6-1-2009

# A Computational Framework for Certificate Policy Operations

Gabriel A. Weaver
*Dartmouth College*

Scott Rea
*Dartmouth College*

Sean W. Smith
*Dartmouth College*

# A Computational Framework for Certificate Policy Operations*

Dartmouth Computer Science Technical Report TR2009-650

Version of June 1, 2009

Gabriel A. Weaver, Scott Rea, Sean W. Smith
Department of Computer Science/Dartmouth PKI Lab Dartmouth College
Hanover, New Hampshire USA

**Abstract.** The trustworthiness of any Public Key Infrastructure (PKI) rests upon the expectations for trust, and the degree to which those expectations are met. Policies, whether implicit as in PGP and SDSI/SPKI or explicitly required as in X.509, document expectations for trust in a PKI. The widespread use of X.509 in the context of global e-Science infrastructures, financial institutions, and the U.S. Federal government demands efficient, transparent, and reproducible policy decisions. Since current *manual* processes fall short of these goals, we designed, built, and tested *computational* tools to process the citation schemes of X.509 certificate policies defined in RFC 2527 and RFC 3647. Our *PKI Policy Repository*, *PolicyBuilder*, and *PolicyReporter* improve the consistency of certificate policy operations as actually practiced in compliance audits, grid accreditation, and policy mapping for bridging PKIs. Anecdotal and experimental evaluation of our tools on real-world tasks establishes their actual utility and suggests how machine-actionable policy might empower individuals to make informed trust decisions in the future.

**Keywords:** PKI; Certificate Policy Formalization; XML

## 1 Introduction

The fundamental purpose of PKI is to allow relying parties to trust users based upon a set of credentials the user has proven they have control over. Confidence in these user credentials requires the relying party to evaluate the trustworthiness of an association between a public key and one or more attributes. These attributes may serve to identify a person, machine, or organization, or they may simply associate an arbitrary property with a public key.

**Foundations for Trust in a X.509 PKI** In X.509, these associations are expressed in a machine-actionable document called a *certificate* and a *certificate authority (CA)* attests to the validity of these associations. The *certificate policy (CP)* of an organization contains a set of expectations that define its notion of a trustworthy public key certificate and how it may be used. The *Certification Practices Statement (CPS)* states how a CA actually implements a CP. In principle, the trustworthiness of a certificate is a function of the similarity of

* This work was supported in part by the NSF (under grant CNS-0448499) and the U.S. Department of Homeland Security (under Grant Award Number 2006-CS-001-000001). The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of any of the sponsors.

stated policy to personal trust expectations (measured in the *Policy Comparison* process) and the similarity of stated policy to actual practice (*Compliance Audit*).

Evaluating the similarity of policies depends upon the comparison of CPs. Ideally users should evaluate trusted roots based on their personal expectations. In actual practice, the average user blindly adopts the trust expectations of their employers, application vendors, or of companies like Thawte and VeriSign. Until user-friendly policy becomes a reality, requirements for trust will continue to be expressed for organizations, by organizations.

Currently, people manually compare the two policies involved, reading both documents line by line. Bridge CAs relieve organizations of much of this burden, employing this same manual process to establish trust relationships between member organizations. Bridge CAs attest that the expectations of trust for all of its member organizations are logically consistent. If a member organization issues a certificate, that certificate is viewed as trustworthy among all members of the bridge at a predetermined level of assurance [17].

A policy is probably useless unless it is followed in practice. *Compliance audits* determine whether a CA issues certificates according to a CP in actual practice. Federations and bridges ensure that the requirements for trust are met through such audits, whether for accreditation or cross-certification respectively. For example, the current approach to accreditation at the *International Grid Trust Federation (IGTF)* defines a mostly manual process; one step involves *Policy Management Authority (PMA)* members reviewing a CP in detail, comparing its contents against requirements in an *authentication profile (AP)* [18]. Auditing PKIs within financial institutions also involves following a manual process described in ISO 21188 [19] [22]. Cross-certification, the operation necessary to establish a bridge, similarly requires comparing CPs.

We claim that informed trust decisions should be based on processes that consistently estimate actual organizational behavior. *Consistent processes* occur frequently in a manner transparent to the user; their results are both reproducible and auditable. Currently-practiced, manual CP operations are costly, time-consuming, lack end-user transparency, and are difficult to reproduce. We claim that computationally processing machine-actionable certificate policies would be more efficient and consistent.

**Our Contributions to Certificate Policy Formalization** Our *PKI Policy Repository*, *PolicyBuilder*, and *PolicyReporter* codify certificate policy processes used in PKI compliance audits, grid accreditation, and bridging PKIs. We claim that these tools improve the efficiency and consistency of policy retrieval, creation, and comparison. This section presents our contributions in the context of previous work on the identification, representation, and manipulation of certificate policies.

*Identification* Our tools identify certificate policies using a hierarchical, human-readable, machine-actionable reference string called a *Canonical Text Services Uniform Resource Name (CTS-URN)* [29]. CTS-URNs emerge from some of our previous work on the *Multitext of Homer Project* [14] to perform

computations on Classical Greek texts [31]. One of this paper's contributions is to use CTS-URNs encoded as Object Identifiers (OID)s to identfy individual CPs, arbitrary sets of policy documents, or other versions or translations of that same document.

CTS-URNs also identify sets of policy requirements organized within a citation (or reference) scheme. RFC 3647, Section 6 [12] explicitly defines a citation scheme organized in terms of policy requirements to facilitate the comparison of CPs and CPSs. CTS-URNs enable the tools we built to compute with respect to a policy's reference structure, identifying individual security provisions[1] or the document in its entirety.

Our identification scheme increases the expressiveness of policy related certificate extensions by encoding CTS-URNs as OIDs. Accepting or rejecting CPs is currently a binary decision; policy documents must be "completely accepted or forbidden" [21]. CTS-URNs let people and machines reference arbitrary sections of a policy; rather than accepting or rejecting an entire policy document, users may whitelist or blacklist agreeable or offending provisions.

*Representation* We encode certificate policies using *Text Encoding Initiative (TEI) P5 Lite*, an XML standard for representing texts in digital form [5]. Like previous efforts to encode policies using XML [8] [7], we model a security policy as a tree.[2] Given a policy's text, we only mark up its citation scheme, the outline of provisions defined in Section 6 of RFC 3647 or Section 5 of RFC 2527 [11]. This results in a *semi-formal* [7] policy representation that is both machine-actionable and human-readable. This approach simplifies the overhead of encoding a policy.

The *Federal PKI Policy Authority (FPKIPA)* Technical Specification recommends writing CPs and CPSs in a natural language [12]. Our representation honors that recommendation and leaves the natural language of the policy unchanged. Alternate representations of policies such as data-centric XML[3], matrices, or ASN.1 require a person to read the source text and fit their interpretation of that text to a data format. Such representations are unsuitable for relying parties to easily understand the meaning of a policy [7] [16] [12]. Our contribution is a policy representation that humans can use as a primary source for informed policy decisions and which computers can process.

*Manipulation* Computing on certificate policies overcomes many of the drawbacks and limitations of manual processes. We claim that implementations of algorithms are efficient and *consistent* processes for explicitly imposing a model on policy content. Algorithms may be run frequently. Unlike deriving a model of policy content from text and encoding it by hand, algorithms can be run on demand any time as CPs change. Algorithms are unambiguous and when their implementations are open-source, the underlying process is transparent to the user. Finally, the output of algorithms may be reproduced by running the same input and interpreted to make informed trust decisions.

---

[1] Trcek et. al's DNS-like system organized the set of all possible security requirements hierarchically into domains that were referenced by human-readable, machine-actionable strings [33].

[2] Our approach is inspired by current approaches to digitizing Classical texts [14] [13].

[3] *Document-centric* XML is the type of documents "written by hand, by an author" like a letter or chapter. *Data-centric* XML is used to transport data between computers [27].

**This Paper** This paper is organized as follows. Section 2 describes some real-world use cases for certificate policy and the reliability of current practices. Section 3 introduces the design and implementation of the tools we built for processing certificate policy and illustrates how they address real-world needs while improving actual practice. We evaluate these tools in Section 4, using anecdotal and experimental evidence. Section 5 reviews related work. Section 6 describes future research directions building upon this work, and Section 7 concludes.

## 2 Problems with Manual Certificate Policy Processes

Certificate policy defines the trustworthiness of a CA and therefore is fundamental to the trust one places in a PKI. This section discusses three real-world X.509 processes which directly use certificate policy: PKI compliance audits, IGTF accreditation, and policy mapping for bridging PKIs. In each case, we'll describe the process in principle and its importance to the trustworthiness of a PKI. Then we'll describe how the current implementation of this process limits trustworthiness and motivate the need for *consistent* operations on certificate policies.

**PKI Compliance Audits** Like accreditation processes, PKI audits verify that the certificate policies and certification practice statements are consistent with a "framework of requirements." In the financial services industry, ISO 21188 specifies such a framework that evolved from WebTrust and ANSI X9.79 [22]. Audits for WebTrust compliance should occur at least every 6 months [15]. During these audits, PKIs are evaluated with respect to five objectives: *security*, *availability*, *processing integrity*, *confidentiality*, and *privacy*. Each of these objectives are defined using principles, "broad statements of objectives," and criteria, "benchmarks that should be objective, measurable, complete, and relevant" [34]. Through Assurance Services, a *Certified Public Accountant (CPA)* may express a WebTrust opinion on whether an objective is met. By keeping objective observations separate from opinion, WebTrust audits are an important and objective process for certifying that actual practice reflects written policy. Additionally, the *International Collaborative Identity Management Forum (ICIDM)* and *Four Bridges Forum (4BF)* [1] are working to define a standard PKI audit process. The IGTF also publishes a set of auditing guidelines [32].

*Frequency* Compliance audits are expensive in time and money, and limited by the bottleneck of human observation. Although compliance audits like WebTrust are supposed to occur at least every 6 months, in actual practice they usually happen less often. Even when these audits occur on schedule, auditors' observations only sample a glimpse of much larger, continuous business processes. By necessity, PKI audits require human intervention; expressing an opinion based upon observed criteria requires human judgement. However, the criteria themselves, measureable and objective by definition, do not require human judgement to measure.

*Transparency* Current auditing schemes lack transparency. Users cannot evaluate an organization's observed behavior in terms of their own trust requirements. Instead users implicitly delegate trust decisions to a CPA or other audit professional. Furthermore, because ISO 21188 and ANSI X9.79 are not freely available, the average user has no way of knowing the trust decisions being delegated.[4] These institutional and economic walls ensure that trust evaluation resides at the organizational level in spite of trust's personal nature. Through combining documentation with computational methods, measurements of trust criteria could be made accessible to users to individually decide the degree of trust they place in an organization.

*Reproducibility* Compliance audits, as currently practiced, are difficult to reproduce because they are so dependant upon auditors' individual observations. Certainly audits attest to an organization's trustworthiness at the time of the last audit; however they say nothing about the current state of the organization. Were an auditor to try to reproduce an audit, the conditions under which the original audit occurred may be extremely difficult or impossible to reproduce because organizations are dynamic, changing entities. Audits rely upon the past as the sole indicator of current and future performance.

**IGTF Accreditation** Researchers using computational grids employ many thousands of distributed nodes to solve complex computational problems by sharing resources. *Grids* often group users under very large *Virtual Organizations (VOs)* which usually reflect real-world collaborations between researchers and institutions. Computational power, data storage, and network bandwidth all must be shared between members of a VO. Since these resources are valuable, access is limited based on the requested resource and the user's identity. Each grid must enforce these limits by providing secure authentication of users and applications. Unauthorized access to resources is unacceptable, especially given the large size of a VO [26]. The IGTF uses X.509 PKI to ensure that grid authentication mechanisms meet a defined level of assurance.

A distributed architecture like the grid requires compatible, non-contradictory policies among member organizations. An IGTF-accredited member should issue policy consistent with all other members, and thereby satisfy the IGTF's standard for trust. The purpose of the IGTF is to "harmonize the work on authentication for e-Science production infrastructures."[18] It accomplishes this by establishing common policies and guidelines between PMAs as well as ensuring compliance to the resulting Federation Document amongst the participating PMAs. Currently, the IGTF maintains a set of authentication profiles (AP) which specify the policy and technical requirements. During accreditation the prospective member sends the Certificate Policy (CP) around to other members for comments and asks multiple PMA members to review it in detail. Eventually this CP, along with recommendations from the reviewers, is presented at the PMA meeting for immediate approval or deferral.

[4] Anecdotally, we believe the costs of ISO 21188 and ANSI X9.79 may discourage organizations from following these standards in actual practice.

The IGTF defines accreditation in terms of manual procedures, institutionalizing bottlenecks that might be eliminated through technology. Accreditation should be defined in terms of functional requirements, not in terms of a particular implementation of those functions.

*Frequency* The IGTF's requirement that any change in policy requires reaccreditation may penalize organizations that change policy to reflect actual practice. Since organizational practices change rapidly and policies should reflect practice, policies need to be able to change rapidly to mirror the actual organization. So as not to penalize organizations for reporting their actual practices, reaccreditation should be defined to accommodate frequent organizational changes. Furthermore, when the IGTF changes an AP, all members have just 6 months to re-certify that they are compliant with the new profile.

*Transparency* Users authenticating to the grid are unable to see for themselves how well member institutions satisfy APs and are implicitly forced to trust IGTF accreditation. While users may not be able to learn every nuance of an accreditation process, the evidence leading to an accreditation should be readily available to grid users. All members are equally trusted under the current scheme. In reality some of them may satisfy the AP or portions of the AP better than others. Grid authentication will only be as strong as its weakest member.

*Reproducibility* Since IGTF accreditation is defined in terms of a manual process, reproducing or auditing an accreditation decision is difficult. Members review a CP in private and then present their findings at an IGTF meeting. The process places much burden on volunteers who may not have much experience in the accreditation process. These reviewers then present their own opinions on the compatibility of the CP with the federation's AP. Since CPs are large documents, other IGTF members may trust the reviewers' opinion rather than reading the CP on their own. While reviewers' opinions determine much of the accreditation decision, the criteria used in forming that opinion may not be captured for future reference.

**Policy Mapping for Bridging PKIs** Bridge CAs, though not themselves anchors of trust, establish relationships with different PKIs so that users from different PKIs can decide whether to trust one another. Bridges exist to mediate trust in several areas including the pharmaceutical industry, the U.S. Federal government (FPKIPA), the defense and aerospace industry, and higher education (HEBCA). Creation of these bridges requires mapping policies between member PKIs. When a new organization wishes to join a bridge, the bridge CA will compare the candidate organization's CP to its own. If suitable, the bridge CA will then sign the certificate of the candidate organization's trust root. Sometimes a bridge CA may even use the policy mapping certificate extension to establish an equivalence between a member organization's policy and one of its own CPs.

*Frequency* Policy mapping occurs when an organization first requests to join a bridge and becomes invalid when actual practice changes. The timelines over which written policy, policy mapping, and actual practice change are not in sync. Policy mapping happens much more infrequently than actual practice

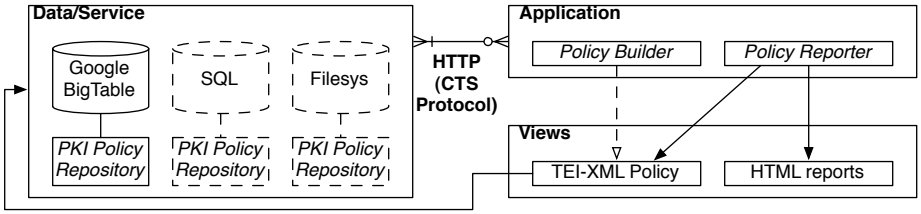


**Fig. 1.** System architecture for our certificate policy framework. (Dashed modules not yet implemented)

changes. As such, diligent organizations keeping their policy statement up to date pose a challenge to bridge CAs who must manually map a member CP into their own. If the policy of the bridge CA changes, it may be important to know how well the CPs of member organizations satisfy the new policy.

*Transparency* The actual evidence used to decide whether an organization belongs to a bridge is not readily available to its users. Without this evidence, users know whether or not an organization meets the bridge requirements but do not know the extent to which those requirements are met.

*Reproducibility* Policy mapping in actual practice does not distinguish between expressed opinion and criterion; there is no way to easily reproduce or evaluate the mapping process for relying parties. Although the policy mapping claimed by the bridge CA may be documented in a certificate extension, these mappings only reference entire policy documents and as such are unsuitable for reconstructing the evidence for membership in a bridge. Mapping matrices are used to document CP compliance, but these are not typically available to the relying party.

## 3 Computational Tools: Design and Implementation

We designed and implemented the *PKI Policy Repository*, *PolicyBuilder*, and *PolicyReporter* to improve the *efficiency* and *consistency* of policy retrieval, creation, and comparison. Each of these tools rests upon our formalization of certificate policy: we identify and reference policy via CTS-URNs and represent policy in TEI-XML. Each tool fully or partially automates one or more of the policy operations and improves their frequency, transparency, and reproducibility. These tools will be released in an open source distribution following publication. Figure 1 illustrates the design and implementation of the tools we will now consider. For each tool, we briefly discuss its relevance to the previous use cases. We then present each of our solutions in the context of current actual practice and prior research on policy formalization.

**PKI Policy Repository** The *PKI Policy Repository* stores certificate policies for retrieval by their reference structure. PKI audits, accreditation, and policy mapping depend upon the reference, and retrieval of certificate policies and yet little work has been done to automate or partially automate these fun-

damental processes. The *PKI Policy Repository* fills this void, and sets the stage for individuals to access and evaluate certificate policy.

*Reference* We extended OIDs to cover the entirity of a certificate policy's reference structure. Furthermore we implemented a bidirectional mapping from a machine-actionable OID to a human-readable and machine-actionable CTS-URN. Certificate policies are reference works by design. The ability to reference these policies motivated RFC 3647, its predecessor 2527, and a parallel reference structure (or citation scheme) in *Certification Practice Statements (CPSs)*. Policy comparison proceeds much more quickly between two policies sharing the same reference structure. Bridge CAs attest to the trustworthiness of PKIs by listing references to equivalent CPs in the policy mapping extension. Unfortunately the OIDs they use only reference the entire certificate policy, causing shallow, very coarse policy mappings. In actual practice, people are interested in referencing meaningful sets of security requirements. This is why policies have a reference structure in the first place! Trcek et al.'s DNS-like system used machine-actionable, human-readable references to security policy domains, allowing one to reference meaningful sets of security requirements [33]. RFC 3647 and 2527 define meaningful sets of security requirements through the reference structure.

We also encoded CTS-URNs as OIDs for referencing arbitrary sets of certificate policies, as well as different editions and translations of the same Certificate Policy. Compliance audits, grid accreditation, and bridging PKIs all attest to the trustworthiness of a CP at a particular point in time. Since policies change over time, we claim such attestations should reference the version of the policy at the time of the audit. In prior work, Grimm proposed using multiple versions of the same policy, one expressed formally for machines to process and one expressed informally for administrators to understand its meaning [16]. Although our policy is *semi-formal*, it would be possible to treat other policies–such as Casola et al.'s highly-structured XML [8] [7] and perhaps even ASN.1 representations–as versions of a policy which can be referenced. Since code is text, we can even reference code claiming consistency with a certificate policy provision as if it were another version of that provision. Using OID-encoded CTS-URNs allow multiple versions of a policy, whether plain text, XML, or code to be uniformly referenced through a parallel citation scheme. This accomplishes Grimm's idea of associating an implementable part of security policy with the "set of specified security measures" realized by the implemented security functions [16].

*Retrieval* Retrieving referenced sections of security policy traditionally involves turning printed pages or scrolling through a PDF. Given a reference to a CP such as section 7.1, a CA typically browses his file system to the appropriate PDF, opens it and then scrolls through it until he gets to the appropriate section. Even when searching the PDF, results tend to be reported in page numbers. In actual practice, the mechanisms for policy retrieval are largely unrelated to the mechanisms for citing (referencing) policy. In prior research on policy formalization, little work has been done with policy retrieval. One exception is *PolicyMaker* which allows one to query policy actions using a database-like syntax [3].

For digital editions of reference works, we claim that page numbers are an unnecessary artifact of print. Digitization allows one to navigate texts by logical reference rather than by page number. The *PKI Policy Repository* is a *Canonical Text Services (CTS)* server[5], loaded with certificate policies. The CTS protocol [30] uses HTTP to provide a simple REST XML web service for retrieving canonical texts. Users or applications can retrieve sections of policy by supplying a CTS-URN and other HTTP request parameters. The *PKI Policy Repository* currently uses a Google AppEngine implementation of CTS, although a Groovy/Java implementation also exists. Programs may process the XML policy fragments from a CTS service and transform it into PDF or HTML. This tool lets CAs query an entire database of policy in terms of how it is traditionally referenced, saving them from having to manage PDFs or other representations on their own. Furthermore, this tool opens the door for algorithms to process the content of certificate policy.

**PolicyBuilder** The *PolicyBuilder* assists CAs in creating new policies from extant ones. In actual practice new certificate policies may be created when a CA wants to join a federation or bridge. Typically CAs copy and paste passages of extant policy into their new policy and selectively edit a few words and phrases as needed. The more similar the new, derivative certificate policy is to older, already accepted policies, the greater the chances for the new policy to be accepted. Under these circumstances, policy creation is quickly followed by policy review. While Klobucar et al. have stated the need for machine-assisted policy creation [21], no tools have been built to fill this need and none have emerged that consider policy creation as a means to streamline policy review.

The *PolicyBuilder* fills the need for machine-assisted policy creation while facilitating the review and evaluation of newly-created policies. Rather than copying and pasting policy statements from PDFs, *PolicyBuilder* imports policy content directly from CPs in one or more *PKI Policy Repositories*. More specifically, the PolicyBuilder initializes an empty reference tree as defined in RFC 3647 and populates it with corresponding content from selected policies. Policy content currently includes assertions, or security requirements qualified by MUST, SHOULD, or other adjectives from RFC 2119. Rather than copying and pasting content, policy assertions are imported into the new document by simply clicking on them. Once a document tree is built to satisfaction, the CA may serialize policy to XML, PDF, or HTML. Since each assertion includes a CTS-URN to its source policy, CAs can see how many security requirments they imported from bridge or grid-approved CPs. Similarly, reviewers can process the XML and measure how much content is original and how much comes from already-approved policies.

**PolicyReporter** The *PolicyReporter* helps users obtain more, higher-quality information useful for comparing certificate policies. Although policy comparison is a fundamental operation in bridging PKIs and grid accreditation, it remains a highly-manual, highly-subjective process, making it difficult to perform consistently. People compare two CPs at a time, line-by-line, evaluating

[5] Again, we helped to develop the CTS Protocol for serving Classical texts in the *Multitext of Homer Project* [14].

their similarity. For a person with a lot of experience, this can take 80-120 hours depending upon the reference structure of the policies compared. The hardest comparisons include policies with non-standard reference schemes.

Prior work in machine-assisted policy comparison encompasses a variety of techniques ranging from using fuzzy theory on highly-formal policy representations [7] to imposing a partial or full order on specific values in policy content [21]. Still others organize sets of security requirements into a tree and compare policies by looking at where they sit within the tree [33]. However none of these approaches automatically process the texts as they were designed to be compared, by reference structure. Certainly not all policies follow RFC 3647 neatly or even at all [28]. However our work illustrates further benefit to following RFC 3647 or even 2527, for it allows a standard set of analyses for comparing CPs to develop.

The *PolicyReporter* aggregates information about a set of policy provisions (the criteria for comparison) into a report by walking the citation structure[6] of each text. Given a reference to a policy, it queries one or more *PKI Policy Repositories*, retrieves the referenced content, and processes it using some algorithm. The results of these algorithms are either human-readable HTML report or an XML document. Although *PolicyReporter* currently operates on *semi-formal* source texts, it could just as easily analyze other, more structured interpretations of the source like the markup proposed by Casola et al. Our immediate goal is to help CAs find large discrepancies between CPs such as the number of security requirements of a certain significance.

Currently three algorithms for extracting information from certificate policies are implemented for the *PolicyReporter*.

*RFC 2119 Analysis* The *RFC2119Analyzer* counts the number of occurrences of words in one of three categories defined in RFC 2119 [4] to indicate the significance of a requirement. Policy statements with the highest importance contain the words MUST, REQUIRED, or SHALL, the next most important provisions contain SHOULD or RECOMMENDED, and the least significant requirements use MAY or OPTIONAL.

Since security provisions in CPs use RFC 2119-like language, counting the number of words in each category indicates how strictly a section of certificate policy needs to be followed. Interpreting the results of this analysis is simple. A large difference in these counts indicates discrepancies in the requirement levels of two sections of policy.

*RFC 2527 Policy Mapper* The *RFC 2527 Policy Mapper* takes a section of text from an RFC 2527 certificate policy and automatically maps it into RFC 3647 TEI-XML. Every mapped provision contains a reference to its source policy. The generated RFC 3647 policy can then be loaded into the *PKI Policy Repository* and used like any other document.

*Source Text Analysis* The *SourceTextAnalyzer* leaves the retrieved passage unchanged. This operation is useful for arbitrarily aggregating sections of the

[6] The FPKIPA recommends *all* members use 3647 format for *all* cross-certified CPs [2].

policy. This analyzer may be used to assist auditors. For example, WebTrust auditors must evaluate "the procedures to add new users, modify the access level of existing users, and remove users who no longer need access" [34]. Extracting the content from CP sections[7] pertaining to issuing new certificates (RFC 3647, Section 3.1), revocation requests (3.4), certificate issuance (4.3), and certificate modification (4.8) would quickly provide auditors with relevant policy information. Other reports could give an overview of PKI processes, or aggregate information on user enrollment [28].

## 4 Evaluation

This section demonstrates that our tools, in particular the *PolicyReporter*, actually address current limitations of compliance audits, IGTF accreditation, and policy mapping. Empirical results are combined with anecdotal evidence to argue improvements in the *efficiency* and *consistency* of certificate policy operations in actual practice.

Our experimental evaluations compare the duration of two common certificate policy operations when performed manually and when using our *PolicyReporter*. The first experiment measures the time needed to aggregate information used to compare two CPs while the second measures the time needed to map a policy from 2527 format into 3647 format. Our hypothesis is that the automated processes take less time and provide better-quality information than the non-automated processes.

**Experiment 1: Aggregating Information for Policy Comparison** The first experiment assumes that the policies being compared are readily available on disk in PDF format for the manual case and are sitting in the *PKI Policy Repository* as TEI-XML in the automated case. Since comparing policies is subjective and varies widely across CAs, we decided to measure the time required to aggregate the information needed to perform a comparison. Additionally, we used a highly-experienced certificate authority operator so that we could compare our approach to the fastest manual times possible. In the automated case, we timed the steps necessary to generate a report using the *SourceTextAnalysis* and *RFC2119Analysis* and to view each of the sections in that report. In the manual case, we timed the steps needed to load the two PDFs, position them, and view each of the sections in that report side by side. We also timed how long it took to manually count the words appearing in RFC 2119. Since the steps done to perform these tasks may vary from person to person, we explicitly defined each of the steps to be followed.

We performed ten time trials to control for variables which could affect the time necessary to collect the information specified by the evaluation *criteria*[8]. More *policy units (p-units)*, defined as passages of depth 3 within the policy's citation tree (such as Section 3.1.4), within the *criteria* will require more information to be collected and potentially require more time. The length of the

[7] We are assuming the CP is structured according to RFC 3647 in this example.

[8] In this context, *criteria* consist of the policy sections to be compared and the algorithms to run on them.

| Trial Information | | | Manual | | | *Policy-Reporter* | **View Difference** | **Total Difference** |
|---|---|---|---|---|---|---|---|---|
| Id | Sections | #P-units | View | Count | Total | | | |
| 1 | 3.1.2 | 1 | 01:03 | 01:41 | 02:44 | 00:38 | **00:25 (40%)** | **02:06 (76%)** |
| 2 | 6.1.1 | 1 | 00:46 | 00:59 | 01:45 | 00:44 | **00:02 (04%)** | **01:01 (58%)** |
| 3 | 5.4.5 | 1 | 01:02 | 00:31 | 01:33 | 00:39 | **00:23 (37%)** | **00:54 (58%)** |
| 4 | 7.1.2, 7.13 | 2 | 00:56 | 01:28 | 02:24 | 00:50 | **00:06 (11%)** | **01:34 (65%)** |
| 5 | 5.3.1, 5.1.2 | 2 | 01:18 | 03:51 | 05:09 | 00:51 | **00:27 (35%)** | **04:18 (83%)** |
| 6 | 4.5, 6.1.5 | 3 | 01:22 | 04:32 | 05:54 | 00:50 | **00:32 (39%)** | **05:04 (86%)** |
| 7 | 6.7, 6.2.6, 5.7.1 | 3 | 01:44 | 04:04 | 05:48 | 01:01 | **00:43 (41%)** | **04:47 (82%)** |
| 8 | 2.1, 1.5.4, 4.4.1, 9.1.4 | 4 | 02:18 | 03:28 | 05:46 | 01:01 | **01:17 (56%)** | **04:45 (82%)** |
| 9 | 8 | 5 | 01:15 | 03:28 | 04:43 | 00:46 | **00:29 (39%)** | **03:57 (84%)** |
| 10 | 5.3, 1.3.4, 3.1.4 | 10 | 01:56 | 14:52 | 16:48 | 01:01 | **00:55 (47%)** | **15:47 (94%)** |
| **Total** | | **32** | **13:40** | **38:54** | **52:34** | **08:21** | **05:19 (39%)** | **44:13 (84%)** |

**Table 1.** Timing results of experiment 1

| Trial | 4 | 7 | 8 | 9 | 10 | 10 |
|---|---|---|---|---|---|---|
| 2119 Category | SHALL | MAY | MUST | MUST | MUST | SHALL |
| Manual | 0 | 0 | 2 | 9 | 30 | 9 |
| *PolicyReporter* | 1 | 1 | 0 | 10 | 22 | 10 |

**Table 2.** Discrepancies in manual and automated RFC 2119 keyword counts

*policy unit* passages will affect how long it takes to analyze the text. The proximity of passages within a policy will affect how much a person or machine must navigate through the text. Finally, since the time it takes to manually gather this information decreases with experience, we had a very experienced subject perform the manual and automated tasks. Tables 1 and 2 display the times for each trial and the accuracy of the RFC 2119 key word counts respectively.

The timing and counting results reveal a great deal about the efficiency of manual versus automated certificate policy operations. An initial glance at trials 8, 9, and 10 in the timing data reveals that the speed of consolidating information for policy reviews depends less on the number of *unit policies* to compare and more upon the proximity of those passages to one another in the text. We suspect that the length of the policy unit passages will affect the time required to actually compare the passages, though that is out of the scope of this experiment. Additionally notice that the times to view each passage manually are significantly less than those required to view and count the RFC 2119 words. This indicates that machine-actionable policy comparison makes it possible to gather useful information, such as the significance of a policy requirement, for making policy decisions that are intractible using current methods. Overall the automated policy comparison saved our CP analyst 5 minutes (39%) of his time when he was just viewing passages, and 44 minutes (84%) of his time when working more closely with the material by counting keywords.

Efficiency is also a function of accuracy. In this experiment we measured the accuracy of manual versus automated comparison of policy using in terms of the RFC 2119 Analysis. Table 2 shows that manual and automatic methods disagreed with each other in 50% of the trials. In trials 4 and 7, our CP analyst missed 1 occurrence of a keyword. In trial 8, our CP analyst counted *requires* as belonging to the *required* category while our algorithm did not. This example highlights how encoding such textual analyses resolves subtle differences in evaluation that occur in manual policy comparison. Finally in trial 10, although our

subject missed a MUST occurrence in section 5.3 of the ULAGrid policy, our analyzer missed a number of occurrences as Section 3.1.4 of the TACC policy refers to Sections 3.1.1 and 3.1.2. Future versions of *PolicyReporter* will have to resolve such references.

Overall however these results indicate that automated PKI operations are more efficient. Additional analyses such as the RFC 2119 Analysis that are manually costly but informative take 84% less time and provide better quality information when automated, allowing these operations to occur more frequently in a manner that can be reproduced. Since the *PolicyReporter* generates HTML reports, the information upon which policy decisions are based can be saved, leading to more transparency. We have experimentally demonstrated that the *PolicyReporter* does make CP comparison more efficient and *consistent*. We can only posit how much more efficient automated policy becomes when comparing dozens of security provisions rather than no more than ten at a time. Furthermore, the *PolicyReporter* makes it possible to easily compare more than two policies at a time whereas current manual operations make such comparisons impractical.

**Experiment 2: Timing Automated Policy Mapping** The second experiment makes the same assumptions as the first; PDF policies are available on disk for the manual case and TEI-XML policies are available in the *PKI Policy Repository* in the automated case. The design of the second experiment was much simpler as we simply timed how long it took to automatically map the set of *policy units* in Section 1 of an RFC 2527 policy into the RFC 3647 citation scheme. We used the mapping defined in RFC 3647 and in three time trials the *PolicyReporter* enabled us to complete the mapping in 50, 39, and 35 seconds respectively.

These results highlight the speed of machine-actionable certificate policy. In under *one minute*, *policy units* from *one* section of a certificate policy were automatically mapped. It is estimated that mapping 2527 to 3647 requires 20% more effort than a direct mapping between 3647 CPs. Considering that the average mapping takes 80-120 hours, although the comparison is not exact, we claim that our results indicate a significant time savings in policy mapping.

In preparation for the experiment, automation of the mapping process immediately revealed an error in RFC 3647's mapping framework: RFC 3647 maps 2527, section 2.1 to 3647, section 2.6.4. A closer look at RFC 3647, Section 6 revealed that section 2.6.4 does not exist in the outline of provisions! Automatic mapping allows one to easily change a mapping and rerun the process as frequently as desired. Our approach also increases the transparency of the mapping process because generated RFC 3647 policies contain references to the source RFC 2527 provisions from which they are mapped. Finally, automatic policy mapping is easily reproduced; generated policies can be compared to other policies by loading them into the *PKI Policy Repository*. It takes roughly 1 minute to load a policy into the repository depending upon the size of the document.

## 5 Related Work

Chadwick developed various XML-based *Role-Based Access Control (RBAC)* authorization policies so that domain administrators and users can manage their own resources [9] [10]. SAML [6] and XACML [24] formalize authentication and authorization policies in XML.

Previous work in certificate policy formalization focuses less on human-readable, machine-actionable representation. Blaze [3], Mendes [23], and Grimm [16] all use ASN.1 to model properies inferred from the policy's source text. Others like Casola [8] [7], have developed data-centric XML representations, suitable for machines but not for readily understanding policy semantics [16]. Recent work by Jensen [20] encodes the reference scheme of a certificate policy using DocBook [35]. His work closely relates to our work in policy formalization and is compatible with our approach.

We reference and compute upon the citation scheme of certificate policies to drive tools that we have empirically verified to increase the *efficiency* and *consistency* of certificate policy operations. Our work builds upon established standards and mature technologies. TEI P5 [5] represents 15 years of research in encoding texts with XML. The CTS Protocol [30] has been in development for 5 years and is based upon over 20 years of experience [13] in computing with a variety of digitized texts.[9]

## 6 Future Work

In future work, we plan to package our tools for release in an open source distribution hosted at OpenCA Research Labs [25]. Most urgently we need to extend the automated policy mapping to include all security provisions, not just *policy units* in Section 1. Our first experiment also revealed the need to extend the tools to resolve references to other sections of text that occur within a policy statement.

In general this work sets the stage for two complimentary problems: *extracting* useful information from natural language policies and *designing* usable certificate policies for man and machine. For the former problem, we intend to design, build, and test additional algorithms for processing policy text. Just as we filtered for RFC 2119 key words, so could we filter phrases indicating the size of the CA key in Section 6.1.5 or extract SHA-2 hash requirements. We also may investigate more sophisticated *Natural Language Processing (NLP)* algorithms. Algorithmic analyses of a policy's source text may lay the foundation for usable security policy metrics with real-world utility.

Computing on certificate policies informs the design of security policies that can be processed by human and computer. Certificate policies encoded in XML formats like TEI and DocBook are currently compatible with the *PKI Policy Repository*. However, other texts such as configuration files or code intended to

[9] We used this experience in designing the CTS Protocol, requiring compatibility with texts encoded in TEI, DocBook, or any other valid XML format encoding a citation scheme.

be consistent with a policy statement might be aligned through the common citation framework that CTS-URNs provide. Given that high-level policy statements can be mapped into software, we also plan to investigate how they might be explictly mapped into hardware. Finally, some of Chadwick and Sasse's research on controlled languages might prove helpful in making policies useable for humans and computers [10].

## 7 Conclusions

To conclude, our *PKI Policy Repository*, *PolicyBuilder*, and *PolicyReporter* make real-world CP operations more *efficient* and *consistent*. We have empirically demonstrated their utility in aggregating information for policy comparison and policy mapping, two common tasks performed in compliance audits, grid accreditation, and bridging PKIs. Our tools streamline these processes, making them more efficient and provide auditors with more, higher quality information. Our tools allow people to to specify a set of provisions that mimic how they actually make trust decisions. Instead of forcing people to accept or reject a policy in its entirety, CTS-URNs encoded as OIDs allow one to blacklist or whitelist arbitrary policy provisions. While we hope that our tools will reduce the costs associated with creating and maintaining a PKI, more importantly we hope to empower individuals to make their own trust decisions through a usable policy framework.

## References


1. 4BF - Four Bridges Forum. Retrieved May 29, 2009 from `http://www.the4bf.com/`.
2. Peter Alterman. Reformatting Entity CP's into RFC 3647 Format, November 2006. Retrieved May 30, 2009 from `http://www.cio.gov/fpkipa/documents/PolicyMemoRFC3647v1.pdf`.
3. M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *IEEE Symposium on Security and Privacy*, pages 164–173, 1996.
4. S. Bradner. RFC 2119: Key words for use in RFCs to Indicate Requirement Levels, March 1997.
5. Lou Burnard and Syd Bauman. *TEI P5: Guidelines for Electronic Text Encoding and Interchange*. 5th edition, 2007.
6. Scott Cantor, John Kemp, Rob Philpott, and Eve Maler. *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*. 2005.
7. V. Casola, A. Mazzeo, N. Mazzocca, and M. Rak. An Innovative Policy-Based Cross Certification Methodology for Public Key Infrastructures. In *EuroPKI*, 2005.
8. V. Casola, A. Mazzeo, N. Mazzocca, and V. Vittorini. Policy Formalization to Combine Separate Systems into Larger Connected Network of Trust. In *Net-Con*, page 425, 2002.
9. David W. Chadwick and Alexander Otenko. RBAC Policies in XML for X.509 Based Privilege Management. In *SEC*, page 39, 2002.
10. David W. Chadwick and A. Sasse. The Virtuous Circle of Expressing Authorization Policies. In *Semantic Web Policy Workshop*, 2006.

11. S. Chokhani and W. Ford. RFC 2527: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework, March 1999.
12. S. Chokhani, W. Ford, R. Sabett, C. Merrill, and S. Wu. RFC 3657: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework, November 2003.
13. Gregory Crane. The Perseus Digital Library. Retrieved May 29, 2009 from `http://www.perseus.tufts.edu/hopper/`.
14. C. Dué, M. Ebbott, C. Blackwell, and D. Smith. The Homer Multitext Project, 2007. Retrieved May 29, 2009 from `http://chs.harvard.edu/chs/homer_multitext`.
15. Robert Gold. WEBTrust / client FAQ, 1997-2004. Retrieved May 29, 2009 from `http://www.webtrust.net/faq-client.shtml`.
16. R. Grimm and T. Hetschold. Security Policies in OSI-Management Experiences from the DeTeBerkom Project BMSec. *Computer Networks and ISDN Systems*, 28:499, 1996.
17. Russ Housley and Tim Polk. *Planning for PKI: Best Practices Guide for Deploying Public Key Infrastructure*. Wiley Computer Publishing, 2001.
18. International Grid Trust Federation Charter. Retrieved May 29, 2009 from `http://www.igtf.net/new-doc/IGTF-Federation-20051005-1-igtf.html/`.
19. ISO 21188: Public Key Infrastructure for Financial Services—Practices and Policy Framework, 2006.
20. J. Jensen. Presentation for the CAOPS-IGTF session at OGF25, March 2009.
21. Tomaz Klobucar and Borka Jerman Blazic. A Formalisation and Evaluation of Certificate Policies. *Computer Communications*, 22:1104, 1999.
22. Ronald Koorn, Peter van Walsem, and Mark Lundin. Auditing and Certification of a Public Key Infrastructure. *Information Systems Control Journal*, 5, 2002.
23. S. Mendes and C. Huitema. A New Approach to the X.509 Framework: Allowing a Global Authentication Infrastructure without a Global Trust Model. In *Network and Distributed System Security*, pages 172–189, 1995.
24. T. Moses. *eXtensible Access Control Markup Language (XACML) Version 2.0*. 2005.
25. OpenCA Research Labs. Retrieved May 29, 2009 from `http://www.openca.org/`.
26. M. Pala, S. Cholia, S. Rea, and S. Smith. Extending PKI Interoperability in Computational Grids. In *IEEE International Symposium on Cluster Computing and the Grid*, pages 645–650, 2008.
27. G. Powell. *Beginning XML Databases*, page 260. Wiley Publishing, 2007.
28. Klaus Schmeh. A Critical View on RFC 3647. In *EuroPKI*, page 369, 2007.
29. D. Neel Smith. CTS-URNs: Overview, December 2008. Retrieved May 29, 2009 from `http://chs75.harvard.edu/projects/diginc/techpub/cts-urn-overview`.
30. D. Neel Smith and G. Weaver. Canonical Text Services (CTS). Retrieved May 29, 2009 from `http://cts3.sourceforge.net/`.
31. D. Neel Smith and G. Weaver. Applying Domain Knowledge from Structured Citation Formats to Text and Data Mining: Examples Using the CITE Architecture. In *Text Mining Services*, page 129, 2009.
32. Y. Tanaka, M. Viljoen, and S. Rea. Guidelines for Auditing Grid CAs version 1.0, February 2009. Retrieved May 30, 2009 from `http://www.ggf.org/Public_Comment_Docs/Documents/2009-02/AuditGuideline%s-Feb26_2009.pdf`.
33. D. Trcek, B. Jerman-Blazic, and N. Pavesic. Security Policy Space Definition and Structuring. *Computer Standards & Interfaces*, 18(2):191 – 195, 1996.

34. Trust Services Principles, Criteria and Illustrations for Security, Availability, Processing Integrity, Confidentiality, and Privacy, 2006. Retrieved May 29, 2009 from `http://infotech.aicpa.org/NR/rdonlyres/05A9970C-A574-406D-BE82-5BE60D17%F90F/0/Trust_Services_PC_10_2006.pdf`.
35. Norman Walsh and Leonard Muellner. *DocBook: The Definitive Guide*. Jul 1999.