IEEE Communications Surveys and Tutorials, Fourth Quarter 2000, http://www.comsoc.org/pubs/surveys/

A Survey of Trust in Internet Applications

Tyrone Grandison, Morris Sloman

Imperial College, Department of Computing
180 Queen's Gate, London SW7 2BZ, UK
{tgrand, m.sloman}@ doc.ic.ac.uk

24 January 2001

Abstract

Trust is an important aspect of decision making for Internet applications and particularly influences the specification of security policy i.e. who is authorised to perform actions as well as the techniques needed to manage and implement security to and for the applications. This survey examines the various definitions of trust in the literature and provides a working definition of trust for Internet applications. The properties of trust relationships are explained and classes of different types of trust identified in the literature are discussed with examples. Some influential examples of trust management systems are described.

Keywords:

Trust specification, trust management, security policy, authorisation, authentication

1 MOTIVATION

Internet services are increasingly being used in daily life for electronic commerce, web-based access to information and inter-personal interactions via electronic mail rather than voice or face-to-face, but there is still major concern about the trustworthiness of these services. There are no accepted techniques or tools for specification and reasoning about the trust. There is a need for a high-level, abstract way of specifying and managing trust, which can be easily integrated into applications and used on any platform. Typical applications requiring a formal trust specification include content selection for web documents [1], medical systems [2], telecommuting [3], mobile code and mobile computing [4-6], as well as electronic commerce [7-14]. Our main motivation in studying trust specification and management is to use this as the starting point for subsequent refinement into security policies related to authorisation and management of security [15]. However, there are additional reasons as to why trust is an important concept for modern systems.

The migration from centralised information systems to internet-based applications will mean that transactions have to span a range of domains and organisations [16], not all of which may be trusted to the same extent. Inconsistencies in current trust relationships highlight the need for a flexible, general-purpose trust management system that can navigate these (possibly) complex

trust domains. A domain may need to support a range of different trust relationships and hence be capable of supporting different types of security policy [17].

Trust decisions are currently hard-coded into an application, which adds to the complexity of the application and the inability to adapt to changes in trust and lack of flexibility when setting up new relationships. A separation of the application's purpose and its trust management framework will offer a more scalable and flexible solution for the distributed environment.

Trust is a vital component in every business transaction. Customers must trust that sellers will provide the services they advertise, and will not disclose private customer information (name, address, credit card details, purchases etc.). Trust in the supplier's competence and honesty will influence the customer's decision as to which supplier to use. Sellers must trust that the buyer is able to pay for goods or services, is authorised to make purchases on behalf of an organisation or is not underage for accessing service or purchasing certain goods. Thus, for Internet commerce to achieve the same levels of acceptance as traditional commerce, trust management has to be an intrinsic part of e-commerce.

Trust is usually specified in terms of a relationship between a *trustor*, the subject that trusts a target entity, which is known as the *trustee* i.e. the entity that is trusted. Trust forms the basis for allowing a trustee to use or manipulate resources owned by a trustor or may influence a trustor's decision to use a service provided by a trustee. Thus, trust can form an important factor in decision-making [18-20]. The level of trust has an approximate inverse relationship to the degree of *risk* with respect to a service or an e-commerce transaction [21-23], but there has been very little work on using risk management frameworks for trust management or on the analysis of the exact relationship between risk and trust. In many current business relationships, trust is based on a combination of *judgement* or *opinion* based on face-to-face meetings or *recommendations* of colleagues, friends and business partners. However, there is a need for a more formalised approach to trust establishment, evaluation and analysis to support Internet services, which generally do not involve human interaction.

In this survey we focus on trust in the context of networked and distributed computing systems, in which it is not just the remote system that needs to be trusted, but also interactions over underlying services such as communication services. Our focus is on modelling trust, so that it can be used in automated systems. Work on the creation of computer frameworks of the entire social concept of trust, though pertinent, is not our emphasis.

There is considerable variation in the meaning of *trust* as used in the literature. In the following section we review the various definitions and suggest a definition applicable to internet services and then discuss trust properties and relationships in more detail in section 3. In section 4 we classify the different types of trust we have identified in the literature. In section 5 we introduce the concept of trust management and in section 6 we describe examples of trust management solutions in more detail. We elaborate on some of the application areas for trust management in section 7, followed by a summary of the key ideas and future research directions.

2 DEFINING TRUST

Trust is a complex subject relating to belief in honesty, truthfulness, competence, reliability etc. of the trusted person or service. There is no consensus in the literature on what trust is and on

what constitutes trust management [21, 23], though many research scientists recognise its importance [24, 25]. The significance of incorporating trust in distributed systems is that trust is an enabling technology. Its inclusion will enable Internet commerce and seamless, secure agent-based applications. Despite the need to standardize trust and its related concepts, many researchers simply use and assume a definition of trust in a very specific way relating to topics such as authentication, or ability to pay for purchases. However, a few authors have tried to view trust in a generic way.

Kini and Choobineh [26] in their considerations on the theoretical framework of trust, examine it from the perspectives of personality theorists, sociologists, economists and social psychologists. They state that trust, as defined in the Webster dictionary, is:

- An assumed reliance on some person or thing. A confident dependence on the character, ability, strength or truth of someone or something.
- A charge or duty imposed in faith or confidence or as a condition of a relationship.
- To place confidence (in an entity).

They highlight the implications of these definitions and combine their results with the social psychological perspective of trust to create their definition of trust in a system – "a belief that is influenced by the individual's opinion about certain critical system features" [26]. Their discussion, though general in concept, concentrated on human trust in Electronic Commerce, but did not address trust between the entities involved in an E-Commerce transaction.

The European Commission Joint Research Centre defines trust as "the property of a business relationship, such that reliance can be placed on the business partners and the business transactions developed with them" [27]. This view of trust is from a business management perspective and offers an interesting analysis of what must be done to enable trust in E-Commerce. They state that the issues of the identification and reliability of business partners, the confidentiality of sensitive information, the integrity of valuable information, the prevention of unauthorized copying and use of information, guaranteed quality of digital goods, availability of critical information, the management of risks to critical information, and the dependability of computer services and systems (specifically the availability, reliability and integrity of infrastructure, the prevention of unauthorised use of infrastructure, guaranteed level of services and the management of risks to critical infrastructure) are key to the emergence of E-Commerce as a viable commercial activity.

The Oxford Reference Dictionary states that trust is "the firm belief in the reliability or truth or strength of an entity". A trustworthy entity will typically have a high reliability and so will not fail during the course of an interaction, will perform a service or action within a reasonable period of time, will tell the truth and be honest with respect to interactions and will not disclose confidential information. *Competence* is a better term than *strength* for the environment related to services and computing system, i.e. an entity should be capable of performing the functions expected of it or the service it is meant to provide correctly and within reasonable timescales. Thus, trust is really a composition of many different attributes – reliability, dependability, honesty, truthfulness, security, competence, and timeliness, which may have to be considered depending on the environment in which trust is being specified.

Trust is a vast topic that incorporates trust establishment, trust management and security concerns. The lack of consensus with regards to trust, has led authors to use the terms trust, authorisation and authentication interchangeably. The outcome of a trust decision is based on many things such as the trustor's propensity to trust, its beliefs and past experiences relating to the trustee.

Authorisation can be seen as the outcome of the refinement of a more abstract trust relationship. For example, if I develop a trust relationship with a particular student, I may authorise him to install software on my computer and hence set up the necessary access control rights to permit access. We define *authorisation* as a policy decision assigning access control rights for a subject to perform specific actions on a specific target with defined constraints.

Authentication is the verification of an identity of an entity, which may be performed by means of a password, a trusted authentication service or using certificates. There is then an issue of the degree of trust in the entity, which issued the certificate. Note that authorisation may not be necessarily specified in terms of an identity. Anonymous authorisation can be implemented using capabilities or certificates [28].

We define *trust* as "the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context" (assuming dependability covers reliability and timeliness).

Distrust may be a useful concept to specify as a means of revoking previously agreed trust or for environments when entities are trusted, by default, and it is necessary to identify some entities which are not trusted. We define *distrust* as "the lack of firm belief in the competence of an entity to act dependably, securely and reliably within a specified context".

We now examine some of the properties of trust relationships in more detail.

3 PROPERTIES OF TRUST RELATIONSHIPS

In general, a trust relationship is not absolute – A will never trust B to do any possible action it may choose. A trustor trusts a trustee with respect to its ability to perform a specific action or provide a specific service within a *context*. For example, a person is only trusted to deal with financial transactions less than $2000 in value. Even trust in oneself is not usually absolute and there is a need to protect resources you own from mistakes or accidents you may cause. Examples include protecting files from accidental deletion or mechanisms to prevent a person driving a car when under the influence of alcohol.

A trust relationship can be one-to-one between two entities, however it may not be symmetric. A's trust in B is not usually the same as B's trust in A. It may be a one-to-many relationship in that it can apply to a group of entities such as the set of students in a particular year. It can also be many-to-many such as the mutual trust between members of a group or a committee, or many-to-one such as several departments trusting a corporate head branch. In general, the entities involved in a trust relationship will be distributed and may have no direct knowledge of each other so there is a need for mechanisms to support establishment of trust relationships between distributed entities.

There have been suggestions that trust relationships should not be transitive [23], however, some trust scenarios do exhibit transitivity. The concept of trust delegation is a prime example of the application of trust transitivity. When I delegate my trust decisions to another, for example John, I authorize John to make trust decisions on my behalf. Thus, when I delegate to John and John trusts an unknown entity (say Tim), John is essentially stating that I trust Tim. According to Christianson and Harbison in [29] the concept of transitivity should be avoided, as it can result in entity B adding trust assertions to an entity A's trust base without A's explicit consent leading to *unintentional transitivity*. We agree that transitivity of trust may have unexpected and adverse results if it implies updating the trust base of a trustor to include derived assertions, but it is may be necessary in some situations. We consider transitivity to be inherent in some relationships and so should be considered in the analysis of trust systems in order to determine which undesired side effects should be prevented.

There is often a *level* of trust associated with a relationship [30]. Some entities may be trusted more than others with respect to performing an action. It is not clear whether this level should be discrete or continuous. If discrete values are used, then a qualitative label such as high, medium or low may be sufficient. Some systems support arithmetic operations on trust recommendations so numeric quantification is more appropriate. It is also possible to provide a mapping from qualitative to numeric labels. However, there is still a problem relating to representation of ignorance (or the unknown) with respect to trust.

Jøsang's Opinion Model, based on subjective logic, may be a suitable technique for assigning trust values in the face of uncertainty [31-34]. An opinion is a representation of a belief and is modelled as a triplet, consisting of: b (a measure of one's belief), d (a measure of one's disbelief) and i (a measure of ignorance); such that $b + d + i = 1$. It is assumed that b, d and i are continuous and between 0 and 1 (inclusive). This model's strength lies in the ability to reason about the opinions (on a mathematically sound basis) and its consensus, recommendation and ordering operators [31]. However, its major weakness is that it cannot be guaranteed that users will accurately assign values appropriately.

From the literature it is clear that there are many different types of trust, which relate to the specific purposes or nature of a trust relationship. In the following section we provide a classification of the types of trust.

4 TRUST CLASSIFICATION

We have identified different forms of trust in the literature relating to whether access is being provided to the trustor's resources, the trustee is providing a service, trust concerns authentication or it is being delegated. This is not meant to be an exhaustive taxonomy, but merely a useful way of classifying the literature relating to trust in internet services.

4.1 Access to a Trustor's Resources

A trustor trusts a trustee to use resources that he owns or controls, which could be a software execution environment or an application service [35-37]. Abrams and Joyce [35] highlight the fact that resource access trust has been the focus for security specialists for many decades, although the emphasis has mostly been on mechanisms supporting access control.

There is an obvious distinction between trusting an entity to read or write a file on your server and trusting an entity to execute code within your workstation. Simple file access requires that the trustee will follow the correct protocol, will not divulge information read, and will write only correct data etc. Allowing an entity to execute code on your workstation implies a much higher level of trust. The code is expected not to damage the trustor's resources, to terminate within reasonable finite time and not to exceed some defined resource limits with respect to memory, processor time, local file space etc.

In [35, 36], the authors implicitly map trust decisions to access control decisions. Generally, resource access trust can form the basis for specifying authorisation policy, which then is implemented using operating system or database access control mechanisms, firewall rules etc. The trust relationship can be refined into authorisation policies that specify actions the trustee can perform on the trustor's resources and constraints that apply, such as time periods for when the access is permitted.

Examples of Resource Access Trust

- Fred is trusted to do Linux installations and Joe is trusted to do NT installations on our section workstations.
- Third year and above students are trusted to use the parallel processing service.
- I trust XY Cleaners to send someone to clean my house even when I am not there.
- I distrust AB Garage so I will not take my car to be repaired there.

These rather abstract specifications of trust and distrust would need to be refined into specific authorisations policies that define permitted operations to specific resources.

4.2 Provision of Service by the Trustee

The trustor trusts the trustee to provide a service that does not involve access to the trustor's resources. Note this may not be true of many services such as web services that download applets and cookies, and so do require access to resources owned by the trustor.

Service bureaux and application service providers (ASPs) [38-40] are prime examples of entities that would require service provision trust to be established. Currently, in these domains, trust is often an unstated implication of establishing a relationship, which is difficult to enforce or monitor. Mobile code and mobile agent based applications obviously must trust the execution environment provided by the remote system (provision of service trust) but the execution environment should not be damaged by the mobile code (access to resources trust).

Examples of Service Provision Trust

- I trust a film recommendation service to only recommend films that are not pornographic.
- I trust website xyz to provide information that is non-offensive.
- I distrust sexy-Susan website.

The above examples are a form of *confidence trust* in that the trustor has confidence in (or specifically distrusts) the standard of service provided by the service provider. This type of trust maps into a form of access control, which is subject-based, in that the subject is only permitted to access trusted services. This type of access control can be implemented by some web browser as a means of screening sites visited by children [1, 41-45].

Some forms of service trust relate to *competence* of the trustee:

- I only trust fourth year students who have an aggregate A grade to do this project.
- I will only purchase PCs from Company ABC.

A trustor's trust in the competence of the trustee's ability to provide a service differs from confidence trust in that, confidence applies to entities the trustor will use and competence applies to entities that perform some action on behalf of the trustor.

Another form of service trust relates to *reliability* or *integrity* of the trustee. In E-Commerce and E-Banking, the customer trusts the vendor or bank to support mechanisms that will ensure that passwords are not divulged and to prevent transactions from being monitored. The vendor or bank is also trusted to maintain the *privacy* of any information such as name, address and credit card details, which it holds about the customer. There have been some high-profile incidents in the UK recently where this trust has been broken. Examples of this form of service trust are:

- I will store these critical files on Groucho (as it has a RAID file system and it is archived every 2 hours). Note that in this case the trustee does have access to the trustor's resources.
- I trust the newsagent to email me an electronic newspaper every morning before 8am.
- I trust my internet bank not to divulge my name and address to companies for electronic marketing.

4.3 Certification of Trustees

This type of trust is based on certification of the trustworthiness of the trustee by a third party, so trust would be based on a criteria relating to the set of certificates presented by the trustee to the trustor. Certificates are commonly used to authenticate identity or membership of a group in Internet applications [28, 46-51]. This may imply competence if the identity is a well-known organisation. However, professional certification is a common technique used to indicate competence in the medical world, commerce and engineering so could be applied to Internet services [23].

Trustee Certification Examples

- I trust Dr. Tom's medical advice site as he is registered with the BMA.
- I will only use downloaded software updates, which have Microsoft certificates.
- I trust only VeriSign to certify programs that can run on my machine.
- I trust anyone with a PGP certificates signed by two people I trust (each must have an average level of trust in my view).
- I trust the identity of anyone authenticated by the Kerberos server in my domain.

Note that the certification authority is in fact providing a trust certification *service* so this is a special form of service provision trust but involves a third party in establishing the trust. There are many papers discussing this specific form of trust service, which is the reason we define it as a separate classification.

4.4 Delegation

A trustor trusts a trustee to make decisions on its behalf, with respect to a resource or service that the trustor owns or controls [52]. This is also a special form of service provision – a trust decision-making service.

Ding and Peterson [52] illustrate a novel way of implementing delegation, with hierarchical delegation tokens. Their work relies heavily on cryptography. They propose a classification of delegation schemes, with appropriate protocols, which they analyse, based on efficiency, and compare with related work. The ideas they express represent lower-level mappings from our concept of delegation in that they concentrate on access control.

Delegation Examples

- I trust my database manager to decide who has access to my database.
- I delegate all decisions concerning my investments to my financial advisor.
- I accept anonymous authorisation certificates for access to my resources issued by the WXYZ authorisation service.

4.5 Infrastructure Trust

This refers to the base infrastructure that the trustor must trust [35-37]. He must trust himself (implicit trust). He should be able to trust his workstation, local network and local servers, which may implement security or other services in order to protect his infrastructure.

It was recognised in early computing that in order to incorporate security (actually resource access trust) into applications, there was a need not only to implicitly trust the reference monitor, but also the administrative procedures that kept the monitor working. The culmination of this work was the U.S. Department of Defense specification for a set of resources, known as the Trusted Computing Base (TCB) [53] that had to be trusted by all applications executing on a machine to support the required security policy. The TCB can be viewed as the set of hardware, firmware and software elements, which are used to implement the reference validation mechanism i.e. the "validation of each reference to data or programs by any user (or program) against a list of authorized types of reference for that user". The TCB was seen as the primary component of a trusted computer containing all of the system elements supporting the isolation of objects (code and data) on which the protection is based. It was aimed more at centralised systems implementing information labelling and preventing information flow to unauthorised users, rather than commercial or networked systems.

Over the years, the TCB has increased in terms of number of components, and therefore size, leading to a higher probability of it being compromised. To make the PC platform more trustworthy, an initiative was launched to develop and formalize a trusted PC framework [24].

This initiative is being led by the Trusted Computing Platform Alliance, an amalgamation of several leading technology companies and research centres.

Infrastructure Trust Examples

- I trust hardware that has been certified by the Trusted PC Computer Base Certification Board.
- The PC's application software trusts the operating system.

5 TRUST MANAGEMENT

The paper by Blaze et. al. [54] was one of the first to introduce the term *trust management*, although prior security solutions for networked applications had an implicit notion of trust management based on PGP [47] or X.509 public-key certificates [46, 48], which are discussed in section 6. Blaze et. al. defined trust management as "a unified approach to specifying and interpreting security policies, credentials, relationships which allow direct authorisation of security-critical actions" [54]. They have implemented several automated trust management systems of note, namely: PolicyMaker [55-58], KeyNote [59-61] and REFEREE [41, 42] (in conjunction with Yang-Hua Chu and the W3C Consortium at MIT). These are discussed in more detail in section 6.

A common flaw with all these solutions is that they are used to identify a static form of trust, usually at the discretion of the application coder (that is, the programmer inserts code to evaluate trust, often at the start of a session). However, trust can change with time. Typically a customer uses an unknown service provider with some trepidation but if the service provided is high quality over a period of time, the customer's trust in the service provider increases. Current solutions do not address this problem of trust changing with time. In order to handle this dynamic property of trust, solutions should have a notion of learning. They should be able to adapt to the changing conditions of the environment in which the trust decision was made. However, current solutions have no notion of incorporating their own experiences (or that of others) in their decision-making process.

Additionally, these systems unconditionally accept credentials offered by the trustee (client) and then decide what the client is permitted to do. However, that may not be what the client currently wants to do. Even though there may be a relationship between the trustor and trustee, the trustee may wish to function in some other capacity than previously agreed upon. Thus, there is a need for negotiation in the trust decision process.

Systems change and evolve so there is a need to monitor trust relationships to determine whether the criteria on which they are based still apply. This could also involve the process of keeping track of the activities of the trustee and of determining the necessary action needed when the trustee violates the trustor's trust.

Trust management is thus concerned with collecting the information required to make a trust relationship decision, evaluating the criteria related to the trust relationship as well as monitoring and re-evaluating existing trust relationships. None of the existing systems cover monitoring and re-evaluation of trust.

6 TRUST MANAGEMENT SOLUTIONS

Most trust management systems focus on protocols for establishing trust in a particular context. Some make use of a trust policy language to allow the trustor to specify the criteria for a trustee to be considered trustworthy.

6.1 Public Key Certificates

A digital certificate is issued by a certification authority and verifies that a public key is owned by a particular entity [47]. The certification authority does not vouch for the trustworthiness of the key owner, but simply authenticates the owner's identity. This is necessary to establish a resource access or service provision trust relationship and may implicitly reduce the trustor's risk in dealing with the trustee [23]. However, the policy governing what resources or services the trustee is permitted to access is not handled by the certificate infrastructure, but is left up to the application. Two of the main certificate systems dealing with authentication, PGP and X.509, are described below.

The PGP trust model [47] is used for authentication relating to electronic mail type of applications between human users. It supports a *Web Of Trust* model in that there is no centralised or hierarchical relationship between certification authorities as with X.509. The underlying assumptions of the model are that a trustor may trust other entities, may validate certificates from other entities or may trust third parties to validate certificates. An *introducer* is an entity that signs someone else's public key (and thus vouches for a name-public key binding). A *meta-introducer* can sign keys as well as specify who is a (trusted) introducer. Thus, any entity can function as a certification authority. Every key that a user trusts or signs has to have a degree of trust associated to it, namely: unknown, untrusted, marginally trusted or completely trusted. It is also assumed that a user has an implicit trust (the highest form of trust in this model) in her own key. It is possible to use these labels to specify complex criteria about the trustworthiness of keys. For example, a user can specify that she only completely trusts a key if it is marginally trusted by a meta-introducer and completely trusted by a (trusted) introducer. Once keys are registered (along with their degree of trust) with the PGP system, then it computes a validity score (this measures how sure are we that this key belongs to this person). It is now the responsibility of each entity to query the system and to acquire keys as they are needed.

The X.509 trust model [46, 48] is a strictly hierarchical trust model for authentication. Each entity must have a certificate that is signed by the central certification authority or another authority, which has been directly or indirectly certified by it. This model assumes that certification authorities are organised into a universal "certification authority tree" and that all certificates within a local community will be signed by a certification authority that can be linked into this tree [54].

It is important to note that neither of these models can be used to model trust in all domains. Due to PGP's lack of official mechanisms for the creation, acquisition and distribution of certificates it is considered unreliable for E-Commerce, but appropriate for personal communication. X.509's rigid hierarchical structure may lead to unnatural business alliances between competing companies that violate the natural order of trust. Some applications, such as the reference information distribution systems described in section 7, need certificates to have a lifespan longer than is currently allowed by either scheme.

6.2 Platform for Content Selection (PICS)

PICS was developed by the World Wide Web Consortium (W3C) as a solution to the problem of protecting children from pornography on the Internet without infringing on one's right to freedom of speech. The basic idea behind PICS is that there needs to be a filter between the potential viewer and web documents. It is a result of the "Censorship of the Internet" debate that took place in the US legislature. PICS defines standards for the format and distribution of labels which are meta-document describing a web document. It does not specify a vocabulary that a label must use, nor does it state which labels are important for a particular circumstance. It is similar to stating "where on a package a label should appear, and in what font it should be printed, without specifying what it should say" [1] or what part of the label is important. A PICS-compliant application should be able to read PICS labels and use the user-defined filtering rules to decide whether to accept or reject the document. PICS makes no assumptions about the number of labels that can be attached to a document. In concept, a document may have several labels that may be issued by different organisations. A user has the right to choose any PICS filtering software and any label source (this entity is called a rating service).

A rating system defines the label attributes and their corresponding range of values used by the rating service. The following example is adapted from the W3C Recommendation on rating services and rating systems [44].

```
(   (PICS-version1.1)
    (rating-system "http://www.doc.worldwide.com/ratings/")
    (rating-service "http://www.doc.worldwide.com/descrip.html")
    (icon "icons/good.gif")
    (name "The Computing Department Rating System")
    (description "All about the rating of the pages offered by computing departments all over the world")
    ( category
        (transmit-as tc)
        (name "Teaching Material Content")
        (min 0.0)
        (max 5.0)
    )
    ( category
        (transmit-as rc)
        (name "Research Content")
        (label (name "very little") (value 0) (icon "icons/little.gif") )
        (label (name "a lot") (value 1) (icon "icons/lots.gif") )
    )
    ( category
        (transmit-as subject)
        (name "Document Subject")
        (multivalue true)
        (unordered true)
        (label (name "SE") (value 0) )
        (label (name "AI") (value 1) )
        (label (name "PC") (value 2) )
        (label-only)
    )
    ( category
        (transmit-as ref)
        (name "Number of references to other computing sites")
        (integer)
    )
```

```
    ( category
        (transmit-as importance)
        (min 0)
        (max 100)
    )
)
```

The first section identifies the version of PICS being used, the rating system and the rating service. The URL in the rating-system clause specifies the location of the document that has the human-readable description of the rating system. The URL in the rating-service clause identifies the document with the human-readable description of the rating service. This URL will also be included in all labels created by this service. The icon, name and description clauses are self-explanatory. The rest of the example specifies five label attributes; each identified by the keyword *category*. Each category has a transmission name and may be followed by clauses that define the attribute's allowable values. The BNF for the syntax of a rating system can be found in [44] and a sample PICS label that uses the above rating system is shown below.

```
(    (PICS-version1.1)
     "http://www.doc.worldwide.com/descrip.html"
     labels on "1998.11.05T08:15-0500"
     until "1999.09.32T23:34-0000"
     for "http://www-dse.doc.ic.ac.uk/~per/index.html"
     by "Joe Green"
     ratings (tc 1.0 rc "a lot" subject "SE" ref 19 importance 90)
)
```

The above example identifies the rating service that created the label, sets the lifespan of the label, identifies the page being labelled, the person labelling the page and the actual values of the label's attributes. More complex labels can be constructed using the PICS label syntax described in [44].

Miller et. al. [44] states that there are three ways that labels can be distributed, namely: they can be embedded in web documents (through the use of a META tag); they can be requested by a user (the HTTP GET is used to request both the label and the web document) and they may be requested separately from label bureaux.

The W3C has also published the PICSRules recommendation, which describes a rule-based, filtering policy language. We now give some policies expressed in PICSRules (version 1.1), adapted from [45].

```
(PicsRule-1.1
(
     Policy (RejectByURL (  "http://*@www.doc.ic.ac.uk*/*"
                            "http://*@www.yahoo.com*/*" )
          )
     Policy (AcceptIf "otherwise")
     )
)
```

The above example states that access to any Yahoo web page or any site at the department of computing at Imperial College is forbidden, but access to any other page is permitted. It does not use PICS labels.

```
(PicsRule-1.1
      (
            ServiceInfo (
                        name “http://www.raters.org/ratings/v1.html”
                        shortname “serv”
                        bureauURL “http://labelbureau.raters.org/Ratings”
                  )
            Policy (RejectUnless “(serv.pics)”)
            Policy (AcceptIf  “( (serv.pics > 3) and (serv.nudity = 0) )” )
            Policy (RejectIf “otherwise”)
      )
)
```

The above example sets the rating service in the ServiceInfo and uses the labels from the service to select pages. It states that all pages with labels must have the *pics* attribute in order for them to be viewed. Additionally, the pages must have a *pics* attribute with value of three or more and also the *nudity* attribute should be equal to zero in order to be allowed. All other pages will be rejected.

```
(PicsRule-1.1
      (
            name (
                        rulename “More Complex”
                        description “Highlight more features of PICSRules”
                  )
            source (
                        sourceURL
                              “http://www.complex.com/complex.html” )
                  )
            ServiceInfo (
                        name “http://www.doc.ic.ac.uk/ratings/v1.html”
                        shortname “DOC”
                  )
            ServiceInfo (
                        name “http://www.raters.org/ratings/v1.html”
                        shortname “serv”
                        bureauURL “http://labelbureau.raters.org/Ratings”
                  )
            Policy (RejectByURL (“http://*@www.badnews.com:*/*”
                                    “http:// *@www.baddernews.com:*/*”)
                  )
            Policy (AcceptByURL “http://*good-entertainment.org/plays*”)

            Policy (AcceptIf  “(DOC.educational = 1)”
                              Explanation “Always allow educational content”
                  )
            Policy (RejectIf  “(DOC.violence >= 4)”
                              Explanation “This is too scary”
                  )
            Policy (RejectUnless “(serv.graphics < 4)”)
            Policy (AcceptIf “otherwise”)
      )
)
```

In the above example, the name clause defines a human readable name for the rule and a description. The source clause specifies where the rule came from. The source URL may contain a document that has information about the rule. The first ServiceInfo clause specifies a rating service that the user wishes to use, giving it an alias. The absence of the bureauURL tag means that only embedded labels will be used. The other clauses reject pages from two sites, accept good plays, allow educational documents, reject documents with too much violence

(unless they are educational), block any page with too many graphics (with the exceptional of educational documents) and allow all other pages.

The BNF of the PICSRules syntax is outlined in [45]. PICSRules is a powerful tag-based language that allows resource-access trust on the Internet. The effectiveness of the PICS framework lies in the expressiveness of the filtering languages and the quality of the rating services.

6.3 AT& T PolicyMaker and KeyNote

PolicyMaker is a trust management application, developed at AT&T Research Laboratories, that specifies what a public key is authorised to do [54]. Traditional certificate frameworks such as PGP and X.509 do not bind access rights to the owner of the public key within the certificate framework. Schemes such as these require a two-step process: a) the binding of a public key to its owner, which occurs within the certificate framework, and b) the binding of access rights to the identified key owner, which occurs outside the certificate framework. In PolicyMaker, both occur in a single step that binds access rights to a public key.

The PolicyMaker system is essentially a query engine which can either be built into applications (through a linked library) or run as a "daemon" service. It evaluates whether a proposed action is consistent with local policy [1]. The inputs to the PolicyMaker interpreter are the local policy, the received credentials and an action string (which specifies the actions that the public key wants to perform). The interpreter's response to the application can either be yes or no or a list of restrictions that would make the action acceptable. A policy is a trust assertion that is made by the local system and is unconditionally trusted by the system. A credential is a signed trust assertion made by other entities and the signatures must be verified before the credentials can be used. Policies and credentials are written in an assertion language. The syntax of an assertion is:

Source ASSERTS *AuthorityStruct* WHERE *Filter*

Source represents the source of the assertion, *AuthorityStruct* represents the public key(s) to whom the assertion is applicable and *Filter* is the predicate that action strings must satisfy for the assertion to hold. Filters are interpreted programs that can accept or reject action strings. Note that Policymaker does not stipulate that a particular filter language (or assertion language) has to be used. Any safe interpreted language can be used to implement either of these languages. Filter programs take as input, the current action string and the environment, which contains information about the current context (e.g. date, time, application name, etc.). This environment can be used by the filter to enforce contextual constraints such as expiration times. A filter also has access to information about the rest of the chain in which it is being evaluated, which makes it possible to design certificates that limit the degree to which their authority can be deferred. Although the filter language interpreter is external to PolicyMaker, the name of the language is given in assertions and must be known by anyone who needs to use the assertion. Any unknown or unsupported filter languages are ignored by PolicyMaker.

The prototype for PolicyMaker had three associated assertion languages: AWKWARD (a safe version of AWK), Java and Safe-TCL. It was hoped that leaving the assertion language an open issue would mean flexibility and greater programmability for PolicyMaker. However, it was realised that the choice of an assertion language can affect the decision processing in

PolicyMaker. For a local policy, the source is always *policy*. The following policy specifies that any doctor who is not a plastic surgeon should be trusted to give a check-up.

```
policy
  ASSERTS doctor_key
  WHERE filter that allows check-up if the field is not plastic surgery
```

For Policymaker to make a decision there must be at least one policy in the input supplied to it from the trust base. The following credential states that the BMA asserts that the person with key "0x12345abcd" is not a plastic surgeon.

```
BMA_key
  ASSERTS "0x12345abcd"
  WHERE filter that returns "not a plastic surgeon", if the field is not plastic surgery
```

An assertion (whether policy or credential) states that the source trusts the public keys in the authority structure to be associated with action strings that satisfy the filter. It is important to note that assertions can modify the action strings that they accept, through the use of *Annotations*. Annotations are essentially a mechanism for communication between assertions (inter-assertion communication), as well as communication between the application and the credentials. This allows PolicyMaker to append conditions to the action strings, if necessary. A query to the PolicyMaker interpreter has the following format:

key_1, key_2, key_3, …… REQUESTS *ActionString*

To check if "0x12345abcd" is allowed to give me a check-up, the interpreter is asked:

"0x12345abcd" REQUESTS "do check-up"

The semantics of the action string is not known to PolicyMaker. The processing of the action strings, as well as signature verification, is left entirely up to the calling application. Action strings are generated and interpreted by the calling applications. The filters however should have knowledge of the action strings. The fact that signature verification is done by the calling application means that any signature scheme can be used, once the application provides the appropriate programs to perform the verification. This allows Policymaker to exploit existing signature schemes. PolicyMaker uses the credentials given to it to prove that the requested action complies with the policy (this process is referred to as compliance checking [55]). In summary, an application gives the PolicyMaker engine, a (set of) requested action(s), a set of credentials and a policy and the engine tries to prove that the credentials contain a proof that the requested action(s) complies with the policy.

KeyNote, the successor to PolicyMaker, was developed to improve on the weaknesses of PolicyMaker by AT&T Research Laboratories. It has the same design principles of assertions and queries [56, 60, 61] but includes two additional design goals, namely: standardisation and ease of integration [61]. In KeyNote, more is done in the trust management engine, rather than in the calling application (as was the case in PolicyMaker). Signature verification is done in the KeyNote engine and a specific assertion language is used. PolicyMaker allowed any choice of assertion language, which made its compliance checker difficult to integrate with applications. KeyNote's predefined assertion language allows simpler integration with its compliance checker. The KeyNote engine is passed a list of credentials, policies, the public keys of the requester and an "Action Environment" (which is essentially a list of attribute-value pairs) by the calling application. The action environment is generated by the application and contains all the

information relevant to the request, and so accurately reflects the application's security requirements. Identifying the attributes of this environment is the essential task in integrating KeyNote into any application. The result of the KeyNote evaluation process is an application-defined string, the simplest response being "authorized". The KeyNote assertion format is similar to email headers and is outlined in [61]. An example of a KeyNote assertion, taken from [56], is:

```
KeyNote-Version: 1
Authorizer: rsa-pkcs-hex:"1023abcd"
Licensees: dsa-hex "986512a1" || rsa-pkcs1-hex:"19abcd02"
Comment:  Authorizer delegates read access to either of the Licensees
Conditions: ($file == "etc/passwd" && $access == "read") -> {return "ok2}
Signature: rsa-md5-pkcs1-hex:"f00f5673"
```

As in PolicyMaker, assertions can either be policies or credentials. POLICY in the Authorizer field identifies policies which are locally trusted and so do not need a signature. The Licensees field specifies the principal(s) to which authority is given. A simple, lightweight assertion language with no loops or recursion is used in order to enforce resource usage restrictions, to allow the assertions to be easily understood by humans and easily refined from high-level languages, etc. [56]. Compliance checking in PolicyMaker required repeated evaluation of assertions, along with an arbitrated "blackboard" for storage of intermediate results and communication between assertions, while compliance checking in KeyNote involves a depth-first search that tries (using recursion) to satisfy at least one policy. KeyNote has no inter-assertion communication mechanisms. Satisfying an assertion entails satisfying both the Conditions and Licensees fields. The current implementation of the KeyNote Toolkit is written in C. Neither system addresses the problem of how to discover that credentials are missing, and neither system supports negative assertions. The authors claimed that both these systems are a more general solution to the trust management problem than public-key certificates. However, they address only authorisation based on public keys, which still does not comprehensively cover the entire trust management problem. They focus on establishing resource access trust, and possibly service provision trust.

6.4 Rule-controlled Environment For Evaluation of Rules and Everything Else (REFEREE)

REFEREE is a trust management system for making access decisions relating to Web documents developed by Yang-Hua Chu based on PolicyMaker [41, 42]. It considers a PICS label as the stereotypical web credential and uses the same theoretical framework as PolicyMaker to interpret trust policies and administer trust protocols, which are represented as software modules. Like PolicyMaker and KeyNote, REFEREE is a recommendation-based, query engine so it needs to be integrated into a host application. It evaluates requests and returns a tri-value and a statement-list, which is the justification for the answer. A tri-value is either true, false or unknown. *True* means "yes, the action may be taken because sufficient credentials exist for the action to be approved", *false* means "no, the action must not be taken because sufficient credentials exist to deny the action" and *unknown* means "the trust management system was unable to find sufficient credentials either to approve or deny the requested action". Boolean operators were modified to allow reasoning about tri-values and special operators were added to create a complete logical framework for tri-values. For example, true-if-unknown and false-if-unknown operators were defined to simulate negation of the *unknown* value. An ordered statement-list specifies information acquired during the execution of modules. They are the means by which inter-

module communication takes place. All statements are "two element s-expressions", similar to attribute value pairs. The first item specifies the context of the statement and the second stating the statement's content. For example, the statement that John is untrustworthy in a certification REFEREE module would be:

```
(  ( "certification module") ( "John" (untrustworthy yes) ) )
```

The REFEREE system is essentially a collection of modules as basic building blocks, each dealing with a particular policy decision. A module can delegate subtasks to other modules and make decisions based on the returned assertions. All modules have the same interface as REFEREE – they accept inputs and return a tri-value and a statement-list. The inputs are an action name and other arguments that provide information about the action and form the module's trust base. For example, a content selection module may have either a URL or the keys of the raters that make assertions about the URL as its input and its output is a tri-value with a statement-list. At the implementation level, a module consists of a policy and zero or more interpreters. A policy is a code segment written in a trust policy language and the interpreters are programs for interpreting the policy or other interpreters. The set of interpreters in a module is hierarchical; the module policy is interpreted by the highest-level interpreter; which in turn is interpreted by a lower-level interpreter and so on.

REFEREE goes through two phases in its lifetime. In the bootstrap phase, the host application gives it the unconditionally trusted assertions and a module database. A module database is a repository of action names, similar to a DNS server in that it allows a module to be referred to by an action name. In the query phase, the host application provides the action and other arguments such as credentials, which are passed onto the appropriate module from the module database. REFEREE runs the module's interpreter with the policy and list of arguments, which may result in other modules being invoked, then returns an answer to the host application.

Profiles-0.92 is rule-based trust policy language, designed to work with REFEREE, in which each rule is an s-expression with an operator as the first element followed by operands. Rules are evaluated top down and the returned value of the last rule is the policy's returned value. Rules return tri-values and statement-lists. The BNF for the syntax can be found in [42]. The following policies highlight some features of this language.

```
(     threshold-and
          2
              (not (url-match URL ("http://www.cam.ac.uk" "http://www.bath.ac.uk")))
              (url-match URL ("http://www.ic.ac.uk"))
              unknown
)
```

The above policy states that all material from Cambridge University and the University of Bath will be blocked, and only material from Imperial College will be automatically downloaded. The user will be prompted about all other material.

```
(     invoke "load-label" STATEMENT-LIST URL
                    "http://web.mit.edu/ratings/CodeSafety.html"
                    ("http://bureau.mit.edu" "http://bureau.cmu.edu")
)
```

```
(     match
                ((“load-label”)
                (((version “PICS-1.1”) *
                  (service “http://web.mit.edu/ratings/CodeSafety.html”) *
                  (ratings  (RESTRICT > virus 8))
                )))
 STATEMENT-LIST
)
```

This policy states that labels from the MIT and CMU bureaus should be used and only pages with labels that state that the document has been thoroughly checked for viruses can be downloaded. For this example, the invoke clause runs the load label module, which loads the labels from the bureaus. The match clause searches all the labels for the pattern described.

6.5 IBM Trust Establishment Framework

IBM views trust establishment as the enabling component of E-Commerce [62]. They state that the underlying trust implications involved in an e-business transaction can be solved using certificates. Certificates can be issued by various bodies, vouching for an entity in a particular role, for example vouching for someone’s status as a buyer or seller or both. IBM has developed a role-based access control model that uses certificates, a Java-based Trust Establishment module and a Trust Policy Language (TPL). Their system is similar to PolicyMaker, but permits negative rules preventing access. The default certificate scheme used is X.509 v3, though other certificate formats are supported. The Trust Establishment module validates the client’s certificate and then maps the certificate owner to a role. The certificate need not bind to a user’s identity, but could just state that the user is an employee of Company XYZ or a public key can be used to map onto an anonymous role. Local policy specified in their TPL defines what a role is permitted to do. The syntax for TPL is written in XML and is described in [62, 63]. The primitive structure in TPL is a group. For each group, there are rules governing group membership. These rules essentially specify which certificates to check. The following example is taken from [63].

```
<POLICY>

    <GROUP NAME=”self”>
    </GROUP>

    <GROUP NAME=”partners”>
        <RULE>
            <INCLUSION ID=”partner” TYPE=”partner” FROM “self”>
            </INCLUSION>
        </RULE>
    </GROUP>

    <GROUP NAME=”departments”>
        <RULE>
            <INCLUSION ID=”partner” TYPE=”partner” FROM=”partners”
            </INCLUSION>
        </RULE>
    </GROUP>
```

```
    <GROUP NAME="customers">
        <RULE>
            <INCLUSION ID="customer" TYPE="employee" FROM="departments">
            </INCLUSION>
            <FUNCTION>
                <GT>
                    <FIELD ID="customer" NAME="rank"></FIELD>
                    <CONST>3</CONST>
                </GT>
            </FUNCTION>
        </RULE>
    </GROUP>

</POLICY>
```

The first group defined is the originating *retailer*. Then, it is stated that entities that have *partner* certificates, signed by the original retailer, are placed in the group partners. The group *department* is defined as any user having a *partner* certificate signed by the partners group. Finally, the *customer* group consists of anyone that has an *employee* certificate signed by a member of the departments group who has a rank greater than 3. In summary, the policy states that a customer is an employee of a department of a partner company. After the Trust Establishment module has determined that an entity can be assigned to a particular role, it then sends this information to another module, which stipulates the access rights that are bound to the particular role.

6.6 Logic-Based Formalisms of Trust

Trust involves specifying and reasoning about beliefs. Forms of first order predicate logic [64-67] or (modified) modal logic [68-71] have been used to represent trust and its associated concepts. A logic used to formalise trust must represent actions and interactions to cater for distributed agents [72, 73].

Simple relational formalisms are used to model trust with statements of the form $T_a\ b$, which means "*a* trusts *b*" [64-67]. Each formalism extends this primitive construct with features such as temporal constraints and predicate arguments. Given these primitives, traditional conjunction, implication, negation and disjunction operators, these logical frameworks express trust rules (such as trust is not transitive) in their language and reason about these properties. In our opinion these simple formalisms are not capable of modelling the complex trust relationships found in the Internet.

Burrows, Abadi and Needham [65] propose a language to specify the steps followed in the authentication process between two entities (resource access protocol analysis). The language is founded on cryptographic reasoning with logical operators defined to deal with notions of shared keys, public keys, encrypted statements, secrets, nonce freshness and statement jurisdiction (for authentication servers and certificate authorities). It should be possible to answer the following questions about a protocol specified in this language:

- What does this protocol achieve?
- Does this protocol need more assumptions than another one?
- Does this protocol do anything unnecessary that could be left out without weakening it?
- Does this protocol encrypt something that could be sent in clear without weakening it?

The language can be used to specify protocol assumptions and interactions and appears to be suitable for modelling trust establishment protocols. In designing the language many simplifying assumptions were made. As stated in [65], "Since we operate at an abstract level, we do not consider errors introduced by concrete implementations of a protocol, such as deadlocks, or even inappropriate use of cryptosystems. Furthermore, while we allow for the possibility of hostile intruders, there is no attempt to deal with the authentication of an untrustworthy principal, nor detect weaknesses of encryption schemes or unauthorised release of secrets. Rather, our study concentrates on the beliefs of trustworthy parties involved in the protocols and on the evolution of these beliefs as a consequence of communication." Also, analysis using this language requires a language expert to read through the logical statements, look at the implications and reason about them.

The Authorization Specification Language (ASL) by Jajodi, Samarati and Subrahmanian [66] is used to specify authorization rules and makes explicit the need for the separation of policies and mechanisms. ASL supports the specification of the closed policy model (all allowable accesses must be specified) and the open policy model (all denied accesses must be explicitly specified) using a common architectural framework. It also supports role-based access control. The separation of the concepts of policy and mechanism allows the specification and implementation of more flexible systems, as the access control model need not be hard coded into the system. This language is an excellent tool for the specification and analysis of resource access trust.

Modal logics can be used to express possibility, necessity, belief, knowledge, temporal progression and other modalities [68-71]. It is an extension of traditional logics (propositional and predicate). The □ (necessity) operator and the ◊ (possibility) operator are added to the traditional syntax. The notion of possible worlds (or multiple words) is fundamental to the interpretation of modal logics and simply states that a statement will have a different meaning depending on the world it is in. Kripke structures are used to represent possible worlds, where a Kripke structure consists of the set of all possible worlds and an accessibility relation (which may be referred to as a possibility relation depending on the modal logic). The accessibility relation states the conditions for which an agent can access a world.

In [70], Jones and Firozabadi address the issue of the reliability of an agent's transmission. They use a modal logic of action developed by Kanger, Porn and Lindahl to model agent's actions. For example, $E_i\ p$ means "agent *i* brings it about that *p*". They use a variant of a normal modal logic of type KD45 as the foundation for their belief system. For example, $B_i\ A$ means "agent *i* believes that *A*". The topic of institutional power is incorporated through the use of a *counts as* operator. Institutional power refers to the fact that a person performing an act in a particular institution will lead to the formation of an institutional fact. In a different institution, this fact cannot be established. For example, a minister performing a marriage ceremony at a church leads to the fact that two people are married; yet in a different church this fact will not exist. They adopt the relevant axiomatic schemas into their formalism and use their composite language to model various trust scenarios. For example, *b*'s belief that *a* sees to it that *m* is expressed as:

$$B_b\ E_a\ m$$

They also use their language to model the concepts of deception and an entity's trust in another entity. In their own words "We do not investigate the formal representations of procedures by means of which trust-relations between agents can be established. Assuming the existence of a

trust-relation, we try to make explicit the reasoning patterns characteristic of a trusting agent." This formalism can be easily modified to express and reason about establishing trust, of any particular context.

Rangan [71] views a distributed system as a collection of agents communicating with each other through message passing in which the state of an agent is it's message history (all sent and received messages). The state of the system is the state of all the agents in the system. He then devises conditions that function as his accessibility relation (in this context, possibility relation is a more accurate term). His model consists of simple trust statements (for example $B_i\ p$, which means "agent i believes proposition p") and properties such as transitivity, Euclidean property, etc. are defined. These constructs are then used to specify systems and analyse them with respect to the property (or properties) of interest. Rangan's model more fluently follows the traditional lines of modal logics of beliefs than does Jones and Firozabadi's, but the model is simpler in the sense of the non-treatment of actions and their effects.

After outlining the various attempts made to model trust, we take a look at how these attempts have been applied to specific domains.

7 APPLICATIONS OF TRUST MANAGEMENT

Most of the literature relating to trust applications really discusses security requirements relating to authentication, confidentially, data-integrity or non-repudiation rather than trust as we defined it. We have selected a few application domains that highlight some specific trust management requirements.

7.1 Medical Information Systems

Medicine has many sub-disciplines each with its own set of trust issues but based on a sound ethical foundation. The difficulty in computerising some aspects of medicine lies in transferring these ethics to the computer systems. The following discussion only considers Clinical Information Systems and Medical Document Distribution.

Clinical Information Systems is a relatively mature field. Countries such as the United States [74], Iceland [75] and Britain [76-79] have been developing and deploying these systems for many years. However, many of the systems in use, do not accurately reflect the trust issues in their particular field. There are three major problems with emerging medical systems, namely: electronic trust relationships not matching the relationship in the real world, a focus on centralisation and putting too much power in the hands of one body, and the lack of sufficient mechanisms to de-identify records.

Britain's National Health System has proposed a cryptography strategy to secure clinical systems. Anderson [80] describes the problems in their proposal, which are: insufficient attention to the safety aspects of clinical messaging, the use of an incorrect threat model, the inconsistency between the trust relationship in professional practice and that of the electronic trust relationship, the underestimation of encryption implementation costs, the prominence of key escrow in initial prototypes despite denial of its importance, the relegation of medico-legal issues to the background, the use of weak encryption mechanisms, problems with the protocol and certification mechanisms, and the lack of security expertise. Anderson also gives a more

general discussion of the security issues in medical practice in the United Kingdom in [81], where he highlights examples of compromises of deployed clinical information systems.

In an effort to define a reference security model for clinical information systems [82], Anderson looks at the ethical basis of clinical information systems, looks at the threat model and defines a security policy. In his security policy [83], he proposes nine principles that should be in every system that will ensure medical systems match their off-line counterparts. These principles are:

i. *Access control*: each identifiable clinical record shall be marked with an access control list naming the people or groups of people who may read it and append data to it. The system shall prevent anyone not on the access control list from accessing the record in any way.

ii. *Record opening:* a clinician may open a record with herself and the patient on the access control lost. Where a patient has been referred, she may open a record with herself, the patient and the referring clinician(s) on the access control list.

iii. *Control:* one of the clinicians on the access control list must be marked as being responsible. Only she may alter the access control list, and she may only add other health care professionals to it.

iv. *Consent and notification:* the responsible clinician must notify the patient of the names on his record's access control list when it is opened, of all subsequent additions, and whenever responsibility is transferred. His consent must also be obtained, except in emergency or in the case of statutory exemptions.

v. *Persistence:* no one shall have the ability to delete clinical information until the appropriate time period has expired.

vi. *Attribution:* all accesses to clinical records shall be marked on the record with the subject's name, as well as the date and time. An audit trail must also be kept of all deletions.

vii. *Information flow:* information derived from record A may be appended to record B if and only if B's access control list is contained in A's.

viii. *Aggregation control:* there shall be effective measures to prevent the aggregation of personal health information. In particular, patients must receive special notification if any person whom it is proposed to add to their access control list already has access to personal health information on a large number of people.

ix. *Trusted computing base:* computer systems that handle personal health information shall have a subsystem that enforces the above principles in an effective way. Its effectiveness shall be subject to evaluation by independent experts.

These principles address the majority of trust issues in the clinical information systems. However, issues like de-identification of patient records and centralisation require more focus.

WAX is "a system for publishing electronic medical books containing information such as treatment protocols, drug formularies and government regulations to which healthcare professionals need frequent access in support of clinical decision-making" [84]. The main security issue in WAX is to guarantee that the users can verify the author and publisher of the books. The initial solution involved using X.509 certificates as the primary trust mechanism and hashes of the catalogues as the secondary trust mechanism. In the development phase of the project, they discovered that because X.509 certificates lifetimes were shorter than the lifetime of a book. Issues arose as to what would happen when the certificate became obsolete. What does

certificate obsolescence mean? Does it differ from certificate revocation? Would another certificate have to be issued for a book, whose certificate is no longer valid? And if certificate re-issue was chosen, what would the effects be on the system? Given these questions, a decision was made to use the catalogue hashes as the primary trust mechanism. Anderson et. al. [85] defined a mechanism to incorporate this catalogue-based trust into the World Wide Web framework, by defining special HTML tags.

7.2 Information Retrieval Systems

From sections 6.2 and 6.4, we can see that attempts have been made to perform trust management in this domain. The primary trust questions that should be asked in this field are "Does this piece of information meet my viewing needs?", "Will I get the information I requested?" and "Will the information have any effects on my system?"

The issues arising from the first question are "How does one label a particular piece of information?" and "How does one specify one viewing requirements?" For the Internet, the de facto standard for describing information is a PICS label. To specify one's viewing tastes one can use any PICS-compliant filtering language, such as PICSRules or profiles-0.92. However, for other information retrieval systems, standards are yet to be established.

"Will I get the information I requested?" requires clarification. This question more specifically asks "Will I get an authentic copy of the information that I requested?" and "Will I get only the information I requested?" Many solutions to the former question have been created, using public key cryptography. However, we see the WAX system [85] taking a different approach and using hashes instead.

The latter question is a much more difficult question to answer and we have not come across any solutions. The question of the effects that information will have on one's system relates directly to active content. Mechanisms exist to minimise the risk of active documents, namely the use of signatures [86] so that only active documents from trusted sources are downloaded, but little has been done with respect to the enforcement of the policy of not allowing active content to affect a system.

It is also necessary to ensure that information retrieval systems do not disseminate information that is not for general distribution. Myers and Liskov in [87] describe a model that allows for the protection of private and secret data using labels, which state the flow of information in a decentralized system. They describe a language, Jif (Java Information Flow), which permits static checking of these labels. Their model assumes mutual distrust amongst the entities in the systems and Jif is assumed to run on a trusted execution platform that enforces the rules of the model (as stated in the labels). The emphasis in this work is on the prevention of information leakage from one level to another.

The only examples of systems that directly address trust in Information Retrieval Systems are REFEREE and PolicyMaker, which was integrated with PICS [1], as discussed in section 6.

7.3 Mobile code

Mobile agents migrate code and data from one machine to another to perform tasks on behalf of a user. There have been suggestions that this can be used for network management but one of

the most promising applications is a local agent which performs actions such as message filtering, email to voice translation, compression or information filtering on behalf of mobile users with limited capability computing devices and communicating via a wireless link. There are many applications that download mobile code, such as an applet or a servlet, to perform a complex transaction on behalf of a user on a remote machine. A mobile agent can be considered mobile code plus mobile data

The primary trust concern for mobile code relates to access to trustor resources as the mobile code could steal information from the server and damage or make excessive use of local resources. Another valid concern is the threat that servers pose to mobile agents (service provision trust) as they could modify or steal information contained in agents [4], delay their migration or even modify their code.

Emphasis in the literature is mostly on techniques for preventing malicious agents from harming their execution environment. The issue of trust has been sparsely mentioned. Wilhelm et. al. [5] gave one of the more comprehensive discussions on the issue of trust in mobile systems. They identified what they referred as the four foundations of trust, namely: blind trust, trust based on (a good) reputation, trust based on control and punishment and trust based on policy enforcement. Their solution to the trust in mobile agent systems problem was the CryPO protocol, based on tamper-proof hardware to provide tamper-proof environments, which are the foundation for the agent executor. Agents assert which environment manufacturers they trust. The protocol uses certificates and encryption technology to ensure security and is essentially an extension of the certification framework.

Mobile agents will not be expected to roam in only one domain. Realistically, there are a number of security domains, with differing structures. Thus, a mobile agent will be expected to be able to handle navigation between security domains. This raises the issue of secure interoperation. In an effort to solve this problem, Gong and Qian [88] describe a mathematical framework for secure interoperation. Swarup and Schmidt [17] also briefly discuss this issue, highlighting the trust management mechanisms, policy negotiation protocols and mobility protocols. No formal trust management framework has been discussed, designed or implemented in this field. Research in this field [5, 6, 89] is focused on formulating the best protocol to ensure that the mobile agent does not cause the server any harm. These protocols define the process of trust establishment, but the other components of trust are totally ignored.

Another trust issue concerns agent-to-agent interaction. The situation may arise where it is necessary for agents to cooperate to complete a task. How do agents trust each other? In [70], applied modal logic is used to define the characteristics of a trusting agent. This logical formalism of interacting trusting agents was discussed in Section 6.6. It is an interesting discussion on trust in agent-to-agent interactions.

8 SUMMARY AND FUTURE WORK

Our definition of trust as the firm belief in the competence of an entity to act dependably, securely and reliably within a specified context seems to cover most of the attributes needed for trust in internet applications. We have also provided a taxonomy of different categories of trust but often applications and solutions fall into more than one category. Trust management is concerned with collecting the information required to make a trust relationship decision,

evaluating the criteria related to the trust relationship as well as monitoring and re-evaluating existing trust relationships. Systems that evaluate trust are quite closely tied to systems that implement access control or authentication. Current solutions to the trust problem do not acknowledge that trust changes over time and thus have no mechanism for monitoring trust relationships to re-evaluate their constraints. Most solutions do not recognize the need for entities to learn from their experience in order to dynamically determine and adjust trust levels. Evaluation of trust, only when a relationship is set up does not cater for the evolution of the relationship from its stated initial purpose to accommodate new forms of interaction, which is common in business relationships. We intend to build a system that deals with these issues.

We already have a policy specification language called Ponder [15], which can be used to define authorisation and security management policies. We hope to extend this to allow the specification of more abstract and potentially complex trust relationships between entities and across organisational domains. We will use the policy refinement tools, being developed, to generate the Ponder policy specification. This can be translated into implementation mechanisms such as Windows security templates, firewall rules or Java security policy. It can also be used to generate event triggered obligation policies for security management, which can for example choose particular encryption techniques for connections based on trust requirements.

There is also a need for a toolkit to aid the establishment, analysis, reasoning and monitoring of trust relationships. The toolkit must support the concepts of trust quantification from third parties and delegation of trust decisions, which may be likely in automated trust systems. The analysis and reasoning must address issue such as is the trust consistent with application constraints, are there conflicts in the specification of what trust relationships apply to particular entities and what trust relationships are implicit? Current work on trust analysis has taken a simplistic approach and are not rigorous or complete enough to model current systems. We envisage the need to translate the trust into a logic framework in order to perform this analysis. Our current effort is in the development of a trust specification and analysis language called SULTAN for specifying and reasoning about trust relationships. Reasoning in this context involves querying the entities, relationships and the trust network to view the ramifications of trust decisions and highlight and eliminate trust conflicts. We intend to produce a comprehensive trust toolkit as part of a new collaborative initiative in Trusted *e*-service within the UK. Another objective of this initiative is to integrate concepts of risk-management, from the financial world, with our concepts of trust management.

9 ACKNOWLEDGEMENTS

We gratefully acknowledge our colleagues in the Policy Group at Imperial College who provided comments on early drafts of this report, namely, Nicodemos Damianou, Naranker Dulay, Emil Lupu, George Mournos and Alessandra Russo. We also acknowledge funding of a studentship from Microsoft and British Telecom.

10 REFERENCES

1. Blaze M., Feigenbaum J., Resnick P. and Strauss M., *Managing Trust in an Information-Labelling System,* European Transactions on Telecommunications, 1997, 8: p. 491-501. http://www.si.umich.edu/~presnick/papers/bfrs/Paper.ps

2. Blaze M., Feigenbaum J. and Lacy J., *Managing Trust in Medical Information Systems*, 1996, AT&T, http://citeseer.nj.nec.com/did/35925

3. Harrington S. J. and Ruppel C. P., *Telecommuting: a test of trust, competing values, and relative advantage,* IEEE Transactions on Professional Communication, 1999, 42(4): p. 223 - 239, http://ieeexplore.ieee.org/iel5/47/17506/00807960.pdf

4. Ordille J. J. *When agents roam, who can you trust?* in *First Annual Conference on Emerging Technologies and Applications in Communications*, 1996, http://ieeexplore.ieee.org/iel4/3740/10938/00502505.pdf

5. Wilhelm U. G., Staamann S. and Buttyan L. *On the problem of trust in mobile agent systems*, in *IEEE Symposium on Network And Distributed System Security*, 1998, San Diego, California. http://citeseer.nj.nec.com/pdf/139207; http://icawww.epfl.ch/buttyan/publications/NDSS98.ps

6. Feigenbaum J. and Lee P. *Trust Management and Proof-Carrying Code in Secure Mobile Code Applications: Position Paper*, in *DARPA Workshop on Foundations for Secure Mobile Code*, 1997, http://www.research.att.com/~jf/pubs/darpa-mobile.ps

7. Clark T. H. and Ho G. L. *Electronic intermediaries: trust building and market differentiation*, in *32nd Annual Hawaii International Conference on Systems Sciences*, 1999, Hawaii, http://ieeexplore.ieee.org/iel5/6293/16785/00772939.pdf

8. Holland C. P. and Lockett A. G. *Business trust and the formation of virtual organizations*, in *31st Annual Hawaii International Conference on System Sciences*, 1998, Hawaii, http://ieeexplore.ieee.org/iel4/5217/14260/00654821.pdf

9. Jøsang A. *Trust-based decision making for electronic transactions,* in *The 4th Nordic Workshop on Secure IT Systems (NORDSEC'99)*, 1999, Stockholm, Sweden: Stockholm University Report 99-005, 1999.

10. Iacono C. S. and Weisband S. *Developing trust in virtual teams*, in *13th Hawaii International Conference on System Sciences*, 1997, http://ieeexplore.ieee.org/iel4/5350/14595/00665615.pdf

11. Jarvenpaa S. L., Tractinsky N., Saarinen L. and Vitale M., *Consumer Trust in an Internet Store: A Cross-Cultural Validation,* Journal of Computer-Meditated Communication, 1999, 5(2), http://www.ascusc.org/jcmc/vol5/issue2/jarvenpaa.html

12. Jiawen S. and Manchala D. W. *Trust vs. threats: recovery and survival in electronic commerce*, in *19th IEEE International Conference on Distributed Computing Systems*, 1999, http://ieeexplore.ieee.org/iel5/6307/16865/00776513.pdf

13. Ketchpel S. P. and Garcia-Molina H. *Making trust explicit in distributed commerce transactions*, in *16th International Conference on Distributed Computing Systems*, 1996, http://ieeexplore.ieee.org/iel3/3771/11006/00507925.pdf

14. Su J. and Manchala D. *Trust vs. Threats: Recovery and Survival in Electronic Commerce*, in *19th International Conference on Distributed Computing Systems*, 1999.

15. Damianou N., Dulay N., Lupu E. and Sloman M. *The Ponder Policy Specification Language*, in *Policy 2001: Workshop on Policies for Distributed Systems and Networks*, 2001, Bristol, UK: Springer-Verlag LNCS 1995, http://www-dse.doc.ic.ac.uk/policies/

16. Khare R. and Rifkin A., *Trust Management on the World Wide Web,* Peer-reviewed Journal on the Internet, 3(6), http://www.firstmonday.dk/issues/khare/index.html

17. Swarup V. and Schmidt C. *Interoperating between Security Domains*, in *ECOOP (European Conference on Object-Oriented Programming) Workshop on Distributed Object Security*, 1998, Brussels, Belgium.

18. Amoroso E., Nguyen T., Weiss J., Watson J., Lapiska P. and Starr T. *Toward an approach to measuring software trust*, in *IEEE Computer Society Symposium on Research in Security and Privacy*, 1991, http://ieeexplore.ieee.org/iel2/349/3628/00130788.pdf

19. Jøsang A. and Knapskog S. J. *A metric for trusted systems*, in *21st National Security Conference*, 1998, http://www.idt.ntnu.no/~ajos/papers.html

20. Manchala D. W. *Trust metrics, models and protocols for electronic commerce transactions*, in *18th International Conference on Distributed Computing Systems*, 1998. http://ieeexplore.ieee.org/iel4/5583/14954/00679731.pdf

21. Konrad K., Fuchs G. and Bathel J. *Trust and Electronic Commerce — More than a Technical Problem*, in *The 18th Symposium on Reliable Distributed Systems*, 1999, Lausanne, Switzerland, http://ftp.informatik.rwth-aachen.de/dblp/db/conf/srds/srds99.html

22. Povey D., *Developing Electronic Trust Policies Using a Risk Management Model*, 1999 http://security.dstc.edu.au/staff/povey/papers/CQRE/123.pdf

23. Povey D., *Trust Management*, 1999, http://security.dstc.edu.au/presentations/trust/

24. *Building a Foundation of Trust in the PC*, 2000, The Trusted Computing Platform Alliance, http://www.trustedpc.org

25. Frank N. M. and Peters L. *Building Trust: the importance of both task and social precursors*, in *International Conference on Engineering and Technology Management: Pioneering New Technologies - Management Issues and Challenges in the Third Millennium*, 1998, http://ieeexplore.ieee.org/iel4/5884/15675/00727781.pdf

26. Kini A. and Choobineh J. *Trust in Electronic Commerce: Definition and Theoretical Considerations*, in *31st Annual Hawaii International Conference on System Sciences*, 1998, Hawaii, http://ieeexplore.ieee.org/iel4/5217/14270/00655251.pdf

27. Jones S., *TRUST-EC: requirements for Trust and Confidence in E-Commerce*, 1999, European Commission, Joint Research Centre.

28. Gerck E and Group M.-C., *Overview of Certification Systems: X.509, CA, PGP and SKIP*, 1997, Meta-Certificate Group, http://www.mcg.org.br/cert.htm

29. Christianson B. and Harbison W. S. *Why Isn't Trust Transitive?* in *Security Protocols International Workshop*, 1996, University of Cambridge.

30. Mayer F. L. *A Brief Comparison of Two Different Environmental Guidelines for Determining 'Levels Of Trust' (Computer Security)*, in *Sixth Annual Computer Security Applications Conference*, 1990, http://ieeexplore.ieee.org/iel2/319/3856/00143781.pdf

31. Jøsang A. *Artificial Reasoning with Subjective Logic*, in *2nd Australian Workshop on Commonsense Reasoning*, 1997, http://www.idt.ntnu.no/~ajos/papers.html

32. Jøsang A. *Prospectives for Modelling Trust in Information Security*, in *Australasian Conference on Information Security and Privacy*, 1997: Springer-Verlag, http://www.idt.ntnu.no/~ajos/papers.html

33. Jøsang A. *A subjective metric of authentication*, in *5th European Symposium on Research in Computer Security (ESORICS'98)*, 1998: Springer-Verlag, http://www.idt.ntnu.no/~ajos/papers.html

34. Jøsang A. *The right type of trust for distributed systems*, in *ACM New Security Paradigms Workshop*, 1996, http://www.idt.ntnu.no/~ajos/papers.html

35. Abrams M. D., *Trusted System Concepts*, in *Computers and Security*, Joyce M. V., Editor, 1995, p. 45 - 56.

36. Abrams M. D. and Joyce M. V., *Trusted Computing Update,* Computers and Security, 1995, 14(1): p. 57 - 68.

37. Abrams M. D. and Joyce M. V., *New Thinking About Information Technology Security,* Computers and Security, 1995, 14(1): p. 69 - 81.

38. Dzubeck F., *Application Service Providers: An old idea made new*, 1999, http://www.nwfusion.com/archive/1999b/0823dzubeck.html

39. Morency J., *Application Service Providers and e-business*, 1999, http://www.nwfusion.com/newsletters/nsm/0705nm1.html?nf

40. Nix M., *Entering the Application Service Provider market*, http://www.developer.ibm.com/library/articles/nixasp.html

41. Chu Y.-H., Feigenbaum J., LaMacchia B., Resnick P. and Strauss M., *REFEREE: Trust Management for Web Applications*, 1997, AT&T Research Labs, http://www.research.att.com/~jf/pubs/www6-97.html

42. Chu Y.-H., *Trust Management for the World Wide Web*, 1997, Massachusetts institute of Technology, http://www.w3.org/1997/YanghuaChu/

43. Krauskopf T., Miller J., Resnick P. and Treesee W., *PICS Label Distribution Label Syntax and Communication Protocols Version 1.1*, http://www.w3.org/TR/REC-PICS-labels

44. Miller J., Resnick P. and Singer D., *PICS Rating Services and Rating Systems (and Their Machine Readable Descriptions) version 1.1*, http://www.w3.org/TR/REC-PICS-services

45. Evans C., Feather C. D. W., Hopmann A., Presler-Marshall M. and Resnick P., *PICSRules 1.1*, http://www.w3.org/TR/REC-PICSRules

46. *X.509 Certificates and Certificate Revocation Lists (CRLs)*, , Sun Microsystems Inc. http://java.sun.com/products/jdk/1.2/docs/guide/security/cert3.html

47. *An Introduction to Cryptography*, in *PGP 6.5.1 User's Guide*, Network Associates Inc. p. 11 - 36, http://www.fi.pgpi.org/doc/pgpintro/

48. Adams C. and Farrell S., *RFC2510 - Internet X.509 Public Key Infrastructure Certificate Management Protocols*, 1999, http://www.cis.ohio-state.edu/htbin/rfc/rfc2510.html

49. Galvin P., *Are you certifiable?*, 2000, http://www.sunworld.com/sunworldonline/swol-10-1997/f_swol-10-security.html

50. Gerck E., *Certification: Extrinsic, Intrinsic and Combined*, 1997, http://mcg.org.br/cie.htm

51. Rivest R. L. *Can We Eliminate Certificate Revocation Lists?* in *Financial Cryptography*, 1998, http://theory.lcs.mit.edu/~rivest/revocation.ps

52. Ding Y. and Petersen H., *A new approach for delegation using hierarchical delegation tokens*, 1995, University of Technology Chemnitz-Zwickau Department of Computer Science.

53. *Department of Defense: Trusted Computer System Evaluation Criteria*, 1983, http://ftp.std.com/obi/DOD/orange.book/

54. Blaze M., Feigenbaum J. and Lacy J. *Decentralized Trust Management*, in *IEEE Conference on Security and Privacy*, 1996, Oakland, California, USA, http://www.crypto.com/papers/policymaker.pdf

55. Blaze M., Feigenbaum J. and Strauss M. *Compliance Checking in the PolicyMaker Trust Management System*, in *Financial Cryptography: Second International Conference*, 1998, Anguilla, British West Indies.: Springer-Verlag. http://www.crypto.com/papers/pmcomply.pdf

56. Blaze M., Feigenbaum J., Ioannidis J. and Keromytis A. D., *The Role of Trust Management in Distributed Systems Security*, in *Secure Internet Programming: Security Issues for Mobile and Distributed Objects*, Vitek and Jensen, Editors, 1999, Springer-Verlag, http://www.crypto.com/papers/trustmgt.pdf

57. Blaze M., Ioannidis J. and Keromytis A. D. *Trust Management and Network Layer Security Protocols*, in *Cambridge Protocols Workshop*, 1999, Cambridge, http://www.crypto.com/papers/networksec.pdf

58. Feigenbaum J. *Overview of the AT&T Labs Trust Management Project: Position Paper*, in *Proceedings of the 1998 Cambridge University Workshop on Trust and Delegation*, 1998: Lecture Notes in Computer Science.

59. Blaze M., Feigenbaum J. and Keromytis A. D. *KeyNote: Trust Management for Public-Key Infrastructures*, in *Security Protocols International Workshop*, 1998, Cambridge, England, http://www.cis.upenn.edu/~angelos/Papers/keynote-position.ps.gz

60. Blaze M., *Using the KeyNote Trust Management System*, 1999, AT&T Research Labs, http://www.crypto.com/trustmgt/kn.html

61. Blaze M., Feigenbaum J., Ioannidis J. and Keromytis A. D, *RFC2704 - The KeyNote Trust Management System (version 2)*, 1999, http://www.crypto.com/papers/rfc2704.txt

62. Herzberg A., Mass Y., Mihaeli J., Naor D., and Ravid., *Access Control Meets Public Key Infrastructure, or: Assigning Roles to Strangers*, in *IEEE Symposium on Security and Privacy*, 2000, http://www.hrl.il.ibm.com/TrustEstablishment/paper.asp

63. IBM, *IBM Trust Establishment Policy Language*, http://www.hrl.il.ibm.com/TrustEstablishment/PolicyLanguage.asp

64. Abdul-Rahman A. and Hailes S. *Supporting Trust in Virtual Communities*, in *Hawaii International Conference on System Sciences 33*, 2000, Maui, Hawaii, http://www.cs.cs.ucl.ac.uk/staff/F.AbdulRahman/docs/

65. Burrows M., Abadi M. and Needham R. M., *A Logic of Authentication,* ACM Transactions on Computer Systems, 1990, 8(1): p. 18-36, http://citeseer.nj.nec.com/details/burrows90logic.html

66. Jajodia S. S., P. and Subrahmanian V. *A Logical Language for Expressing Authorizations*, in *Security and Information Privacy*, 1997, http://ieeexplore.ieee.org/iel3/4693/13107/00601312.pdf

67. Marsh S. P., *Formalising Trust as a Computational Concept*, in *Computing Science and Mathematics*, 1994, University of Stirling, p. 170.

68. Firozabadi B. S. and Sergot M. *Power and Permission in Security Systems*, in *7th International Workshop In Security Protocols*, 1999, Cambridge, UK: LNCS, Springer-Verlag.

69. Genesereth M. R. and J. N. N., *Logical Foundations of Artificial Intelligence*, 1987, California, U.S.A.: Morgan Kaufmann Publishers Inc, 405.

70. Jones A., J. I. and Firozabadi B. S. *On the characterisation of a Trusting agent - Aspects of a Formal Approach*. in *Workshop on Deception, Trust and Fraud in Agent Societies*, 2000.

71. Rangan P. V. *An Axiomatic Basis of Trust in Distributed Systems*, in *Symposium on Security and Privacy*, 1988, Washington, DC: IEEE Computer Society Press.

72. Elgesem D., *The Modal Logic of Agency,* Journal of Philosophical Logic, 1997, 2(2): p. 1-46.

73. Kakas A. and Miller R., *A Simple Declarative Language for Describing Narratives with Actions,* Journal of Logic Programming, 1997(Special Issue on Reasoning About Actions).

74. Safran C., Sands D. Z. and M. R. D., *On-Line Medical Records: A Decade of Experience,* Methods of Information in Medicine, 1999, 38: p. 308-312.

75. Anderson R. J., *The DeCODE Proposal for an Icelandic Health Database*, 1998, http://www.cl.cam.ac.uk/users/rja14/iceland/iceland.html

76. Anderson R. J., *NHS wide networking and patient confidentiality: Britain seems headed for a poor solution,* in British Medical Journal, 1995, 311: p. 5 - 6, http://www.bmj.com/cgi/content/full/311/6996/5

77. Anderson R. J., *Patient Confidentiality - At Risk from NHS Wide Networking,* Proceedings of Healthcare 96, 1996, http://www.cl.cam.ac.uk/ftp/users/rja14/hcs96.ps.Z

78. Anderson R. J., *An Update on the BMA Security Policy,* Proceedings of the 1996 Cambridge Workshop on Personal Information – Security, Engineering and Ethics, 1996: p. 217 - 234, http://www.cl.cam.ac.uk/ftp/users/rja14/bmaupdate.ps.Z

79. Anderson R. J., *Remarks on the Caldicott Report*, 1998, http://www.cl.cam.ac.uk/~rja14/caldicott/caldicott.html

80. Anderson R. J., *Problems with the NHS Cryptography Strategy*, 1997, http://www.cl.cam.ac.uk/users/rja14/zergo/zergo.html

81. Anderson R. J., *Information Technology in Medical Practice: Safety and Privacy Lessons from the United Kingdom*, 1998, http://www.cl.cam.ac.uk/users/rja14/austmedjour/austmedjour.html

82. Anderson R. J., *Clinical System Security - Interim Guidelines,* British Medical Journal, 1996, 312: p. 109 - 111. http://www.ftp.cl.cam.ac.uk/ftp/users/rja14/guidelines.txt

83. Anderson R. J. *A Security Policy Model for Clinical Information Systems*, in *IEEE Symposium on Security and Privacy*. 1996, Oakland, California, USA: IEEE Computing Society Press, http://www.cl.cam.ac.uk/ftp/users/rja14/oakpolicy.ps.Z

84. Anderson R. J., Matyas V., Jr, Petitcolas F. A. P., Buchan I. E. and Hanka R., *Secure Books: Protecting the Distribution of Knowledge.* Proceedings of the Security Protocols Workshop 97, 1997, http://www.medinfo.cam.ac.uk/miu/papers/paris97/wax_sec.htm

85. Anderson R. J., Matyas V., Jr. and Petitcolas F. A. P. *The Eternal Resource Locator: An Alternative Means of Establishing Trust on the World Wide Web*, in *3rd USENIX workshop on Electronic Commerce*, 1998, Boston, Massachusetts, USA, http://www.cl.cam.ac.uk/~fapp2/papers/ec98-erl/

86. Kesterson H. L., II. *Digital Signatures - Whom Do You Trust ?* in *Aerospace Conference*, 1997, http://ieeexplore.ieee.org/iel3/4328/12521/00577506.pdf

87. Myers A. C. and Liskov B., *Protecting Privacy using the Decentralized Label Model,* To appear in ACM Transactions on Software Engineering Methodology, http://www.cs.cornell.edu/andru/papers/iflow-tosem.pdf

88. Gong L. and Qian X. *The Complexity and Composability of Secure Interoperation*, in *IEEE Symposium on Security and Privacy*, 1994, Oakland, California, USA, http://ftp.csl.sri.com/reports/postscript/sri-csl-93-13.ps.gz

89. Abdul-Rahman A. and Hailes S., *Security Issues in Mobile Systems*, 1995, http://www.cs.ucl.ac.uk/staff/AbdulRahman/docs/

Tyrone Grandison (tgrand@doc.ic.ac.uk) obtained his BSc (Hons) in Computer Studies and MSc in Software Engineering from the University of the West Indies, Jamaica. He is currently a doctoral student in the Department of Computing at Imperial College. His primary research interest is in trust in Distributed Systems.

Morris Sloman (m.sloman@doc.ic.ac.uk) obtained his BSc (Eng) in Electronic Engineering from University of Cape Town, South Africa and a PhD in Computing from University of Essex, U.K. Professor Sloman leads the Distributed Systems Group, and has been in the Department of Computing, Imperial College since 1976. He has given courses on distributed systems, computer networks, operating systems and computer architecture. In addition, he has given talks at conferences and invited seminars to various organisations in many countries. Professor Sloman has been working on Policy based management for about 10 years. He has managed many research projects funded by the UK Engineering and Physical Science Research Council (EPSRC), European Union and various industries on management, security and design of distributed systems, multimedia systems and mobility. He is editor of a reference book on Management of Network and Distributed Systems published by Addison Wesley, and a member of the editorial board of the Journal of Network and Systems Management. Professor Sloman was chairman of the UK EPSRC Multimedia and Network Applications Funding Programme and currently is a member of the Research Assessment Panel for Computing Departments in the UK. He was program co-chair of the 1999 IEEE/IFIP Integrated Management Conference (IM99), is on the advisory board for the IM/NOMS conferences and on the steering committee for the EDOC conferences. He is conference chair for Policy 2001: Workshop on Policies for Distributed Systems and Networks. See http://www-dse.doc.ic.ac.uk/~mss for more details and selected papers.