TO APPEAR: *Web and Information Security*, E. Ferrari and B. Thuraisingham (eds), Idea Group Inc., Hershey, PA (*forthcoming*).

Creating a Policy-Aware Web: Discretionary, Rule-based Access for the World Wide Web

Daniel J. Weitzner[1], Jim Hendler[2], Tim Berners-Lee[1], Dan Connolly[1]
[1] CSAIL, Massachusetts Institute of Technology, Cambridge, MA {djw,connolly,timbl}@w3.org
[2] Computer Science Dept, University of Maryland, College Park, MD hendler@cs.umd.edu

ABSTRACT

In this paper, we describe the motivations for, and development of, a rule-based policy management system that can be deployed in the open and distributed milieu of the World Wide Web. We discuss the necessary features of such a system in creating a "Policy Aware" infrastructure for the Web, and argue for the necessity of such infrastructure. We then show how the integration of a Semantic Web rules language (N3) with a theorem prover designed for the Web (Cwm) makes it is possible to use the Hypertext Transport Protocol (HTTP) to provide a scalable mechanism for the exchange of rules and, eventually proofs, for access control on the Web. We also discuss which aspects of the *Policy Aware Web* are enabled by the current mechanism and describe future research needed to make the widespread deployment of rules and proofs on the Web a reality.

INTRODUCTION

Inflexible and simplistic security and access control for the decentralized environment of the World Wide Web have hampered the full development of the Web as a social information space because, in general, the lack of sufficiently sophisticated information controls leads to unwillingness to share information. This problem is greatly exacerbated when information must be shared between parties that don't have pre-existing information sharing policies, and where the "granularity" of the information to be shared is coarse - that is, where access is granted to an entire website or data resource because policy control mechanisms for access at a finer-grained level aren't available. Even large intranets and controlled-access Webs, face these problems as the amount of information and the number of information seekers grow. Thus, despite ever-greater amounts of useful information residing on the Web in a machine-retrieval form, reluctance to share that information remains and is likely to increase.

In this chapter, we will argue that a new generation of *Policy-Aware* Web technology can hold the key for providing open, distributed and scaleable information access on the World Wide Web. Our approach provides for the publication of declarative access policies in a way that allows significant transparency for sharing among partners without requiring pre-agreement. In addition, greater control over information release can be placed in the hands of the information owner, allowing discretionary (rather than mandatory) access control to flourish.

The technical foundation of our work focuses on developing and deploying the upper layers of the "Semantic Web layer-cake" (see Figure 1 - based on Berners-Lee, 2001,

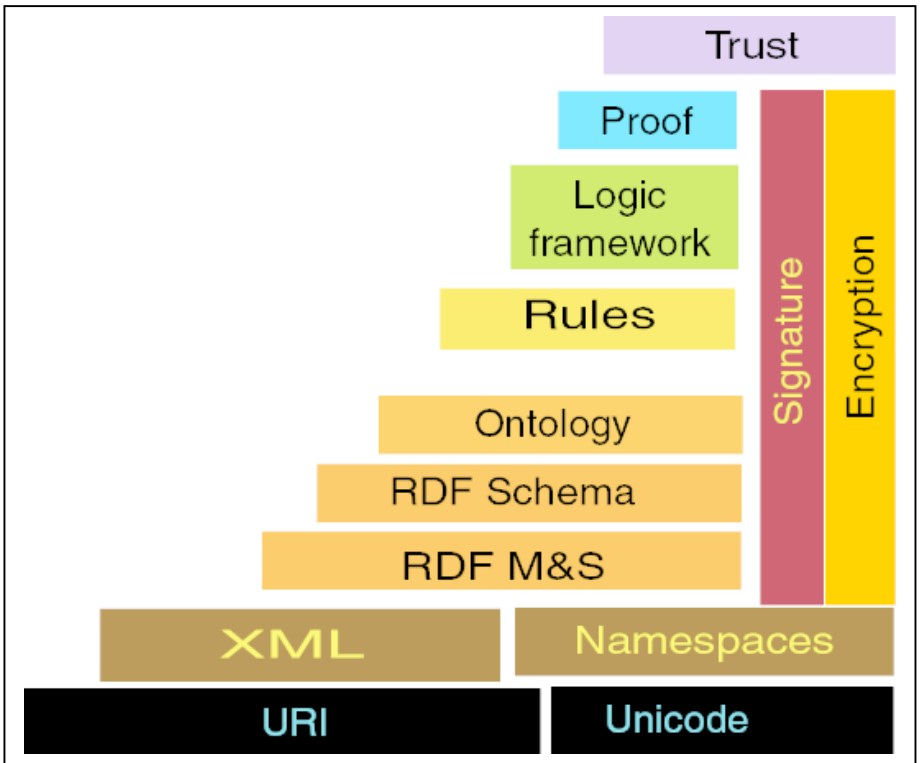


Figure 1: The Semantic Web Layer Cake *ca. 2002*

Swartz and Hendler, 2001) in order to enable Policy Aware infrastructure. The ambition of the Semantic Web is to enable people to have richer interactions with information online through structured, machine-assisted integration of data from all around the Web (Berners-Lee et al, 2001). We will show that it is possible to deploy rules in a distributed and open system, and to produce and exchange proofs based on these rules in a scaleable way. These techniques, properly applied by taking crucial Web architecture issues into account, will extend Semantic Web technology to allow information resources on the World Wide Web to carry access policies that allow a wide dissemination of information without sacrificing individual privacy concerns.

The ultimate success of the Semantic Web, however, will depend as much on the *social* conditions of its use, as on the underlying technology itself. Much of the power of the Semantic Web lies in its ability to help people share information more richly and to discover subtle information linkages across the Web that are not visible in today's relatively flat online information environment. However, people will not share information freely in an environment that is threatening or antithetical to basic social needs such as privacy, security, the free flow of information, and ability to exercise their intellectual property rights as they chose. Though today's Web falls short in many of these areas, the descriptive and logical functions of the Semantic Web can offer the ability to help people manage their social relationship on line, in addition to just

managing the traditional information content found on the Web today. We describe here the framework for, and first steps toward, a *policy aware* Web.

As an integral part of the Semantic Web, policy aware infrastructure can give users greater transparency in their online interactions, help both people and machines to 'play by the rules' relevant to social interactions in which they participate, and provide some accountability where rules are broken. The policy aware Web is the logical continuation of the "user empowering" features of the Web that have, in the Web's first decade, been critical in shaping the delicate relationship between Web technology and the surrounding legal environment (Berman and Weitzner, 1995).

In this paper, our primary focus will be on the use of Semantic Web technologies to provide a rule-based access mechanism in a style that is consistent with current and expected future Web Architecture. First, however, we describe what we mean by policy-awareness, and the needs of bringing it to the online world.

BEING POLICY AWARE

By any measure, today's World Wide Web has been extraordinarily successful at meeting certain social goals and rather disappointing at others. The Web has enhanced dissemination of, and access to, information in both commercial and non-commercial contexts. We've seen great ease of publishing relative to mass media and constantly improving search and discovery. The Web has even provided relatively robust responses to the great diversity of opinion about what constitutes good, bad, moral, immoral, legal and illegal content (cf. Reno v. ACLU, 1987). Yet for all of the Web's success at meeting communication and information exchange goals, it has failed in equal measure at satisfying other critical policy requirements such as privacy protection, a balanced approach to intellectual property rights, and basic security and access control needs. We worry about these problems not only because they implicate fundamental human rights, but also because the failure to solve them renders this medium that we all care about that much poorer, and causes people to feel alienated in their online interactions, even as the appreciate the unprecedented benefits of the Web.

As these problems fall into the category of law & public policy, the general impulse is to look to the law to solve them. Law is certainly a necessary part of making the Web a humane environment, but it is not alone sufficient. For as much as there are real deficiencies in the laws that govern online interactions, the absence of technical capacity to share basic context information between users and services providers, and amongst users, is a fundamental impediment to the web being an environment in which people will feel comfortable and confident to conduct a full range of human activities. Indeed, the focus on law as a solution to the policy-related problems on the web risks obscuring the deep technical and functional gaps that prevent us from having normal social interactions online.

To illustrate these gaps, consider the differences in policy-awareness regarding the flow of sensitive personal information between browsing in your local library and browsing an online digital library repository. In either case your browsing habits may be tracked, perhaps even in a way that associates your name with the information collected. The similarity ends there because offline, if an over-eager librarian follows you from aisle to aisle looking at which books you pick up and whether you open the pages or not, you would both know that this was happening and have a variety of understated but clear techniques for stopping the behavior, or at least making your displeasure known. Our sense of vision (to notice the snooping) and mastery of simple gestures (the quizzical or displeasing look over the shoulder) help us to be aware of and resolve this awkward situation. Only in the oddest of circumstances would recourse to law be required or even useful. A simple exchange of social clues would more than likely solve the problem.

When this scenario is replayed in an online library, however, the user doing the browsing is at a distinct disadvantage. First, it is quite unlikely that the online browser will even be aware of the tracking behavior (or lack of it) unless she has found a privacy policy associated with the site and managed to read and understand it. Even with that, the policy is likely to describe what the site might do, not what actually happens in the case of a given browser on a given visit. Second, even if the online library browser ascertained that unwanted tracking was occurring, what could she do? We have no online equivalent of shooting the snooper a 'dirty look' or sneaking down another aisle.

This gap between what is possible in the online and offline environment has a critical impact on the degree which people feel comfortable interacting online. As the library example illustrates, in most human interaction, we rely on various feedback loops to establish what is acceptable versus unacceptable behavior. Online environments that lack the channels for such feedback thus need to replace these mechanisms with other, more Web appropriate ways of maintaining our mastery over our personal information space. In order to make the Web a more socially-rich environment we can take advantage of the rich representational framework offered by the Semantic Web to help people manage not just the traditional Web content, but also the social context and cues that around any information-related activity.

Consider the simple desire to share photographs amongst friends. Offline, if you want to share a picture with a friend or colleague, you have an easy way to give them the picture and its very likely that the context of that interaction and your relationship will give the recipient of the photo a pretty good clue about the social rules to be associated with the use and sharing of that picture. Of course today we can email pictures around and many of the same social conventions are likely to apply. But try to use the Web to share pictures with the informally-defined communities in which we all participate, and problems soon emerge. While the Web allows us to access and transport pictures all around the world to hundreds of millions of potential recipients, the inability to specific even very simple rules for sharing information forces us into an uncomfortably

inflexible set of choices: share with everyone, share with no one, or engage in the arduous task of managing access via IP addresses or assigning names and passwords.

The lack of *policy awareness* in today's Web infrastructure makes it difficult for people to function as they normally would in informal or ad hoc communities. Thus, policy awareness is a property of the Semantic Web that will provide users with readily accessible and understandable views of the policies associated with resources, make compliance with stated rules easy, or at least generally easier than not complying, and provide accountability when rules are intentionally or accidentally broken. So, in building Policy Aware services, we seek to meet the following requirements:

- **Transparency:** both people and machines needs to be able to discover, interpret and form common understandings of the social rules under which any given resource seeks to operate. Can it be shared, copied, commented upon, made public, sold, etc? Encoding social rules in the formal mechanisms described below will provide a level of transparency currently unavailable on today's Web (Weitzner, 2004). What remains is to develop the social practice of using these mechanisms in consistent ways to communicate about social context and expectations. Related work has been done in the context of existing Web standards such as the Platform for Privacy Preferences (P3P), and XML markup languages such as SAML, EPAL and XACML. However, research is still required to enable the development of local-community-specific policy description frameworks and tools to help users evaluate policy rules, especially when various rule-sets interact.

- **Compliance mechanisms :** We'd like it to be just as easy to comply with rules expressed in a policy aware environment as it is to use the Web today. Thus, most users must be largely unaware of the underlying formalisms in which the policies are expressed and maintained, and mechanisms built into the structure of the Web (protocols, browsers, etc) should support the policies thus expressed. The mechanism we describe in this paper uses rules and transportable proofs as the communications channel through which the user establishes compliance with a given rule set with the discovery and use of the rules built into the Web infrastructure. Expression of social rules in a formal, machine-readable manner will enable end-user software (including browsers and other user agents) to make it easier for users to comply with the rules of the environment in which they participate.

- **Accountability**: Rules, no matter how well described or carefully enforced, may be broken. Whether the breach is inadvertent or intentional, a policy aware environment will help participants to spot and track infractions. In some cases, there may have been a misunderstanding or inadvertent error. Or, in large user communities such as the Web, it is certainly possible that the breach was malicious. The individuals and communities involved will respond in different ways depending on the social and legal context of the breach. Policy awareness seeks to

identify the fact the rule violation with adequate accountability and context sensitivity so that those involved can take whatever action is appropriate.

Based on these principles, a key difference between policy aware access control, of the sort that we describe in this chapter, and traditional access control approaches developed in the computer security and cryptography community is that we stress description over enforcement. In current systems, often the description of the policies is intertwined with the enforcement thereof. Cryptographic enforcement mechanisms generally require a high degree of pre-coordination on policy terms, and demand that users and system administrators bear the costs of maintaining a local public key management infrastructure. While these costs may be acceptable to certain environments which must protect high value assets (commercial financial transactions or intelligence information, for example), they are entirely beyond the means of small and ad hoc communities. In these cases, most users will continue to live with virtually no access control mechanisms at all. Our aim is thus to give people the ability to have highly descriptive security policies with a relatively low enforcement burden placed on the individual web client. Hence, we concentrate our energies on describing access control policies and providing the tools to enable policy aware systems to assess compliance with rules based on good faith assertions from all involved. The policy aware approach can work well with more robust cryptographically-enforced security as well, as we will describe later in this paper, but our current emphasis is at the high description end of the spectrum, rather than at the high enforcement end.

One notable piece of past work in the area of highly descriptive access on the Web is that of the REI system (Kagal et al, 2004). REI extends a rule-based policy mechanism developed for distributed processing applications. REI is based on an "agent-based" computing approach, in which agents (realized primarily as web services) are able to control access and information sharing via policies encoded in OWL ontologies. Our work is closely related to ideas in REI, but is focused on going beyond their multi-agent, service-based paradigm and building rule-based access into the web protocols themselves, with an emphasis on application to the decentralized environment of the Web.

RULE-BASED ACCESS AND THE WORLD WIDE WEB

Research in the security area has recently been exploring mechanisms that allow the requirements above to be realized by the use of "rule-based" access policies, shifting away from the identity- and role- based mechanisms that are the primary mechanisms used on the Web today (where any access control is used at all). Our work focuses on extending rule-based access to be used in the open and distributed World Wide Web, which is necessary for achieving the policy-awareness goals described above. In this section, we provide some background on past work and define the goal of our research, as well as identifying some of the key pieces of work that we build on.

Most Web access today is performed using identity–based approaches (Shamir, 1985) where access to all or some of the data is granted based on pre-existing agreements negotiated between the data owner and those accessing the data resource. A simple example of this is password-based access to a protected web site – a user who identifies him or herself by providing the correct login/password combination is allowed in, others aren't. Identity schemes are also used in many database systems for both online and offline access, with more recent work focused on using public key certificates, rather than passwords, to add more security (*cf.* Boneh and Franklin, 2001). Role-based access (cf. Ferraiolo et al, 2003) is similar to identification-based access, except that instead of identifying a particular user, an access policy is created to allow users of a particular class (i.e. those who play some "role") to access various parts of the data. Thus, for example, the World Wide Web Consortium (W3C) web site has an access policy that (simplifying somewhat) allows users to be assigned to three classes by their roles – *team,* which has access to all files, *member,* which has access to all files accept those marked team, and *public,* which has access to all files except those marked team or.*

There are several problems with identity- and role- based schemes. First, in most cases the classes must be defined in advance. Creating a temporary class is difficult, if not impossible, in most implementations of these policies. For example, in preparing this chapter one author, Hendler, needed access to a W3C document that was labeled *team*, but he only had *member* access rights. Giving Hendler *team* access would have meant letting him see other documents he did not have the right to view, moving the document to *member* would have risked letting it be seen by others who served the same role as Hendler, but were not entitled to see this particular document. In the end, moving the document to a different site where we could set up a temporary (password-based) scheme was more trouble than it was worth, and instead we had to resort to emailing the document to each other (a workaround which bypassed the entire security system).

A second problem with these schemes is that they tend to be difficult to set up in a fine-grained way as web-based schemes generally work at the file-directory level. It is difficult, for example, to give someone access to a part of your page or to particular data in a specific context.† Our goal is to be able to write rules that describe policies at the level of individual URIs, thus grounding the system in the smallest externally nameable Web resources. Our decision to base our approach on RDF, rather than XML, is largely

---

* Actually, a newer version of the W3C access system is built on the rule-based approach and uses some of the technology we describe later in this paper – see http://www.w3.org/2001/04/20-ACLs for details.

† Most XML-based security work, for example, focuses on tagging individual parts of a document with a list of applicable roles and the use of XML signatures for checking whether an incoming request matches those roles. Although this approaches the issue of finer-grained access, it does not address the other problems we discuss, and primarily focuses works on documents, not other web resources.

based on the fact that RDF assigns individual URIs to instances and classes, seemingly making it ideal for this purpose. (It is worth noting, however, that current Web protocols still return an entire document, rather than the individual named entity, when URIs containing "fragIDs" are used. It is our hope, however, that RDF query languages currently under development will allow delivery of finer-grained query responses from RDF stores, thus helping to alleviate this problem. For non-text resources such as individual photos within a photo collection, current web protocols allow appropriate experimentation with finer-grained access.)

A third limitation of these schemes is that it is usually extremely difficult to have access change over time in a precise. For example, a better solution to the access problem described previously would have been to temporarily create a "*team+hendler*" role and to have the document in question be limited to *team+hendler* until some specific date, at which time the new role could go away and the document could revert to its previous state. Defining time-sensitive rules is difficuly in role-based schemes.

The ability to specify access policies that don't have to be defined in advance, have fine grained access, and allow fairly dynamic change is a current focus of research in the database (Kyte, 2000), Programming Language (Pandey and Hashli, 1999; Bauer et al, 2004), Operating System (Ott, 2001), Artificial Intelligence (Barbour, 2002) and multiparty security (cf. the Portia, SDSI, and SPKI projects) areas. This work largely focuses on a switch from role-based authentication to what is known as *rule-based* access policies (*cf*. Didriksen, 1997), an approach which has been gaining popularity since the late 1990s. In rule-based access, a declarative set of rules is used to define finer-grained access to resources with requests for data providing a "demonstration" that they satisfy the policy encoded in the rules. The demonstration of meeting these rules can be fairly simple – for example, most commercial implementations of rule-based access have only simple antecedents that can match information in (public key) certificates to features in the data.

To date rule-based access has been primarily associated with "Mandatory Access Control" (MAC) systems, especially those used to provide multi-level access to documents. MAC systems are those where the owner of the information does not get to control protection decisions, but rather the system is designed to enforce *a priori* protection decisions (i.e., the system enforces the security policy possibly over the wishes or intentions of the object owner). In these systems, now in common use in both industrial and government applications, every "information object" is tagged with a sensitivity level, and every "subject" (generally a process which can cause information to flow – i.e. something which can remove data objects from the system) is also given a tag. A lattice of subject/object pairs is used and a simple set of rules implemented that will only allow a subject access to an object if its tag has a position in the lattice that is equal to or higher than that of the object.

Rule-based systems have been less successful, however, in "Discretionary Access Control" (DAC) systems, where the information owner can locally determine the access

policy. The reason for this is that rule-based access generally requires the subject to "prove" that they have access, and for the objects in the system to have a finer-grained control of the information than is typical in MAC schemes.

For example, the provider of some data may wish to have an access policy that would match a rule such as

```
User can-access DATA.Associated-record.{decision,signees}
  IF   User.attribute-certificate:originator="W3C"
    And DATA.type-designator="Member"
      And Employer=User.employer
        And Employer.member:status ="current"
```

i.e. that users who are authorized by the W3C (i.e. providing an attribute certificate* signed by an authorized W3C entity) can see particular documents (in this case, the decision and who signed it) for those data elements which are allowed to be seen by those in the "member" group and where the employer associated with the user is identified as being a current W3C member organization.

It is this ability to create a discretionary, rule-based access control on the Web that we are trying to achieve – that is, we believe such rule-based mechanisms will be a necessary component of the *policy-aware Web,* as being able to control access will be an integral part of the privacy and sharing controls described previously. Our goal, therefore, is to show how rule-based access methods can be brought to the Web using the same principals of openness, distribution and scalability that have allowed the Web to grow into the pervasive application that it is today.

Technical Challenges

There are many challenges inherent in bringing rule-based, discretionary access control to the Web. Contrast the Web access problem to the typical database (or OS) access issues, and it becomes clear why this is so:

i. Current rule-based schemes use specialized access control languages generally designed to work in a specific application. Such proprietary approaches rarely work on the Web, due to the need for openness and shared use – if my application cannot read your rules, or yours cannot read mine, then we don't get interoperability. Indeed, it is not enough to have a standard for writing the rules –it is critical that the mechanisms by which the rules governing access to an entity can be expressed in a flexible manner, discovered easily, and applied in a reliable manner, preferably within the scope of the Hypertext Transfer Protocol (HTTP) itself.

---

* Attribute certificates are generally used in PKI schemes for aspects of a transaction that have a shorter lifetime than the public-private key pair (*cf.* http://ospkibook.sourceforge.net/docs/OSPKI-2.4.6/OSPKI/pkix-concepts.htm).

ii. In a closed, controlled system, a pre-defined set of subject tags checked against a predefined set of object tags is sufficient – in fact, this is why rule-based MAC has become viable for many organizations. On the Web, however, there is no simple set of tags that will be sufficient for all applications in all domains. Instead, a mechanism must be provided that can evaluate the rules in the policy against information provided by the subject – this information can be in the form of a signed access certificate or other such identity provider, a web based proof, or some combination thereof.

iii. When access certificates and other such identifiers are not sufficient, there must be a general mechanism for providing a *proof* that one system is allowed to access the information in some resource using published rules (Bauer, 2003). On the Web, the subject must provide some sort of grounded and authenticatible proof that an object's access policy can be met, and the proof must be exchanged using Web protocols. In addition, there must be a mechanism by which the system receiving the proof can check its correctness with respect to only those rules of logic that it accepts. On the Web, we cannot assume that every user will employ the same piece of proof-checking software, so a set of standards is required be sure that all participants evaluate proofs on the semantic web in a consistent manner. Some tools may developed that only use a few simple rules (perhaps limiting expressivity for efficiency), some applications may accept non-standard rules of inference specialized to some particular application class or type, and some users may prefer rules that seem "illogical" to other users (such as "I will assume that anything my mother says is true, is true"). The policy aware web, to be able to accommodate the wide range of users and applications it will need to support, must be able to tolerate many kinds of different "proofs" being used for many different purposes.

iv. Inconsistency must be handled in some way that does not cause the downfall of the Web. In many rule- and proof- based systems, anything can be derived from an inconsistency, and thus these systems are generally defined in a way that no inconsistency can be tolerated. On an open system like the Web, inconsistency is inevitable and the policy-aware web must have means to deal with it. This is particularly mandated for privacy and security applications where it can be assumed that some users will try to "raid" information sources. If it was possible to defeat the policy aware web by simply asserting "X" at one point and "not X" at another, the system would certainly not survive long in any useful state. Past work has defined "paraconsistent" logics that handle inconsistency in logic programming languages and deductive databases and a similar approach to handling inconsistency will be needed for the Web.

All of the capabilities above have been explored as separate research topics in a number of fields. However, to date no end-to-end approach that can combine all of the above has been developed. In the remainder of this chapter, we describe the steps we have been taking to provide these capabilities and present an example of how the mechanisms we describe can be realized using tools that we have been developed.

Future work focuses primarily on the issue of dealing with inconsistency, with the scaling of these tools to work on the Web, and with the development of a prototype environment (controlling access to personal photographs) that we are building to explore these issues.

Rules Engines as a Foundation of the Policy Aware Web

Recalling the layer-cake diagram from the beginning of this paper, the Semantic Web, and the functionality we want, requires that a "stack" of standard languages be designed for facilitating the interoperability of tools – just as the Web itself required the definition of a markup standard (HTML) so too does the Semantic Web stack necessitate shared languages. More importantly, by building on already existing Web standards, policy-awareness will not require changing the basic architecture of the Web – after all, our goal is for the policy awareness to eventually be built right into the user's web client (and displayed through their web browser).

For a rule language to meet our needs it will therefore have to be realizable in a form where the rules can be published, searched, browsed and shared using well-known HyperText Transfer Protocol (HTTP 1.1). The rule language must therefore be defined in a way that it can take advantage of the web protocols enabled by being realized in XML documents and exploiting the document tagging properties thereof, using the linking capabilities provided by the Resource Description Framework (RDF), the class definitions enabled by RDF Schema, and the more powerful ontological agreements enabled by the Web Ontology Language OWL.†

Currently, designing a rule language that is syntactically realized in XML and, preferably, compatible with RDF is an active research area. Current approaches include a proposal for RuleML, an XML-based rule standard (Boley et al, 2001) a recent proposal called SWRL, which builds the rules on top of OWL (Horrocks et al, 2003), and a very powerful logic language, called the SCL (Standard Common Logic, Menzel and Hayes, 2003) that is intended as a web-based successor to the earlier KIF language. Given our concern for transparency, it is clear that a "human readable" form of the language is important –RDF/XML format is often overly verbose and difficult for humans to interact with. Therefore, we base our work on "Notation 3" (or N3 as it is more commonly known) which was designed by Berners-Lee (2000) and is now actively supported by a growing open-source development community. N3 is a RDF-based rule language that was designed based to be consistent with a number of Web Architecture principles. N3 is also designed to work closely with Cwm, an RDF-based reasoner specifically defined for web use, which we discuss in the next section.

---

† The documents describing all of these languages can be found at http://www.w3.org -- the W3C web site.

Providing the details of the N3 rule language is beyond the scope of this paper, but a simple example should suffice to show a couple of the special features of the language. The simple N3 rule

```
{?x  cs:teaches ?y.  ?y.cs:courseNumber math:greaterThan "500"} =>{ ?x a cs:professor} .
```

states that if *X* teaches *Y*, where *Y*'s course number is greater than 500, then *X* must be of type professor. Note that qnames are used (and would be defined elsewhere in the document) to denote the unique URIs of each of the entities in the formula, and that the special qname "log" is used to denote logical properties while, in this case, the qname "math" is being used to invoked mathematical functions. This rule can be rendered into RDF/XML in an automated way, and in that form, although ungainly, the N3 becomes a valid XML document (the importance of which we return to below).

The N3 rules language has co-evolved with the design of a reasoner which can process the rules and evaluate them appropriately in a Web context – that is, a Web-based prover must be able to handle those procedural attachments crucial for working on the Web and must have a means for reasoning about a set of assertions that can be accessed on the Web using standard Web protocols. Cwm* (Berners-Lee et al, 2000 is a reasoner that has been specifically developed to work in the Web environment. Cwm is a forward-chaining reasoner that can be used for querying, checking, transforming and filtering information on the Web. Its core language is RDF, extended to include N3 rules. One of the key features of Cwm is its ability to include a number of specialized modules, known as built-ins, which allow a number of different functions to be evaluated during rule processing – with the specific procedures being those needed for processing information on the Semantic Web ranging from simple functions like

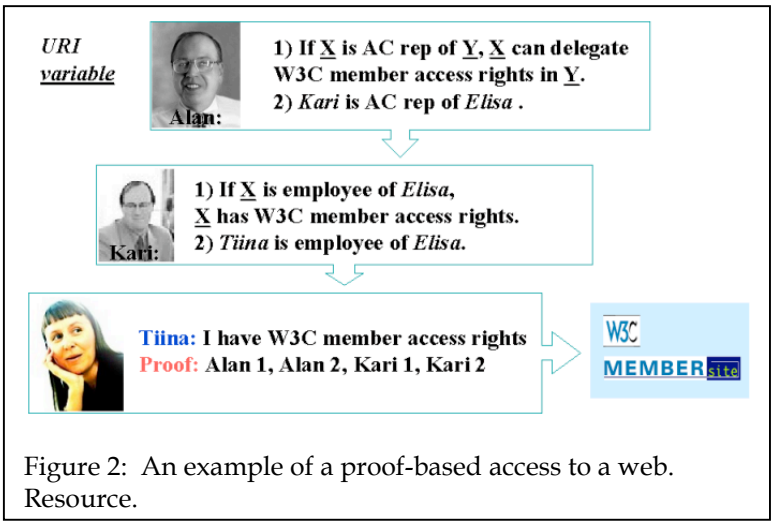


Figure 2: An example of a proof-based access to a web. Resource.

* CWM originally was an acronym for "Closed World Machine" but it rapidly developed into a more powerful, open-world tool that is being used for the sort of distributed, open rule sets we describe in this proposal. Rather than renaming CWM, the development team dropped the acronym and kept going.

math:greaterThan which invokes a mathermatical function to *log:semantics* which allows information to be fetched from the Web and parsed or *crypto:verify* which verifies a digital signature. Indeed, in Cwm the integration of the Web and inferencing goes even further: the inference engine can look up symbols on the Web to discover information which may directly or indirectly help to solve the problem in question. Predicates can be looked up to find OWL ontologies or queried so as to find the specific properties of individuals and classes defined elsewhere on the Web.†

Cwm's Web-specific built-ins, which are integrated into its inferencing algorithms, make it a useful tool and will serve as the primary tool used for checking rules, handling certificates, generating and checking proofs, and controlling access. Cwm has been used for prototyping the capabilities discussed in this paper. (Current work is exploring how to scale Cwm. Approaches include the development of a new RETE-based algorithm for Cwm and an analysis as to whether it is possible to use deductive database techniques to improve Cwm's performance. In particular, we are exploring whether a recent approach to "magic sets" (Behrend, 2003), can be used to provide database like scalability to Cwm under certain circumstances, despite the fact that it is more expressive than datalog.)

Implementing Rule-Based access with a Semantic-Web Proof Engine

Cwm allows us to implement rule-based access control on the Web in several ways. First, Cwm is able to check whether an access request can be granted in the "base case" where either a signed certificate or a grounded assertion is presented (a grounded assertion is one where a URI is used to point to an assertion that can be checked on the Web using HTTP-Get). In these cases, Cwm checks that the antecedent of a policy rule indeed matches the subject's access (similar to the approach used in rule-based MAC). In more complex cases, Cwm can check a set of such assertions to make sure they are all valid, and then check that they form a "proof" that the rule or rules for access have been met.

Consider the example shown in Figure 2 (which is a more complicated version of the Web file-access rule shown previously). In this case, access to some files on the W3C web site will be granted to a user if that user can prove they work for a member of the W3C. Further, the W3C can delegate the certification of users to an individual at a member organization. In this example, when Tiina requests access, she can prove that she meets these rules by showing that Alan had the right to delegate the authority, that Alan delegated the authority to Kari, and that Kari certified that Tiina is an employee of

---

† At the time of this writing Cwm has about 100 built-ins. The list can be found at http://www.w3.org/2000/10/swap/doc/CwmBuiltins.html.

his organization. Given the rules shown, and the grounded assertions (i.e. the rules with the variables replaces by the instance data), Cwm is able to demonstrate that the assertion that Tiina has access is consistent with the policy, and can grant her access.

A working version of this example is part of an on-line Cwm tutorial which can be found at http://www.w3.org/2000/10/swap/doc/Trust. The example combines rule-based reasoning with the use of built-ins for cryptographic access to prove that access should be given. The key rule in the system is

```
this log:forAll :d, :k, :k2,.
{ :request a acc:GoodRequest } is log:implies of
{
   :request acc:forDocument :d;
         acc:requestSupportedBy :k.

 []      acc:certSupportedBy :k2,
           log:includes { :k a acc:RequestKey }.

 []      acc:certSupportedBy [a acc:MasterKey];
           log:includes { :k2 a acc:MemberKey }.
```

Which states: if a request is supported by a key, and there is a certificate -- signed itself with k2 -- which says k is a good request key, and that there is some other certificate, signed with the master key, that says that k2 is a member key, then the request is a good request.

A problem with our current use of Cwm in this example is that although it correctly meets the rules stated in the figure, it requires the bulk of the reasoning to be done by Cwm on the server's side. Thus, while the rules as to who can generate what certificates is somewhat distributed, the proof as to the trust-worthiness of the certificates is generated by Cwm on the site being accessed. We are exploring a system that is both more scalable and distributed by using cwm to generate proofs on the client side and then to transmit these (via http) to the server, which then only has to check the proof to see if it is both grounded and consistent.

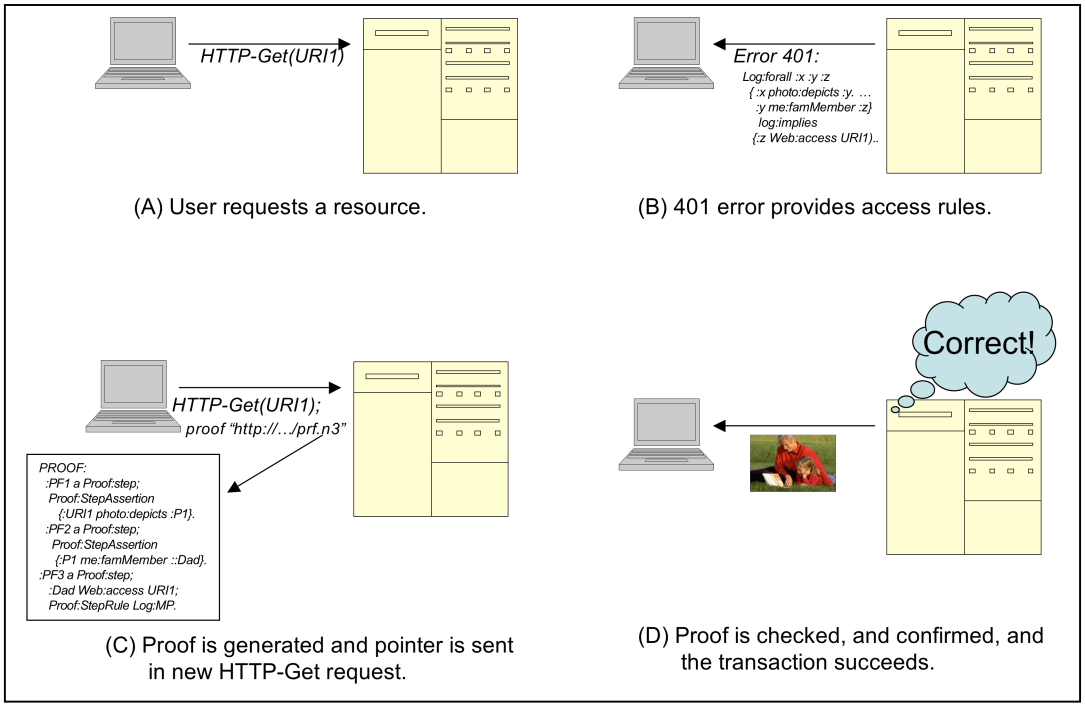
The protocol for doing this is quite straightforward and uses the standard web protocol. A user attempts to access a site as usual, clicking on a web link. This initiates an HTTP-GET request to the URI in question. Assuming the requested URI is protected by a set of access rules, the user will receive a 401 "Unauthorized" response. 401 errors, defined in IETF RFC 2617 (http://www.ietf.org/rfc/rfc2617.txt), are extensible by the addition of new tokens that define specific the authentication schemes, and we take advantage of this. We return, as a part of the 401 response, the N3 description of the access rules. The client can then generate a proof and transmit the URI for that proof to the server as part of a follow-up HTTP-GET requesting authorization. Figure 3 illustrates this use of the use of the 401 protocol for rule and proof exchange.

Although the protocol is straightforward, the transmittal of proofs requires its own syntax and semantics. Several research groups (cf Pinhiero da Silva et al, 2003; Hendler, 2004) have been working on developing languages for the exchange of proofs on top of the OWL language (essentially, simple proof ontologies). These ontologies are relatively straightforward, and allow the proof to be represented as a set of steps, each containing a list of previous steps they depend on and the rational used to produce the new clause. Thus, for example, a step in a proof might look like (in N3):

```
:PF8 a Proof:Step;
   Proof:StepAssertion
     {:Auth :Kari :URI1};
   Proof:StepRule log:ModusPonens;
  Proof:StepDependsOn (:PF6 :PF7).
```

stating that the assertion containing the fact that Kari authorized access to some URI followed by the logical rule modus ponens from two previous steps of the proof.

One of the more interesting aspect of proof checking on the Web is that the proofs presented may contain not just traditional logics, but also extended (higher-order) logics or even proof steps grounded in "non-logical" justifications. One of the most important examples of the use of "non-standard" logics on the Web is Proof-Carrying authentication (PCA - Appel and Felten, 1999;Bauer, 2003). The Princeton team working on PCA has designed and implemented a general and powerful distributed authentication framework based on a higher-order, constructive logic and postulated that higher-order logics can be used as a bridge between security logics in a way that would enable authentication frameworks based on different logics to interact and share resources. We believe this is an important technology for the Policy-Aware Web, and we are working on extending our proof language (and Cwm's processing thereof) with



a “quasi-quoting” facility to handle higher-order constructs such as those used in PCA in the open and distributed framework we are advocating.

In many cases, “proof steps” will actually be justifications that must be shared between the parties without the ability to appeal to a formal model theory or other proof of logical correctness. Thus, on the web, steps in a proof may be made by reference to an agreed upon “oracle” rather than to a logical mechanism. For example, suppose we have a rule that says you can only have access to an entire passenger roster if you can authenticate that you work for a Federal Agcncy and that you can produce the name of at least one passenger who has purchased a ticket for the flight. The former can be validated by the sort of keys and authentication discussed above, but the validation that passenger you have named has actually purchased a ticket may require that a separate airline system check its purchase database and make the answer available on the Web. One of the steps in the proof is to essentially say something like “you can find this asserted at the URI

> http://owl.mindswap.org/people/pages/pages.py?person=%7B%27link%27%3A+%27http%3A%2F%2Fowl.mindswap.org%2F2003%2Font%2Fowlweb.rdf%23JimHendler%27%7D”

which is clearly a non-standard logical mechanism – however, if the system checking the proof agrees that that Web Page is on a trusted server, and the assertion can be found there, then this can be a valid (and important) justification. Different users, of course, may have trust in different servers, may be willing to accept different sets of axioms in the proofs conveyed, etc.

Our policy aware approach to access control is a response, in part, to the observation that typical security architectures involve the requesting party doing very little computation -- typically, just providing a username/password, or perhaps computing a message digest and/or digital signature -- and the party providing and controlling access being obliged _and_ trusted to derive a justification based on the request credentials and some access control policy data (e.g. the file permission bits). Execution of even relatively inflexible policies described above epends on enormous trusted computing bases. At W3C, the trusted computing base starts with the entire linux and solaris kernels, apache, php, mysql; and we are constantly developing custom web-based tools; a bug in any of them puts our access control at risk. And the decision of how much software to trust can go beyond the boundaries of our organization: if W3C wants to prove to the satisfaction of some outside party that our policies have not been violated, we would need to audit this entire computing base to the outside party's satisfaction.

If we shift the burden of deriving the access justification to the requesting party, who transmits that justification to the controlling party, who need only check it, the resulting system (a) has a much smaller trusted computing base (only the part that verifies

justifications) and (b) is much more transparent: any third party can audit that the justification is valid.

We contend that such "social" proof mechanisms willy be a critical part of the Web access mechanism and we must handle them. Our work on conveying proofs therefore needs to be able to do more than say that "X is true." Rather, we must represent that "X is asserted to be True at Location Y" (or possibly a set of locations as in the Tiina example above). SHOE, an early Web Ontology Language developed at the University of Maryland, used a "claims logic" (Heflin et al, 1998;Heflin, 2001) to differentiate between a statement found on the Web and the resource asserting it. (The OWL language, the current web ontology standard, chose to use a more traditional model theoretic approach (based on description logics) rather than to use a less-standard claims logic). However, several features of the claims logic turn out to be powerful for proof checking on the Web, and we are revisiting these in our work. In particular, the appear to be three kinds of "proof steps" that are atypical in the standard proof checking literature:

- In the case we've been calling "grounded" above, the truth value of an assertion is checked by a "sensing action" (an HTTP-GET or checking a certificate) on the Web,

- The second case is where one proof checker may invoke another – for example, a specialized reasoning component may choose to delegate a complex piece of a proof to a more general system to check whether some rule holds, and

- The third case is where some sort of "meta reasoning" is needed about the assumptions. For example, if "Site X claims P", and if I believe that Site X is trustworthy, then I am willing to believe that P is true (even though there is no logical theory backing it up).

As we continue our development of a Cwm-based proof checking tool that can handle the protocol shown earlier (Figure 3), we are exploring how best to handle these cases.

FUTURE WORK

There are several key challenges to extending this work and making it practical for Web deployment. First, coherent representation and operation with inconsistency inherent in an open system such as the Web remains an unsolved problem. Most logic-based systems built to date are very intolerant of inconsistency. Most research has therefore focused on removing inconsistency by limiting expressivity, controlling data entry to disallow entries that could cause inconsistency, and/or by strongly enforcing integrity constraints and other similar mechanisms. Second, while we have demonstrated the design and implementation of one Policy Aware application (W3C site access control), there still remains substantial work to do in developing protocols and user interface strategies to enable the full range of transparency, compliance management, and accountability required for the Policy Aware Web.

Inconsistency

Unfortunately, in an open system such as the Semantic Web it is inevitable that inconsistency will occur and we contend that none of these approaches are likely to work: for our system to be powerful enough to control access to non-structured documents, web services and multimedia it must have at least the expressivity of OWL (which allows inconsistency), we cannot control data entry in an open system, and integrity constraints are difficult to maintain, let alone enforce, in a distributed and extensible system. Social mechanisms for enforcing consistency are also likely to fail, as inconsistency may be the result of error (for example, putting data in the wrong field on a form), serious disagreement (*ie*, the web sites of abortion supporters and opponents would be unlikely to have consistent ontologies) or maliciousness (the deliberate introduction of inconsistency to attempt to circumvent the very policies we are trying to enforce). Thus, developing an approach where inconsistency can be tolerated, and kept from causing harm, is one of the key areas of research in our work.

The primary problem with inconsistency is that in classical logics, not only are inconsistent statements false, but they entail every other statement whether related or not. Thus, the mere presence of an inconsistency in such systems renders everything meaningless -- rooting out inconsistency becomes essential as nothing useful can be done once the knowledge base becomes inconsistent. Paraconsistent logics are logics that tolerate inconsistency by blocking the inference from a contradiction to arbitrary conclusions. In essence, these logics are constructed so that the effects of contradictions are localized and do not propagate – thus if X & -X are asserted, it will not cause a system to believe Y, Z, Q, etc. unless these are specifically affected by the contradiction. Different paraconsistent logics localize contradictions in different ways: non-adjunctive logics (da Costa et al, 1977; Schotch and Jennings, 1980) prevent contradictory assertions from automatically forming self-contradictions (i.e. the truth of X and the truth of Y does not necessarily imply the truth of X AND Y), relevance logics (Routley et al, 1982; Restall, 1993) prevent explicit self-contradictions from entailing conclusions that are not directly related to the contradiction, and multivalued paraconsistent logics (Asenjo, 1966; Dunn, 1976) permit assertions to have truth values other than 1 or 0. Although all of these have been explored in the literature, few examples of paraconsistent reasoners have been implemented.

One notable exception is in the area of annotated logics for logic programming languages and especially the work of Kifer and Subrahmanian (1987; Subrahmanian, 1994). Annotated logics are an effective paraconsistent formalism for a number of reasons: they have clear semantics and a proof theory; they are a clean extension of FOL; and they are reasonably intuitive to work with. From a Semantic Web viewpoint they are also desirable as annotated logics fit well with the Semantic Web's focus on "triples" -- the natural locus for annotations (in fact, the claims logic of SHOE, described above, was implemented as an annotation framework in XSB). One difficulty with bringing annotation logics to the web is in determining what set of annotations (and logic) offers the right balance of user transparency, scalability, and expressivity. While

annotated logics allow the non-destructive presence of inconsistency, they often offer many incompatible ways of localizing the inconsistency and the effects of these on security policies have not been carefully explored. Annotated logics also have tended to work in a centralized and controlled framework, so integrating them into an open and multi-perspective framework like the Web produces a number of challenges. We are exploring how to develop an instantiation of the Kifer and Subrahmanian framework that is implementable in Cwm so that we can test various different annotation theories for their efficacy and usability.

Transparency, Compliance and Accountability revisited

The technical approach described in this paper has focused on the use of, and extensions to, N3 and Cwm for use in rule-based access on the Web. However, our goal of creating a policy-aware infrastructure for the Web includes more than just these basic infrastructure components. Achieving the triple goal of transparency, compliance management, and accountability requires exploration of the process of developing and agreeing upon policy vocabularies, and addressing a variety of complex user interface challenges in order to represent policy aware information to the general user. By exploring application models to enable communities to take advantage of policy description we believe we will be able to extend the reach of Semantic Web tools to meet policy aware requirements.

Policy Awareness begins with *transparent* access to rules associated with any given resource. While we have shown that it is possible to put rules on the Web and to use HTTP and RDF infrastructure to exploit them, we are still far from the full realization of Policy Awareness as described in Section 2. Making the rules explicit, publishable, and exchangeable via the HTTP provides a significant improvement in transparency, and from a "programmer's point of view" meets our stated goals. However putting this capability into the hands of end users will require much more work to determine how to build user interfaces (Ackerman et al, 2001) that provide usable access to social rules, and to tools that communities can use to decide on and develop rules.

The access control mechanism described in Section 3 illustrates a simple case of the *compliance management*, the second important attribute of the Policy Aware Web. While this access control mechanism demonstrates that rules engines can be used on the Web to mediate access, there is much more to be done to enable full Policy Awareness. Consider the example of a set of people exchanging photographs on the Web. A person posting a photo to a site might wish to know what the site's policy is with respect to sharing the photos. Similarly, a user wishing to share personal information might wish to publish a set of photos but control who can see them. Publishing a picture and saying "my friends" can see it seems simple, but actually raises complex issues. This is because we expect rules to be evaluated in a multiparty, multi-transaction social setting.

Social rules require careful consideration of unanticipated "transitive closure" when applied in more sophisticated, but likely more typical, community applications.

Consider the case where the user took a potentially embarrassing photo (say the unlikely case of a picture of someone drinking too much at the WWW conference). Publishing this to friends seemed straightforward to the user, but he forgot that some of his friends also worked at his company.

One of these people saw the photo, and republished it to his "business associates," which violated the original intent of the photographer. Further, the photographer (whose identity is encapsulated in the EXIF information in the photo) is unable to demonstrate that he was not the one who shared the photograph in the first place, earning the enmity of the photo's subject and other such social detriment. In this case, building the 3rd component of Policy Awareness, *accountability*, would help the community of photo-sharers to figure out how and even why a photo got shared beyond the intended constraints. Perhaps someone made a mistake, or perhaps someone played a malicious joke. Accountability mechanisms that reconstruct the proofs presented to gain access and establish what policy statements were associated with the image when it appears outside the boundaries established by the community can help to establish whether the act was intentional or inadvertent.

CONCLUSION

The infrastructure discussed in this paper is a starting place for exploring this important problem and for allowing the greater sharing of personal information. However, it is just a start, and much work remains to be done if we are to eventually see a truly Policy-Aware Web. We have described the development of rule-based policy management system that can be deployed in the open and distributed milieu of the World Wide Web. Combining a Semantic Web rules language (N3) with a theorem prover designed for the Web (Cwm) we have shown that it is possible to apply rules on the Web using the Hypertext Transport Protocol (HTTP) to provide a mechanism for the exchange of rules and, eventually proofs. We have also shown how this mechanism can provide a base for a Policy Aware infrastructure for the Web, and have argued for the necessity of such infrastructure.

We anticipate that Policy Awareness tools will enable the Semantic Web to address a wide range of policy requirements: questions such as who owns the copyright to a given piece of information, what privacy rules apply to an exchange of personal information, and what licensing terms apply to a particular piece of genetic information are all examples of social needs that Policy Awareness can help mediate. In testimony to the United States Congress in 2000, Daniel Weitzner argued:

> "This same interactivity, the bi-directional ability to exchange information from any point to any other point on the Net has brought about significant threats to individual privacy. For the same communications mechanisms that give individuals the power to publish and access information can also be used, sometimes without the user's knowledge or agreement, to collect sensitive personal

> information about the user … Our goal is to use the power of the Web, and enhance it where necessary with new technology, to give users and site operators tools to enable better knowledge of privacy practices and control over personal information" Daniel J. Weitzner, Testimony to US Senate Commerce Committee, May, 2000

Policy awareness is not alone sufficient to solve the pressing public policy problems raised by the interaction of the Web and society, but we believe that Policy Aware infrastructure is a necessary part of enabling human institutions and communities to adapt to this new environment.

ACKNOWLEDGEMENTS

The authors thank Joe Pato, Hewlett-Packard Laboratories and Poorvi Vora, George Washington University for insightful discussions of trust in ad hoc online communities, and Ted Wilson of Hewlett-Packard for initial exploration of policy capable infrastructures.. Some or all of the authors were supported in part by grants from the DARPA Agent Markup Language program, DARPA Fast C2 Applications project, and funding from the Army Research Laboratory, the National Institute for Standards and Technology and other Department of Defense agencies as well as a grant from the National Science Foundation which directly supports this work. We also thank Fujitsu Laboratory of America College Park, NTT Corp., and Lockheed Martin for funding of the University of Maryland MIND Laboratory.

REFERENCES

Ackerman, M., Darrell, T., & Weitzner, D. J. (2001). Privacy in Context. *Human-Computer Interaction*, 16, pp. 167-176

Appel, A. and Felten, E., (1999). Proof-Carrying Authentication, *6th ACM Conference on Computer and Communications Security*, November, 1999.

Asenjo, F.G. "A Calculus of Antinomies", (1986). Notre Dame Journal of Formal Logic, Vol. XVI, pp. 103-5, 1966.

Barbour, G. (2002). *Program Modeling: A Machine Learning Approach to Intrusion Detection*, PhD Thesis, Computer Science Dept, University of Maryland. 7, 2002.

Barbuti, R., Bernardeschi, C. and De Francesco, N. (2004), Analyzing Information Flow Properties in Assembly Code by Abstract Interpretation, *The Computer Journal,* Volume 47, Issue 1, January 2004: pp. 25-45

Bauer, L. (2003). Access Control for the Web via Proof-carrying Authorization.. Ph.D. Thesis, Princeton University, September 2003.

Bauer, L., Schneider, M. and Felten, E. (2002). A General and Flexible Access-Control System for the Web, *Proceedings of the 11th USENIX Security Symposium*, San Francisco, CA, August 2002.

Behrend, A. (2003). Soft stratification for magic set based query evaluation in deductive databases, Proc. International Conference on Management of Data and Symposium on Principles of Database San Diego, California, 2003, 102 – 110

Berman, J. & Weitzner, D. (1995). Abundance and User Control: Renewing the Democratic Heart of the First Amendment in the Age of Interactive Media, 104 Yale L.J. 1619 (1995)

Berners-Lee, T. (2000) Primer: Getting into RDF & Semantic Web using N3, October, 2000 (and updated) http://www.w3.org/2000/10/swap/Primer.

Berners-Lee, T., "Semantic Web," (2000) presentation at XML 2000, Washington, DC, December, 2000. (http://www.w3.org/2000/Talks/1206-xml2k-tbl/slide10-0.html)

Berners-Lee, T., CWM – A general purpose data processor for the Semantic Web, 2000, http://www.w3.org/2000/10/swap/doc/cwm.html

Berners-Lee, T., Fielding, R. and Frystyk, H. (1996). "Hypertext Transfer Protocol - HTTP/1.0," HTTP Working Group, Feb. 1996.

Berners-Lee, T., Hendler, J., and Lassila, O. (2001). The Semantic Web: When the Internet gets Smart, *Scientific American*, May, 2001.

Boley, H., Tabet, S. and Wagner, G. (2001). Design Rationale of RuleML: A Markup Language for Semantic Web Rules, Proc. SWWS'01, Stanford, July/August 2001.

Boneh and Franklin, M. (2001). Identity based encryption from the Weil pairing, *Crypto 2001* (LNCS 2139), 213-229.

Burrows, M., Abadi, M. and Needham, R. (1989). A logic of authentication. *Proc. of the Royal Society of London*, volume 426 of LNCS, pages 233-271. Springer-Verlag, 1989.

da Costa, N.C.A. and Dubikajtis, L. (1977). On Jaskowski's Discussive Logic, Non-Classical Logics, Modal Theory and Computability, A.I. Arruda, N.C.A. da Costa and R. Chuaqui (eds.), North-Holland Publishing Company, Amsterdam, pp.37-56.

Didriksen, T. (1997). Rule based database access control - a practical approach. ACM Workshop on Role-Based Access Control 1997: 143-151

Dunn, J.M. (1976). Intuitive Semantics for First Degree Entailment and Coupled Trees, *Philosophical Studies*, Vol. XXIX, pp. 149-68, 1976.

eXtensible Access Control Markup Language, OASIS Technical Committee, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

Ferraiolo, D., Kuhn, D. and Chandramouli, R. (2003). *Role-Based Access Control,* Artech House.

Heflin, J. (2001). *Towards the Semantic Web: Knowledge Representation in a Dynamic Distributed Environment*, Ph.D. Thesis, University of Maryland, College Park, 2001.

Heflin, J., Hendler, J. and Luke, S., (1988). Reading Between the Lines: Using SHOE to Discover Implicit Knowledge from the Web, by Jeff Heflin, James Hendler, and Sean Luke. *AAAI-98 Workshop on AI and Information Integration*.

Hendler, J. (2004) An OWL Full Ontology for creating proof instances on the World Wide Web (submitted for Publication, 9/04).

Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Macgregor,R., Grosof, B. and Dean, M. (2003). SWRL: A Semantic Web Rule Language Combining OWL and RuleML, Version 0.5, November 2003, http://www.daml.org/rules/proposal/

Kagal, L., Paoucci, M., Srinivasan, N., Denker, G., Finin, T., and Sycara, K. (2004). Authorization and Privacy for Semantic Web Services, IEEE Intelligent Systems (Special Issue on Semantic Web Services), July, 2004.

Kifer, M. and Subrahmanian, V.S. (1989). On the Expressive Power of Annotated Logic Programs. In E. L. Lusk and R. A. Overbeek, editors, *Proc. North American Conf. on Logic Programming*, pages 1069--1089.

Knowledge Interchange Format, http://logic.stanford.edu/kif/kif.html

Kyte, T. (2000). Fine Grained Access Control (aka DBMS_RLS), http://govt.oracle.com/~tkyte/article2/index.html, August, 2000.

Menzel, C. and Hayes, P. (2003). Standard Common Logic, http://ceur-ws.org/Vol-82/SI_paper_12.pdf

Ott, A. (2001) The Rule Set Based Access Control (RSBAC) Linux Kernel Security Extension, *Proc. International Linux Congress*, 2001.

OWL Web Ontology Language Semantics and Abstract Syntax, W3C Recommendation 10 February 2004, http://www.w3.org/TR/owl-semantics/

Pandey, R., and Hashii, B. (1999). Providing Fine-Grained Access Control for Java Programs, *Concurrency: Practice and Experience*.

Pinheiro da Silva, P., McGuinness, D. and McCool, R. (2003). Knowledge Provenance Infrastructure. *IEEE Data Engineering Bulletin* Vol.26 No.4, pages 26-32.

PORTIA: Privacy, Obligations, and Rights in Technologies of Information Assessment, http://crypto.stanford.edu/portia/ -- see also: Dan Boneh, Joan Feigenbaum, Avi Silberschatz, and Rebecca Wright, To appear in the *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering*, 2004

RDF Semantics, W3C Recommendation 10 February 2004, http://www.w3.org/TR/rdf-mt/

Reno v. ACLU, 521 US 844 (1997) (see http://supct.law.cornell.edu/supct/html/96-511.ZO.html)

Restall, G. (1993) Simplified Semantics for Relevant Logics (and some of their rivals), Journal of Philosophical Logic, Vol. XXII, pp. 481-511, 1993.

Routley, R., Plumwood, V., Meyer, R.K., and Brady, R.T. (1982). Relevant Logics and Their Rivals, Atascadero, Ridgeview, CA, 1982.

Sandhu, R., Coyne, E., Feinstein, H. and Youman, C. (1996). Role-based access control models. *IEEE Computer*, 29(2):38-47.

Schotch, P.K. and Jennings, R.E. (1980). Inference and Necessity, Journal of Philosophical Logic, Vol. IX, pp. 327-340, 1980.

Schunter, M. (ed,), The Enterprise Privacy Authorization Language, IBM Corp. http://www.zurich.ibm.com/security/enterprise-privacy/epal/Specification/index.html

Shamir, A. (1985). Identity-based cryptosystems and signature schemes, *Advances in Cryptology - CRYPTO '84*, Lecture Notes in Computer Science, vol. 196, Berlin: Springer Verlag, pp. 47--53.

Simple Distributed Security Infrastructure, http://theory.lcs.mit.edu/~cis/sdsi.html

Simple Public Key Infrastructure (open-source project) http://sourceforge.net/projects/spki/

Subrahmanian, V.S. (1994). Amalgamating Knowledge Bases. TODS 19(2): 291-331.

Swartz, A. and Hendler, J. (2001) The Semantic Web: A Network of Content for the Digital City, *Proceedings Second Annual Digital Cities Workshop*, Kyoto, Japan, October, 2001.

The Platform for Privacy Preferences 1.1 (P3P1.1) Specification, W3C Working Draft 20 July 2004, http://www.w3.org/TR/P3P11/

The Rule ML Initiative, http://www.ruleml.org/; See also [BoTW01]

The Security Assertion Markup Langauge, OASIS Technical Committee, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

Weitzner, D. (2000) Testimony before the United States Senate Commerce Committee Hearing on Online Privacy. May 2000.

Weitzner, D. (2004). The Transparency Paradox: Privacy design strategies for open information networks, (Extended Abstract) http://www.w3.org/2004/05/loc-priv-transparency-extab.html]