically excluded). Some of these trust assumptions (or viewed from the other side, inherent vulnerabilities) are common to multiple policy models. Others vary from model to model.

password-based model is that it requires minimal infrastructure and support. It is extremely lightweight (given the assumption that TC platforms are ubiquitous in the environment of interest) and easy for users to understand. Users can password-protect documents and communicate these to intended recipients without requiring support from the organization's IT (information technology) infrastructure. Users can set up a group password, which is shared amongst a select group of users, quite conveniently and on their own initiative. Password-based sharing can be seamlessly done across organizational boundaries and across large distances. Conversely password-based sharing is not very scalable since the management, memorization, and entry of passwords become cumbersome beyond a certain point. But then how much scalability is really appropriate for sharing of sensitive information? These aspects of password-based sharing are well known and collectively make it an attractive mode. Nonetheless, as we will see in Section 5 the classic approach to password-based information sharing is fundamentally insecure since it reveals the password by off-line dictionary attacks. We will also see how TC enables a much higher degree of security for this very attractive mode of information sharing.

The second means of authorizing legitimate access is via the device (that is, the personal computer or PC) itself. TC technology establishes a unique cryptographic identity for each PC (more accurately for the TPM chip on the motherboard of the PC). In device-based sharing the protected content is viewable only on a designated individual PC.[17] The key to unlock the content is bound to the TC credentials of the PC and can only be rendered on that PC by means of a trusted viewer. A user who has access to that specific PC will be able to view the content.[18] Device-based sharing ties into the sense of ownership that modern computer users typically have with respect to their PC. It is appropriate in environments where there is a sole user for every PC. It is also appropriate in environments where the specific PC possesses specific characteristics that make it appropriate to render content there, possibly for multiple users. For instance, physical location of the PC could be such that it is appropriate to render the content since physical controls limit the user population who can access the PC. Device-based sharing lacks some of the convenience of password-based sharing but offers higher assurance as compensation. Access to the protected content on multiple PCs is much easier with password-based sharing, as is the ability to have group passwords. Unlike password-based sharing, device-based sharing makes it possible to prevent re-dissemination of protected content by simply propagating the password along with the protected content. The difficulty is to ensure that the content is protected for the correct device and not inadvertently for a rogue device under control of an attacker. Nevertheless, like password-based sharing, device-based sharing has aspects of being lightweight, requiring minimal additional infrastructure, and the ability to operate across organizational boundaries. So it appears to be a worthwhile model for TC-based secure information sharing.

[17]More generally we could consider policies where the content is viewable on a class of PCs, says PCs issued by an organization. This case would be covered by credential-based sharing where the credential is a class of devices presumably identified by a root or intermediate Certificate Authority.

[18]How user access to the specific PC is controlled and enabled is left unspecified in this paper. We simply assume that an adequate level of assurance will be provided for this purpose. Alternately, a combination of device-based and password or credential-based policies can be used to deny access to a unauthorized user even if he has access to the PC. For example, a secret can not only be tied to a specific PC, it can also be sealed so that it is unlocked only if a user supplies a specific password.

The final policy model we consider recognizes legitimate users based on credentials. We do not specify the nature of these credentials. They could include identity credentials, attribute credentials, role credentials, etc. Clearly there will be a need for additional infrastructure to support the issuance, verification, and maintenance of these credentials. The sharing policies that are enabled will depend on the nature of these credentials. Thus if the credentials are entirely identity-based then identity-based information sharing will be the only form of sharing that can occur. With role-based credentials we will get role-based sharing.

In the following sections we will consider each of these three policy models in turn and discuss enforcement and implementation models for each one. The high-level at which we have defined these policy models is adequate for our purpose in this paper. Nevertheless in practice we would need to flesh out a number of additional details at the policy model layer before implementing a real system. Below, we briefly identify issues (in no particular order) that will need to be resolved at the policy models layer.

**Revocation policy.** Revocation has too often been the Achilles heel of authorization policy and mechanism. Revocation must be addressed at the policy models layer. While it may not be possible to resolve all revocation issues at this layer it should be explicitly stated what the overall approach to revocation is, in order to provide guidance for the enforcement and implementation layers. Some of the questions to be addressed are: Can authorized access be revoked? What is the delay in revocation? More generally, can authorized access be changed? (For example, changing the password or changing the role credential required for access.)

**Usage policy.** In traditional access control the usage policy is fairly simple. If a user is authorized to access an object the access can be performed as many times and for as long as the user desires. Essentially there is no usage control. The concept of limiting usage was first emphasized in recent years by retail DRM where limits on how often or how long an access is permitted is often viewed as a desirable feature. The UCON model proposes to integrate usage controls into the foundation of next generation access control [23] in support of many policies beyond DRM that require it. Limiting the number of times a protected document can be opened is useful for keeping the legitimate recipient from showing it to too many colleagues while displaying on a TC-enabled authorized platform. Similarly limits on the overall rate at which a user can access protected content could allow humans to work within the limits of human capacity while protecting against automated attacks. These considerations should be explicitly addressed at the policy models layer, even if the decision is not to impose any controls.

**Re-dissemination policy.** Re-dissemination of protected content is an essential component of secure information sharing. The simplest policies are either not to allow re-dissemination or to place no limits on it. The former is too rigid and the latter too liberal so there will be situations which require something between these two extremes. One possibility is to limit the number of recipients to whom re-dissemination is authorized. Another possibility is to limit to whom re-dissemination can be done. For example, re-dissemination to faculty is authorized but not to students.

**Distribution policy.** Distribution policy is concerned with how the protected content is distributed. Two basic alternatives are to require protected content to be obtained from a server every time the content is accessed versus permitting persistent storage of the protected content on the client PC.[19] A more detailed analysis of alternatives is given in [24]. A related issue is how the policy

[19]Note that TC technology that is used to implement trusted and confined viewers can also prevent local persistent storage of protected content just as it prevents leakage of cleated content.

that controls access to protected content is itself distributed, that is, the distribution policy of security policies. Requiring download of fresh policy on every use ensures that the applied policy is up-to-date and reflects any revocations that may have occurred. On the other hand it makes provision of off-line access more difficult.

**Accessibility policy.** The basic question here is whether or not the user needs to be on-line to access content. This question has some relationship to the distribution policy, since some form of persistent local storage of protected content is required to facilitate truly off-line access. Some systems provide off-line access by default while others prohibit it except by special preparation where the normal controls are bypassed for specific documents.

Resolution of these issues will require reference back to the topmost objectives layer as well as consideration of implications for the enforcement layer. The issues themselves are somewhat orthogonal to the manner in which legitimate users are authorized to access protected content, but this does become relevant to reconciling conflicts and trade-offs. There are also mutual dependencies amongst these various issues.

It is beyond the scope of this paper to give a thorough analysis of these issues and their dependencies. For our purpose we will make some simple assumptions and proceed accordingly. Specifically we assume that revocation is not possible, there are no usage controls, re-dissemination is not possible except in the password-based case where it is authorized without any controls, local persistent copies of protected content can be kept on client PCs, and the degree of off-line access depends on the details of the enforcement and implementation models. This is a reasonable set of assumptions for purpose of illustrating the PEI framework in context of information sharing and Trusted Computing. More generally this is a reasonable set of assumptions to begin detailed investigation of alternatives. Once some details have been developed these assumptions can be revisited to accomplish additional analysis at the enforcement and implementation layers with more realistic assumptions. As we depart from simple assumptions for the issues enumerated above it is increasingly likely that we will need formal or quasi-formal policy models to rigorously and precisely articulate these policies.

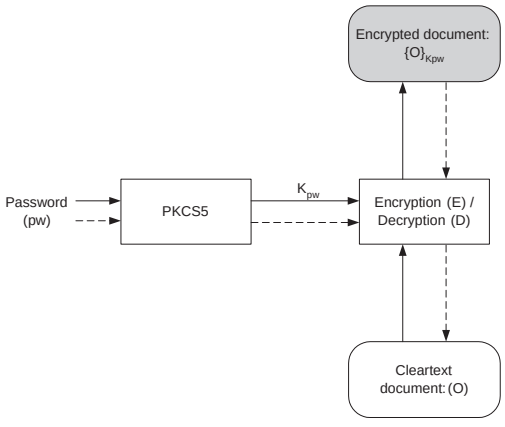


**Figure 2: Password-Based Enforcement Model: Password Encryption**

# 5. PASSWORD-BASED MODELS

We now turn to more detailed consideration of enforcement and implementation models for each of the three policy models identified in the previous section. We start with password-based information sharing.

## 5.1 Password-Based Enforcement Models

The ability to password-protect electronic documents is widely available in products from leading vendors such as Microsoft and Adobe. Due to lack of a TC foundation traditional password-based information sharing systems have a very simple enforcement model shown in Figure 2. Basically the password is used to encrypt the document as well as to decrypt it. Figure 2 shows two protocol flows, one in solid lines and one in dashed lines.[20] The solid lines indicate the flow when the protected document is constructed, while the dashed lines show the flow when the protected document is accessed. The solid lines in Figure 2 show a password being entered by the user preparing the protected document into a box labelled PKCS5, indicating that the password is transformed using the well-known PKCS5 standard [16] to produce a symmetric encryption key $K_{pw}$. This key is then used to encrypt the cleartext document to produce the encrypted document (indicated by the notation $\{O\}_{K_{pw}}$).[21] Conversely the dashed lines show the password being entered by a user who is accessing the protected document which is used to regenerate the same key, this time for decrypting the document.

The problem with the enforcement model of Figure 2 is that it is fundamentally insecure due to dictionary attacks. An attacker who has possession of the encrypted document will not take guesses for $K_{pw}$ but rather will guess likely passwords. This is the well-known dictionary attack first made famous by a classic paper of Morris and Thompson [20]. There are any number of tools available on the Internet to crack such documents, empirically confirming the intrinsic weakness of this enforcement model.

How is TC going to improve the situation? In Figures 3 and 4 we show two different enforcement models for the use of TC for password-based sharing. First consider Figure 3. Following the solid lines, a strong key K is generated and used to encrypt the cleartext document (middle part of the figure). K itself is sealed (left part of the figure) to a trusted viewer TV (indicated by $[K]_{TV}$).[22] Finally, on the right side of the figure the hashed password is encrypted. The three items at the top of the figure collectively make up the protected document. Recovery of the cleartext by an authorized user is shown in dashed lines. First K is recovered by unsealing it, an operation that TC guarantees can be performed only by an unaltered TV program in the approved configuration. K is then used to decrypt the hashed password which is compared with the hashed password obtained by hashing the user provided password. If there is a match the user has provided the correct password. TV can then go ahead and decrypt the encrypted document using K. If the match fails the password is incorrect. Since K is a strong random key an off-line dictionary attack on the encrypted hashed password or on the encrypted document is not possible. Note that the password is not used to encrypt the document but is used only

[20]For clarity the dashed lines are always shown to the left of the solid lines in lines that run vertically, and below the solid lines in lines that run horizontally.

[21]By convention cleartext documents are shown at the bottom while encrypted documents and ancillary material, if any, are shown at the top.

[22]The seal operation is a standard operation in TC which encrypts K in such a manner that K can only be decrypted by a program and environment whose hash-value matches that of the trusted viewer TV. The operation to recover a sealed K is called unseal and will succeed only if requested by a unaltered trusted viewer running on a TC platform in an approved configuration. See [25] for additional details.

to authenticate the user. TV has the ability to decrypt the document without the password but is trusted not to do so. This is the essential paradigm of trusted computing. We emphasize that in Figure 3 the key K used to encrypt the document has no relationship to the password whereas in Figure 2 it is directly computed from the password. Hence the tremendous gain in security in Figure 3. Also in Figure 3 the key used to encrypt the hashed password could be a different key from K, in which case it would be separately sealed to TV.

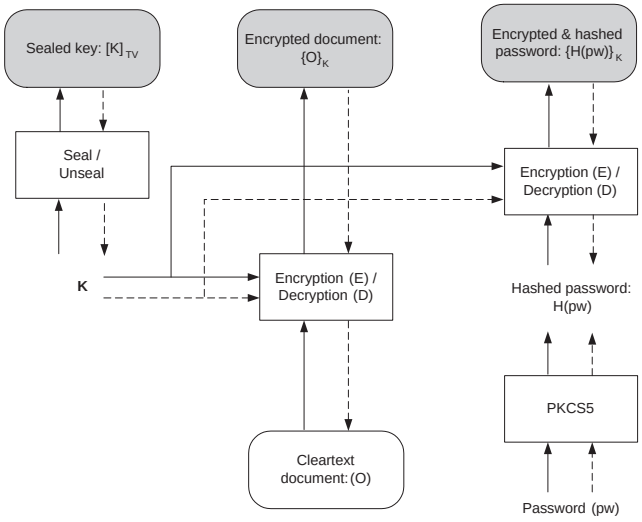


**Figure 3: Password-Based Enforcement Model: Trusted Viewer Seal with Password Authentication**

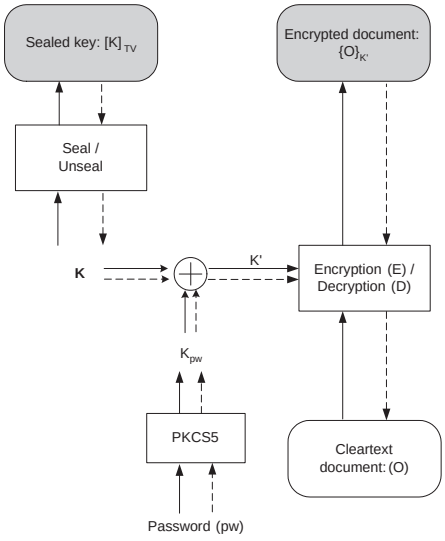


**Figure 4: Password-Based Enforcement Model: Trusted Viewer Seal with Password Encryption**

A possible concern with Figure 3 is that a failure of the seal and unseal operation may reveal K and make the protected content accessible to unauthorized users who need not bother about finding the password. Figure 4 shows an alternate enforcement model that seeks to address this concern. Rather than using K to directly encrypt the document, we instead combine K with a key $K_{pw}$ generated from the password to get $K'$ and then use $K'$ as the encryption key. (The combining operation is simply shown by the + circle in the figure and should be cryptographically robust.) Failure of the seal/unseal operation will reveal K bit not $K'$ so the attacker would need to do additional work to get to the content. However, knowledge of K would enable an off-line dictionary attack on the encrypted document. The encrypted hashed password is not explicitly required in Figure 4 since the authentication is indirectly achieved by successful decryption of the document.

At this point we have three enforcement models for password-based information sharing. Figure 2 makes no use of TC and is intrinsically insecure. It also represents the state-of-the-art today in widely-deployed commercial products. Figures 3 and 4 make good use of TC and are considerably more secure than Figure 2. It appears that Figure 4 is more secure than Figure 3 but this should be demonstrated by some degree of formal analysis. Moreover each of these figures needs a careful security analysis which may require providing additional details about the protocol flows.

## 5.2 Password-Based Implementation Models

Figures 3 and 4 both protect against an off-line dictionary attack. Possession of the protected content (depicted by the shaded boxes on the top of these figures) does not permit an attacker to discover the password by trying guesses without involving the TV. However, both do allow for on-line password guessing attacks. An attacker who has access to a legitimate TV but does not know the password for a specific document may repeatedly submit password guesses to the TV till such time as the TV successfully renders the clear content. (Note that TC can use its trusted path mechanism to make sure that the password is actually being entered from a trusted device so there are significant barriers for an attacker to try and automate an on-line guessing attack.)

To mitigate this on-line password guessing attack the TV must limit the rate at which passwords can be guessed. There are a number of ways this can be accomplished. A simple approach would be to simply limit the overall rate at which the TV can be used on a specific TC device. This will require the TV to maintain state on the device which can be securely accomplished using the seal and unseal operations. A more nuanced approach may limit the overall rate but also put additional limits on repeated attempts against the same protected content. Identification of multiple copies of the same protected content could be achieved by comparing hash values. We believe that this level of detail should be developed in the implementation model layer. At the enforcement model layer we simply decide that some throttling mechanism is needed to thwart on-line password guessing and leave the details to the implementation model. Clearly there are many different ways that throttling can be implemented. That might be sufficient reason by itself to delegate the details to the implementation layer.[23] We also need to articulate criteria for selecting amongst different implementation alternatives. For intellectual property and similar reasons certain mechanisms may be ruled out and others strongly favored. Alternately the selection may be left to purely technical criteria. The enforcement-implementation layer boundary gives us an opportunity to make such criteria explicit and apply them in the selection process. In general there are a whole lot of details about Figures 3 and 4 that need to be further articulated at the implementation layer.

[23]Development of detail within the implementation layer may itself benefit from a layered approach. We would consider such layering to be internal to the implementation layer and not at the same level as PEI.

We feel it is important to keep the enforcement layer models lean and highly abstract so as to focus on the really important alternative without getting bogged down in detail that is better pushed into the implementation layer models. At the enforcement layer we are looking at the "big picture" and it is important to get it correct. At the implementation layer we will be looking at very specific aspects and it is important to be able to focus on each aspect in deep detail.

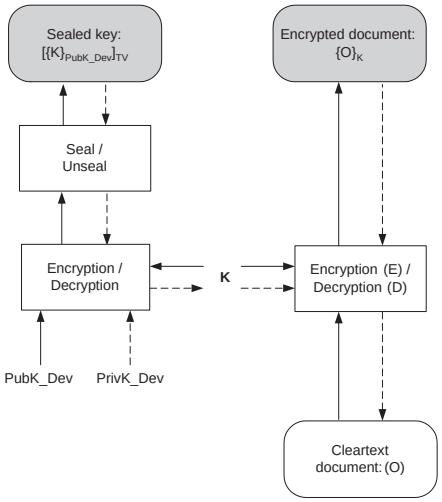


**Figure 5: Device-Based Enforcement Model: Trusted Viewer Seal with Device Encryption**

## 6. DEVICE-BASED MODELS

Turning to device-based secure information sharing we consider the single enforcement model shown in Figure 5. The right half of the figure shows use of a strong random key K to encrypt and decrypt the protected content. The left half of the figure shows that this key K is first encrypted with the public key of the device and then sealed to the trusted viewer TV. To access the content the TV can unseal the encrypted K but must further decrypt it which is possible only on the device where the matching private key is available. This is again a straightforward application of TC.

The main difficulty lies in being sure that the correct public key is used in the encryption operation. In general there are three approaches that could be taken. One is to use an identity-based public key infrastructure for users so Alice can reliably inform Bob as to what her device public key really is. But this is an infrastructure-heavy approach, in which case it may be better to use credential-based information sharing. Lightweight protocols for reliable exchange of device public keys could be based on device proximity using channels that are physically limited (such as bluetooth or infra-red). Lightweight protocols that run over large distances would require some form of human visual confirmation by means of hash values or graphic displays.

The decision of which one of these three approaches to follow probably belongs at the enforcement model layer. The deep details of how to pursue the specific selected approach definitely belong to the implementation layer. Where exactly to draw the line between enforcement and implementation layers is ultimately a matter of judgement in applying the PEI framework. The precise place where we draw this line is not terribly important, and inevitably there will be some gray areas at the boundary.

## 7. CREDENTIAL-BASED MODELS

Finally we consider the credential-based case. We show two enforcement models in Figures 6 and 7. Figure 6 uses the credentials purely for authentication, whereas Figure 7 uses the credentials purely for encryption.

Figure 6 shows that a credential policy is encrypted as part of preparing the protected content.[24] This policy is decrypted by the TV using the sealed key and then credentials of the user are examined to determine if they satisfy the stated policy. The TV will exercise its ability to decrypt the protected content only if it is satisfied that the credential requirements have been met. Figure 3 from the password-based case can be regarded as a special case of Figure 6 where the required credential is a password. Similarly there can be other instantiations of Figure 6 depending upon the actual credential in use. We would regard these alternatives as implementation models for the enforcement model of Figure 6.

Figure 7 shows that the public key of the recipient is used to encrypt the key K before sealing it to TV. This is very similar to Figure 5 except the public key belongs to an individual human being rather than a TC device. (We assume a suitable public-key infrastructure is in place to deliver the correct public key.) Clearly this approach can be taken only if the credential is usable for encryption.

We can imagine combining Figures 6 and 7 so that some credentials are used for authentication and others for encryption. In any case details about the credentials will lead to different implementation models.

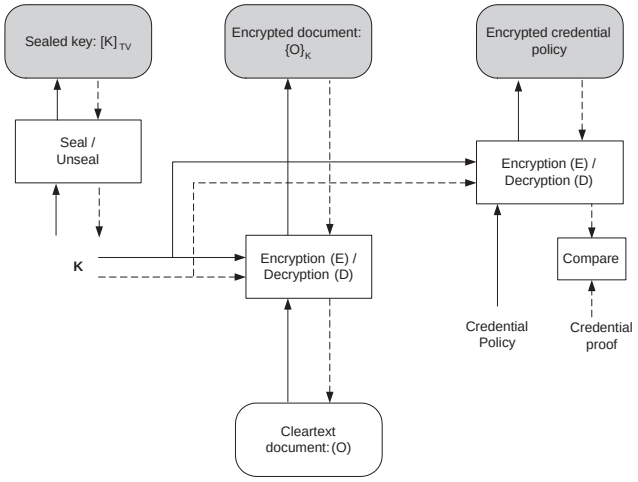


**Figure 6: Credential-Based Enforcement Model: Trusted Viewer Seal with Credential Authentication**

## 8. CONCLUSION AND FUTURE WORK

In this paper we have shown how modern Trusted Computing (TC) technologies can facilitate secure information sharing in a manner not available using pre-TC technology. We have developed the PEI framework of policy, enforcement and implementation models, and demonstrated its use in analyzing this problem

[24]Depending on the credential it may be necessary to encrypt the credential policy (say, if the policy requires presentation of a specific password) whereas in other cases it may suffice to sign the credential policy and leave the policy itself in cleartext (say, if the policy is to require a certificate chaining up to a specific trusted root). For simplicity we only discuss the former case here.

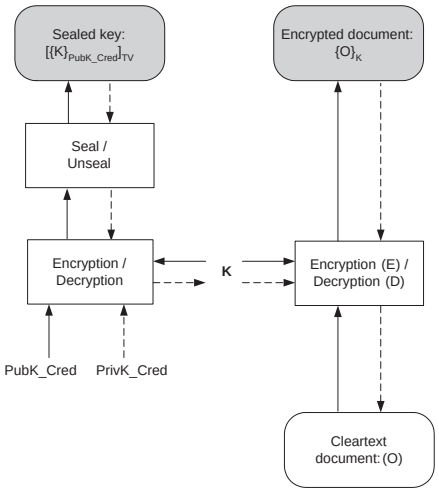


**Figure 7: Credential-Based Enforcement Model: Trusted Viewer Seal with Credential Encryption**

and synthesizing solutions for it. A number of open questions for future work have been identified along the way. This framework will allow us to investigate potential applications of TC for secure information sharing in greater depth in future work. Other applications of TC beyond information sharing can also be investigated. Finally the PEI framework can also be used in future work which does not require TC.

## Acknowledgement

The authors thank Michael J. Covington of Intel Corporation, Portland, Oregon for valuable discussion and input. George Mason University gratefully acknowledges the collaborative and financial support of Intel in pursuing this research.

## 9. REFERENCES

[1] *LaGrande technology architecture*. Intel Developer Forum, 2003.

[2] M. Abrams, J. Heaney, O. King, L. LaPadula, M. Lazear, and Ingrid. Olson. Generalized framework for access control: Toward prototyping the Orgcon policy. In *Proceedings of 14th NIST-NCSC National Computer Security Conference*, pages 257–266, 1991.

[3] Gail-Joon Ahn and Ravi Sandhu. Role-based authorization constraints specification. *ACM Transactions on Information and System Security*, 3(4):207–226, November 2000.

[4] S. Balfe, A. D. Lakhani, and K. G. Paterson. Trusted computing- providing security for peer-to-peer networks. In *Proceedings of IEEE International Conference on Peer-to-Peer Computing*, pages 117–124, Konstan, Germany, August 31 - September 2 2005.

[5] D.E. Bell and L.J. LaPadula. Secure computer systems: Unified exposition and Multics interpretation. Technical Report ESD-TR-75-306, The Mitre Corporation, Bedford, MA, March 1975.

[6] K.J. Biba. Integrity considerations for secure computer systems. Technical Report TR-3153, The Mitre Corporation, Bedford, MA, April 1977.

[7] D.E. Denning. A lattice model of secure information flow. *Communications of the ACM*, 19(5):236–243, 1976.

[8] Department of Defense National Computer Security Center. *Department of Defense Trusted Computer Systems Evaluation Criteria*, December 1985. DoD 5200.28-STD.

[9] P. England, B. Lampson, J. Manferdelli, and B. Willman. A trusted open platform. *IEEE Computer*, 36(7):55–62, July 2003.

[10] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security*, 4(3):224–274, August 2001.

[11] J.A. Gougen and J. Meseguer. Security policies and security models. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 11–20, Oakland, CA, 1982.

[12] G.S. Graham and P.J. Denning. Protection – principles and practice. In *AFIPS Spring Joint Computer Conference*, pages 40:417–429, 1972.

[13] R. Graubart. On the need for a third form of access control. In *Proceedings of NIST-NCSC National Computer Security Conference*, pages 296–303, 1989.

[14] V. Haldar, D. Chandra, and M. Franz. Semantic remote attestation - a virtual machine directed approach to trusted computing. In *Proceedings of the Third virtual Machine Research and Technology Symposium*, pages 29–41, San Jose, CA, USA, May 6-7 2004. USENIX.

[15] M.H. Harrison, W.L. Ruzzo, and J.D. Ullman. Protection in operating systems. *Communications of the ACM*, 19(8):461–471, 1976.

[16] B. Kaliski. *PKCS #5: Password-Based Cryptography Specification Version 2.0*, September 2000. RFC 2898.

[17] B.W. Lampson. Protection. In *5th Princeton Symposium on Information Science and Systems*, pages 437–443, 1971. Reprinted in *ACM Operating Systems Review* 8(1):18–24, 1974.

[18] R. Levin, E. Cohen, W. Corwin, F. Pollack, and W. Wulf. Policy/mechanism separation in Hydra. In *5th ACM Symposium on Operating Systems Principles*, pages 132–140, 1975.

[19] C.J. McCollum, J.R. Messing, and L. Notargiacomo. Beyond the pale of MAC and DAC - defining new forms of access control. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 190–200, Oakland, CA, May 1990.

[20] Robert Morris and Ken Thompson. Password security: A case history. *Communications of the ACM*, 22(11):594–597, 1979.

[21] Sylvia Osborn, Ravi Sandhu, and Qamar Munawer. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Transactions on Information and System Security*, 3(2), May 2000.

[22] Jaehong Park and Ravi Sandhu. Originator control in usage control. In *Proc. 3rd IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 60–66, Monterey, CA, June 5-7 2002.

[23] Jaehong Park and Ravi Sandhu. The $\text{UCON}_{\text{ABC}}$ usage control model. *ACM Transactions on Information and System Security*, 7(1):128–174, February 2004.

[24] Jaehong Park, Ravi Sandhu, and James Schifalacqua. Security architectures for controlled digital information dissemination. In *Proceedings of 6th Annual Computer Security Application Conference*, pages 224–233, New Orleans, LA, December 11-15 2000.

[25] Siani Pearson, Boris Balacheff, Liqun Chen, David Plaquin, and Graeme Proudler. *Trusted Computing Platforms: TCPA Technology in Context*. Prentice-Hall, 2003.

[26] A. Sadeghi and C. Stuble. Taming trusted platforms by operating system design. In *Proceedings of the 4th International Workshop for Information Security Applications, LNCS 2908*, pages 286–302, Berlin, Germany, August 2003.

[27] R. Sailer, T. Jaeger, X. Zhang, and L. van Doorn. Attestation-based policy enforcement for remote access. In *Proceedings of ACM Conference on Computer and Communication Security*, pages 308–317, Washington, DC, USA, October 25-29 2004.

[28] Ravi Sandhu. The typed access matrix model. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 122–136, Oakland, CA, May 1992.

[29] Ravi Sandhu. Lattice-based access control models. *IEEE Computer*, 26(11):9–19, November 1993.

[30] Ravi Sandhu. Engineering authority and trust in cyberspace: The OM-AM and RBAC way. In *Proceedings of 5th ACM Workshop on Role-Based Access Control*, pages 111–119, Berlin, Germany, July 26-28 2000. ACM.

[31] Ravi Sandhu, Venkata Bhamidipati, and Qamar Munawer. The ARBAC97 model for role-based administration of roles. *ACM Transactions on Information and System Security*, 2(1):105–135, February 1999.

[32] Ravi Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, February 1996.

[33] Ravi Sandhu and Qamar Munawer. How to do discretionary access control using roles. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*, pages 47–54, Fairfax, VA, October 22-23 1998. ACM.

[34] Ravi Sandhu and Xinwen Zhang. Peer-to-peer access control architecture using trusted computing technology. In *Proceedings of the 10th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pages 147–158, Stockholm, June 1-3 2005.

[35] Manoj R. Sastry and Michael J. Covington. Attribute-based authentication using trusted platforms. In *Proceedings of Wireless Personal Multimedia Communications*, Aalborg, Denmark, September 18-22 2005.

[36] S. Schechter, R. Greenstadt, and M. Smith. Trusted computing, peer-to-peer distribution, and the economics of pirated entertainment. In *the Second International Workshop on Economics and Information Security*, College Park, MD, USA, May 29-30 2003.

[37] E. Shi, A. Perrig, and L. Van Doorn. Bind: a fine-grained attestation service for secure distributed systems. In *Proceeedings of IEEE Symposium on Security and Privacy*, pages 154–168, Oakland, CA, USA, May 8-11 2005.

[38] Sean Smith. *Trusted Computing Platforms: Design and Applications*. Springer, 2005.

[39] Roshan Thomas and Ravi Sandhu. Towards a multi-dimensional characterization of dissemination control. In *Proc. 5th IEEE International Workshop on Policies for Distributed Systems and Networks*, pages 197–200, Yorktown Heights, NY, June 7-9 2004. Springer-Verlag LNCS.

[40] Xinwen Zhang, Songqing Chen, and Ravi Sandhu. Enhancing data authenticity and integrity in P2P systems. *IEEE Internet Computing*, 9(6):42–49, Nov-Dec 2005.

[41] Xinwen Zhang, Francesco Parisi-Presicce, Ravi Sandhu, and Jaehong Park. Formal model and policy specification of usage control. *ACM Transactions on Information and System Security*, 8(4):351–387, November 2005.