A TRUST-RELATIONSHIP MANAGEMENT FRAMEWORK FOR FEDERATED VIRTUAL ORGANIZATIONS

by

JILL GEMMILL

PURUSHOTHAM V. BANGALORE, COMMITTEE CHAIR
GARY J. GRIMES
JOHN K. JOHNSTONE
HELMUTH F. ORTHNER
ANTHONY SKJELLUM

A DISSERTATION

Submitted to the graduate faculty of The University of Alabama at Birmingham,
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

BIRMINGHAM, ALABAMA

2006

A TRUST-RELATIONSHIP MANAGEMENT FRAMEWORK
FOR FEDERATED VIRTUAL ORGANIZATIONS

JILL GEMMILL

ABSTRACT

Research collaborations today are virtual organizations (VO's), formed dynamically and crossing institutional and administrative boundaries. Electronic communication is essential for VO members who work at a distance and in different time zones. VO shared resources are owned by and attached to networks managed by enterprise administrative groups who are typically unaware of the VO's existence or requirements. Assembling a computing environment that provides the required tools and data in a manner that is both accessible and secure is a central challenge in distributed computing. This dissertation addresses that challenge through a novel combination of existing enterprise infrastructure and emerging middleware components that results in a new scalable, VO-centric trust model.

Middleware provides a layer of services between applications and underlying systems' administrative domains, user identity methods, and use policies. Middleware-enabled applications should make it possible for enterprises to manage resources they own while collaborators share communications, resources and data unaware of the organizational boundaries below. In this dissertation recent grid security and federated identity middleware solutions are combined to accomplish a consistent security context in a distributed environment. A novel aspect of that environment is the sharing of identity and attributes across systems and applications without use of a central repository or portal.

A workable framework is developed and used to demonstrate these capabilities in

the form of a prototype VO Service Center that provides VO's of any size the ability to manage their own memberships and role assignments and to control access to their own information while simultaneously adhering to multiple enterprise policies. Identity management occurs through federated identity and as a result each VO member can access any VO resource using their enterprise authentication system. The VO Service Center creates a new middleware service that is needed in addition to Identity Provider and Service Provider. The VO Service Center is useful for web-based applications as well as for grid computing environments, providing both types of applications with attributes that are consistent in format and semantics and a known source for otherwise widely distributed attributes needed for secure access control.

DEDICATION

Words can not adequately express the appreciation due to my family for the love, support and humor provided throughout my lifetime. Thank you.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation by grant ANI-0330543 "NMI Enabled Open Source Collaboration Tools for Virtual Organizations" (Gemmill, PI). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation. Additional partial funding was provided by the National Library of Medicine, National Institutes of Health, under Contract No. N01-LM-3-3513.

Many colleagues contributed to the work and results described in this dissertation, including John-Paul Robinson, Jason L. W. Lynn and graduate students Prahalad Achutharao and YiYi Chen. These colleagues were funded via ANI-0330543 and worked under my supervision in my role as Principal Investigator. Special recognition is due to co-PI John-Paul Robinson who contributed many original ideas to the initial proposal as well as to the decision process and architecture as work progressed.

Dissertation advisor Puri Bangalore provided excellent manuscript critiques and other guidance throughout the dissertation process. I appreciate his guidance as well as direction provided by additional dissertation committee members Tony Skjellum, Gary Grimes, John Johnstone, and Helmuth Orthner. My thanks to the entire UAB Computer Science faculty for my early training in this field and for the years of stimulating dialog that have followed, with special thanks to Warren Jones and Ken Sloan for that Silvertron

pep talk.

Colleagues from Internet2, Comité Réseau des Universités, and the National Center for Supercomputing Applications are recognized for their impact on this dissertation through conversations, working groups, and feedback on numerous occasions during the past three years. Members of the Internet2 MACE-MLIST Working Group contributed to the description of the mailing list's interface with middleware. Special thanks to Steve Olshansky, flywheel extraordinaire. Shibboleth developers Ken Klingenstein, Scott Cantor, and Bob Morgan stimulated some initial thoughts that led to this dissertation; Sympa developers Serge Aumont and Oliviér Salaun were tremendously cooperative in explaining their code's inner workings and in collaborating on modifications to support Shibboleth; and GridShib developers Von Welch, Tom Barton and Tom Scavo are thanked for their interest and collaborative work on myVocs integration with GridShib.

Finally, appreciation is expressed to the UAB Information Technology Division and to the Director of the Department of IT Academic Computing for their support of this research activity.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF ABBREVIATIONS

AAA	authentication, authorization, accounting
CA	certificate authority
DN	distinguished name
ePPN	eduPersonPrincipalName
GSI	grid security infrastructure
IdP	identity provider
LDAP	lightweight directory access protocol
MLM	mailing list management system
NETID	network identity
P2P	peer-to-peer
PKI	public key infrastructure
SAML	security assertion markup language
SOA	start of authority
SP	service provider
SSO	single sign on service
VO	virtual organization
XML	extensible markup language

MOTIVATION AND BACKGROUND

Assembling a distributed computing environment that provides seamless, shared access to distributed tools while also providing appropriate access controls remains an important challenge in distributed computing. The recent interest in grid computing (Foster & Kesselman, 1997; Foster, Kesselman, Nick, & Tuecke, 2002; Foster, 2005) is because of its success as a partial solution to this problem. The first grid implementations were designed to meet the requirements of large and scientifically prominent distributed teams of collaborators. This dissertation began from an observation that the "big-science" scenario did not fit the requirements of a more typical university collaborative team that is smaller and less well funded.

New Computing and Work Paradigms Require Revised AAA

Authentication, authorization, and accounts (AAA) are at the core of computing security and resource management. Numerous AAA implementations have been developed to address a distributed computing environment, but designs to date have used an administrator-centric model. In a distributed computing environment, if "the network is the computer"[1], then where is the root authority, where are identity and attributes stored, and where does session information reside? Single root models were adequate for the computing environment that existed prior to commercialization of the Internet, when the

[1] Sun Microsystems corporate slogan

number of people using computers was relatively small and each computer user needed to access only a few computers. A dramatic change in computer usage began with the introduction in 1993 of the Mosaic graphical web browser (Lemelson-MIT Program, 2001), a technology built upon Tim Berners-Lee's "WorldWideWeb" text browser program (Berners-Lee & Groff, 1992). The Mosaic browser made it possible for everyone to easily access data and images stored on any computer. Perhaps most significantly, Mosaic enabled hundreds of thousands of people to begin creating and sharing their own content. Initially web content consisted of public information, creating a growing community of computer users based on the new model of accessing content wherever it happened to be located.

Commercialization of the Internet in the mid-1990's was accompanied by the explosion of e-business and its more demanding security requirements; financial transactions needed to be private and secure and certain content began to be offered by paid subscription. These developments led to encrypted network connections, new search technologies, and hundreds of millions of people using the Internet to access information culled from millions of systems. In less than one decade, the typical computer usage scenario changed from that of one user with a handful of login accounts to complex, many-to-many, consumer-producer relationships. Unfortunately, identity theft and unauthorized access to information have accompanied this change in the Internet, requiring great care in protecting information while making it accessible to authorized users.

AAA has had a difficult time keeping pace with these rapid developments. As noted in the recent "Cyber Security: A Crisis of Prioritization" report by the President's Information Technology Advisory Committee (PITAC) (President's Information Tech-

nology Advisory Committee, 2005) most cybersecurity research to date has assumed a perimeter defense model, as if there were a clear distinction between the "inside" and the "outside" of the Internet. In summary, existing AAA models no longer reflect real world requirements.

What Are Virtual Organizations?

The Internet has also affected the way research is done. New approaches to scientific investigation and medical care are now feasible because of the possibility of easy access to data anywhere and aggregation of large data sets explored using data mining techniques. Scientific inquiry that used to be conducted by a single investigator is being replaced with small teams of research collaborators from different disciplines who collectively have the expertise needed to attack a specific problem. These teams are formed dynamically and cross many administrative and institutional boundaries. Collaborations having these characteristics are called Virtual Organizations (VO's) (Foster, Kesselman, & Tuecke, 2001). An excellent VO definition appeared in the introductory article by the editors of the "Special Issue on Virtual Organizations" of the *Journal of Computer-Mediated Communication* (DeSanctis, 1998) and it is worth quoting at length:

> A virtual organization is a collection of geographically distributed, functionally and/or culturally diverse entities that are linked by electronic forms of communication and rely on lateral, dynamic relationships for coordination. … The result is a "company without walls" that acts as a "collaborative network of people" working together, regardless of location or who "owns" them. … In some cases, the entities composing the organization may participate in several virtual organizations simultaneously … [and] will appear less a discrete enterprise and more an ever-varying cluster of common activities in the midst of a vast fabric of relationships.

The VO is a participant-centric organization and there should be no surprise that the in-

frastructure provided by an administrator-centric AAA architecture is not a good match for the AAA required by VO's.

Existing AAA solutions are out of sync with everyday use of the Internet by millions and with the new approaches to scientific discovery. In the old computing model, institutions were concerned only with information inside the enterprise and that information was shared only with other members of the same institution: an intra-institutional model. The new computing model consists of frequent, cross-institutional collaborations where information is shared across these administrative boundaries: an inter-institutional model.

Virtual Organizations Are Important in Research and Health Care

The U.S. Department of Health and Human Services National Institutes of Health (NIH) is the source of federal funding for medical and clinical research in the U.S. and has a direct influence on research directions in these areas. NIH has recently published the "NIH Roadmap" (Zerhouni, 2004), a document calling for new multidisciplinary approaches to analyzing large and complex data sets and for new directions in clinical medicine, stating "the most remarkable feature of this twenty-first century medicine is that we hold it together with nineteenth-century paperwork" (Thompson, 2004). As an example in the area of clinical research, the Roadmap states "behavioral scientists, molecular biologists, and mathematicians might combine their research tools, approaches, and technologies to more powerfully solve the puzzles of complex health problems such as pain and obesity." The scientific problems described in the Roadmap are so large that even one institute at NIH can not adequately undertake the investigation single-handedly;

therefore, a large number of investigators in multiple locations must undertake some aspect of the problem while sharing their insights and results with everyone else working on the problem.

The President's Information Technology Advisory Committee (PITAC) is a Federal Advisory Committee chartered by Congress and is responsible for reviewing federal programs mainly funded by the National Science Foundation (NSF) in networking and IT research and development. The PITAC report "Revolutionizing Health Care Through Information Technology" (President's Information Technology Advisory Committee, 2004) found health care to be an area in critical need of an information management overhaul, especially one that would make it possible to share a single health record appropriately among patient, physicians, clinical researchers, and insurers.

The Virtual Organization Infrastructure Problem

A VO consists of people and resources spread across administrative domains and institutions. Distributed collaborations require infrastructure that can support both intra- and inter-institutional information access. The architectural challenge is to provide VO members with an experience that provides easy access to all resources they need while enforcing policy appropriately, and to do so without requiring the manual intervention of an expert systems programmer. Figure 1 illustrates the desired VO environment as experienced by its members. The shared collaboration space appears easily accessible to each VO member and provides all the data, tools, and other resources needed to work as a team. Setting up their shared tools and establishing appropriate access control was easy and they are confident that the necessary security is in place.

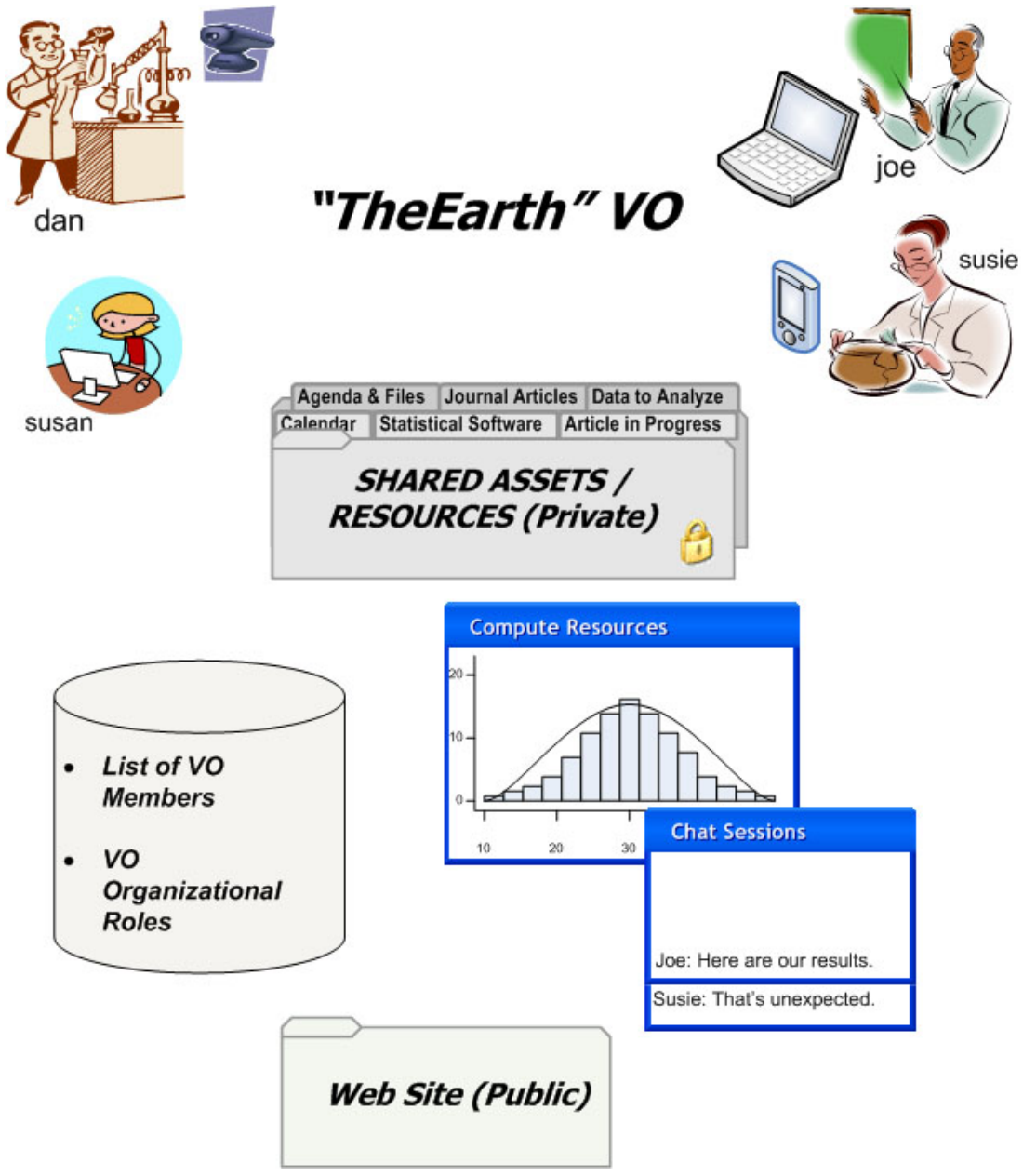

Figure 1. Illustration of "TheEarth" Virtual Organization. Resources function as if they were dedicated for use by this group and are easy to access.

This scenario is unfortunately difficult to implement today, requiring that special login accounts are created by an administrator who may request a policy review if the account is for someone not affiliated with the institution whose resources will be accessed. Internet social network software such as Yahoo Groups and Google Groups provide user self-management for login and group creation, but require that all information and services being shared reside at the Yahoo or Google site. Their approach does not solve the problem of sharing data and resources owned by an enterprise that has responsibility for maintaining and managing access to those resources.

A typical solution provided today is to create a special login account and password in each tool for each participant. Applications today are frequently accessed using web browsers, but data transferred via the Hypertext Transfer Protocol (HTTP) (Fielding, Gettys, Mogul, & et al, 1999) is stateless and has no associated information regarding who owns or has permission to read or modify the file. An account local to the web server or other password mechanism is one way to associate an identity with a particular HTTP request. While the per-application or per-service login approach works, it is not scalable. Since people participate in numerous VO's they soon find themselves owning numerous, unrelated login/password pairs which can lead to confusion about which pair to use at some specific resource. A natural human reaction is to reuse the same login/password combination for every site, a solution which may be convenient but is also insecure since the password is stored in many unrelated security domains; too many people with unknown trustworthiness have access to that password. Another scalability issue is the added burden for system administrators who must maintain these lists, provision accounts and storage resources, reset forgotten passwords, and terminate access in a

timely manner.

The complex set of identity management, authentication, and access control systems that can be required to support cross-domain access for VO's is illustrated in Figure 2. VO assets are distributed among University A, University B, a Federal laboratory, Antioch College, and Corporation C. VO data sources (dashed black lines) are actually a distributed file space; compute resources (dashed blue lines) are administered in several unrelated locations; and separately administered licensing, usage or access policies may exist. The institutions shown in the figure each have their own identity management and authentication services; the authentication systems may be incompatible and quite often at least one collaborator is located at an "infrastructure poor" institution. Specific challenges to be addressed include (a) absence of a common AAA root; (b) distributed and unrelated identity providers; (c) distributed and unrelated usage policies; and (d) institutional licensing and policy issues that must be enforced when accessed using the institution's AAA infrastructure.

Introduction to Middleware

Middleware has been described as a set of services available to the application layer where the services cross operating systems boundaries (Tanenbaum & van Steen, 2002). More recently, middleware has been categorized as a layered stack of services, including host infrastructure middleware, distribution middleware, common middleware services, and domain specific middleware layers (Schantz & Schmidt, 2005). Schantz and Schmidt further describe the host infrastructure middleware layer as closest to the underlying operating systems and their respective communications protocols. Examples of this

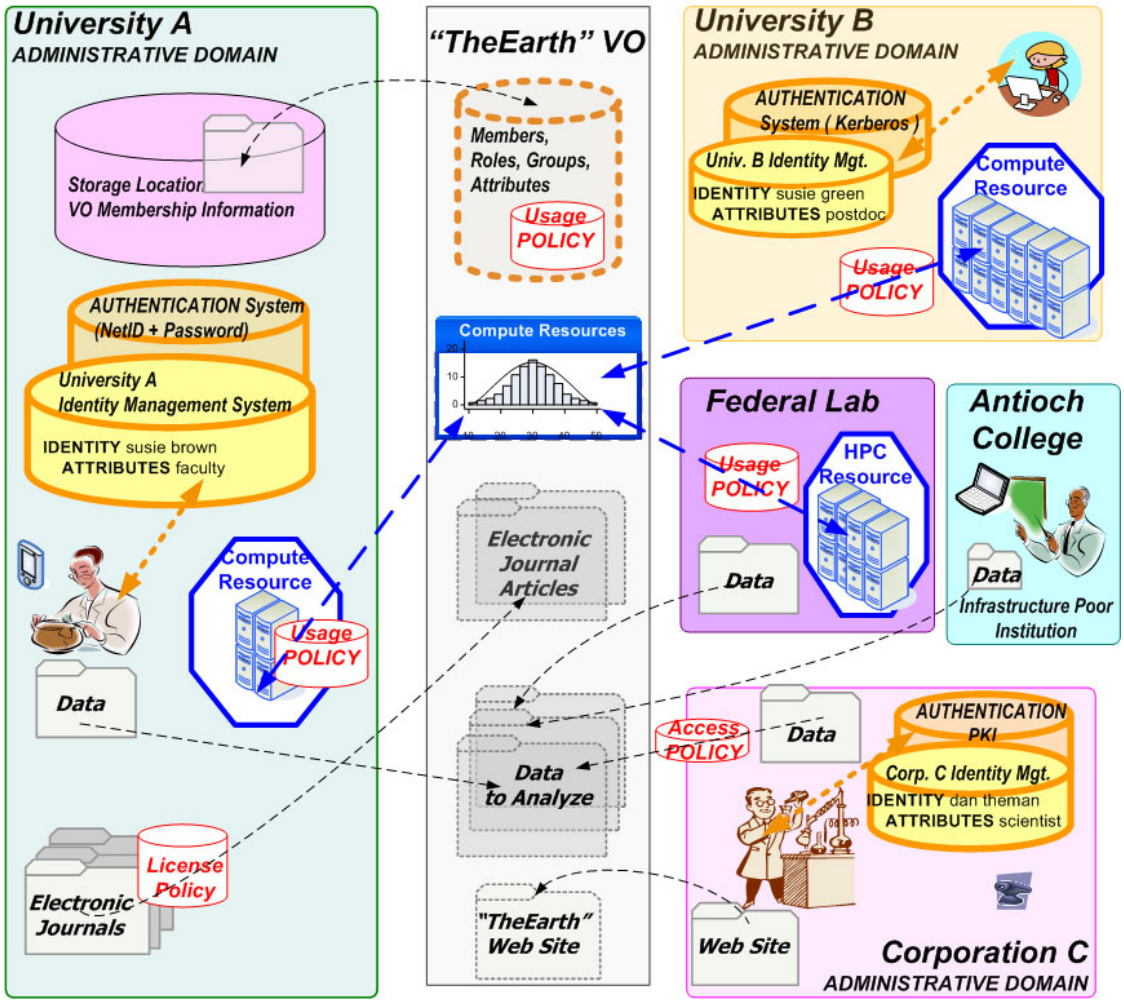

Figure 2. The simple VO members' view (center of figure) is actually a complex arrangement of distributed information, resources, software systems, and policies

middleware layer include the Sun Java Virtual Machine (Lindholm, 1997) and Microsoft .NET (Thai & Lam, 2001). Distribution middleware consists of reusable application program interfaces (APIs) that allow client applications to be written for a generic rather than specific target service, and examples in this category include the Common Object Request Broker Architecture (CORBA) (Object Management Group, 2000), Sun's Java Remote Method Invocation (RMI) (Wollrath, Riggs, & Waldo, 1996), and the simple object access protocol (SOAP) (World Wide Web Consortium, 2000). The common middleware services layer defines domain independent services that allow programmers to concentrate on business logic rather than "plumbing" issues, and examples of this layer include Sun's Enterprise JavaBeans (Thomas, 1998) and Microsoft's .NET Web Services. The final middleware layer is designated domain-specific middleware; it is highly domain specific and is exemplified by the Health Level 7 (HL7) (American National Standards Institute, 2005) standard for storage and exchange of clinical and administrative data.

Security functions are notably absent from these descriptions because security functions can be implemented in any of the layers from network to application, and there is no consensus yet on which layer is best for which function. In fact, multiple layers may simultaneously be implementing secure connections because by design, protocol layers are unaware of functions occurring at other layers. Additional challenges in implementing secure systems include management of cryptographic keys and user account addition/removal. Design of a well-defined, flexible security service that is available to all communication layers is quite challenging and current implementations are quite immature. This dissertation applies emerging security middleware in the form of heterogeneous

authentication systems; distributed authorization mechanisms; and automated account provisioning.

Identity Management and Authentication

Authentication is the process of proving ownership of a unique digital identity. Ownership of a specific digital identity is proven using some previously issued credential, and uniqueness is guaranteed only over the domain associated with that authentication system. Unique naming is important to applications so that the appropriate environment, resources, and access control can be assigned for this particular user; the reliability of the authentication system is important for deciding how much trust can be placed in the digital identity presented.

The legal basis for electronic signatures under United States federal law was established by the Electronic Signatures in Global and National Commerce Act passed in 2000 and also known as ESIGN (U.S.Government, 2000). ESIGN ensures the validity and legality of contracts that are digitally signed and also protects certain consumer rights. ESIGN is technology neutral, resulting in a broad definition of "electronic signature" that includes use of a certain date or name of a pet as a unique identifier; use of a PIN or password to authenticate banking transactions or merchandise purchases; and text appearing at the end of an e-mail message or a digital image can qualify as electronic signatures under ESIGN. Because of ESIGN's legal implications many enterprises using these weak types of digital security began to see the importance of improving their identity management systems.

An enterprise-wide identity management system can be used to build a single au-

thentication service; this approach requires that all enterprise applications use the one central authentication service and is called single-sign-on (SSO). In the SSO scenario, independent systems within the enterprise all trust the central authentication service to add and remove users in a timely manner, and to provide some authentication credentials. At UAB, for example, the "BlazerID"[2] authentication service can be used for access to Oracle applications, the campus wireless network, campus email service, and more. These systems are hosted on different operating system platforms and users must logon to each system separately, but each system has been configured to trust the one campus authentication service. SSO can be further leveraged so that a single authentication instance is shared across many enterprise applications using a single logon, and this type of service is called Web Initial Sign On (WebISO). The NMI component Pubcookie (University of Washington, 2005) is an example implementation of this type of service. Authoritative and accessible directories are a key component in an identity management system. Most universities and large corporations are now building such central directories that aggregate information about who is associated with the institution, their institutional affiliations and roles. The aggregated information can be used as the core of a central Identity Management System. An Identity Provider (IdP) practices consistent procedures for adding and removing persons from its identity store, assigns the enterprise identifier, and issues/revokes the security credential used for authentication.

A goal of this dissertation is to leverage enterprise authentication for use in locations outside the enterprise. Leveraging services already provided by existing IdPs is more efficient and cost-effective than attempting to replicate this infrastructure on a per-

[2] UAB calls its SSO identifier a "BlazerID", named after the university mascot and sports teams.

application basis. However, leveraging enterprise managed identity and credentials is challenging because the credentials are technically and politically incompatible. Authentication credentials used include username/password, Kerberos password; digital certificate; or biometric device. The absence of a global, uniquely identifiable name space is another challenge for distributed authentication. With a global naming scheme and a method for global recognition of identities established by authentication at the home enterprise it would be possible to extend WebISO beyond the enterprise and into the Internet.

Authorization: Applying Access Control Rules

Authorization is the process of determining "who" is allowed access to "what". "Who" refers to the attributes associated with a process, including a user identifier, group membership, or role definition. A user's identity should be considered an attribute, one of many possible attributes associated with that user. This perspective is important so that the act of authenticating successfully can be considered separately from the act of gathering attributes about the requestor. "What" refers to the action requested, typically read, write, or execute, as well as to the specific object that is the target of the action. "Allow" refers to the process of locating the policy associated with the requested object and the requested action and examining available attributes to determine if the policy requirements have been met, resulting in a decision as to whether the action should proceed or fail. The "allow" process is generally referred to as access control.

An interesting problem for VO's concerns how to combine important attributes that may be defined by unrelated enterprises. For the highest degree of data confidence,

each attribute ideally should be created and managed by an authoritative source. A human resources department is authoritative for the attribute `employee`, for example; a university registrar is authoritative for the attribute `student` and an individual is authoritative for their preferred email address. Within a single enterprise, it can be beneficial to gather all these attributes from their authoritative sources and centralize them in a directory so that the information is accessible to enterprise applications for use in authorization decisions. There are also instances where it might be useful to combine attributes from two unrelated enterprises. It might be useful, for example, to combine the UAB attribute `faculty` with the IEEE `member` attribute in order to determine all UAB faculty who are also members of IEEE. A second goal of this dissertation is to identify attribute management that is best suited for VO operations.

Accounting and System Specific Account Provisioning

Accounting is the process of provisioning system-specific resources to authorized users. For example, an enrolled student may be authorized to use the university's email service, but a "mailbox" (mail account and at least temporary use of disk space) and a "mailbox name" (email address) must be provisioned in order for the SMTP application to be able to deliver email. Provisioning is often referred to as "setting up an account," meaning allocating some uniquely named resources such as disk space and the user's unique identifier is often selected for use as the account identifier. It is not necessary for user identifier and account name to be identical, however. This practice can cause confusion when considering the abstract concept "identifier", a user name, as distinct from the concept "account" a name referring to a specific set of provisioned resources. Accounting

also refers to the process of tracking and reporting on the utilization of the assigned resources; in order to summarize and report utilization by different users of the system; unique account names allow users to be distinguished from one another. A third goal of this dissertation is to enable the automated provisioning of resources for VO's in a manner that minimizes system administration and maximizes VO member independence.

Trust Relationships and Trust Management

Security is Based on Trust

Trust issues are central to distributed AAA. A trust relationship refers to one system or domain trusting information managed in a separately administered system/domain. The term 'reliable' means that a domain always provides accurate information that is released only according to some pre-determined policy. Use of a person or process's Internet identifier by unrelated administrative domains depends entirely on the relying party's degree of trust in the processes and personnel of the domain issuing the identifiers. The trusting party, called the relying party, must know how identity is established, the type of authentication in use, the management processes for following established policies, and the competence of the technical staff. Policies define the precise conditions and procedures to be followed, but without reputation there is no reason to have confidence that stated policies are practiced. Since it is not practical for each domain to directly assess all these items for every other domain, the trust established is often based upon the domain's general reputation. A person may be willing to give their credit card number to Amazon.com, a large and well-known on-line business, but probably should not be willing to give that information to some unknown "mybusiness.com." Identity thieves often abuse a

domain's reputation in order to obtain a user's credentials; this practice is called "phishing"[3] and usually involves an email that appears to have been sent by a person's bank or other trusted business asking that they "verify" their credentials by providing them again. In summary, non-technical issues are the essence of trust relationships. User education, instead of technology, is the best protection against these schemes.

Trust is Implemented Using Cryptography

Cryptography provides a mathematical implementation of trust for computer systems. Cryptography depends on the use of secret numbers to verify that a communication channel is connected to the right location and not an imposter host, that incoming communications are really from the person claiming to be the contactor, and assurance that a message can be received only by the person for whom it is intended. The techniques for accomplishing these assurances are digital signature and encryption, and both techniques require a pair of cryptographic keys.

Symmetric cryptography involves a pair of identical keys; using a certain key to encrypt a message means that only holders of a copy of that same key can decrypt those messages. The greatest challenge in symmetric cryptography is one of distributing those keys securely; how can one be sure who is receiving an electronically transmitted key, that the key has not been intercepted or copied in route, and that both sides have stored the key so that it is accessible only to authorized persons or processes?

Asymmetric cryptography (Diffie & Hellman, 1976; Rivest, Shamir, & Adleman,

[3] The term phishing is defined in Webopedia, http://www.webopedia.com/TERM/p/phishing.html accessed April 11, 2006

1978) involves the pairing of a private key with a public key. To implement a digital signature a sender uses their private key to sign messages and only the associated public key will verify that signature, thus assuring that the signature is valid. Digital signatures are also used to prove that message contents have not been modified after having been signed; this message integrity is accomplished by constructing a hash of the message and digitally signing that hash. A hash can be calculated from the arriving message at the receiving side, and if the two hashes are identical no message tampering has occurred. Cryptography is also used for message encryption. The public key of the intended recipient is located and used to encrypt a message; only the recipient's private key can decrypt the message, thus assuring that only the intended recipient can read the message.

As compared to symmetric cryptography, which requires that secret keys are somehow securely transmitted, asymmetric cryptography has the advantage of simplifying the secrecy required since private keys do not need to be transmitted at all. The challenge in asymmetric cryptography is one of reliably associating an identity with each private key. The identifier contained in a public key might claim to represent "George W. Bush, President of the United States", and this public key can be used to verify that a message was signed by whoever owns the associated private key. Unfortunately, this mathematical validation does not mean that that President George W. Bush is actually the key owner. Therefore, trust in the accuracy of an identity associated with cryptographic keys must be established some other way. As the next chapter will explain, there are several approaches that can be taken to reliably associate identity with some cryptographic key. What differentiates the various approaches is the trust model used to establish this association, and not the mathematical mechanics of the cryptography.

Trust Management

Consideration of security credentials, security policies, and trust relationships within a single framework is called "trust management," a term first defined by Blaze (Blaze, Feigenbaum, & Lacy, 1996). A trust management system combines authentication, credentials, policy, authorization and access control into a coherent system that can be used in a distributed environment. The first implementation of a trust management system was known as Keynote (Blaze, Feigenbaum, Ioannidis, & Keromytis, 1999). Keynote was notable in requiring local control of trust relationships and avoiding hierarchies of certifying authorities. Without any root authority, domains disclosed digital credentials to each other and could enforce access control based on local policy. These credential exchanges were called "assertions." Keynote also provided a language for expressing policies and credentials. Other early attempts to build trust management systems are described and compared in Bertino (Bertino, Ferrari, & Squicciarini, 2004). A fourth goal of this dissertation is to understand which of the trust models described in the next chapter are best suited for supporting inter-institutional collaboration activities. The requirements for that environment include the ability to establish trust relationships across independent IdP's that use different authentication systems.

Summary of Dissertation Contributions

The motivation for this dissertation was a desire to discover a system architecture supporting cross-domain single sign-on for Virtual Organizations. To achieve scalability, existing enterprise authentication systems need to be leveraged for use in other domains. Although enterprises can be authoritative for identity and other attributes associated with

a person's affiliation with that enterprise, it is the VO that is authoritative for VO memberships and roles. Therefore, the desired architecture must be able to combine enterprise-authoritative and VO-authoritative attributes in a manner that is acceptable to both. To minimize manual system administration and maximize VO member independence, resources needed by the VO should be provisioned automatically. Finally, the trust model selected must be able to establish trust relationships across independent identity providers who may use incompatible authentication systems.

The framework described in this dissertation contributes the following new approaches in grid computing:

1. Federated identity is used, permitting many independent root identity authorities and non-interoperable authentication methods to be used for accessing any resource. No central repository is required.
2. Attributes maintained at multiple locations and by multiple authorities are combined for use in authorization decisions.
3. Accounts are provisioned automatically for authorized users, increasing scalability by eliminating the need for a system administrator to add new users or services manually.
4. Preliminary guidelines are developed explaining the process for middleware-enabling existing applications. These guidelines can be used when converting useful open source software from self-contained software islands to pluggable collaboration environment components.
5. Identity and attributes are shared across distributed systems without requiring use of a portal or central repository to manage a security context.

6. A prototype virtual organization service center is demonstrated supporting autonomous self-management for VO's.
7. A solution for sharing identity and attributes between web applications and grid applications is presented.

RELATED WORK

Network Layer Trust Models

The design of any trust management solution is highly influenced by the problem being addressed. Internet service providers (ISP's), for example, have relied on AAA solutions that are implemented at the network layer. These protocols were designed to control who can access networks managed by the ISP's and also to collect revenue for that access. User-to-network style solutions such as Remote Authentication Dial-In User Services (RADIUS) (Rigney, Willens, Rubens, & Simpson, 2000); IP Security (IPSec) (Kent & Atkinson, 1998); and most recently, Diameter (Calhoun, Zorn, Spence, & Mitton, 2005) allow network subscribers to access the Internet via their service provider from any location. Some protocols in this category are network-to-network in nature, and are used to support business relationships between ISP's. The Common Open Policy Service Protocol (COPS) (Durham et al., 2000), for example, can be used to express and implement rules for exchange of network traffic between ISP's. The Secure Socket Layer (SSL) (Dierks & Allen, 1994), since renamed as Transport Layer Security (TLS), was designed so that web browsers could establish secure, encrypted connections to any web server. Whereas ISP's sign up subscribers in advance and have already established a payment mechanism, e-commerce requires "just in time" security so that any unknown visitor can establish a secure connection and provide payment without any prior arrangement between customer and service provider.

The network layer trust model is the foundation for e-business and certainly works well for that purpose. However, the trust model required for VO collaborations is characterized by a user-to-user or more typically user-to-group communication that is implemented at the application layer, and applications have no access to any credentials or information that may be available at the network layer because of the Internet Protocol's strict separation of communication layers. The remainder of this chapter will therefore focus on trust models that are useful at the application layer.

Direct Trust Model

One straightforward approach to managing trust is to trust only yourself. This is referred to as the Direct Trust model or peer-to-peer trust model, as exemplified the Pretty Good Privacy (PGP) protocol (Garfield, 1994). A PGP user generates an asymmetric cryptographic key pair and provides his public key by physical transfer to someone he knows, for example by manually handing him a disk containing the key. Trust exists because the two people recognize each other. A somewhat more scalable public key transfer model requires that the sender provides the public key by email, and the recipient then calls the sender and reads some numbers stored inside the key that the sender can compare to what was sent so that trust is based on proof of receipt plut voice recognition. These solutions work only for people who already know each other, so PGP also supports a type of social software validation process that provides for multiple signatures on PGP public key certificates. Without knowing a correspondent directly, a user can employ their own judgment to assess the trustworthiness represented by the number or quality of individuals who felt the public key was authentic and indicated their conviction by sign-

ing the key. PGP has become quite popular because of its reliance on existing trust relationships, *i.e.* a person's set of acquaintances. This model might meet a VO's requirements for simple identity management, but will not be suitable identification for a resource owner such as a university on-line library that may have licensing policy requirements. For universities where an identity management infrastructure is already in place, using that infrastructure is likely to be more desirable than relying on the PGP self-identifying approach.

Single Root Trust Models

Trust among a set of separate domains has been implemented in the past by establishing a single root authority and subordinating all participating domains underneath the one root. This root authority is referred to as a "trusted third party" or a "trusted root authority."

Kerberos

Project Athena occurred at the Massachusetts Institute of Technology during the period 1983-1989, focusing on how to secure the emerging client-server computing environment. Project Athena addressed the following issues in distributed AAA: secure encryption of credentials transported across the network; single sign on used to access available network resources; and identification of both servers and users. These requirements were met by introducing a trusted third-party authority into the network called the Kerberos root that provides an authentication service and also per-application server session keys. Kerberos security credentials are referred to as Kerberos tickets. The root, application servers and participating clients together are referred to as a Kerberos realm,

which corresponds to an administrative domain (Kohl & Neuman, 1993; Garman, 2003; Stallings, 2003). Upon logging in to a Kerberos server using a username/password pair, the user has access to any application provided in the realm until the time stamp in the original login ticket expires. Symmetric encryption is used for secure ticket transfer and verification. The most recent version of the Kerberos protocol added the concept of inter-realm authentication accomplished by cascading trust relationships through a single "über" root.

The first commercial Kerberos implementation was introduced in 1989 and was called the Distributed Computing Environment (DCE) (Open Software Foundation, 1993). DCE introduced many new and important concepts in distributed computing that were added to its Kerberos core. Many of today's key distributed computing concepts such as remote procedure call, heterogeneous computing environment, directory services, and distributed data management were first developed for DCE. Unfortunately, DCE was difficult to install and administer and was also buggy so it never received wide adoption. Kerberos finally matured as a commercial product when it was adopted by Microsoft as the basis for its Windows 2000 / Active Directory architecture.

The Kerberos trust model was an important step in developing a secure distributed computing environment, and it is still in use today. Its single root trust model, however, has some significant shortcomings. It is difficult for one domain to participate in more than one Kerberos realm; therefore, all persons using those services must belong to that domain. This requirement may make it difficult to share administrative responsibilities so that users can, as happens frequently at UAB, find themselves belonging to two or more Kerberos domains and must remember which one to log into for each service used. This

awkwardness is because of Kerberos's bundling of domain administration, identity, and resource access control in a single architectural function. These limitations became important limits for e-commerce because of the incompatibility of domain-specific Kerberos ticket in a web-based environment of independent, unrelated domains.

Microsoft attempted to address Active Directory's limitations in a short-lived project from 2001 to 2003 called Microsoft Passport. Passport was to be a Single Sign-On service providing a single identity and authentication for Internet users that would be accepted everywhere and provide a common identity shared across all Internet and Microsoft applications. The European Union's (EU) reaction to Passport was an almost year-long investigation into whether or not Passport violated EU privacy laws, and Passport also suffered from discovery of a series of serious security flaws (Electronic Privacy Information Center, 2005; McWilliams, 2005). Eventually Microsoft abandoned the idea of Passport as a global solution for authentication.

Public Key Infrastructure

Public Key Infrastructure (PKI) refers to an architecture built around the X.509 protocol. A PKI binds the identities of key owners to public/private key pairs that are signed by a trusted third party. The keys are used to create trust relationships and to secure communications between any of the PKI participants. X.509 is an International Telecommunications Union (ITU) (International Telecommunication Union (ITU), 2000) and also Internet Engineering Task Force (IETF) (Chokhani, Ford, Sabett, Merrill, & Wu, 2003) standard providing authentication services associated with use of an X.500 directory (Telecommunication Standardization Sector of ITU, 2004). Asymmetric crypto-

graphic key pairs signed by a Certificate Authority (CA) are used to sign, encrypt, and verify data that traverses the Internet. As in Kerberos, scalability is achieved by a hierarchical arrangement of authorities. Each CA level is signed by the private key of the authority one level above, thus establishing a chain of trust, referred to as the Start of Authority (SOA), that can always be traced back to the root CA. PKI was intended to provide identity, only, and it was hoped that by adhering to this simple concept a global PKI could be achieved.

Any application using PKI should independently locate the authentic public key for that third party and use that key to verify the certificate's validity and also make sure that the key has not been revoked since being issued. That requirement raises the question of where to locate the appropriate key so as to be sure of its authenticity. By original design the X.500 directory was to be used to store the CA's public keys and to list any revoked certificates. Although the global PKI has not yet appeared, X.509 has received broad adoption because it has enabled e-commerce by use of SSL. In SSL, PKI is used to establish a one-way trust where it is the DNS registered name of the web server that is of interest. The browser's role is to validate the server name by following the chain of trust through a certificate bundle that has been pre-installed with the operating system and then negotiate a secure communication channel. The browser user does not need to be identified to establish this connection.

Although the encryption technology used in PKI is quite strong and X.509 has been widely adopted, PKI has serious shortcomings as a solution for providing identity. The vision of a global PKI has not proven to be practical for a number of reasons. Key management by end users has been one of the obstacles; for PKI to work as intended,

each person on the Internet would need to have a private key and to understand how to properly manage their private key, because anyone in possession of this key has essentially stolen their identity. The rapid proliferation of mobile devices has made the private key management issue even more complex. PKI imposes major management challenges for system administrators; for example, outdated public and private keys should be escrowed so that a stored, encrypted message is readable at any time in the future, or for future validation of a digital signature. In addition to all the key management difficulties, there is also no standard specifying the contents of an X.509 certificate. The X.509 standard describes only the packaging, transport, and functional type of key in use, and does not specify any data that is to be transported inside the certificate. Finally, the association of PKI with identity has been troublesome for those who are concerned with privacy and also for applications where user attributes or roles are more important that user identity.

The Globus Toolkit (Foster et al., 1997; Foster, Kesselman, Tsudik, & Tuecke, 1998) makes use of X.509 digital certificates for establishing grid user identity and for communication among distributed grid components. Globus did not, however, include a CA in its distribution until the release of Globus Toolkit Version 4 (Foster, 2005). Prior to that addition Globus assumed that some process external to Globus would manage identity and issue digital certificates. As a result, participants in prominent national projects such as Teragrid (National Science Foundation, 2005) or Department of Energy have been issued project-specific certificates, while those who are not participants in these projects may have no source for certificates at all. The distribution of certificates depends on personal knowledge of all project participants directly or some trusted party such as a Principal Investigator who vouches for her project team members by letter or

email. This approach works for a few grid users but is not scalable for managing a large user population.

Bridged Certificate Authorities

Organizations that valued PKI for user authentication eventually recognized that the vision of a global PKI with a single SOA would never become reality, so they began to explore alternatives. An EDUCAUSE project called Higher Education Bridged Certificate Authority (HEBCA) (Educause, 2005), and its affiliated HEBCA-Federal Bridge project (Alterman et al., 2002) were the first to bridge independent CA roots. The bridged CA model is a scalable hybrid combining the traditional single root authority with a peer-to-peer model at the SOA level. To bridge an unrelated set of SOAs, a bridge CA authority is created and cross certifies itself with each of the independent SOAs. If there are N SOAs there are $\Omega(n)$ cross-certifications to be accomplished; that is certainly an improvement over having each SOA cross-certify itself with each of the SOAs, an $\Omega(n^2)$ operation, as is currently done for the Teragrid project.

Cross-certifications at the bridge level allow a certificate issued by CA-1 to be accepted by a user who trusts only the unrelated CA-2 authority. By using a bridged CA, a federal agency such as the National Institutes of Health (NIH) whose employees and servers might be using Verisign™ signed certificates, for example, could directly verify the digital signatures on grant applications, no matter which CA served as root for that signature. Another driver for the federal government's interest in bridged CAs was that several agencies had already implemented incompatible PKI commercial solutions, making it difficult to conduct inter-agency communications.

The bridged CA solution requires establishing some degree of mutual trust among CA owners since the architecture depends on accepting identity that was established by someone else. Therefore, most of the work in bridging CAs occurs in the policy space, establishing mutual understandings of procedures, expectations, and practices in order to exercise the common trust. As a result of the HEBCA-Federal Bridge project the federal government and other interested parties are developing joint specifications for levels of identity assurance and what level of assurance is required for each type of signed electronic transaction.

Bridged certificate authorities (BCAs) have also been constructed for grid computing (Jokl, Basney, & Humphrey, 2004) and the first cross-certification occurred between the BCA at University of Virginia (UVa) and the UABgrid CA for use in the regional activity known as SURAGrid (Southeastern Universities Research Association, 2005). An interesting feature of the BCA architecture is that the trust chain is always rooted in the organization that is doing the certificate path validation. Therefore, when Jane at UAB is presented with a certificate representing a resource at UVa, Jane's client will follow a trust path from the resource certificate through the bridge to UAB's SOA.

Federated Identity and Federated Administration

Federated identity is the newest approach to distributed identity management. Federated administration describes a trust relationship among independent domains, where each entity is trusted to correctly identify its community using whatever its local authentication mechanism may be. Federation members are expected to accurately document the processes used to identity a person, the type of authentication system used, and

any policies associated with authentication credentials such as annual expiration. Other members of the federation may examine these policies to determine whether to trust information provided by this domain. Rather than using X.509 certificates, information is exchanged among federation members using Extensible Markup Language (XML) (World Wide Web Consortium (W3C), 2005) definitions related to security.

The XML messages are digitally signed and thus could suffer from many of the shortcomings of PKI mentioned above. However, federation is distinguished by five significant advantages. First, digital certificates are issued only to servers, not to individuals; as compared to PKI, this reduces use of digital certificates by at least an order of magnitude. In federation, an authentication service creates and signs messages asserting someone's identity thus eliminating most key management problems, especially those associated with a mobile population to be identified. Secondly, federation separates the act of authenticating from the object asserting identity; thus, it is possible for participants to use incompatible authentication systems while agreeing on the format used for asserted identity. Third, federation includes multiple user attributes in addition to identity, such as group memberships or roles, recognizing that identity alone may not be sufficient for all authorizations. Fourth, XML is intended to be used for a variety of purposes and there are well-established methods for developing and publishing a standard vocabulary. Therefore, federation can involve quite complex information exchange with reasonable expectation that the content will be meaningful. Finally, federation eliminates the existence of any root. The federation makes all participants' root bundles available but does not vouch for any of these credentials; only the issuing party does this.

Project Liberty

The Liberty Alliance is a consortium of over one hundred fifty companies, non-profit and government organizations committed to developing an open standard for federated network identity (Liberty Alliance Project, 2001). The Liberty Identity Federation Framework (ID-FF) describes a standardized, multi-vendor, web-based single sign-on built by federating identities based on commonly deployed authentication technologies (Wason & Cantor, 2005). Liberty describes each user as having a network identity that consists of all attributes associated with all the user's Internet accounts. Whereas user accounts are today isolated and spread across unrelated Internet sites, the basic idea is that these attributes (the various login names, passwords, PINS, and other information associated with each of those accounts) collectively constitute a person's network identity. The design of ID-FF is based on federating identity and also federating service providers. Affiliated identity and service providers can, using Liberty based architecture, conduct business in a secure manner and, most importantly, in a manner that appears seamless to a user inside that environment. Based on the ID-FF architecture, Liberty has developed an early web services specification called the Identity Web Services Framework (ID-WSF) that enables single sign-on using a federated network identity (Landau et al., 2005).

WS-Security, WS-Federation, WS-Resource Framework

The Web Services Security (WS-Security) standard was developed by Microsoft™, IBM™, Verisign™ and Sun Microsystems™ (Nadalin, Kaler, Hallam-Baker, & Monzillo, 2002) and was adopted as a standard by OASIS in 2004 (OASIS, 2004) so that WS-Security is now an open standard. WS-Security provides XML signatures, encryption

and key-management using the Simple Object Access Protocol (SOAP). SOAP bindings for Security Assertion Markup Language (SAML) (OASIS, 2005b) message exchanges are also described by WS-Security.

The Web Services Federation Language (WS-Federation) is a set of vendor developed specifications defining mechanisms for different security realms to federate using heterogeneous authentication and authorization systems. The federating parties establish trust among themselves. There is an associated set of specifications for exchange of identity, attributes, and authentication between participating Web services (Bajaj et al., 2003). This set of specifications is collectively referred to as WS-*. WS-Federation provides a closed, proprietary environment and is not an open standard.

The WS-Resource Framework is a set of specifications for six services that manage state in a Web services environment. WS-Resource Framework was intended to converge Grid and Web services (Baker, 2004) and was designed by IBM™ and the Globus toolkit developers as they worked together to migrate Globus from X.509 to web-services based communications (Siebenlist, Nagaratnam, Welch, & Neumann, 2004). The first working web services implementation of Globus (Version 4) was released in June 2005 as part of NMI Release 7. The WS-Resource Framework has been submitted to OASIS and is currently considered to be a draft under review as a potential open standard.

Evaluating Federation Standards

All the approaches to federation described above share several design features. Each approach depends on the availability of distributed identity providers and their authentication services, and a principal role of the federation is to establish policies that al-

low each participant to trust user identity that has been established out of realm. Another commonality is use of digitally signed SAML assertions. In addition to the reliability, privacy, and non-repudiation provided by this approach there is a less obvious advantage obtained by moving from X.509 certificates to assertions signed by servers. Moving away from PKI avoids the problems associated with user private key management and mobility; servers are rarely mobile and are usually managed by knowledgeable system administrators. A final advantage in each approach is that use of SAML provides tremendous flexibility in providing the information needed for many different types of authorization decisions. A drawback that is common to all these approaches is their current state of immaturity and the continuously shifting standards. Shibboleth and the newly issued Globus Toolkit are some of the only working software available for implementing federations.

Distributed Authorization Systems

Access control involves decisions made after comparing the policies associated with a target resource and the attributes associated with the requestor. Policy-based access control in a distributed environment is quite challenging, especially since the policies and attributes involved are likely to involve several administrative domains. Challenges in designing a solution include deciding where to store the attributes and how to make them available in a manner trusted by the consuming system. Another design choice concerns how to express policies and where to store them. Because of the difficulties involved, software developers tend to build rules and application-specific attributes into their applications rather than using external sources. However, if a set of policies is

to be applied consistently across distributed resources and domains then these rules also need to be shared across applications.

Attribute Managers

Shibboleth

The Shibboleth Project is an Internet2 initiative to develop an open, standards-based solution for exchange of information among institutions participating in a Federation. The developers of Shibboleth have also been active leaders in the Liberty Alliance, so there are many similarities in approach, except that the Shibboleth developers have focused on the AAA requirements in higher education. In particular, Shibboleth has been concerned with protecting user privacy and anonymity, largely because of federal regulations such as the Family Educational Rights and Privacy Act (FERPA) mandating protection of student education records, including identity and enrollment information (U.S.Department of Education, 2005). Important features of the Shibboleth architecture include federated administration, heterogeneous authentication systems, access control based on attributes, and a strong emphasis on user-managed privacy.

Shibboleth is an attribute transporting mechanism; it does not authenticate users, nor does it provide any user accounts. Shibboleth message verification and validation is provided by Security Assertion Markup Language (SAML), a set of XML definitions related to security. The assertion messages are digitally signed by the issuing server. SAML support is provided by the OpenSAML (University Corporation for Advanced Internet Development, 2005) library which uses protocol bindings based on Organization for the Advancement of Structured Information Standards (OASIS) (OASIS, 2003). A SAML

profile describes a specific use scenario; current SAML profiles begin with user authentication at their identity provider and presentation of credentials received as a result of that authentication to one of many possible service providers.

Shibboleth's attribute exchange process is reliable, secure, and privacy-preserving (Erdos & Cantor, 2002). Shibboleth transports attributes stored by the identity provider in a "Just in Time" manner to policy decision points. The current version of Shibboleth software is an implementation of SAML Version 1.1 with the addition of two distinguishing features: (a) protection of privacy by optional use of an anonymous "handle" in place of a name-revealing identifier, and (b) a service-provider first profile description. Whereas SAML requires the user to authenticate first, Shibboleth is designed for the scenario where a user first accesses some resource and may wish to remain anonymous when accessing that resource. Shibboleth is intended to be used to determine if a person has permissions to access a resource at a remote service provider based on information such as being a member of a particular institution or some specific class of users. An electronic publisher, for example, uses Shibboleth to determine whether a visitor to the publisher's web site is eligible to view publications that are licensed for use by different universities. In the past, this access control may have been accomplished by examining the IP number of the visitor, a scenario that works only if the visitor is actually on the university campus or knows how to access the university VPN from other locations. Since IP numbers can be easily spoofed this is not an ideal solution. Shibboleth releases only the attributes needed; in the licensed publications scenario, the handle itself representing "an authenticated user from university XYZ" may be sufficient information.

User attributes including identity, group memberships, and roles supplied by the

home institution can optionally be transported to a service provider site; agreements regarding which attributes will be released and under what circumstances are negotiated in advance as part of the federation process. Multi-party agreements are established via "Shib Clubs." A Shibboleth Club might be considered a loose VO, usually based around some common service; for example, a Shib Club for the purpose of accessing Elsevier documents online according to the appropriate licensing rules, which may vary from one institution to another. State information is maintained by the web server, and transparent browser movement is managed through URL redirects. Attributes are exchanged over encrypted channels that are secured using bi-directional Transport Layer Security (TLS). An example of the Shibboleth architecture is depicted in Figure 3. Each identity provider has an authentication service, and heterogeneous authentication services are possible. A user browses to the Shibboleth service provider; in this example the service consists of electronic journals provided to licensed users by EBSCO publishers. The user is redirected to the WAYF (Where are you from?) service to select the name of their home institution and are then redirected to the home Shibboleth Identity Provider. The Shibboleth IdP makes sure the user is authenticated and then assigns an anonymous handle that is returned to the service provider. The service provider references the handle when requesting additional attributes from the IdP whose address is now known to the service provider. Once the attributes are delivered, a policy decision can be made and policy enforcement occurs.

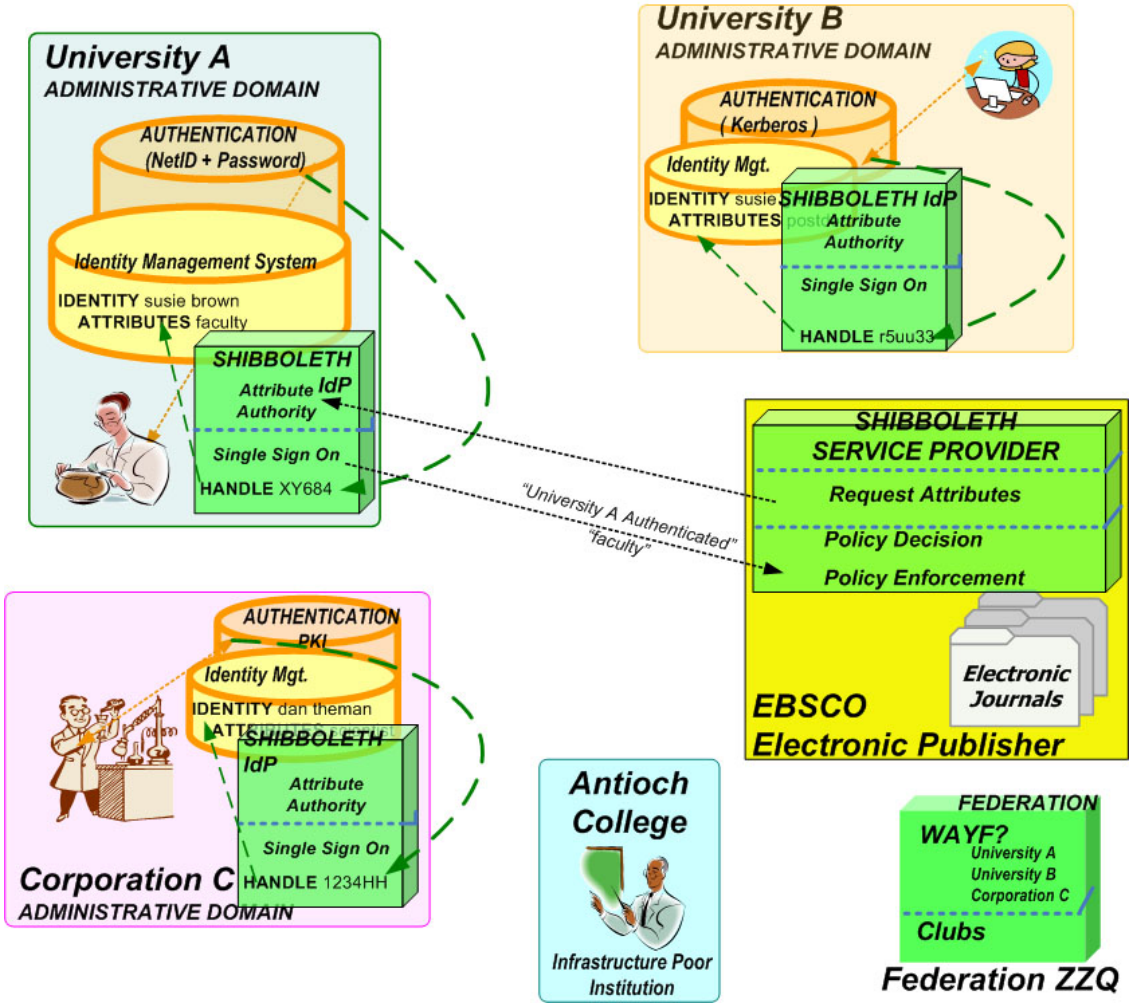

Figure 3. An example Shibboleth Federation

Additional Attribute Management Systems

The Privilege Management Infrastructure (PMI) was first introduced as a PERMIS component and became part of the X.509 standard as ofVersion 4. A PMI is used for authorization after authentication has occurred. An attribute authority uses the X.509 data structure to bind a set of attributes to their holder, and this type of certificate is called an Attribute Certificate. Attribute Certificates are digitally signed by the issuing attribute authority rooted at a Source of Authority. A single root common to identity and attribute authorities is not required; this means that identity and attributes are associated by the user's name. As with PKI, the attribute naming and semantics are not part of the standard. PMI has not been adopted beyond the PERMIS infrastructure with the exception of VOMS.

The Virtual Organization Membership Service (VOMS) (Alfieri et al., 2003) was developed in the European grid community. Attributes associated with each user are stored in various databases; for example, some attributes such as `faculty` may be assigned and managed by the university, while other attributes such as association with a particular project might be located at a national research lab. Each database capable of providing attributes for VOMS exposes those attributes via a VOMS service. The user is responsible for knowing which VO attributes are needed by each application as well as where each VO Membership server is located, and collects the needed attributes from the appropriate VOMS servers. The collected attributes are packaged into an X.509 certificate along with the user's identity and in that way the information is made available to grid resources.

Grouper (Barton, 2005) and Signet (McRae, 2005) are attribute management

software; each is intended to provide a user-friendly interface for defining and assigning attributes that can be stored in a location associated with the person to whom the attributes are being assigned. Signet is a privilege management and delegation assignment system, while Grouper is a role or group name definition and assignment system. Together Grouper and Signet provide functionality that is similar to the group definition and assignment capabilities in Microsoft Active Directory. The Grouper and Signet developers claim that their intention is to establish trust relationships that would allow anyone or any VO to assign whatever is necessary and that this information would then be stored in each organization's respective enterprise directory. The driving design goals are to have attributes assigned by whoever is authoritative for that attribute while keeping every attribute associated with a particular identity stored in one location. These goals make sense inside a university IT organization that is offering an authentication service and associated directory services; in fact, if the IT organization wants its authentication service to be useful to different units it must enable departments to manage access to their own resources. As to whether this design can scale to federations remains to be seen; implementations were just being introduced in March of 2006.

The GridShib project (Welch, Barton, Keahey, & Siebenlist, 2005) focuses on use of Shibboleth-issued attributes for authorization in Grids built on the Globus Toolkit. The grid community's interest in attribute-based authorization developed from recognition that identity alone was not sufficient information for many authorization scenarios. GridShib is focused specifically on integration of Shibboleth with the grid security infrastructure and consists of two plug-ins: one for Globus Toolkit 4.0 and one for Shibboleth 1.3. When a grid service provider receives a digital certificate identifying the owner of the

request, the Grid Toolkit plug-in extracts the DN from the certificate and forms a SAML query to be sent to a Shibboleth identity provider in order to obtain additional attributes. Unfortunately, a digital certificate's DN does not provide enough information to determine the Internet address of an authority having additional information about that identity. The GridShib developers considered use of the "handle" provided by Shibboleth in place of DN, but were then faced with the problem of how to map that name to existing DNs and grid-mapfiles.

Authorization Systems

While attribute management systems are concerned with managing, collecting, and transporting attributes, authorization systems may be termed Policy Decision Points (PDP). The terms PDP and Policy Enforcement Point (PEP) derive from IETF RFC 3198, entitled "Terminology for Policy-Based Management" (Westerinen et al., 2001). This document describes a glossary of policy-related terms that can be used for consistency in proposing solutions for authorization systems, and a PDP is described as "a logical entity that makes policy decisions for itself or for other network elements that request such decisions." The model described assumes the existence of a Policy Enforcement Point that is protecting a service. When an attempt to access the resource occurs, the PEP sends a description of the access request to a PDP; this message is called an authorization decision request. The PDP evaluates the request in the context of the service provider's policies and requesting user's attributes and produces a decision which is returned to the PEP; the PEP is responsible for enforcing the decision.

eXtensible Access Control Markup Language

The eXtensible Access Control Markup Language (XACML) is an OASIS standard specifying schemas for authorization policies and authorization decision requests and responses (OASIS, 2005a). As a language XACML is not a complete system but does offer important standardization for expressing policy. XACML Version 2.0 includes a profile detailing use of SAML to express XACML.

PERMIS

The Privilege and Role Management Infrastructure Standards (PERMIS) is a system for storing and accessing XML-format policy and attribute information in Lightweight Directory Access Protocol (LDAP) (Harrison, 2005) directories (Chadwick & Otenko, 2002). Developed in Europe, a driving design goal was to develop a domain-wide authorization policy that could be queried by any application, rather than creating multiple access control lists at each service provider. Applications query the PERMIS system to determine whether the current user has permission to execute the requested action at the service provider; the PERMIS privilege allocator component queries the LDAP directory for the relevant policy and role definitions which are compared with attribute, role, or access control information that is delivered inside an X.509 Attribute Certificate. PERMIS can operate as an authorization decision engine, providing "Allow/Deny" decisions. PERMIS authorization is independent of authentication and can be used with any authentication system.

Akenti Distributed Access Control

Akenti was developed at Lawrence Berkeley National Laboratory to address complex authorization problems involving multiple administrative domains; use of Akenti has been primarily within US Department of Energy funded research projects. Design goals included the ability for each resource owner to define and enforce its own access control requirements in near real-time (Johnston, Mudumbai, & Thompson, 1998). Akenti uses digitally signed X.509 identity certificates for authentication; attributes and policies are expressed in XML inside of signed certificates. In the Akenti authorization process, the user requests access to a resource and is authenticated, then a resource gateway contacts Akenti requesting a decision. Akenti locates the global and local usage policy certificates and also the user's attribute certificate; multiple locations may be consulted in gathering this information. Finally, Akenti renders an access control decision. Implementations of Akenti for both web and Globus environments have been developed. A potential Akenti shortcoming is its use of proprietary format policy and attribute certificates. A thorough comparison of Akenti and PERMIS (Chadwick & Otenko, 2005) concludes that although similar in over-all design, major differences can be found in implementation details.

Community Authorization Server

The Community Authorization Server (CAS) system is a "push-model" authorization service developed initially to manage file access control in a distributed environment (Welch, Anaanthakrishnan, Meder, Pearlman, & Siebenlist, 2005). Unlike the Akenti and PERMIS models in which a client's attempt to access a resource triggers an authorization

decision request, CAS is designed so that the client first contacts a CAS server indicating what resources they want to access and by what actions. The CAS server obtains the user's identity, consults the policy of the VO that owns the resource, and returns a signed SAML message containing its decision to the client. The client presents the signed assertion to the resource, which may in addition impose local policy based on its trust in the signing CAS server.

Lionshare

The Lionshare project at Penn State University represents a peer-to-peer authorization service that operates within a federated trust fabric (Open Source Initiative, 2005). Lionshare was designed as an extension of the open source gnutella protocol, a distribution system for discovering and sharing files without use of central servers (Kirk, 2005). Gnutella is best known as a popular system for downloading music files from the on-line collections of other fans, and the wrath of the music industry in pursuing copyright violations has highlighted the role of policy and authorization in distributing intellectual property on the Internet. The goal of the Lionshare project is to enable individual faculty authors to organize, store, and determine access policies for their own intellectual property. The initial design is based on use within a single Kerberos domain, but the developers envision extending their work into federated space. In the peer-to-peer scenario, individuals publish information and users can discover and retrieve files of interest. Descriptions of the information available is made available by use of metadata, and the publisher digitally signs the metadata they've created to indicate their ownership; in addition, the publisher may use metadata to define her own access control policy. Access control is

determined by a list of required attributes. Anonymous retrieval is supported, meaning that a user presenting the appropriate attributes may download information without necessarily revealing their identity. Lionshare credentials consist of a temporarily issued digital certificate pair; one credential contains an anonoymous "handle" and the other certificate contains the owner's name, email address and department. The identifying certificate is used to sign published data. The opaque certificate is used for accessing the metadata, and also for opening an SSL secure channel to the domain's Attribute Authority, which returns the requested attributes associated with the handle provided. Finally, the client opens a bi-directionally verified SSL connection with the "server;" presents an assertion signed by the Attribute Authority and containing the attributes, and may download the file. The "server" is not a DNS registered service but is actually just a portion of a file system that belongs to an individual.

Interfacing Applications with Trust Models

VO's are the core of today's scientific collaborations. Researchers no longer gather together around the library and work in close proximity – that approach describes the first 800 years of the university. The extreme amount and density of knowledge available today has resulted in knowledge so highly specialized that persons having the precise expertise needed for a research team are likely to be found at some distance. Most research teams consist of relatively small numbers of people who spend a lot of time developing ideas and sharing results together. The best collaboration software available today is either proprietary, which means expensive and non-interoperable, or so complex to assemble into a working environment that only national laboratory staff have the required

expertise and time to build them.

The desired result is that groups, whether large or small, whose memberships cross institutional boundaries, should be able to work electronically as a team with their preferred toolset, preserving the desired access control but without introduction of an entirely new set of login/identification procedures and with consistent and transparent management of the attribute and role information required to properly manage access. The utility of this approach is demonstrated by the Yahoo Groups Service that allows any Yahoo subscriber to create a group name and manage group members' use of shared files, photos, databases, calendars, links, and polls; there are hundreds of thousands of groups listed at this service (Yahoo, 2005). The Yahoo Groups service is a proprietary packaging of mailing list, database and other useful tools; one must be a subscriber to Yahoo to use any of the non-mailing list features. Google offers a similar service called Google Groups.

Application Silos

When small research groups turn toward the open source community for collaboration software solutions they find a wide range of software available such as wiki, content management systems, blog, and file sharing solutions. These applications are typically designed for self-contained authentication, have limited role definitions, and authorization is handled by having the system administrator create application instances and accounts manually. The reasons for this "silo" style of application development are many. The first factor is the developer's desire to create a complete, stand-alone solution; this perspective usually results in "feature creep" over time since every feature needed by

anyone must be built into the application. Another factor is that greater skills are required to design modular software with well-described interfaces to other software. Finally, the federated model is fairly recent and is not yet well understood by the developer community. As a result, while many open source collaboration tools are individually useful, they are limited by having separate authentication infrastructures and unrelated user lists and accounts.

Portal Silos

Portals were developed to provide a single point of access to a set of services used by an organization. A portal manages identity and attributes so that this information is shared across applications housed inside the portal. An application is attached to a portal via a portlet, a standardized interface between an application and the portal environment. The Open Grid Computing Environment (OGCE) (OGCE Consortium, 2006) is an emerging portal standard based on the JSR 168 Portlet Specification (Hepper, 2006). JSR 168 defines interoperability standards for portal containers. Other popular open source portals include Sakai (community source, 2005) previously known as Chef, uPortal (JASIG Open Standard, 2005), and GridSphere (Novotny, Russell, & Wehrens, 2004).

While portals are certainly an improvement over per-application AAA, there are still some limitations to consider. Typically, the portal uses a proxy mechanism so that it can authenticate on behalf of the user for each application; grid portals, for example, depend on the existence of MyProxy (Novotny, Tuecke, & Welch, 2001) which authenticates to Grid services on behalf of the user. The implications of this design are that the portal is essentially impersonating that user, and the user loses some control over when

and how their identity is presented. The danger of impersonation is often addressed by use of proxy credentials with short lifetimes, for example 24 hours or less. From a functional perspective, a more serious limitation is that a system architect is required to assemble a portal which means that collaborators are presented with a set of tools that have been selected by someone else. In order to add a new application to a portal, the application must be redesigned and a set of platform-specific portlets needs to be developed; this represents a fairly significant development effort. Finally, there is an irony in realizing that each portal is itself a silo; a user who happens to participate in several collaborations may find herself still facing multiple authentication systems and unrelated accounts.

Peer-2-Peer Silos

The final set of approaches to consider fall into the Peer-to-peer (P2P) classification. P2P design is based entirely on actions taken by end-users and requires no central infrastructure at all. An early well-known P2P implementation was Groove. A set of users were declared a group by action of any one user. User identity was handled by email address. Files belonging in any member's file system space could be shared with any other group member. Unfortunately, anyone who turned off their machine or took their laptop off the network would also be removing access to these documents, and would sometimes even lose their identity. Since being acquired by Microsoft, Groove has been integrated as a friendly front end for Microsoft's Sharepoint. Sharepoint is a file sharing system and can be used only by persons listed in a local Microsoft Active Directory or in an Active Directory having configured some trust relationship with the hosting directory.

Summary

None of trust management solutions described in this chapter provided a solution enabling VO's to easily share data and resources across domains. Where the domain has some ownership or responsibility for what is being shared, peer-to-peer solutions are not useful. Single domain solutions, although secure, were inadequate because they do not work when more than one domain is involved. The mechanisms associated with X.509 digital certificates worked well for identifying computer systems and for establishing secure communication channels, but global PKI never happened as an identity management system. Implementing access control based on membership in a particular VO requires that the VO membership attribute be stored in some consistent format and location, but PKI is concerned only with identity and single domain attribute storage is inaccessible outside that domain. The few existing attribute management systems were designed to serve only a single VO. Finally, grid computing provided a means to execute jobs on remote systems, but the scenario relevant to most collaboration tools is access by a web browser.

Grid computing and federations were the only two approaches to cross-domain AAA. Grids were developed to support research VO's using a PKI based security infrastructure and a working implementation was available in the Globus toolkit. The grid security infrastructure designers concentrated on identity only, and left authorization as a problem to be dealt with later. Unfortunately, attributes that are not closely associated with identity are not useful for access control, and while open source CA's are certainly inexpensive the grid was already stumbling over certificate verification and management issues. Federations were developed mostly to address business to business trust issues

using signed XML messages, but the only working prototype was the Internet2 Shibboleth software. The federation architects recognized that authorization cannot occur without authentication, and the Internet2 architects had established eduPerson for standardized naming and storage of identity information. Unfortunately, their schema design was based on a standardized sharing of only domain-specific attributes, which did not include VO membership.

DISSERTATION OBJECTIVES AND CONTRIBUTIONS

Collaboration Tools for Virtual Organizations

VO's are the core of today's scientific collaborations. An expert's scope of knowledge has become more specialized while funding agencies are demanding a broader, more diverse approach to research. A team of experts is needed to address this demand, and quite often the team members are not all from the same institution. An essential requirement for this dissertation is a trust management solution that is suitable for use by VO's, especially small teams of investigators.

Each VO needs a set of collaboration tools, but predicting which tools might be needed is nearly impossible because the total number of tools available today is staggering. Software tools may be discipline specific and, in addition, personal preferences vary considerably. Use of data collaboration tools in higher education was surveyed by ViDe[4] in 2004 (Trauner, Finken, Hofer, & Krienke, 2004), and 180 people responded. Key findings from that survey analysis included (a) research required data collaboration more than teaching or administrative functions; (b) lack of interoperability negatively impacted the tool's usefulness; and (c) a weak preference for integrated tools was expressed. The survey asked respondents to prioritize the importance of 22 collaboration functions, and respondents wrote in an additional 29 desired functions. Specific scenarios correlated with distinct functional requirements. Additional evidence of the need for great flexibility in

[4] ViDe Video Development Initiative. Retrieved April 1, 2006 from http://www.vide.net/

selecting appropriate collaboration tools exists in the listing of five hundred and nine different open source content management systems at the "CMS Matrix" website.[5] Possible explanations for the current plethora of collaboration software include (a) the field is immature and standards will take some time to emerge, or (b) the ubiquity of web-based access and emerging web services frameworks provide all the standardization that is necessary. In either case, a reasonable conclusion is that establishing a framework supporting a pluggable component approach to tools would be an extremely useful contribution allowing collaborators to easily choose and assemble an environment best suited for their knowledge domain and personal preferences.

VO Collaboration Environment Design Requirements

What characteristics would such a collaboration environment have? In addition to the collaboration surveys mentioned above, design requirements were derived from communications with other professionals interested in collaborative software, experience as a member of several VO's attempting to use some of this new software, and experience in a university information technology service division.

Minimum required design characteristics included: (a) enterprise identity management including enterprise authentication must be used, if available; (b) at least one VO-specific attribute (*i.e.,* member of VO XYZ) must be required; (c) some resources must be managed by the enterprise; and (d) application-specific accounts must be provisioned automatically. The requirement for enterprise resource management and enterprise

[5] CMS matrix web site. Retrieved January 20, 2006 from http://www.cmsmatrix.org/

identity management were linked; while the two enterprises did not need to be the same, each one would have policies regarding identity and use to leverage in the solution. When there is no enterprise involved and members own the resources in use then a P2P trust model might be a better solution choice. The VO membership attribute requirement was necessary in order to have an access control rule based on that membership. Automated account provisioning at the application level would provide a minimum of VO self-sufficiency. The requirement to use existing enterprise IdP's brought along the associated requirement that incompatible authentication architectures would be used.

Desirable design characteristics included: (a) a standardized interface that could be applied to existing applications to enable the concept of pluggable application components; (b) ability to access data and services using a standard web browser; (c) access control based on VO role in addition to VO membership; and (d) automated provisioning of applications and other infrastructure needed. Under ideal circumstances, establishing and managing a VO and its access to resources would require no administrator intervention at all.

Challenges arising from the design requirements included: (a) how to identify and trust the identity of VO members who did not have an enterprise IdP; (b) what to use to create VO's and manage their memberships and roles; and (c) how to associate enterprise identity with VO-specific attributes. These challenges are each related to trust management, choice of cryptographic credentials, and common attribute semantics and descriptors. Solving the VO collaboration problem was a system integration challenge: would it be possible to choose components from existing solutions and integrate them in a way that produced a new outcome?

Selecting a Trust Management Solution for the VO Collaboration Problem

As summarized previously in the chapter on related work, trust systems have employed a single root trust model so that all aspects of trust are internal to that system. A single root is responsible for establishing identity, and a single set of groups, roles, and rules exists to be applied for authorization decisions. The resulting requirement for a "monolithic" hierarchy and its inflexibilities has been a trade-off for the benefit of direct control of all trust issues. Single root models, including Kerberos and PKI, were scalable across an enterprise but have not been scalable to the Internet's many-to-many transactions.

The Grid Security Infrastructure (GSI) was an important contribution to rethinking AAA frameworks and to defining a security middleware. GSI recognized that an externally established identity could be mapped to any local system account so that it was possible to conceive of a globally available computational grid. However, GSI was also based on the hope of some future global PKI because methods for establishing scalable identity were not addressed by GSI. In addition, GSI did not include consideration of any user attributes other than certificate DN and local account name.

Federation, as being developed by Liberty Alliance and Internet2, appeared to have some promising new approaches that could lead to a secure and flexible middleware. As a participant in the National Science Foundation's Middleware Initiative (NMI) (NMI, 2005; Blatecky, West, & Spada, 2002) Testbed program the author became familiar with the concepts and components of the emerging middleware architecture. The NMI program included components from the GRIDS and EDIT developers. The NMI-GRIDS developers were the Globus Toolkit developers; the NMI-EDIT developers were from the

Internet2 community and included enterprise software architects. That group established standardized LDAP directory schemae, pre-standard SAML implementation libraries, and software to manage group and role assignment. These particular components could be used to provide the semantics needed and were intended to address non-hierarchical trust models.

The Federation work looked promising, but consisted of some draft architecture papers, libraries, and configuration guides; the components were not yet integrated with each other and there were no applications that could make use of them. It was difficult for most people to understand what the components might be used for. The Shibboleth architecture seemed especially interesting because it was based on distributed identity management which provided scalability and a certain inherent level of trust. Shibboleth also offered a method for secure attribute transport, and had the additional advantage of being based on standards emerging from the OASIS and Liberty Alliance initiatives. Considering the design requirements, Shibboleth was the only trust model implementation available that had some working software components, did not require a single root authority over participating components and utilized existing identity management architectures. These features suggested Shibboleth would be the best choice in trust models.

Shibboleth and Web Services

Federation loosens the requirement for a single root authority by replacing the root with definitions, policies, and semantics agreed upon out of band. Enterprise participants consist of identity providers and resource providers. Resource providers specify what they need to know in order to authorize access to their resources; identity providers

decide which attributes about their members will be made available and select which resource providers can receive that information. Identity providers use their own identity management procedures and choose their own authentication methods and agree to make those procedures and methods known to other members of the federation. Resource providers can examine the identity management practices of each IdP and decide on the trustworthiness of the information offered.

In order to be able to exchange information among participating Identity Providers (IdPs) and Service Providers (SPs) some standard approach to user attributes is necessary. This requirement was addressed in higher education by Internet2 and EDUCAUSE who developed the eduPerson object class (Hazelton, 2006). The eduPerson object class describes many attributes, and one of the more important ones is `eduPersonPrincipalName` (ePPN). The InQueue and InCommon federations combine NETID and home domain name to form a global identifier referred to as a scoped ePPN. An example of ePPN would be `mary.smith@idp.example.org`. In this example, `mary.smith` represents the local enterprise network identity (NETID). The `@` sign separates the NETID from enterprise domain name, and `idp.example.org` is the globally unique name for the enterprise, typically the enterprise DNS name. To preserve privacy, identifiers for these federations may be more generic such as `student@idp.example.org`. In this case `student` represents that the current anonymous user has a student affiliation with the enterprise `idp.example.org`. The NETID `mary.smith` appears to correspond to a person's name but some random identifier such as `ZYQ43U` could have been assigned instead, as long as that identifier is unique within the `idp.example.org` domain. These semantics and conventions provide a basis for meaningful information ex-

change among members of a federation. The format used to conduct that exchange is a standardized extensible markup language (XML) called Security Assertion Markup Language (SAML) that can describe authentication, identity, and attributes in a text-based, standardized manner. This use of XML is consistent with the current trend towards web services, a more general approach for exchanging information via XML among heterogeneous software, operating systems, and hardware platforms. Consistency with web services also contributed to the selection of the Shibboleth trust model especially because of web services' strengths in heterogeneous systems communications.

SAML describes not only security assertions but the ability to digitally sign these assertions, and in the case of Shibboleth the digital signature is that of a server belonging to either an IdP or an SP. Server-level digital signature approaches provide many powerful advantages over individually managed private key signing schemes. Both types of digital signatures employ an X.509 public/private key pair, but server-level private keys solve many of the known PKI private key management difficulties. A host computer creates and stores its private key locally and has no need to transport its key anywhere, while an individual is highly mobile, uses multiple devices, and must understand key management in this complicated environment. A single server may be used to identify thousands or even hundreds of thousands of people; it is intuitively obvious that making technical arrangements for one server signature to assert the identity of each of these persons is one or more order of magnitudes simpler than handling technical arrangements for thousands or hundreds of thousands of user managed certificates.

Use of SAML also allows for message formats that are considerably more flexible than the content conveyed in an X.509 certificate. XML messages can be highly expres-

sive and may also be used in a query and response format for selective release of attributes, while X.509 certificates can be sure to contain only a DN and perhaps an email address. In conclusion, federation was selected as the best choice for trust management because it was the only cross-domain solution providing both identity and also other attributes needed for authorization in remote domains. The Shibboleth implementation of federated identity was selected because it was the only working implementation available.

EXPERIMENTAL DESIGN

Experimental Setup

While the emerging set of NMI middleware components were being tested for basic working functionality, it was not at all clear that these components could be combined by a system integrator to support cross-domain single sign on. What steps were required to "middleware-enable" applications? Was it possible to integrate NMI components with distributed resources to create a system environment providing identity or group based access control? The experimental setup required to test and evaluate these questions was potentially challenging because of the requirement for separately administered identity providers that would each install some compatible infrastructure and would also be willing to explore approaches to cross-domain authorization. Fortunately the many activities of the Internet2 Middleware and NMI program had resulted in exactly the technology deployment and personal contacts needed. Many institutions, including UAB, had completed a migration of their directories to the eduPerson schema, resulting in a common naming, storage and retrieval strategy for university person attributes. UAB had also implemented an enterprise-wide authentication system. Those elements were a necessary minimum set of components required to install Shibboleth and begin building a VO collaboration environment. The experimental components are illustrated in Figure 4.

Elements colored yellow and marked [D] represent components that were deployed by staff at other Internet2 member universities. Each location serves as an Identity

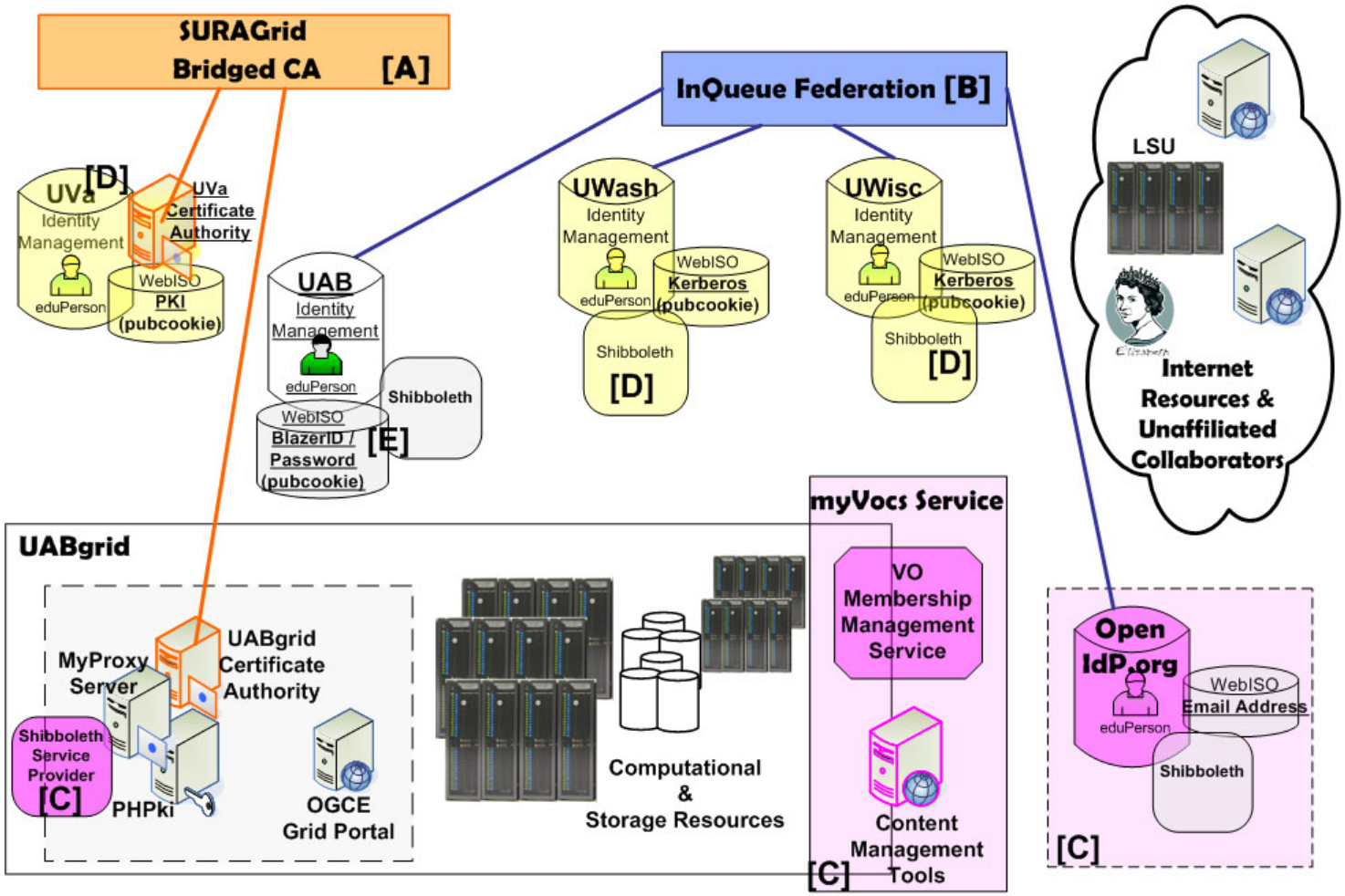

Figure 4. Experimental Design: yellow components were provided at other Internet2 institutions; blue indicates InQueue components; orange indicates a SURAGrid bridge components; gray indicates Shibboleth components installed at UAB for this project; pink indicates challenges addressed in the dissertation.

Provider (IdP) and maintains an authoritative directory of persons affiliated with that institution; each location also provides a central authentication service. The University of Washington and the University of Wisconon (UWisc) had each deployed a Kerberos authentication infrastructure and had also installed pubcookie as their web initial sign on (webISO) service. The University of Virginia (UVa) was an early adopter of PKI technology for its primary authentication service and used X.509 certificates for user authentication. UVa had also installed pubcookie as its webISO.

Elements colored in gray and labeled [E] represent Shibboleth components needed at UAB to participate in Shibboleth's cross-domain authorization architecture; these components needed to be installed and configured in order to complete the experimental setup.

The InQueue Federation, illustrated in blue and marked [B], is an experimental federation established by Internet2 as a sandbox environment so that identity providers could install and test Shibboleth components in a non-production environment. Each entity joining InQueue chooses which attributes to release to other federation members; typically InQueue participants choose to release attributes that have been designated as public by their enterprise. For example, the UAB IdP has determined that the eduPerson attributes `eduPersonPrimaryAffiliation` and `givenName` are public, while `preferredLanguage` and `userCertificate` are not[6]. The dark blue lines indicate that UAB, UWash, and UWisc were participants in the InQueue Federation while UVa was not.

The NMI Testbed activities created a working consortium of organizations col-

[6] UAB LDAP schema V.1 http://www.dpo.uab.edu/US/ldapfields.html Retrieved April 1, 2006.

laborating and combining resources to help bring grid technology to the level of seamless, shared infrastructure; this work became known as SURAGrid. The UABgrid is a security infrastructure leveraging the BlazerID authentication system to identify grid users. SURAGrid was interested in exploring use of a bridged CA for grids. The bridged CA components are represented in orange and labeled [A].

Components represented in pink and labeled [C] illustrate the core problems addressed in this dissertation and will be described at length in the next chapters.

Shibboleth Components

Shibboleth Identity Provider

Shibboleth requires the existence of a campus identity management system, an authoritative data store listing those identities, and an authentication system. The data store can be an LDAP Directory Server or any of the standard relational databases; the authentication system can be any system chosen by the enterprise. Using these elements as a foundation, Shibboleth adds a Single Sign On (SSO) service and an Attribute Authority.

The SSO service is the initial point of contact at the IdP. The SSO service interacts with the authentication service; the SSO can force a user authentication or determine that the user has authenticated recently enough and remains aware of each authenticated user's NETID. The NETID is a network identifier that is unique within an enterprise; in the eduPerson schema, the `eduPersonPrincipalName` (ePPN) attribute is used for NETID. Additional Shibboleth components needed at UAB were an Attribute Authority, a Single Sign On Service (SSO), and one or more federation memberships. Each site also requires some type of webISO solution, and an implementation of pubcookie was already

available for use at UAB. The SSO interacts with the webISO system and also responds to external queries. The SSO is aware of each authenticated user's NETID (and eduPerson attribute) and may assign an anonymous identifier to any user for purposes of preserving privacy. These functions are common in both SAML and Shibboleth; however, Shibboleth adds the capability to hide the user's identity by having the SSO assign an anonymous identifier in place of ePPN. The resulting anonymity preserves privacy. This capability was important to the Shibboleth architects because of the Family Educational Rights and Privacy Act (FERPA) law mentioned on page 46, but the privacy preserving aspects of Shibboleth may be quite useful for health care related applications as well.

The Shibboleth Attribute Authority (AA) processes incoming requests for attributes and issues attribute assertions according to the enterprise Attribute Release Policy (ARP). The role of the AA is to query the authoritative data store for attributes associated with each NETID and convert them into a standardized a SAML assertion (SAML was described in the section on the Liberty Alliance Project, page 31).

Shibboleth Service Provider

The Shibboleth Service Provider (SP) components include an Attribute Requester and an Assertion Consumer Service. Access control in Shibboleth is currently implemented using the mod_shib Apache web server module; in order to run that module, the system must also be running the Apache or IIS web server and the Tomcat Servlet Container. These components work together to protect web content until the specified local access control rules have satisfied; protection is implemented by use of a `.htaccess` file with the following contents:

```
AuthType shibboleth
ShibRequireSession on
Require valid-user
```

The Assertion Consumer Service processes assertions from the IdP's SSO and establishes a security context at the SP for the user. The assertions received by the Assertion Consumer Service are carried by the user's browser which serves as a sort of secure bucket for transporting messages from IdP to Assertion Consumer Service. The Attribute Requester may perform additional query-response dialogues with an IdP's Attribute Authority if additional information is required; this dialog occurs between Attribute Requester and Attribute Authority modules over a secure communication channel instead of using the browser as the communication vehicle.

Shibboleth WAYF Service

Shibboleth is designed to support Service Provider first access requests; this is the second feature distinguishing Shibboleth from SAML. In the Shibboleth flow scenario a browser arrives at a protected resource managed by the Service Provider. The Service Provider needs to request authentication and identifying information from the SSO associated with the current web browser user; that request is in the form of a URL-encoded message and can currently be configured to point to only one location. The Service Provider has no way of knowing yet who the user is or which Identity Provider will have the information needed so the request is first re-directed to a Federation provided service known as the "Where Are You From?" (WAYF) service. At the WAYF the user selects their home institution from a drop-down list of federation participants and is re-directed again to their home SSO. The IdP releases attributes associated with that user. Eventually

attributes associated with that user are transported back to the Service Provider's Attribute Consumer Service so that access can be authorized or denied at the protected resource. The exact flow to each Shibboleth component and sub-component is detailed in Figure 5; the path illustrated with arrows indicates the browser's path from initial attempt to access the protected resource to eventual access. Dotted arrows indicate optional communication flow steps; the blue dotted lines (labeled (7) and (8)) refer to request-reply communications that do not involve the browser.

Open Source Applications

Open Source applications were selected for use in the VO collaboration environment so that the application source code could be modified as necessary. Initially applications were chosen that were considered useful in terms of function provided; the first applications selected to middleware-enable were a mailing list, a content management system, and a simple file sharing system. The exact package to use for each application type was made after evaluating what was available in terms of programming language choice, degree of modularity in the code, developer documentation available, and interest in the project by the application's developers.

The choice of mailing list management software turned out to be a key selection. Mailing List Management software (MLM) has been supporting collaborations for twenty years or more and has become an essential tool for inter-organizational collaborations. It was reasonable to assume that the mailing list application would be an important element in a middleware enabled toolkit and that there might be others in the Internet2 community who would be interested in exploring integration of the mailing list application with mid

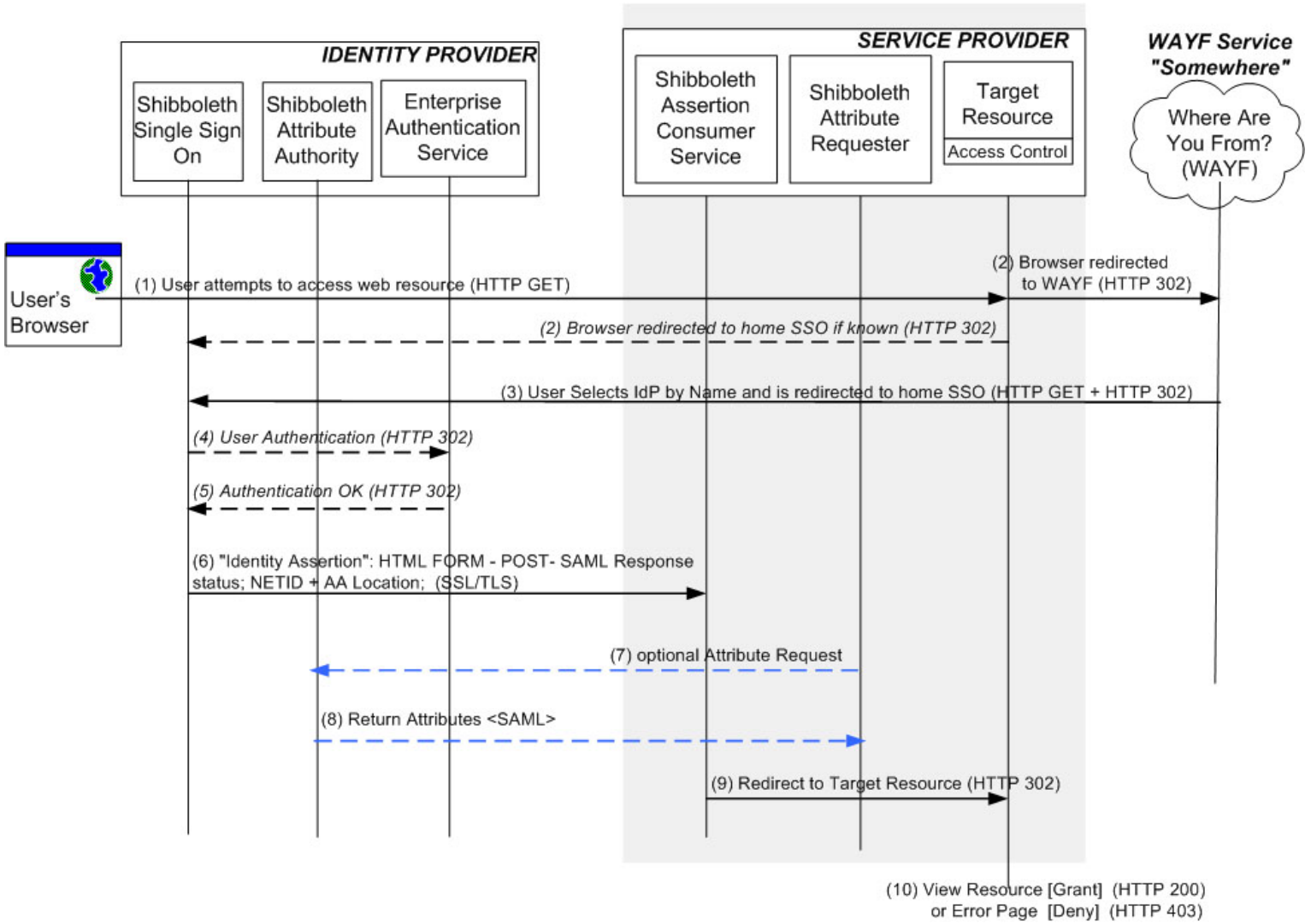

Figure 5. Shibboleth Communication Process

dleware. As a result, a special Internet2 working group was formed around this topic (Internet2 & Gemmill, 2006). After closely examining the interaction of middleware with the MLM application, attention would turn to selection of other applications.

The UABgrid Component

The University of Alabama at Birmingham is a single public institution, but it is also a set of divisions that each has its own management structure, funding sources, and priorities. As is typical in many large universities, each division makes its own decisions regarding purchase of computing resources based on available funding and for purposes they have identified. Legally all items purchased by the university are the property of UAB; in practice, resources are generally viewed as being owned by the researcher, department, or division that manages the funding source from which the purchase was made. This situation creates an environment in which decisions about who may use a specific computational resource are distributed among deans, department chairs, chief librarians, and researchers who have acquired grant funds. Given the reality that computer systems at our campus have highly distributed ownership and administration, the grid architecture was identified as a solution capable of maximizing use of these resources.

UABgrid is a collaboration between academic and administrative units at UAB, as well as a grid architecture designed to provide shared computational resources to UAB. UABgrid partners included the Department of Computer and Information Sciences (CIS), the School of Engineering's Enabling Technologies Laboratory (ETL), and IT Academic Computing. Resources available in UABgrid as of March 2006 include six high perform-

ance computing clusters with a total 862 processors; several terabytes of storage; and two large visualization walls (Viz-walls). Equipment rooms are each directly connected to the gigabit Ethernet campus backbone. All resources are available for use by members of the UAB community and specific usage policies are determined by the owner of each UAB-grid resource.

Grids to date have left the issue of identity management as an open question: a digital identity is needed in order to use the grid, but who should provide that identity? The issue of authentication (proving who you are) is often confused with authorization (what you are allowed to do) and the practice of issuing grid certificates based on the project you are working on is an example of this confusion. Department of Energy (DoE) funded researchers, for example, must have DoE-issued certificates in order to access DoE funded resources, and NASA funded researchers require NASA-issued certificates to access NASA-funded resources. While this approach is fairly straightforward regarding certificate validation, it is not scalable: one user may end up with many certificates and need to know which one to use for every access attempt. By approaching the problem in a new way that separates identity from authorization, identity can be established by any trustworthy identity provider and authorization can be managed by the resource owner.

The grid security architecture requires use of digital certificates and all the many issues related to key management as discussed earlier. It seemed desirable in the UAB environment, where there was little previous exposure to grid computing, to hide the details of key distribution and management from end-users. Much work in simplifying key management had already been accomplished by the Open Grid Computing Environments

Collaboratory (OGCE) project; however, the portal required its own central repository. At the beginning of this dissertation project's timeline none of the grid projects utilizing OGCE had integrated portal authentication with enterprise authentication. In 2005 the University of Virginia accomplished similar work in parallel.

Steps in Architecting a VO Collaboration Environment

Although the set of components needed to create the desired architecture had been identified, many open design issues remained. First steps involved installing and configuring the Shibboleth components and becoming familiar with them. Once familiar with the Shibboleth application interface, some applications useful for collaboration were examined and their authentication and authorization subsystems were re-engineering to match the Shibboleth interface requirements. After several iterations of the re-engineering process some common patterns emerged that provided information needed for further architectural decisions.

One of the patterns that emerged was that once there were a few applications interfacing with middleware it was possible to move between them without re-authenticating, and that observation led to the concept of a collaboration toolkit. Because Shibboleth was useable only with web-enabled applications a decision was made to treat the OGCE grid portal, with the grid behind it, as simply another web-enabled application to be included in the toolkit. Initial implementations were accomplished using pubcookie and UAB domain authentication to demonstrate feasibility. Once there were several middleware-enabled applications available it was straightforward to consider replacing pubcookie authentication with Shibboleth SSO; however, it was not straightforward to de-

termine where and how to store VO membership information. The next chapter describes the experimental methods in detail.

The VO collaboration environment was assigned the name *myVocs* for “my Virtual Organization Collaboration System” and is the first known attempt to add VO membership functionality to a Shibbolized environment. The prototype has captured the imaginations of both the Internet2 and Grid communities especially because it accomplishes a new approach to a well-known cross-domain scenario problem.

EXPERIMENTAL METHODS AND ANALYSIS

The Application / Middleware Interface: Mailing List Case Study

The mailing list management application (MLM) was selected as a case study for understanding how applications should interface with middleware. The MLM was chosen because of its long-time use for collaboration and also because both web-based and non-web-based interaction were provided in the same application. MLMs have traditionally assumed that list members have no organizational affiliation other than their membership in the list itself, and this type of "silo" assumption was precisely what would need to be changed when middleware is used. The case study work was assisted by members of MLIST, an Internet2 Working Group chaired by the author.

All members of MACE-MLIST were familiar with generic mailing list functions and developed two detailed models describing the interface of those functions with middleware. The general approach was to look for the MLM's use of internally stored data and consider its replacement with externally stored data. In addition, data created by the application was examined for potential use by other applications. Two MLM models were developed: the "Domain Model" represented non-ordered process flow and identified middleware interaction points (Phelps, 2004); the "Object Model" represented the hierarchical nature of the mailing list application and identified middleware interaction points (Gemmill, 2004). Both models identified similar points of middleware interaction. Additional model validation was provided by the developers of the Sympa MLM

(Aumont & Salaun, 2005), who found both models consistent with their application's architecture. The Sympa open source MLM is maintained by the Comité Réseau des Universités, an organization providing services to French universities. Sympa is widely used in France and other parts of Europe but was almost unknown in the US before 2005. Having identified points for middleware interaction, use of each external data store was categorized according to its associated AAA service: authentication, authorization, or accounting. Some additional external service possibilities were also identified, for example a "Filter Service" that could screen incoming messages and apply rules for removing messages before they even entered the MLM system.

Once the potential points for middleware interaction had been identified it was time to select an open source MLM package to modify. Sympa was a pleasant discovery as it already made extensive use of external data stores including LDAP and relational databases. The Sympa developers were interested in federation and in Shibboleth so a collaboration was initiated with the Sympa developers. Lessons learned from the case study were used to modify several open source applications. The specific applications were phpwiki (Klapp, Dairiki, Urban, & Wainstead, 2006), drupal (Kessels, 2006), and WEBinsta FM Manager (Feijen, Patial, & Poot, 2006). These tools were each web-based applications providing functions that are arguably essential for collaboration: joint content development, information and blog sharing, and file sharing.

Initially only authentication was addressed in the application modifications, replacing each application's internal login mechanism with an identifier provided by pubcookie. Although that approach may appear to be straightforward, this turned out to not be the case, an experience that provided an interesting introduction to the impact use of

middleware has on application design.

Re-Engineering Application Authentication

The objective in this phase was to replace internal authentication systems with identity provided by an external authentication service, and the Unix system environment served as a model for the desired distributed environment. In UNIX the user is authenticated by the login program, which assigns the appropriate user identifier. Once known, identity becomes part of the context for all subsequent processes. Those processes trust that the login application has sufficiently validated the requestor's identity and do not attempt to re-implement services supplied by the login program.

Authentication Should Establish Identity, and Only Identity

In a middleware rich environment, authentication services are distinct applications focused exclusively on securely validating a user's identity and providing that identity to trusted applications. Identity is simply a unique name that an Identity Provider houses in its central store; this identity will be referred to generically as NETID. During authentication the user presents a shared secret or cryptographic token to the authentication service and the service verifies the credential. Successful credential verification establishes the current user's ownership of a specific NETID. The NETID must be unique among identifiers stored by the IdP. Standardization of a schema housing the NETID is useful in a distributed environment so that each domain has a common definition for identifiers. While there is no global standard for identifier, Internet2 participants who had implemented the eduPerson schema did share a common NETID: the eduPersonPrincipalName (ePPN)

attribute serves this purpose. It is worth emphasizing that the act of authenticating and the identifier assigned as a result of that authentication are distinct. An identifier might be as persistent as an LDAP DN or as transient as a session ID and should be treated as one of possibly many attributes associated with the current session owner.

Identity is neither an Account nor an Account Name

While middleware-enabling authentication it became apparent that authentication and accounts were often confused with or substituted for each other in a manner that was not apparent until attempting to use external authentication. Authentication should never be considered identical to the concept of "having an account". Creating and naming system specific resources is a process distinct from establishing identity. Failure to distinguish identity from account causes problems such as those described below.

An account has two parts: (a) some system specific resources are allocated, and (b) the allocated resources are associated with a name used to reference these specific resources. Account creation is correctly described as the act of provisioning resources for a specific user to enable use of the system. For example, in order to use the email application a mailbox, really some disk space and an address, must be provisioned. This definition of account is more accurately termed system-specific identity. A key observation to make is that the system-specific identity can be different from the identifier. Accounting is the process of using the account name to track utilization of the allocated resources. Processes for mapping that local system to an actual person or organization may or may not be known to that system.

In today's distributed Internet environment with limited working middleware, ap-

plication developers typically design their application to provide its own identity management, authentication and account creation. Each system asks users to self-register (provide self-selected login name and password) or request an account (provide a login name and be assigned a password). An account for each new user is allocated and named, and typically the identifier is used for that purpose. The prevalence of this design blends the concepts of authentication, identity, and account so that they are nearly indistinguishable.

A rose is not a RoSe

An early observation was that naming conventions do not always substitute nicely for one another; the identifier provided by the middleware may violate an application's internal naming requirements. The wiki application, for example, insists on the use of a case sensitive mix of upper and lower case characters in a certain order. This convention allows the application to distinguish names, including page names and user identifiers, from content. An IdP may release the identity "rose" but the wiki would not recognize that string as a name unless it was in the form "RoSe". This was definitely a case where a "rose" is not a "RoSe!" Modifications to the wiki application included removing that particular naming requirement.

Application developers should expect that an IdP's naming conventions may conflict with their preferred account name scheme. Shibboleth provides an identity in the form of `ePPN@domain`: for example, `jgemmill@uab.edu`. This convention has the advantage of providing a name that is globally unique, but the form of that identifier is not suitable for use as an account name in many systems. Application developers should

therefore be rigorous in storing identity separately from account name, mapping from one to the other as needed. This approach is already used in grid software where a grid-mapfile is employed to map certificate DN to some local user account.

Email Address is Not an Identity

The first modifications made to Sympa for use with Shibboleth led to one of the most interesting and heated discussions held by the MLIST working group. The debate topics were who should be considered authoritative for email address and whether it was reasonable to use email address for identifier. MLM software had been in existence for twenty years and, as an application designed to deliver email, was not surprisingly built to consider email address as identifier and to equate email address, identity, and account. Shibboleth had the ability to provide scoped ePPN, email address, and other attributes stored in the enterprise directory; which of these attributes should be used to authorize access to the MLM?

While UAB decided to use `BlazerID@uab.edu` as a working email address, it was known that other institutions made no association of NETID with email address, meaning that `ePPN@domain` could not be assumed to be a working email address. Furthermore, Sympa and MLM's in general were known to use email address as identifier. The eduPerson schema includes an attribute named `mail` so it initially seemed reasonable to use that attribute as provided by Shibboleth in place of `ePPN@domain`. This choice appeared reasonable since it would provide a mapping from identity to another unique identifier that happened to coincide with any MLM's expectations regarding internal account names.

While testing this approach, however, certain participants failed to subscribe successfully to any Sympa lists because their enterprise email attribute was deliberately set to a non-working address. Some people argued that it was ridiculous to assume that an enterprise directory would not contain at least one working email address that could be found in the "mail" attribute. Others argued that they preferred to use a different email address for each list although they were always the same person. After spending a bit of time debating who was being more perverse the group came to a common understanding of these important issues. First, the semantics formed by the SAML XML metadata should not be disturbed; even if some attribute "looks like" a unique identifier (*i.e.*, is of the form `name@domain.edu`) it should never be used as the federation identity. The identity is not just the name instance, but also the metadata infrastructure behind it. Secondly, at least in the higher education community, the authoritative source for preferred email address is the user, not the enterprise. This observation served as an early reminder to consider the possibility that a single identity might need to be associated with multiple attribute authorities.

After this discussion modifications were made to the Sympa database; specifically, scoped ePPN as released by Shibboleth is captured and stored for use as an alternate primary key. In SAML parlance, the MLM was a resource provider member of InQueue that trusted the InQueue federation IdP's to provide a valid identifier. To use this new version of Sympa users were directed through Shibboleth processes to identify themselves at their home IdP and, upon returning to Sympa, were asked on their first visit to provide their preferred email address. Using the Sympa "front door" web page it was now possible to authenticate through the federation and gain transparent access to the applica-

tion. The Sympa developers adopted these changes which were released in Sympa version 5.2 in March 2006.

The Sympa developers were reluctant to further redesign their application since their implementation had been built using email address as their database primary key. Because of mixed use of email address as identifier, account, and mailbox it would not be trivial to now separate those functions. Some situations have been discovered in which this is an issue; one important case is when a user with established Shibboleth credentials arrives via web browser at any Sympa web page other than the "front door", for example a page associated with a specific mailing list. With credentials in place, the user should gain transparent access to the application; what actually happens is that the user is asked to log in again because of the application's dependence on account name (email address) for identity, which is not available in the Shibboleth security context.

In summary, applications were highly likely to assume that identity and account name were identical, even storing both in the same location. For middleware enabled applications, this design should not be expected to work. Developers should therefore be vigilant in separating authentication verification identifier from account definition.

Re-Engineering Application Authorization

Automated Account Provisioning

If identity is already known it should be possible to automatically provision an account for authorized users. This approach enables self-managed, transparent access to applications. Use of both group and role information was explored in this context, an experience that led to the concept of using Sympa as a VO membership management tool.

Each mailing list was considered to be a VO; as members subscribe or leave using usual MLM procedures they are considered to be joining or leaving the VO. The mailing list name becomes a group identifier associated with each member. Using the modified Sympa made it possible for one identity to belong to many groups using their preferred email address for each group. While Sympa itself did not make much use of this advantage it was to be important in leveraging Sympa-defined group memberships. Sympa also defined a small number of roles per VO; these are the usual MLM roles of list administrator, list moderator, and list member. Since these roles were already stored in Sympa's internal relational database in addition to groups it seemed reasonable to use these definitions to begin exploring use of VO membership and role information.

Once a user is able to authenticate successfully at their IdP and if Shibboleth can transport both identifier and group membership to an application, the application should be able to determine if group members are authorized to have accounts and if so, proceed to provision an account for the identifier presented. Today this step is often done manually because of lack of identity management, but by using middleware the identifier, attributes, and permission can be available without having to ask the user to present them again. Applications could add and remove application-specific accounts as needed based on information maintained elsewhere listing active VO members. The phpwiki application was the first application revised to accomplish automated account provisioning.

Once the naming format issues discussed previously were addressed, it was a straightforward process to protect the application with Shibboleth, providing an access control rule that allowed only members of group X to access the site. The phpwiki application was modified so that if a member of group X did appear, the next step was to

check to see if this identity was already assigned an internal account. If not, the application stored the scoped ePPN as account name; because of the event context it was not necessary to use the usual wiki naming convention to identify ePPN as a name. What the user experiences is that the first time they arrive at the wiki the application is already displaying their name and they are able to begin editing pages. This particular case was relatively simple in that group membership was the only requirement for any type of access. The other applications each had some internal concept of role and the objective in re-engineering these applications was to provision an account that was appropriate for their VO role.

Authorization

Authorization is extremely complex in a distributed system. The "big picture" is that a requested action, policy associated with that action, and action owner's attributes must all be located and combined to produce an access control decision; the point of decision is called the Policy Decision Point (PDP). The application serves as a Policy Enforcement Point (PEP), enforcing that decision. Today that decision process is usually internal to an application or system; but in a distributed environment any part of the process, or even the entire process, may be external to the application. Figure 6 illustrates the many components of the authorization model. The left hand side of Figure 6 represents an overview of the entire authorization process, showing action, policy, and attribute inputs to the decision process. Note also that the PDP may be a service that is separate from the PEP. The right hand side shows detailed inputs for each component and possible sources for the required information. The complexity suggested in this model demon

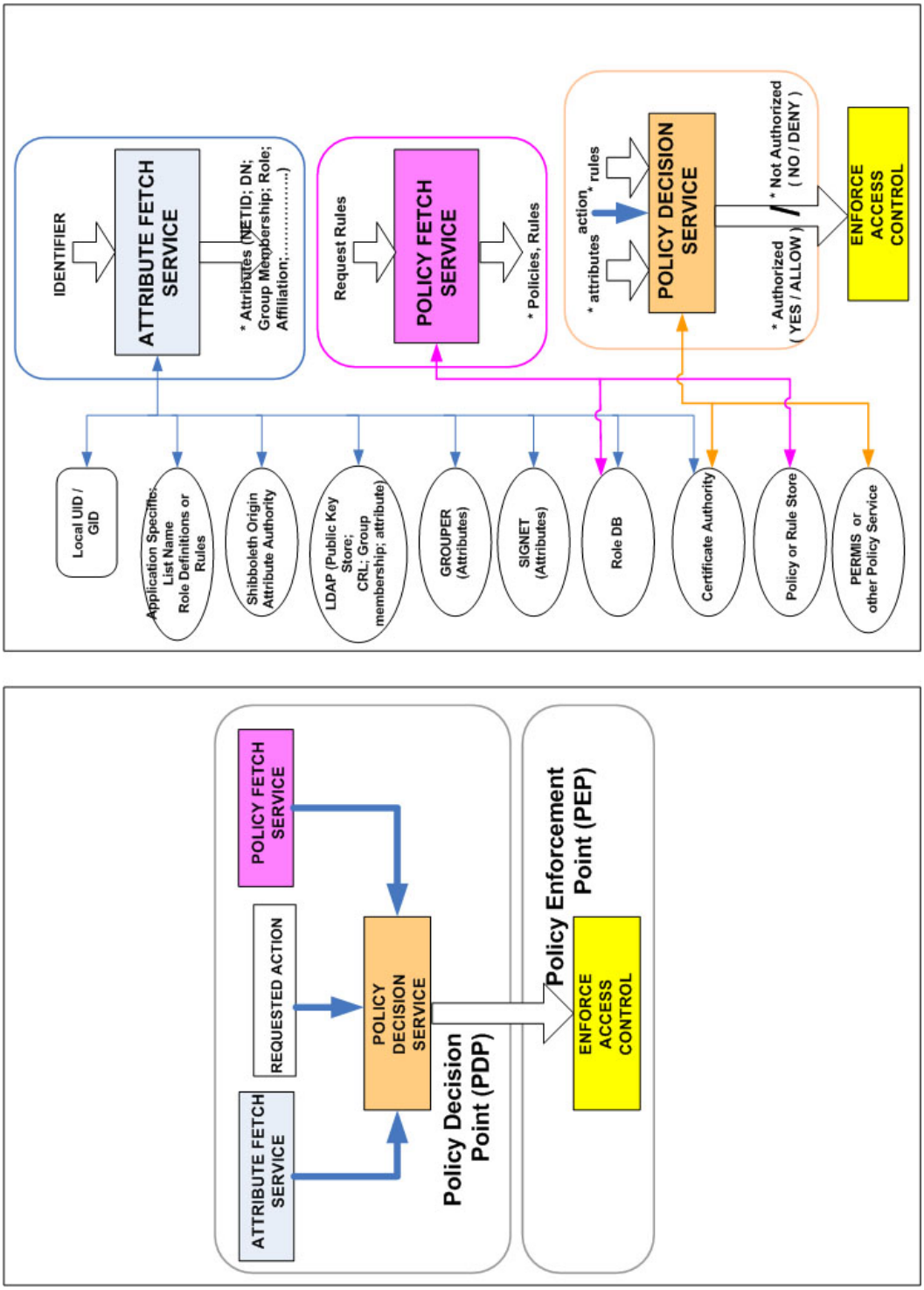

Figure 6. Authorization in a Distributed Environment is Complex

strates many of the challenges in distributed computing. As an attribute transport service Shibboleth does not solve the entire authorization problem. Shibboleth is a vehicle for locating a user's IdP and transporting attributes to some decision process.

Even restricting the problem space to attributes only results in a large problem space. Once possible external sources for attributes are considered one can quickly compile an alarmingly numerous set of potential locations. An example of this dilemma is illustrated in Figure 7. Shibboleth can indeed transport attributes, but is the IdP authoritative for all the attributes needed, and if not where might those other attributes be located? Recall that the Sympa-related discussion regarding who was authoritative for email address was one case already discovered where the IdP was not the authority. If the attribute sources are also distributed, how would Shibboleth locate them? Some applications do use a pluggable-switch model (Samar & Lai, 1997), employing configuration files to enumerate the possible sources and associated query filters. The application tries each source in sequence until receiving a response or exhausting all possibilities. This approach works well but requires the resource provider to enumerate all possible sources in advance and may require that the application is issued special credentials for each source that might be queried. A second approach is one favored by the Internet2 middleware architects (MACE) and involves all authorities storing attributes at the IdP. That certainly makes locating attributes easier, but does require the cooperation of every enterprise directory manager along with some type of rights delegation and group assignment applications. Based on the author's experience it seemed unlikely that enterprise directory managers would permit unaffiliated authorities to write into their attribute stores. These observations resulted in a decision that some type of attribute aggregating or attribute fetch

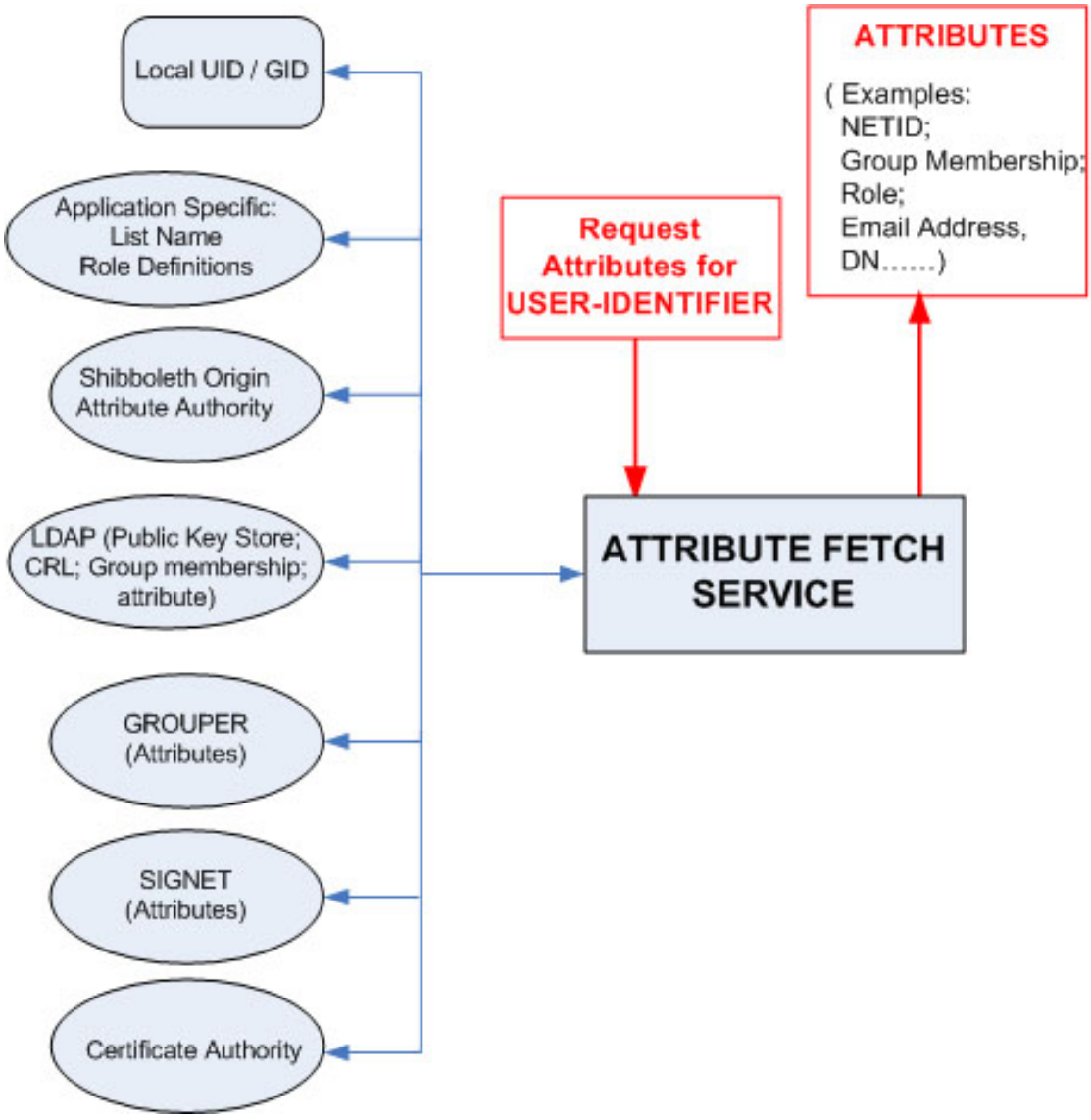

Figure 7. Potential for Multiple Attribute Stores Demonstrates Need for an Attribute Aggregation or Fetch Service

service would be needed. Regardless of where the attributes were stored, it seemed best from the application's perspective to consult what appeared to be a single source for all attributes needed. This question was especially important in determining who is the authority for VO membership and role attributes and where to store those attributes.

Authorization Models Found in the Wild

As might be expected, some applications were easier to modify than others and a factor common to every application was its adherence to or departure from modular programming principles. Well designed, modular software was always preferable to code written with the organization of a spaghetti pile. As applications were modified a pattern of common system authorization models in use emerged that are described below. These models were useful in analyzing how to best middleware-enable each type of application.

Account-Role Model

The Account-Role Model associates a set of permitted actions with specific account names. For example, the Mailman mailing list software (Warsaw, Hylton, & Kikuchi, 2006) requires a special account in order to use list administrator functions such as "moderate." An email address that is a list subscriber cannot make use of any administrative functions without logging out of that account and logging in using an administrative account. When per-account assignments become cumbersome, applications typically move to a scheme that assigns actions to roles and roles to accounts. The Mambo content management software (Mambo Foundation, 2006), for example, defines five roles: administrator, editor, publisher, content developer, and anonymous. These roles are strictly

hierarchical, so that each role includes capabilities of all roles below, and accounts can be assigned only one role. With this approach, changing a person's level of authorization involves assigning a different role rather than re-assigning accounts.

Object-Owner Authorization Model

The Object-Owner authorization model defines internal objects and assigns objects to owners. Owners (accounts) can then be authorized based on object ownership rather than type of account. The Sympa software, for example, authorizes read/write permissions to each owner-created object. Therefore, a single owner (account) can create a list-membership object (an individual subscription) or a list-object (a new mailing list) and exert appropriate control over each object. Role or group definitions can more easily be introduced into this approach, assigning accounts to one or more groups and assigning the group as object owner.

Hybrid Model

The Hybrid Model combines Account-Role and Object-Owner models by separately defining accounts, actions and groups and then combining authorization descriptions flexibly. For example, with accounts `joeuser` and `moeadmin`, action roles `read` and `write`, and `moeadmin` assigned as a member of the `admin` group it is possible to describe permissions such as "any account may read subscriber_list_A" and "members of the admin group may write subscriber_list_A", resulting in `moeuser` allowed to do both. The Unix UID/GID with permissions flags authorization system is an example of this model. When an application's authorization model promotes actions and data abstractions

to the same level, the authorization system consists of enforcing rules based on requested action and owner attributes without having to examine the object's contents.

Application Session State

A state diagram was developed as an aid for application developers and also for use in re-engineering existing applications (Figure 8). This diagram presents a continuum of middleware-enabled application state, proceeding from time of anonymous arrival at a resource provider through authentication, authorization, and account provisioning. Circles labeled A-G at the top of the diagram indicate an application's possible start states; progressing left to right from locations A to G represents progression towards a fully middleware enabled application. An application beginning at A provides its own authentication, attributes, rules, decisions, and accounts. An application beginning at G trusts its environment to provide that context and focuses on the application's special function. Dark squares labeled α through δ represent information that can be provided to an application from remote data stores or external processes that need not be replicated by the application.

Ideally the application developer will someday provide total flexibility for system integrators by building in configuration options for any of the states represented in Figure 8, positions A-G. At a minimum the developer needs to give some thought to which system environments she intends to support in addition to "silo" and identify her perception of those points of integration. It is then left as an exercise for the system integrator to assemble the desired environment.

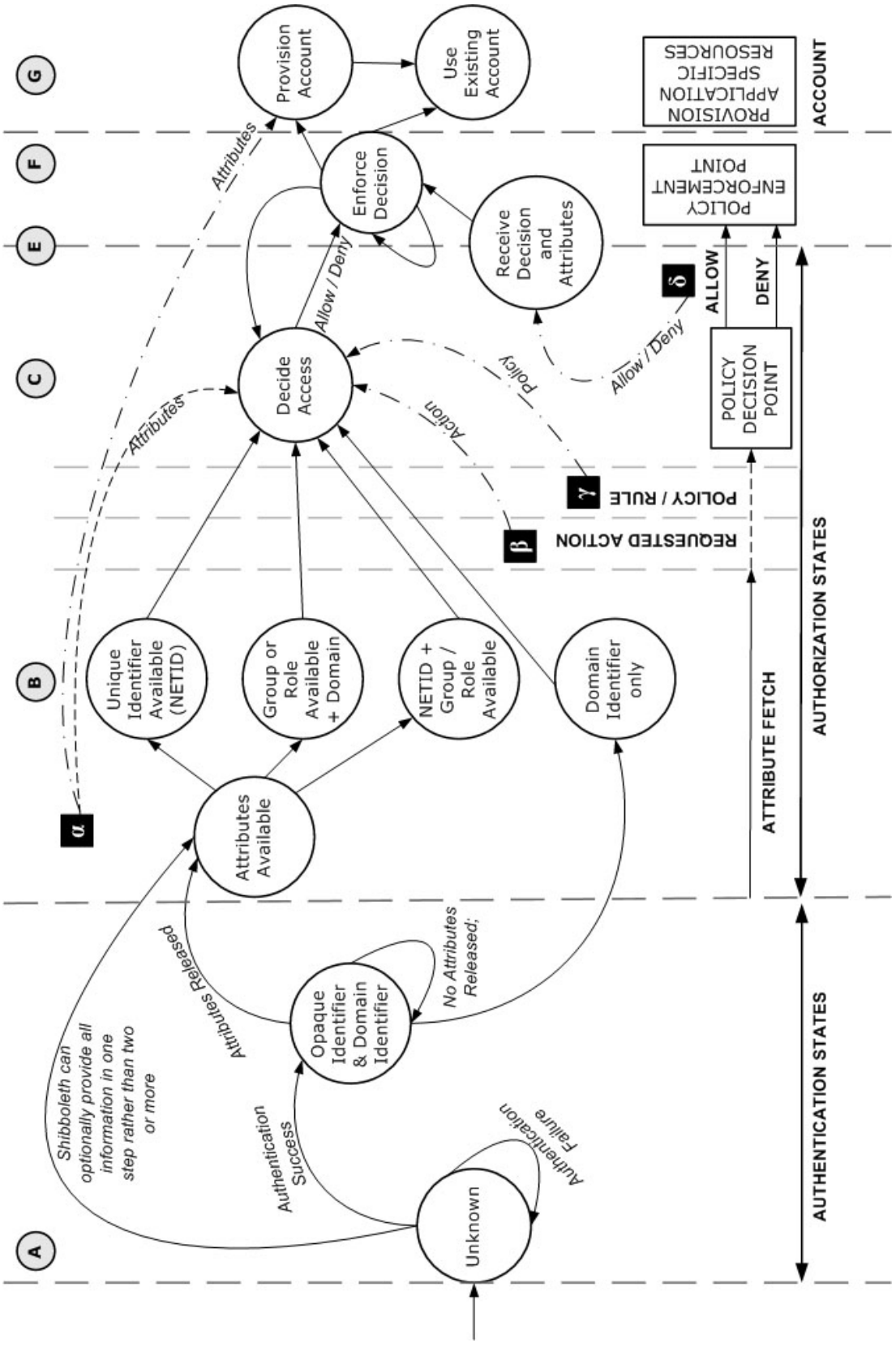

Figure 8. Application Session State Diagram

The Application as an External Store

Middleware introduces the entirely new consideration of the application as iteself the authoritative source of information needed by other applications. A mailing list application contains globally unique names for each list (*i.e.* the list's email address) and information about who is a member of that list. That information could be useful to other applications; by way of example, suppose the people represented by the mailing list membership wanted to share some other collaboration tool; it would be quite useful to query the mailing list application, obtain each subscriber's identifier, and provision those identities with appropriate access. Remembering to consider the application as a provider of information as well as a consumer was one of the more difficult new concepts to keep in mind.

Grid Web Logon Using Pubcookie

Issuing and Managing Grid Certificates

The Globus grid security infrastructure (GSI) uses digital certificates to identify users; one outstanding issue for grids has been the question of who provides these certificates. It is a common practice for each Virtual Organization (VO) to issue certificates to their respective community of users; examples of this approach are the European DataGrid (CERN, 2004) and the Teragrid. This works well enough when grid computing is used by a handful of people in a few high-visibility projects, but the approach is in general not scalable for many reasons. Because no global PKI exists, a Certificate Authority (CA) used to sign certificates may not be unknown and therefore unverifiable at a remote location; therefore whatever certificates are issued are likely to be of use only at re-

sources owned by or closely associated with the CA owners.

The "unknown CA" problem can be fixed by providing each resource with a root bundle used to construct a validation path back to the issuing root CA. Unfortunately, this approach is not scalable because each grid node in the global grid would need to download and install this bundle for every CA, an $\Omega(n^2)$ problem. If certificates are issued on a VO-specific basis, one person may belong to many VO's and own many digital certificates. The user then becomes responsible for knowing which certificate to use for each action requested, a substantial usage and key management hurdle. As CAs proliferate their value as a trusted third party could diminish; a CA's signature is only as trustworthy as the organization who manages the CA. Resource owners do not want to download CA bundles without first being familiar with the CA owner's trustworthiness and identity practices.

As an NMI Testbed participant, UAB IT Academic Computing was introduced to directory-enabled applications, software tools for policy based management, single sign on, federated identity and Globus. The UAB IT department had already established a well-known campus identifier (the BlazerID) to use for authenticating to many campus services. If this identity management process could also be leveraged for grid computing, there could be significant advantages. From a policy perspective, use of the campus identifier means that the Human Resources Division and University Registrar determine who is an active member of the UAB community so that the grid system administrator does not have to make that decision. The university IT department provided many useful BlazerID support services to handle BlazerID creation, password resets, and BlazerID deactivation if necessary. Deferring authentication to the central service would allow the

grid system administrator to ignore the entire issue of assigning and supporting user-names and passwords, a process that can consume considerable time better spent concentrating on grid administration. Finally, leveraging the campus authentication service would provide access to other attributes stored in eduPerson format in the campus LDAP directory. This information could be useful for various authorization decisions in the grid.

The challenge in leveraging the university's existing authentication service for UABgrid access was converting the BlazerID login/password mechanism to a digital certificate suitable for use in GSI. The UABgrid Certificate Authority (UABgridCA) solves that problem by functioning as a gateway from username/password-based identity to digital certificate-based identity and also hides all the details of certificate management from end-users. UABgridCA is constructed from the following open source components: PHPki (Roadcap, 2005), Pubcookie (University of Washington, 2005), MyProxy (Novotny et al., 2001; Basney, Humphrey, & Welch, 2005), and OGCE (OGCE Consortium, 2006). Customizations were made that leverage a web-enabled single sign on solution to provide web-enabled grid logon (Robinson et al., 2005; Robinson, Gemmill, & Bangalore, 2005). The UABgridCA architecture and steps in the logon process are reviewed below.

UABgrid Registration Process

Access to UABgrid begins with a one-time registration process as summarized in Figure 9.

1 . At the location labeled "START REGISTRATION" the user employs a standard web browser and attempts to access the UABgrid registration web page.

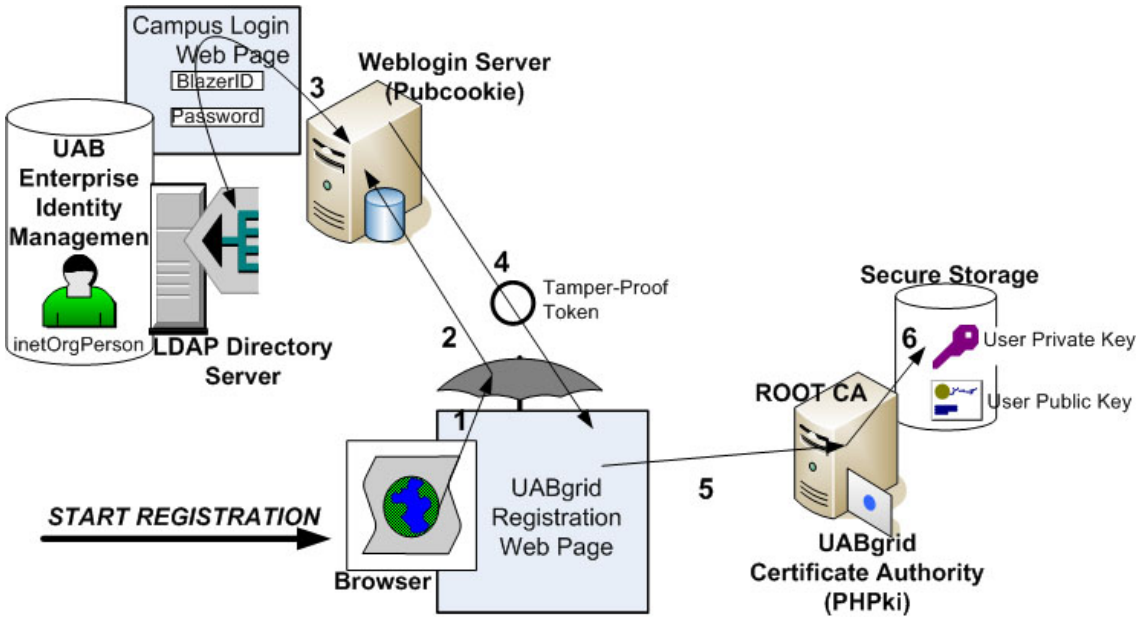

Figure 9. UABgrid registration: umbrellas represent web page protection provided by an add-in the mod_pubcookie web server module. Individual services are illustrated by separate server icons but do not necessarily run on physically separate servers.

2 . The browser is redirected to the weblogin server.

3 . If the user does not have a valid login credential, she is presented with the familiar campus login page. The enterprise identity management system is leveraged for authentication.

4 . Upon successful authentication, the weblogin server provides the browser with a tamper-proof application token and redirects the browser back to the registration web page. The token includes the user's BlazerID, a campus-wide unique identifier.

5 . If the user has a valid token, the PHPki key service creates and digitally signs an attribute. The UABgridCA, the root CA for UABgrid issued certificates, is also used to sign these certificates.

6 . The newly created key pair is stored for the user in a secure storage location.

At the conclusion of this registration process the user has been assigned a hidden public-private key pair with a two year lifetime. Successful BlazerID authentication, represented in the tamper-proof token, authorizes the user to have the X.509 key pair issued. The BlazerID contained within the token is assigned to the certificate's name attribute.

UABgrid logon process

Once this one-time registration step has been completed, users are ready to use UABgrid resources. The UABgrid logon process is how users would usually log into the grid and is summarized in Figure 10.

A. At the location labeled "GRID LOGIN START" the user employs a standard web browser and attempts to access the UABgrid home page.

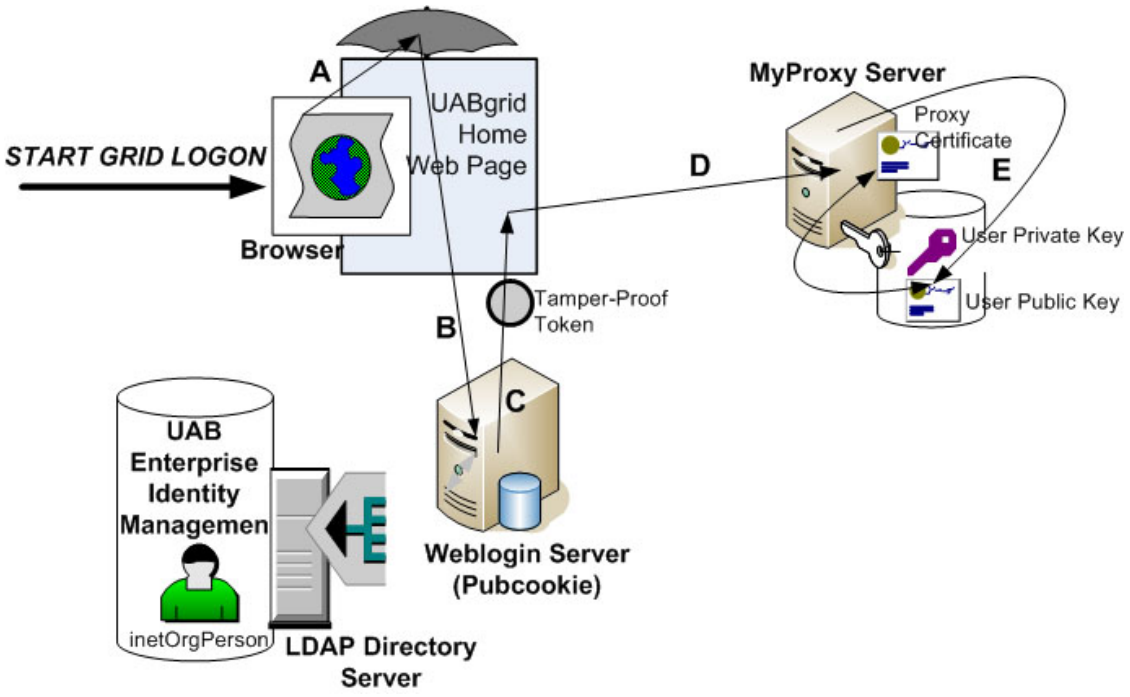

Figure 10. UABgrid logon process.

B. The browser is redirected to the weblogin server. If the user already has a valid login credential she does not need to re-authenticate; otherwise, the user would repeat step 3 described above.

C. The weblogin server provides the browser with a tamper-proof application token, containing the user's BlazerID, and redirects the browser back to the UABgrid home page.

D. The UABgrid home page is a web interface for the myproxy-init command.

E. If the user has a valid token the current user's certificate and key are accessed from the UABgridCA secure repository so that a proxy certificate can be created for the user.

By protecting this application with weblogin the current user's identity is known and trusted, allowing the proxy initialization to occur on their behalf. The proxy certificate is stored using BlazerID for username and a common shared secret as password.

Integrating OGCE Login

The user is now ready to access grid resources. OGCE is used in UABgrid to provide a web-based user interface for job submission. OGCE is a grid computing portal providing a single point of access to a grid system. As distributed, OGCE provides username/password authentication into an internal database; that same usename/password pair is then used to fetch the grid proxy certificate which completes initialization of the grid client environment. The user's proxy certificate is needed to access grid resources. The native OGCE login was replaced with weblogin authentication, as illustrated in Figure 11.

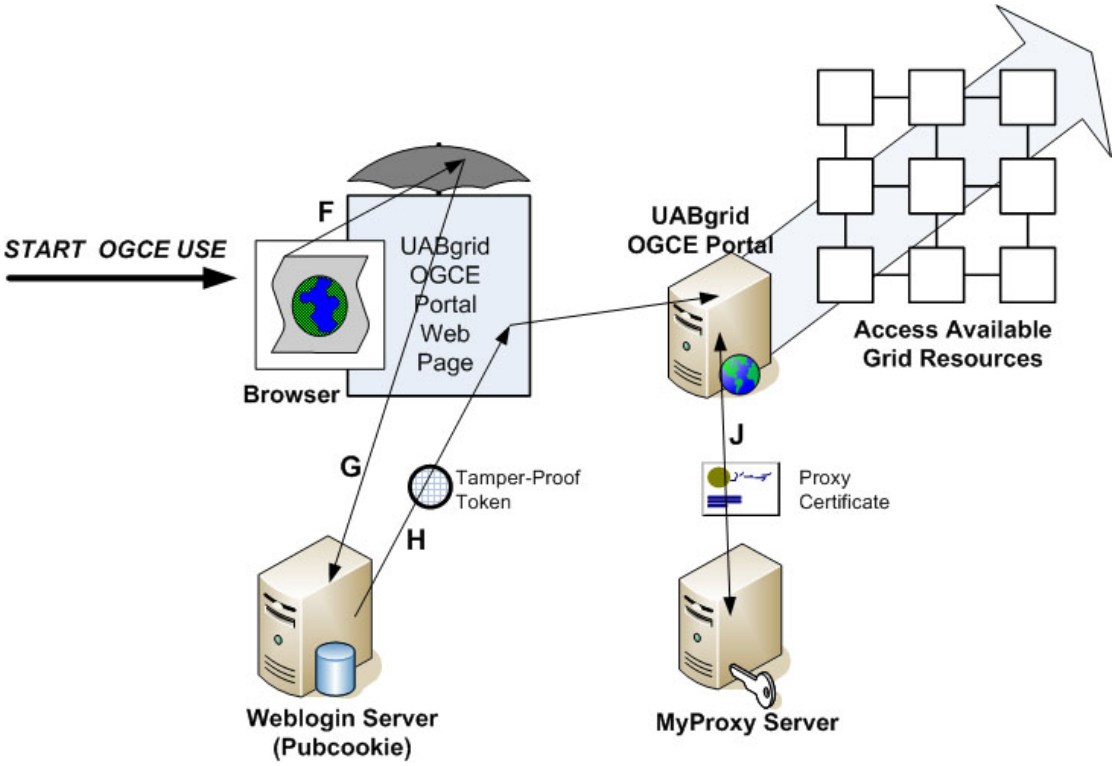

*Figure 11.*Weblogin Server Integration with OGCE portal

F. Having completed the MyProxy initialization in step E above, the user is redirected to the OGCE portal web page. (location labeled “START OGCE USE”).

G. The user is redirected to the weblogin server. The user should have a valid login credential at this point having just visited step B above.

H. Having a valid login credential, the weblogin server provides a tamper-proof application token, containing the user’s BlazerID, and the user can now access the OGCE portal. The user’s BlazerID is also available to OGCE. The OGCE MyProxy portlet knows the shared password; using BlazerID and that password the user’s proxy certificate can be retrieved from the MyProxy repository and is held by OGCE for the rest of that user’s session.

The steps described in Figures 9, 10, and 11 are quite complex but use of weblogin makes most steps totally transparent to the end user. What the user experiences is the familiar BlazerID/password page followed by automatic appearance of the OGCE portal page. The user can know nothing at all about key management, although the implemented solution permits a user to handle their own key if they want to do so. The mod_pubcookie web server module and associated redirects forces the user to adhere to the appropriate event sequence. When this work was submitted for publication it was discovered that a similar approach had been implemented in the same timeframe at UVa (Martin, Basney, & Humphrey, 2005).

Managing Grid User Accounts and Access Control

A user with a valid grid certificate may request access to any grid resource. Whether the requested access is allowed or denied is a matter of access control which

Globus manages with the grid-mapfile. A grid mapfile entry maps a GSI Certificate DN to a local account name. For example, the grid mapfile entry:

`"O=University of Alabama at Birmingham/OU=UABGrid/CN=jsmith" imauser`

specifies that an entity presenting an X.509 certificate containing this DN should be assigned use of a local account named `imauser`. Of course, an account named `imauser` must be provisioned in order for this to work. Since there are approximately 69,000 BlazerIDs contained in the UAB enterprise directory while there are currently at most a few hundred grid users, it was not be practical to create accounts for each of these BlazerIDs at every system.

Creative use of grid mapfiles can provide flexible and even automated account management; several strategies for creating and distributing appropriate grid mapfiles have been considered. A simple global strategy adds a new entry to a master grid-mapfile as each new user completes their one-time registration, mapping each person to local account name `griduser`. Each resource could be provisioned with one `griduser` account and automated scripting could be used at each UABgrid resource to frequently pull a copy of the updated grid mapfile. All registered UABgrid users would at a minimum be able to make use of the one `griduser` at each resource; this was the initial grid account deployment for systems owned by IT Academic Computing because (a) it was a simple way to get started, and (b) ITAC's usage policy provides equal access for any member of the UAB community. ITAC also implements a global file system namespace (of the form `/home/BlazerID`) to provide users with permanent storage for their data.

Future plans for UABgrid call for use of posixAccount[7] as a more elegant solution that utilizes an LDAP directory as a general Network Information Service (NIS) along the lines described by Howard (Howard, 1998). A UABgrid LDAP account directory would contain a posixAccount, UABgrid specific attributes such as DN, and relevant attributes from the enterprise directory. This strategy permits the greatest flexibility for implementing authorization and may make it possible to replace the grid-mapfile mechanism altogether.

Bridged CAs for Scalable, Cross-Domain Grid Services

The goal of grid computing is to establish a computational framework capable of crossing organizational boundaries. A critical step in achieving this goal is availability of user identities that are both useful (*i.e.,* unique) and verifiable across organizational boundaries. Grid projects such as Teragrid address this issue by configuring Teragrid resources to accept certificates signed by any one of nine select CAs. One reason for limiting the number of CAs involved is that each resource must be pre-populated with certificate root bundles for each of the nine CAs so that certificates issued by these CAs can be validated. The current Teragrid model introduces many scalability issues: applications to receive a digital certificate from an approved CA must be requested and approved manually.

Utilizing existing enterprise identity management systems provides a reliable means for reliably identifying large numbers of people, and has many important adminis-

[7] posixAccount is the name of a standardized schema that defines elements of a user account object. It is included in the OpenLDAP distribution and is in use by Novell, Microsoft, and other LDAP implementors.

trative advantages such as relying on that enterprise to have reviewed government-issued identity documents, accurate maintenance of enterprise affiliation, and some existing local support infrastructure that can be leveraged for grid support. While many-to-many root bundle exchanges among enterprises is not scalable, bridged certificate authorities have recently been demonstrated to provide a scalable trust mechanism. The concept of bridged certificate authorities was first implemented by Alterman (Alterman, Weiser, Rea, & Blanchard, 2005) and was first deployed for use in grids by Jokl (Jokl, 2005). Scalability is achieved by limiting the trust configuration of each local CA to a single cross-certification with a bridge CA, Participating CAs inherit all the cross-organizational trusts defined by the bridge. Therefore, grid nodes are required to be aware of only two CAs, their own and the bridge.

The University of Virginia established a grid CA as part of the NMI Testbed Grid and UABgridCA was the first CA to cross-certify with this testbed bridge CA and demonstrate use of UABgrid issued certificates to execute jobs on resources operated by Texas Advanced Computing Center (TACC) and the Center for Computation & Technology at Louisiana State University. This effort contributed to the architecting of a scalable grid trust fabric and the project has grown under the name SURAGrid. Documentation of the cross-certification process and related Globus configuration is available in Jokl. The bridge CA approach carries the added advantage of being well aligned with current efforts within the US higher education IT community to establish a Higher Education Bridge Certificate Authority (HEBCA) (Educause, 2005).

A Virtual Organization Service Center Using Shibboleth

Shibboleth provides many advantages over the grid's end-entity PKI approach. First, Shibboleth provides many user attributes in addition to identifier. One may start thinking about putting attributes into user certificates, but unfortunately there are no standards for naming or storing information in this manner. If all attributes are stored in a publicly readable certificate, an additional shortcoming is lack of privacy. Secondly, Shibboleth solves the IdP discovery problem; SAML standards provide a mechanism for a service provider to discover and communicate directly with an identity provider over a secure channel, lending a higher level of trust to the information obtained. Shibboleth also provides finely grained attribute release control so that only the attributes needed are sent. For licensed library materials, for example, institutional affiliation is all that is needed to authorize access; identity is not needed and does not need to be provided. This feature is important for preserving privacy.

Like pubcookie, the Shibboleth implementation employs a webserver module to securely obtain identity information make that identity available in a web application's environment. Shibboleth is also capable of providing additional user attributes known to the identity provider. As mentioned previously, one approach to attribute aggregation is to gather all attributes at the IdP. Another approach is to gather the attributes elsewhere which introduces the problem of how to associate an identity with these attributes when identity is stored in a separate repository. There were many reasons to consider the VO authoritative for designating membership and VO role. Another desirable goal was to allow VO's to select their own application suite; if many collaborative tools were middleware enabled it could be possible for VO's to build a customized environment based on

their own requirements. This approach to VO support was first described by Gemmill (Gemmill, Robinson, & Shealy, 2003).

Sympa MLM as a Prototype Membership Management Tool

Prior to selecting Sympa as a prototype VO membership management tool, solutions available in the grid community were examined. The closest likely candidate was the VOMS authorization system that provided group membership information by placing attributes into a specialized "pseudo certificate." VOMS had some notable shortcomings including (a) requiring its own central repository and (b) a limitation of only one VO supported per VOMS instance. A great deal of work would have been required to modify VOMS for compatibility with webISO, Shibboleth, and multiple VO capability. Use of Sympa included the benefit of its relational database backend, unusual for a mailing list package. This non-proprietary storage was not only accessible, it also happened to be one of the backend data stores supported by Shibboleth.

Once the components were selected it was time to detail their integration. Identity was to be provided by federated IdPs. Current Shibboleth implementations require that service providers are configured to redirect browsers back to a single identity provider: that provider can be hard-coded into the configuration or the WAYF service can be substituted. The membership and role information were stored at the Sympa service and the challenge was how to integrate that information into the path of redirects. To fit the Shibboleth architecture, the Sympa MLM was configured as a service provider pointing to the InQueue WAYF in order to obtain identity information from the distributed IdPs. The middleware-enabled collaboration applications needed both identity and VO-related at-

tributes in order to provision accounts automatically and control access appropriately. What was needed to complete the architecture was a service that could aggregate attributes from the IdP's who were authoritative for identity and the VO's who were authoritative for VO-related attributes. As mentioned earlier an architecture that depended on enterprises allowing strangers to write attributes into that enterprises directory was a path filled with many, many hurdles. It also seemed right to design the architecture so that VO's were asserting attributes for which they were authoritative. Therefore, aggregating attributes at the VO was a logical choice.

Using Sympa as the VO membership management service, scoped ePPN was already being stored inside Sympa. While the service obtained scoped ePPN as a service provider, its architecture might allow the same service to play the role of identity provider for VO applications. VO service providers would need to trust the membership management service to (a) provide correct group membership and role information and (b) assert the user's identity using transitive trust. In other words, the VO service center would obtain the user's identity via Shibboleth and then assert that identity along with the VO-related attributes to the collaborative applications. Each application would be able to transparently recognize the identity of VO members using Shibboleth's web single sign on and provide access to and control of application content based on their VO roles. This VO Service Provider architecture provided an alternative to bridged CAs for achieving federated identity.

myVocs as a Virtual Organization Service Provider

The collaboration system environment prototype developed along these lines has been named myVocs ("my Virtual Organization Collaboration Service") and it is a VO service center. The myVocs service constructs an environment in which a user can join a virtual organization using their enterprise authentication service and enterprise identifier. After joining, their enterprise identifier becomes an attribute stored at the VO and the VO now has enough information to play the role of Shibboleth IdP, responding to queries with both identity and VO attributes. Recognizing that the VO service is not authoritative for identity, scoped ePPN and other enterprise attributes are refreshed regularly and identity is therefore cached at the VO service for short periods of time. The process of joining a VO is illustrated in Figure 12.

A. A browser attempts to access the VO Service web site, whose privileged functions are protected with the usual Shibboleth access control mechanism described in the previous chapter; having no security context the browser is redirected to the WAYF service.
B. The user selects their home institution from the WAYF and is redirected to their home IdP.
C. Lacking a security context, the user is asked to authenticate at their IdP.
D. The browser is redirected back to the VO Service site, now carrying user's scoped ePPN.
E. Now that the user has a valid NETID (ePPN) they are permitted through the access control mechanism and are eligible to use the MLM's JOIN and CREATE functions.

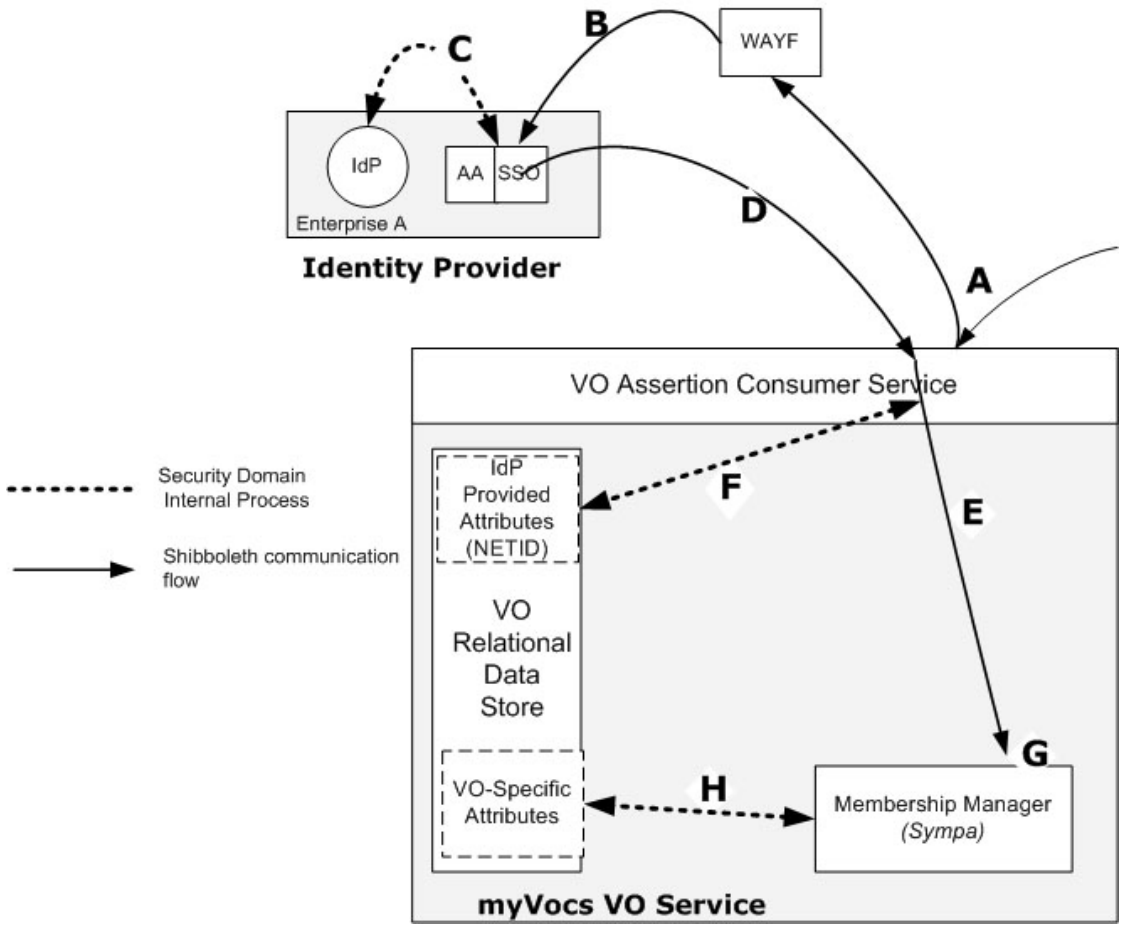

Figure 12. Joining a Virtual Organization

F. The Assertion Consumer Service examines the VO's database to see if this particular NETID is already registered, i.e. a database record exists for this user already. If the answer is yes, no action is taken. If the answer is no, a new database record is provisioned for the new NETID.

G. The registered user can create new VO's or join existing VO's using the corresponding Sympa functions.

H. New VO's and changes in VO membership for this NETID are recorded in the VO database.

myVocs as a Virtual Organization Identity Provider

The myVocs service is a service provider with respect to the federated IdP's, and is also an identity provider with respect to the resources that are to be used by the various VO's serviced. Having the VO service play both roles makes it possible to insert the service into the browser redirect path every time a VO resource is accessed, a situation made possible by transitive trust of VO service center. VO service providers rely on the VO IdP role to provide attributes for authorization at the resource; the VO IdP plays the role of Service Provider and relies on distributed IdP's to authenticate and pass enterprise attributes to the VO. The Service Provider must trust the VO IdP to handle NETIDs correctly, but this is a level of trust equivalent to trusting any IdP: reliability will be judged on reputation and documented procedures in use. Figure 13 illustrates this use case.

A. A user attempts to access a web resource, in this case a private wiki application. The VO Service Provider (myVocs) has be been configured to redirect the user to the VO IdP, also called the VO Single Sign On Service (SSO).

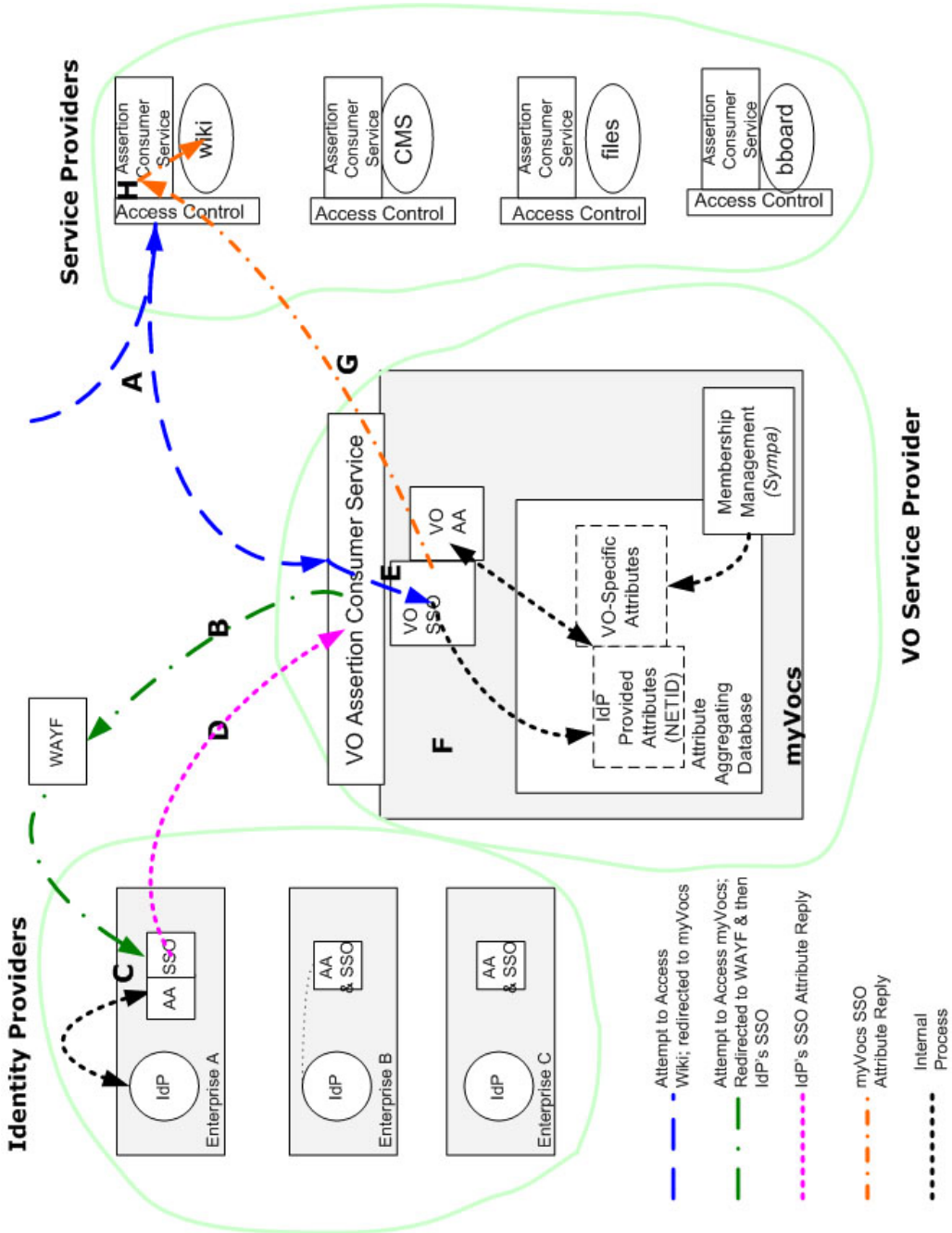

Figure 13. The myVocs Architecture

B. The Single Sign On Service is protected by a SP Target Resource Access Control; therefore, access to the SSO is prohibited until federation credentials are presented. If the browser has no security context it is redirected to the federation's Where Are You From? (WAYF) service so that the user can select their home institution. The WAYF redirects the browser to the user's selected home SSO.

C. At the home SSO the user is to authenticate if she has no security context. The SSO configuration describes which user attributes may be released.

D. The user's browser now carries the security context provided by the IdP and these are delivered to the myVocs Assertion Consumer Service.

E. The security context now satisfies the Access Control mechanism and the browser can continue on through to the protected resource, which in this case happens to be the myVocs SSO service.

F. If there is an existing database entry for this NETID, IdP specific attributes can be refreshed; if there is no database entry a new database record can be provisioned for this user.

G. The VO SSO consults its configuration file to determine what attributes are to be made available to the requesting service provider; these attributes are drawn from the VO's attribute store which has aggregated IdP specific attributes as well as VO specific attributes. The user's browser is directed back to the SP.

H. When the requested user attributes are available at the SP they are examined by the SP's Assertion Consumer Service; if the access control requirements are met the user's browser is permitted to access the protected resource. In this case, the

protected resource is the wiki. If the wiki has been modified so that it is middleware enabled, the application can obtain owner and group information from its environment and provide transparent access to the application. In the event that this user is unknown, application-specific resources such as a database record can be automatically provisioned by the application without any separate registration step.

The process flow described above works identically for any of the participating IdPs, with the exception that the user makes a different selection at the WAYF resulting in redirection to a different IdP. In a parallel fashion, each application protected by a Shibboleth module configured as a VO SP is a member of a set of federated applications shared by members of a specific VO. The applications do not need to run on the same server, or even be managed in the same domain. Using Shibboleth in this way provides a mechanism for sharing identity across applications and for customizing environments for different users; that functionality is usually associated with a portal architecture. This architecture requires no portal, which is quite novel.

VO Roles and Account Provisioning

Flexible delegation of authority and role assignment are complex issues. Fortunately, some tools have been developed in the NMI that provide these services. The long term architecture for myVocs includes integration of these tools with the myVocs database. Since the larger problem was being addressed by other areas of NMI, the initial myVocs implementation explored use of a small set of roles, leveraging role information that is native to an MLM. Sympa assigns roles such as list administrator, list moderator,

and list member; since these roles are stored in the Sympa database and were associated with NETID because of the modifications made, it seemed reasonable to use this small set of roles as a prototype. An earlier section of this chapter discussed authorization models found in use in current applications, and part of the re-engineering process required mapping of the VO role to the application's account model. Therefore, the VO list administrator was assigned ownership of each application used by the VO; all VO list members were assigned application-specific member accounts; and VO list moderators might be assigned to an application's editor role if one existed.

Each application had some different expectations about its own account structure and some thought was needed regarding how to best map the VO defined roles onto the application. For example, the drupal CMS application assigns the administrator/owner role to the first account created. The application allows additional persons with administrative rights, and a decision was made to equate the VO moderator role with drupal administrator. In contrast, the phpwiki application does not have any roles at all except member.

The applications explored each had some type of login function; some offer a separate registration function and others just recognize that the person logging in has no local account yet. Part of middleware enabling these applications meant removing or disabling any registration function and making a decision about how to handle the login function. For a private wiki, for example, the login function was hidden entirely. If instead the application was meant to be publicly readable but writeable only by members, implementors might chose a "login" or "enter" function may be needed. Some thought was required about which directories and files needed to be protected by Shibboleth and

which could be left public. Other application scenarios can occur: the Mambo application, for example, has both a “front end” and “back end” user interface, where the front end is the view of the final web site and the back end is an administrative interface supporting site administration, work flow, and content approval. Mambo is designed so that an administrator or publisher role is always presented with the back end; the user must change roles in order to view the actual web site. Since the Mambo application uses the Account-Role model, the user must login in with a separate identity (and therefore, a separate account and associated role). A middleware enabled Mambo should permit a single identity to simply switch roles. These examples are intended to illustrate the interaction of external role assignments with internal application assumptions and indicate some of the challenges that might be involved in developing a general purpose middleware API for collaborative applications.

OpenIdP.Org: Infrastructure for the Rest of Us

As of early 2006, over 200 locations or organizations had participated in the InQueue sandbox federation. In contrast, there are fewer than a dozen members of InCommon, the first production federation service. This situation suggests that reliable federated IdP services may not be available for most people for some time to come, and it should not be surprising that complex infrastructure experiences uneven development. In recognition of this situation a “free love,” low level of assurance, identity service was developed so that collaboration members who have no identity infrastructure can experience federated identity. This service is called OpenIdP.org, which technically is an instance of drupal that contains only the identity registration component (Gemmill & Robinson,

2006). OpenIdP.org registration uses a verified email address as identifier and a self-selected common name. OpenIdP.org is also registered with the InQueue federation and is a shibboleth enabled identity provider with its own Single Sign On Service and attribute authority. Thus, anyone with a working email address can register with OpenIdP.org and use this service as their Shibboleth IdP.

The addition of an open identity provider completes the set of components needed for the myVocs framework and the full framework is illustrated in Figure 14. In summary, a Shibboleth federation provides a cooperating set of identity providers. A VO's distributed resources can be considered to be a federated set of services with identical security context requirements. The myVocs service provides a bridge between federations of identity providers and a VO's federated services.

myVocs meets GridShib

The GridShib project uses Shibboleth to transport available attributes from identity providers to resources in a grid infrastructure (Welch et al., 2005; Barton et al., 2005). This is an important improvement for grid security because the existing certificate based GSI provides DNs associated with signing authorities but no additional user information. The role of the Shibboleth plug-in is to map the DN from the user's X.509 proxy certificate to ePPN so that Shibboleth transported attributes are available for authorization decisions at the Service Provider. The developers' initial strategy for accomplishing DN to ePPN mapping was to provide mapping files at MyProxy CAs and Shibboleth Attribute Authorities. Unfortunately, this plan did provide a discovery mechanism for the

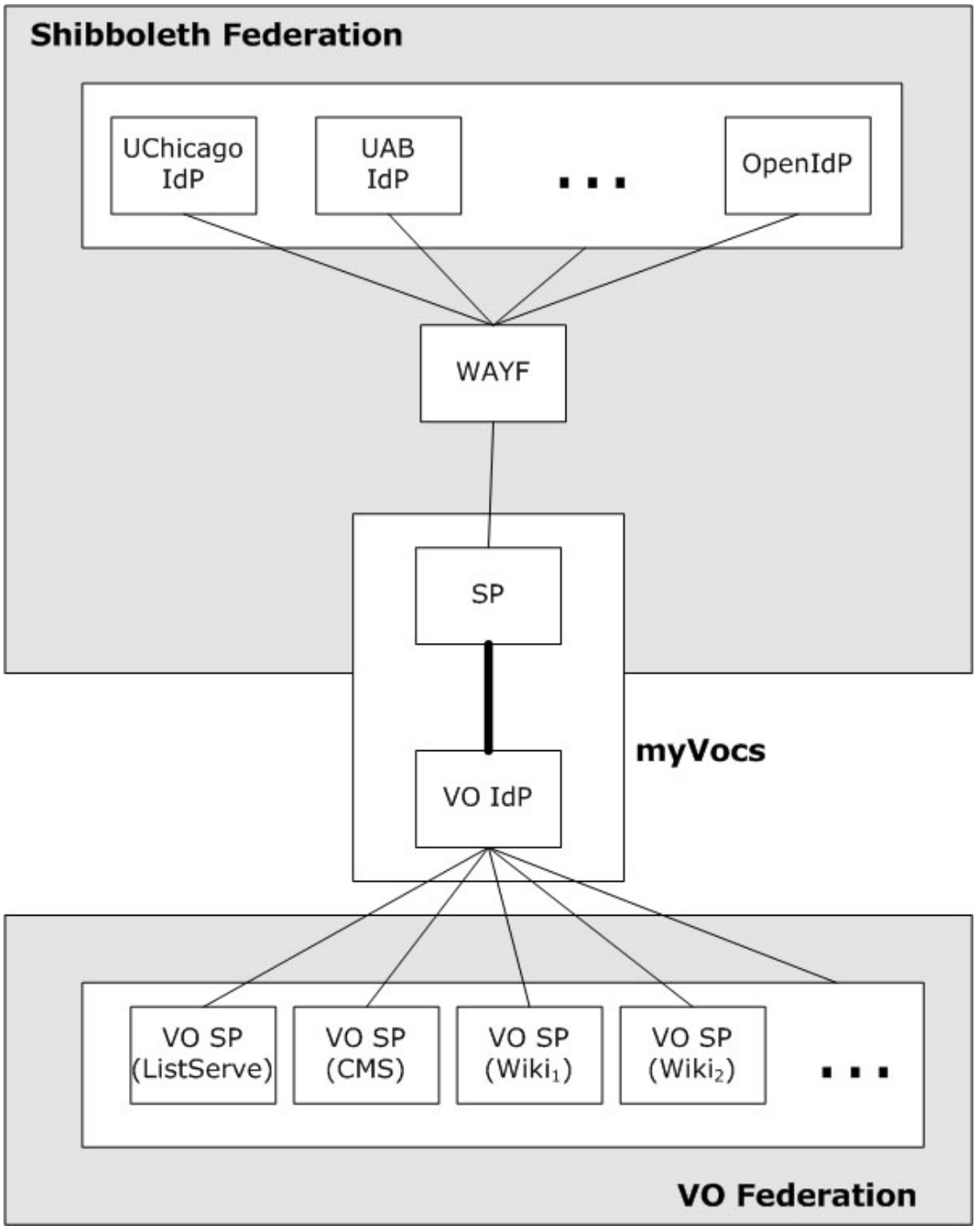

Figure 14. View of the myVocs architecture as a bridge between two types of federations.

appropriate AA so that manually created map files would be required. The Welch paper describes GridShib's powerful capabilities for policy based access control but the authors admit that their design was not scalable because of the need for manually created name maps and the $\Omega(n^2)$ trust relationships required.

The GridShib CA

The GridShib developers first became aware of OpenIdP.org when they were working on methods to use Shibboleth authentication as a means for being authorized to obtain a digital certificate from a grid Certificate Authority; most of the development team did not yet have a shibbolized IdP. After some discussion with the myVocs team, Von Welch of the National Center for Supercomputer Applications completed what is currently being called the GridShib CA.

Using a browser, the user attempts to access the GridShib CA which is configured as a Shibboleth Service Provider and provides Shibboleth based access control. If the user's security context is not adequate Shibboleth makes sure they choose their home IdP via a WAYF, are authenticated with their home IdP, and are returned to the web page. Once their security context meets the access control requirements the user's browser is allowed to view the page. The web page tells the user they are about to create a Grid-ShibCA signed digital certificate and also presents a view of that certificate's DN, which looks something like: `/C=US/O=NCSA-TEST/OU=USER/CN=jgemmill@uab.edu.` Because her security context is complete at this moment the GridShib CA knows her scoped ePPN and can include that information in the CN field of the certificate's DN.

The user pushes a button on the page to request a certificate; the GridShibCA ini-

tiates the download of a Java Network Launching Protocol (JNLP) application to her desktop. The user has already installed Java Web Start in her browser so that the application begins to run as a desktop application once the transfer is complete. The JNLP application creates a public/private key pair on her desktop and generates a certificate request; the request is sent over a secure web channel to the GridShib CA, which returns a signed public key to the desktop. This certificate is an End Entity Certificate which means that it is signed by the GridShib CA. The private key (which has never left the user's desktop) and its associated, signed public key are stored in a desktop location that is well-known to grid toolkit software. The certificate is good for only a few hours, so there is no password protection assigned to the corresponding private key. The user has just been identified using Shibboleth and the resulting security context has been used to generate a certificate useable within the Grid Security Infrastructure.

GridShib / myVocs Integration at the VO IdP

The GridShib developers became aware of the myVocs framework and are currently working collaboratively with the author. Use of myVocs solves many problems for GridShib, including where membership information is stored and what SSO to query. MyVocs serves as an attribute aggregator, providing a single location that applications can refer to for acquiring attributes originating in multiple locations.

The integration of myVocs and GridShib involves name mapping in the myVocs database and configuration of the grid Service Provider's GridShib plug-in to point back to the myVocs IdP. Rather than protecting the GridShib CA with standard Shibboleth access control, the GridShib CA is configured to refer arriving browsers to the VO IdP for

authentication as shown in Figure 15. The integrated GridShib CA will be referred to as the GridShib-CA, and discussions are still on-going as to whether this CA is a service provided by myVocs or a separate VO Service Provider that uses myVocs as its attribute source. Additions to the previous myVocs architecture are highlighted in blue. As described in the section on myVocs as an IdP, the VO IdP is itself Shibboleth protected, causing arriving browsers to be referred to a WAYF (A), then their home institution for authentication (B, C), then back to the VO IdP (D); steps A, B, C, and D have not changed from their earlier description in Figure 12. As a result, at the time the user accesses the GridShib CA web page their scoped ePPN is known at a location that is also aware of that identity's VO memberships and roles (E). At the time the GridShib CA signs the user's End Entity Certificate (F), the CA also writes the DN/scoped ePPN name pair into the myVocs data store (G).

Once the VO CA has issued a user certificate, the user may begin using the grid. As illustrated in Figure 16, grid client software presents the user's public certificate to the grid server software at the compute resource (A). After verifying the signature and negotiating a secure channel with the client, the Globus Toolkit software passes the certificate to the Globus GridShib plug-in (B), where the certificate's DN is extracted, converted to a SAML expression, and handed off to the Shibboleth Assertion Consumer Service. The grid service provider has been configured to always send queries to the VO IdP, so no IdP discovery step is required, and the DN is passed to the Shibboleth Attribute Requester component (C) which is capable of direct communication with the VO IdP's Attribute Authority (AA) over a secure channel. The Attribute Requester queries the VO AA referring to the user by DN (D). The VO AA can use the DN to discover the user's

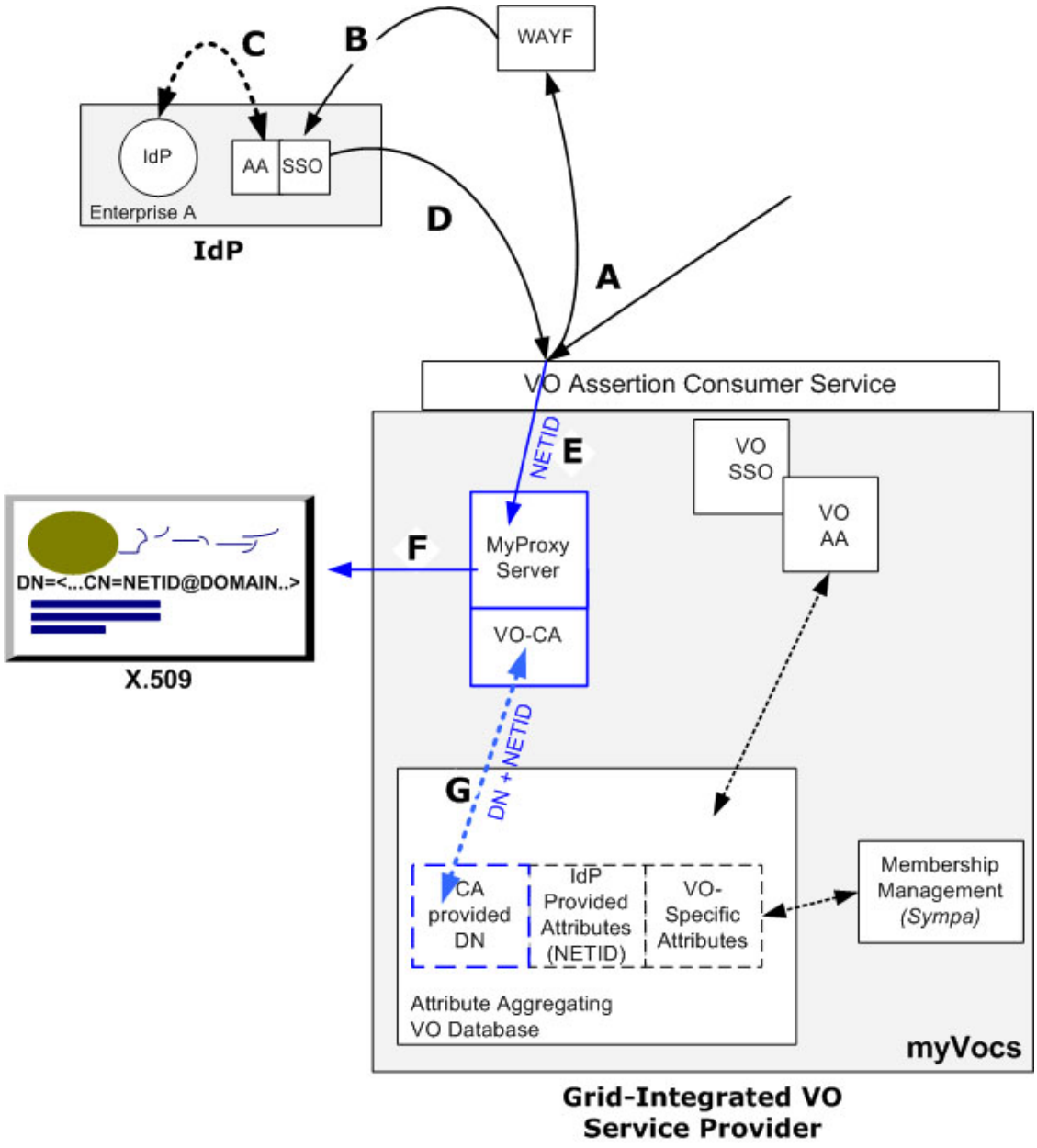

Figure 15. GridShib / myVocs Integration at the Service Provider

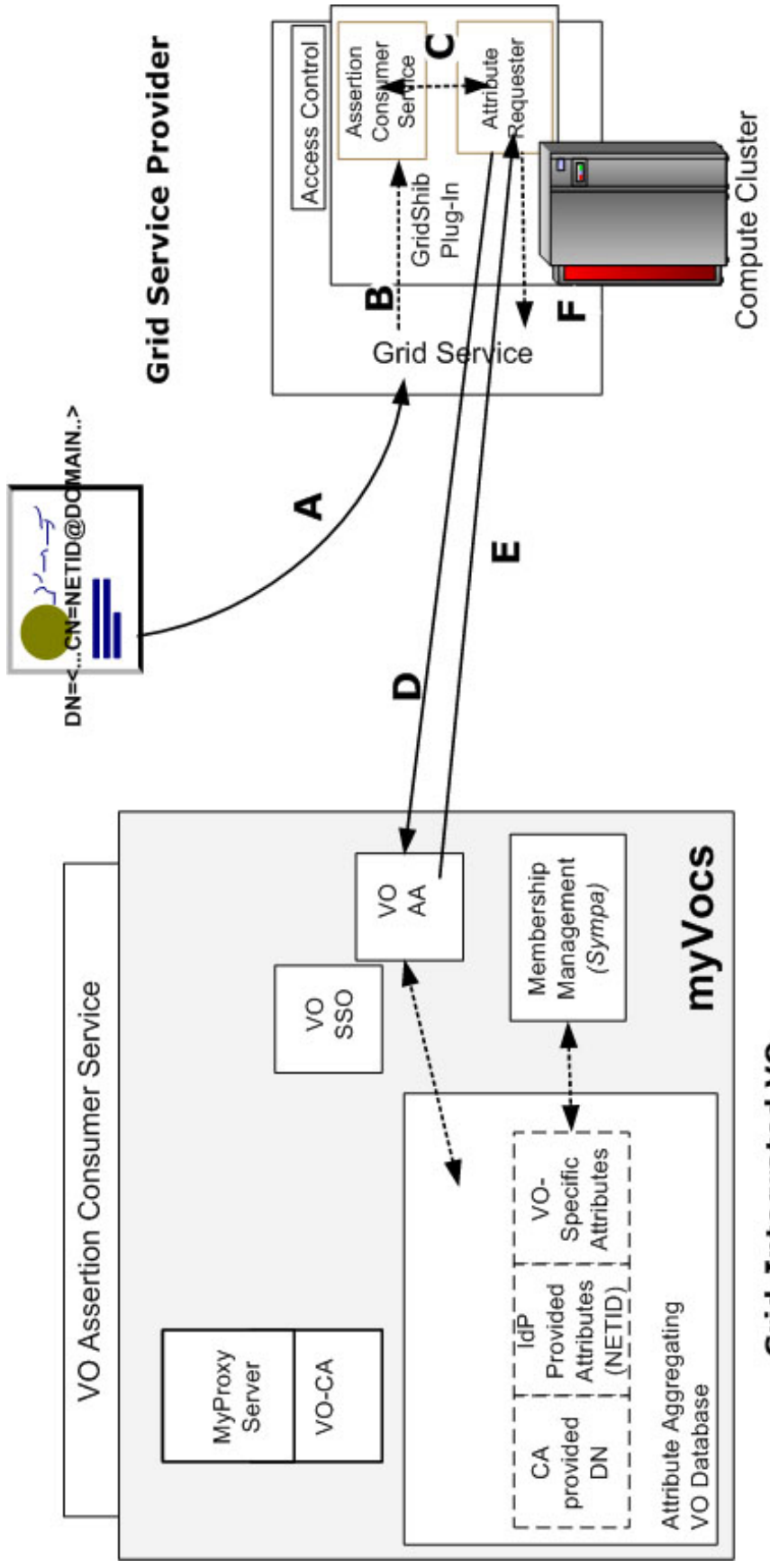

Figure 16. myVocs integration with Grid Service Provider.

scoped ePPN, VO membership and VO role, and these are returned to the Attribute Requester (E). Finally, the Attribute Requester passes the information back to the GridShib plug-in which reformats the reply into a format that can be consumed by the Globus software (F).

Instead of replying to the Service Provider through Shibboleth mechanisms, it is also possible to write the attributes discovered at the VO AA into a short-lived digital certificate using special attribute fields. Use of such a certificate is central to VOMS (another approach to collecting information for grid authorization described on page 31) demonstrating that the myVocs framework can be interoperable with other existing grid authorization schemes.

Scenarios Using myVocs and UABgrid

Two scenarios will be explored to illustrate use of myVocs: (a) a collaboration of environmental scientists with web-accessible resources distributed across several institutions, (b) a non-UAB collaborator who needs to access UABgrid resources. The second scenario can be addressed by either UABgrid or by myVocs.

VO Shared Websites

The environment VO "theearth.org" will be revisited in this first scenario (Figures 1 and 2). The collaborators have a jointly maintained web site located in the Corporation C domain; all members of TheEarth VO need to be able to add content to this site, and the content needs to be read-only for the public. University B provides a web server where collaboration members deposit raw data and early experimental observations; this

information is private to the VO. Collaborating VO members are: Dr. Susie from University A who is the VO lead investigator; scientist Dan from Corporation C; student Susan from University B; and Professor Joe from Antioch College.

Dr. Susie, using her web browser, visits the myVocs service home page. If her security context does not meet the requirements, Shibboleth access control makes sure she is authenticated at her home institution, and she can then access the service. If her scoped ePPN (`NETID@domain`) is not known to the service, a new record will be created. The VO membership management application is used to create a new VO (list) named "theEarthVO," as list creator Dr. Susie is automatically designated as list and therefore VO administrator. She also subscribes herself to the list as a member. Dr. Susie selects her preferred mechanism for adding the remaining VO members; she can invite them to join, or wait until they subscribe themselves.

Each member accessing myVocs is authenticated by their IdP using whatever type of authentication may be used there; each person is identified to myVocs by their scoped ePPN. As each member subscribes to theEarthVO mailing list they are added as a VO member. Professor Joe's institution does not provide an IdP. Professor Joe registers with OpenIdP.org; he provides a self-selected name and a working email address which is verified by OpenIdP.org. Now Professor Joe has a Shibboleth IdP and his scoped ePPN will be `ProfessorJoe@openidp.org`.

Any service provider making resources available to theEarthVO configures Shibboleth to require the following so that only members may access the site.

```
<saml:Attribute
     AttributeName=
      "urn:mace:dir:attributedef:eduPersonScopedAffiliation"
```

```
      AttributeNamespace=
       "urn:mace:shibboleth:1.0:attributeNamespace:uri">
     <saml:AttributeValue>
        member@theEarthVO
      </saml:AttributeValue>
</saml:Attribute>
```

Alternatively, the Service Provider may leave the site open and allow the application to manage access control based on the security context available in its environment. Members of theEarthVO can move around among all federated services and their security context is available at each location.

Using UABgrid for Cross-Domain Grid Resources

A collaborator named Sam at University of North Carolina, Chapel Hill (UNC) needs to use computational resources at UAB. His institution uses Kerberos authentication and has made a KX.509 service available so that his Kerberos ticket can be used to obtain a temporary digital certificate good for use on UNC grid resources. Fortunately, UNC is a member of SURAGrid and UNC's grid Certificate Authority has been cross-certified with the SURAGrid Bridged CA. Sam can use his grid client software to request use of a UAB-owned cluster. The grid client software presents Sam's digital certificate to the grid software at the cluster. The UABgrid CA has also been cross-certified with the SURAGrid Bridged CA and this particular cluster authorizes anyone with a valid SURAGrid credential to use nodes on the cluster if they are available. The cluster's grid software does not recognize the signature on Sam's digital certificate so the Bridge is queried; the Bridge validates the signature on Sam's certificate and makes sure the certificate has not been revoked since issued. Because UABgrid resources trust the SURAGrid Bridge, the cluster's grid software proceeds to establish a secure connection with Sam so

that his job can start.

Using myVocs for Cross-Domain Grid Resources and Web Resources

Scientist Dan is a member of theEarthVO and needs to use compute resources provided by the Pittsburgh Supercomputer Center (PSC). Dan has already subscribed to theEarthVO as a member and has been using that VO's web-based services. Scientist Dan accesses the VO CA web page; the Shibboleth process refers him to the federation WAYF and his corporate IdP before returning as an authenticated user to the VO CA web page. Dan pushes the "Request Certificate" button and within a few seconds he has a digital certificate signed by the VO CA. Dan uses his grid client to request compute cycles at PSC, which causes his new public certificate to be sent to the Globus service at PSC. Using its new GridShib plug-in service, the PSC server queries the myVocs Attribute Authority regarding the identity named in the DN; the myVocs AA database contains a mapping of Dan's DN to his scoped ePPN, Dan's group membership in theEarthVO organization, the time period over which his certificate is valid, and his status as an employee of Corporation C. These attributes are returned over Shibboleth channels to PSC; the Globus server software at PSC sees that theEarthVO has been allocated 2000 hours of computation which are not yet used up and that Scientist Dan is a verified member of that VO, so his job is allowed to run on the resource.

If Dan wants to consult a web page belonging to his VO that shows the results of his computation he can simply point his browser at that location; his security context is already in place and he will transparently be granted access to that web site. Surprised at the numbers he is seeing, Dan points his browser at his VO's private Wiki so he can re-

port his findings to his collaborators. Even though this wiki is on a computer he has not visited in a month, his security context is in place and he can immediately begin entering data into the group wiki without having to log in. Dan and Dr. Susie are happy that theEarthVO did not need to wait for a custom web portal to be built so that their VO can easily and securely get to the tools needed to get their work done.

CONCLUSIONS

The myVocs architecture is a framework providing a consistent and well-defined environment to applications. That environment includes authentication, identity, group, and role information originating from trusted sources; this relieves applications from having to replicate those functions and allows application developers to focus on their application's unique functionality. Furthermore, the myVocs framework allows sharing of that same information across many applications, regardless of their location on the Internet. From the applications' perspective, myVocs provides a single source of information needed to form a handling decision for each action requested.

Leveraging Distributed Identity Management

Existing identity management systems are essential foundation elements for myVocs. Authoritative identity management systems serve to anchor a federating trust fabric; federation begins with a high degree of confidence in the identity being presented. The Shibboleth and related Internet2 projects established some common attributes and directory schema and also a method for establishing federations. Using Shibboleth, the myVocs framework has been shown to support multiple security credentials issued by multiple identity providers. Interoperable, distributed identity management and a framework within trust relationships can be formed, solving many scalability problems in distributed computing.

The myVocs service is itself a participant in the trust fabric, and as currently designed provides identity via transitive trust to service providers. The myVocs framework offers a solution for combining attributes for which different entities are authoritative. As suggested in Figure 7, many possible sources exist for attribute storage locations. The myVocs framework solves this problem by becoming the attribute aggregator. From an application developer's perspective it is desirable to refer to a single location to obtain all attributes needed for an authorization decision. Within a single enterprise this is not much of an issue since the enterprise is authoritative for its own attributes and can collect them in a central attribute store. When handling decisions that involve attributes having multiple authorities, however, the process for locating all necessary attributes was previously undefined. The myVocs framework provides a vehicle for the orderly collection of attributes needed to support virtual organizations, and this framework is the first formal solution proposed for this problem.

A Common System Environment Without Portals

Today's standard approach for establishing a consistent environment for distributed applications is to provide a portal; the portal's functions are to interface applications with an identity management system, share authenticated identity across applications and store shared attributes that can be used to customize the portal's presentation. However, a significant amount of time is required to customize a portal for a particular set of users, including re-writing applications to a standard portlet API. Therefore, portals tend to be built to serve a pre-determined and usually large population and are not generally available for small groups of collaborators. Application integration through a portal could be

described as elevating the silo application development approach to a higher (that is, enterprise) level; the portal may well be designed to use a single IdP or authentication system and still not able to support cross-domain access. In addition to aggregating attributes the myVocs framework provides a common security context for VO members without having to also directly manage their identity. That common security context begins with VO creation and VO membership self-management. Using Shibboleth, the common security context can be shared across applications and services anywhere by simple service provider configuration. Identity providers, service providers, and myVocs are system components held together by the underlying trust fabric woven from federations and their attribute release policies.

As was demonstrated in the OGCE/pubcookie integration described earlier, the myVocs framework permits portals to become middleware-enabled applications. When identity and attribute delivery are independent Internet services and applications are developed to trust and interact with those services it is possible to construct a consistent system environment without using portals; myVocs is certainly a novel architecture in this regard.

The VO Service Center Model

The myVocs/UABgrid framework was designed to provide VO management autonomy. That design goal favored VO-managed attribute definition, assignment and storage for VO-authoritative attributes. A mailing list self-management model was not only followed but was actually modified to provide the VO self management over information for which it is authoritative. The resulting myVocs service, even without the addi-

tion of a GridShib CA, is fairly complex to install and configure; however, once operational it can run as a service requiring little administrative overhead. The myVocs deployment model, therefore, is that myVocs is part of the federation trust fabric which means its administrators and their operational procedures must be made known to the other members of the federation. As a practical matter the framework is too complex for any one VO to stand up on its own.

Applications as Pluggable Components

Many useful open source applications exist but have limited utility when written as a stand-alone silo. The myVocs framework defines a consistent approach by which applications can become middleware enabled. Re-engineering such applications to use middleware has the potential to rapidly produce a set of useful applications that can be easily combined into a customized application suite, providing VO's with flexibility in selecting their toolset

Component-based software development has proven successful in rapid deployment of complex, reliable systems, and this dissertation provides a framework within which applications can be incorporated as pluggable components. The advantage for application developers is that they can concentrate on their application's functionality; the advantage for system architects is the ability to rapidly assemble a customized collaboration environment.

Dissertation Outcomes

UABgrid

The concept of leveraging enterprise identity management for access to grids was first explored by Henderson at the University of California San Diego, in an environment where existing grid users and an existing enterprise Kerberos infrastructure could make use of the NMI K.X509 (Doster, Watts, & Hyde, 2002) component for credential translation (Henderson & University of Southern California, 2004). Through this dissertation, in parallel with work done at UVa by Humphreys and Jokl, that work became generalized to supporting a wider range of campus authentication technologies. The Global Grid Forum held its first workshop on the topic of campus grids in Fall 2005 and Spring 2006, and it was noted that several more campuses, such as the University of Michigan and University of North Carolina at Chapel Hill, were building campus grid infrastructures following the UABgrid model.

SURAGrid Bridged CA

The cross-certification of UABgridCA with the SURA regional grid CA demonstrated the use of a bridge in scalable, multi-domain grid deployments and contributed to the working collaboration that still exists today and is called SURAGrid. This collaboration is significant in providing an actual multi-domain scenario within which to explore cross-domain grid access, and SURAGrid also serves as a community for sharing existing grid expertise.

myVocs

The myVocs prototype has been met with great interest in both the EDIT and GRIDS communities who are interested in providing a reference implementation of myVocs. The Internet2 organization itself consists of many VO's, known as working groups, and the Internet2 staff are currently discussing the possibility of using myVocs to control access to working group wikis and draft papers under development; they are also discussing possible use of OpenIdP.org for member institutions who have not yet implemented their webISO solution. Although the Internet2 staff are reluctant to adopt such a new and untested approach, they admit they have yet to find any alternative that provides the same functionality.

GridShib Integration

Recognition from the GridShib developers of the benefits brought by the myVocs framework was itself evidence that a real problem has been identified and solved in a useful way. The integration of GridShib with myVocs also provides a more flexible approach to integration of grid services than was initially envisioned, adding the aggregation of DN name mappings to the VO-centric data store. It would be possible to implement UABgrid as one or more VO's managed by myVocs.

OpenIdP.org

The OpenIdP.org service was provided simply to allow access to the myVocs framework for people who did not yet happen to have an IdP. Since its introduction, OpenIdP.org has been adopted by many working groups in Internet2 who need it for their project, and was also promoted by the GridShib team when it came to demonstrating their

Shibbolized CA. As federated identity becomes more mainstream and is found in commercial services, there may be a market for an identity provider service that is federated, provides user control over their own attribute release, and supports anonymous identity.

Scientific Merit and Broader Impact

Federated trust fabrics are established by domain administrators; leveraging this fabric to support VO requirements was an important problem to solve. There are many knowledge domains that require enterprise accountability for access to its resources by authorized external partners. For example, The National Cancer Center at the National Institutes of Health has a large project known as caBIG™ (National Cancer Institute, 2005) designed to enable the sharing of data and tools across cancer research centers. This consortium recognized that in order to meet its goals "there was a need to develop a comprehensive grid security infrastructure for managing federated authentication and authorization in caBIG™" and has identified many of the NMI components mentioned in this paper as candidate solution components. A prototype will be built as a next step (cancer Biomedical Informatics Grid (caBIG), 2006). It will be interesting to see what solution will be used to define, manage and assign attributes describing their research groups, subgroups, and their access requirements; the myVocs framework could be a useful solution that also provides an interface to grids. Another example in the health informatics arena is the information requirements of a regional medical consortium: local clinics keep their own records but need to make them accessible to hospitals; hospitals need to make patient information available to home-town physicians for follow-up care; and emergency medicine providers need to track their own activities as well as make them

available to emergency rooms. A federated authentication and authorization architecture is one possible and scalable solution, and the VO service center model makes it possible to consider a working service for a small ambulance company.

Limitations of the myVocs Architecture

Many collaborations consist of a small group of people sharing resources that they themselves own; myVocs is not needed for this scenario, even if all the participants have an enterprise IdP. The myVocs framework is best suited for cross-domain access to resources for which some enterprise is responsible when the access is by large numbers of people working in small, reconfigurable groups. The myVocs framework is an approach to system integration, not a software package; therefore it is more of a configuration and information exposure methodology than a "product," which means that the ability to construct an environment using framework is currently limited to fairly sophisticated system administrators.

A key assumption for the framework is that the transitive trust model is acceptable. A service provider must trust that the myVocs service administrator is releasing correct information – not only information created at the myVocs service regarding VO's, but also and most importantly that the identity information asserted has not been tampered with. For small collaborations with low-level or average security requirements, this model may be acceptable; perhaps if national nuclear secrets are at stake the model's assumptions will not do.

Although the attributes aggregated at myVocs are refreshed each time a user authenticates, there are legitimate concerns regarding the cached data at myVocs. For ex-

ample, if a VO member leaves their home institution and can no longer authenticate, they are still listed as a VO member at myVocs. Although a user cannot authenticate and therefore cannot access VO resources, they would still receive email and could conceivably be in possession of a long-term grid-certificate whose DN is still contained in myVocs. The current design leaves it to the VO administrator to manage these types of changes; perhaps some back channel mechanism from myVocs to IdP's could be added to better synchronize the membership attribute data.

The applications prototyped for the myVocs environment are useable, but certainly could benefit from a well designed user interface and ability to apply some common style elements across applications. Two interfaces have been developed so far: one adds some functionality to the Sympa MLM web interface; that one is easy to use but makes it appear that Sympa is used as a portal to access data when in fact use of Sympa is not required after one has joined a VO. The second interface is modeled after Google, emphasizing the lack of relationships across applications except for one's security context. Neither one of these approaches does a transparent job yet of registering first-time users at myVocs. Each application comes with its own default graphical design look; when several applications are used together it can be jarring to switch from one to another and have the design be inconsistent across applications. These issues are less important from the system integration perspective, but are certainly important in terms of usability, user acceptance and ultimate success.

Role assignment in myVocs is currently simple-minded: myVocs uses the MLM's existing roles of mailing list administrator, moderator, and member directly as roles to be transferred into other applications, but there is as yet no capability to define additional

roles or their associated permissions. The Signet and Grouper applications may be ideal replacements for the current MLM roles model.

Future Directions

This dissertation represents an attempt to build a system environment suitable for use by VO's from newly minted NMI software components. The attempt was successful and has attracted some interest from others beginning to build federated solutions. As an early system integration activity, this dissertation demonstrates that federated authentication and authorization can be made to work from existing components in both web and grid spaces. At the same time, this work also demonstrates that certain scenarios, such as the VO-centric one, may have unmet requirements.

The Internet2 middleware architects are about to release new software called Signet and Grouper. Signet is a privilege allocation service: it is a top-down delegation of rights and rights assignment capabilities. Grouper is intended to be a target for Signet and to permit the definition of groups and roles, assuming one has been designated the ability do so. The first software implementations of Grouper and Signet will be available soon, and these applications appear as promising replacements for VO membership role-assignment functions currently being implemented using Sympa.

Another contribution in this area would be work to define and standardize a core set of VO attributes, roles, and actions that would be available to each VO and that could possibly form the basis for some common approach to authorization for applications using middleware. The new attention to social networking suggests that VO's may be discovered as well as declared. What amount of common interest in a subject signals a pos-

sible new VO and how could a federated VO Service Center be used in this scenario? The same approach might be pursued to discover common VO role definitions.

A federated solution for multimedia is needed. Middleware-enabling multimedia tools, especially video and voice over IP standards-based products, would provide valuable additions to the collaboration toolkit. Previous work in this area discovered two hard problems: (a) the H.323 protocol had a flawed security design and (b) the SIP protocol had only MD5 hash security and TLS defined (Gemmill et al., 2004).

The OpenIdp.Org model could improve the value of identities it asserts through use of some mechanism such as PGP key rings. A great deal of work remains to be done to facilitate user-controlled attribute release; currently the enterprise controls this function.

In summary, the myVocs framework provides an interesting context for exploring new approaches to security, application development, and access control built from Internet services without relying on a central authentication repository or scheme.

References

Alfieri, R., Cecchini, R., Ciaschini, V., dell'Agnello, L., Frohner, A., Gianoli, A., Lorentey, K., and Spataro, F. (2003). VOMS, an Authorization System for Virtual Organizations. [Electronic Version]. In *Proceedings of the 1st European Across Grids Conference*, Santiago de Compostela.

Alterman, Peter, Weiser, Russel, Gettes, Michael, Stillson, Kenneth, Blanchard, Deborah, Fisher, James, Brentrup, Robert, and Norman, Eric (2002). Report: EDUCAUSE - NIH PKI Interoperability Pilot Project. [Electronic Version]. In *1st Annual PKI Research Workshop*.

Alterman, Peter, Weiser, Russel, Rea, Scott, and Blanchard, Deborah (2005). NIH-EDUCAUSE PKI Interoperability Project Phase Three Project Report. [Electronic Version]. In *4th Annual PKI Research Workshop*, Washington, D.C.

American National Standards Institute (2005). Health Level 7 (HL7). Retrieved August 11, 2005 from http://www.hl7.org/

Aumont, S. & Salaun, O. (2005). Sympa Mailing List Manager. [Computer Software]. Comité Réseau des Universités. Retrieved February 19, 2006 from http://www.sympa.org/

Bajaj , S., Della-Libera, G., Dixon, B., Dusche, M., Hondo, M., Hur, M. et al. (2003). Web Services Federation Language (WS-Federation). Retrieved April 14, 2005 from http://www-106.ibm.com/developerworks/webservices/library/ws-fed/

Baker, Mark (2004). Ian Foster on Recent Changes in the Grid Community. *IEEE Distributed Systems Online,* 5, 2. Retrieved March, 2004.

Barton, T. (2005). Grouper. [Computer Software]. University Corporation for Advanced Internet Development (Internet2). Retrieved April 14, 2005 from http://middleware.internet2.edu/dir/groups/

Barton, Tom, Basney, Jim, Freeman, Tim, Scavo, Tom, Siebenlist, Frank, Welch, Von, Ananthakrishnan, Rachana, Baker, Bill, and Keahey, Kate (2005). Identity Federation and Attribute-based Authorization through the Globus Toolkit, Shibboleth, GridShib, and MyProxy. [Electronic Version]. In *PKI 2006 Conference*, Dartmouth, N.H.

Basney, J., Humphrey, M., & Welch, V. (2005). The MyProxy Online Credential Repository. *Software: Practice and Experience, 35,* 801-816.

Berners-Lee, T. & Groff, J.-F. (1992). WWW. *SIGBIO Newsl., 12,* 37-40.

Bertino, E., Ferrari, E., & Squicciarini, A. (2004). Trust negotiations: Concepts, systems, and languages. *IEEE Computational Science and Engineering, 06,* 27-34.

Blatecky, A., West, A., & Spada, M. (2002). Middleware: The New Frontier. *EDUCAUSE Review, 37,* 24-35.

Blaze, M., Feigenbaum, J., Ioannidis, J., & Keromytis, A. D. (1999). RFC 2704: The KeyNote Trust-Management System Version 2. Internet Engineering Task Force. Retrieved August 17, 2005 from http://www.faqs.org/fcs/rfc2704.html

Blaze, M., Feigenbaum, J., & Lacy, J. (1996). Decentralized Trust Management. In *IEEE Symposium on Security and Privacy*, (pp. 164-173). Oakland, CA.

Calhoun, P., Zorn, G., Spence, D., & Mitton, D. (2005). RFC 4005: Diameter Network Access Server Application. Internet Engineering Task Force. Retrieved August 20, 2005 from http://www.ietf.org/rfc/rfc4005.txt?number=4005

cancer Biomedical Informatics Grid (caBIG) (2006). caBIG™ Security Technology Evaluation - White Paper. Retrieved March 14, 2006 from https://cabig.nci.nih.gov/workspaces/Architecture/caBIG_Security_Technology_Evaluation_White_Paper_20060123.pdf

CERN (2004). The DataGrid Project. Retrieved April 14, 2005 from http://eu-datagrid.web.cern.ch/eu-datagrid/

Chadwick, D. W. & Otenko, A. (2002). The PERMIS X.509 role based privilege management infrastructure. In *Seventh ACM Symposium on Access Control Models and Technologies*, (pp. 135-140). Monterey, California.

Chadwick, D. W. & Otenko, S. (2005). A Comparison of the Akenti and PERMIS Infrastructures. Retrieved August 23, 2005 from http://www.cs.kent.ac.uk/pubs/2003/2071/content.pdf

Chokhani, S., Ford, W., Sabett, R., Merrill, C., & Wu, S. (2003). RFC 3647: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. Retrieved April 17, 2006 from http://www.ietf.org/rfc/rfc3647.txt?number=3647

community source (2005). Sakai Project Collaboration and Learning Environment. Retrieved April 14, 2005 from http://www.sakaiproject.org/cms/

DeSanctis, Gerardine Monge Peter (1998). Communication Processes for Virtual Organizations. *Journal of Computer-Mediated Communication,* 3, 4. Retrieved March 11, 2004.

Dierks, T. & Allen, C. (1994). RFC 2246: The TLS Protocol Version 1.0. The Internet Society. Retrieved May 31, 2005 from http://www.faqs.org/rfcs/rfc2246.html

Diffie, W. & Hellman, M. E. (1976). New Directions in Crytography. *IEEE Transactions on Information Theory, IT-22,* 644-654.

Doster, W., Watts, M., & Hyde, D. (2002). The KX.509 Protocol. University of Michigan, Center for Information Technology Integration. Retrieved May 15, 2004 from http://www.citi.umich.edu/techreports/reports/citi-tr-01-2.pdf

Durham, D., Boyle, J., Cohen, R., Herzon, S., Rajan, R., & Sastry, A. (2000). RFC 2748 The COPS (Common Open Policy Service) Protocol. Internet Engiineering Task Force. Retrieved August 17, 2005 from http://www.ietf.org/rfc/rfc2748.txt?number=2748

Educause (2005). Higher Education Bridge Certficate Authority (HEBCA). Retrieved August 28, 2005 from http://www.educause.edu/hebca/

Electronic Privacy Information Center (2005). Sign Out of Passport! Retrieved June 20, 2005 from http://www.epic.org/privacy/consumer/microsoft/

Erdos, M. & Cantor, S. (2002). Shibboleth Architecture Draft V05. Retrieved December 12, 2005 from http://shibboleth.internet2.edu/docs/draft-internet2-shibboleth-arch-v05.pdf

Feijen, C., Patial, V., & Poot, R. (2006). WebInsta FM Manager. [Computer Software]. Retrieved February 20, 2006 from http://www.webinsta.com/fm.php

Fielding, I., Gettys, J., Mogul, J., & et al (1999). RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1. IETF. http://www.w3.org/Protocols/

Foster, I. (2005). Globus Toolkit Version 4: Software for Service-Oriented Systems. In *IFIP International Conference On Network and Parallel Computing*, (pp. 2-13) Springer Verlag.

Foster, I. & Kesselman, C. (1997). Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputer Applications, 11,* 115-128.

Foster, I., Kesselman, C., Nick, J. M., & Tuecke, S. (2002). The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Globus Project. Retrieved April 10, 2006 from http://www.globus.org/alliance/publications/papers/ogsa.pdf

Foster, I., Kesselman, C., Tsudik, G., & Tuecke, S. (1998). A security architecture for computational grids. In *Proceedings of the 5th ACM conference on Computer and communications security*, (pp. 83-92). San Francisco, CA: ACM Press.

Foster, I., Kesselman, C., & Tuecke, S. (2001). The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of Supercomputer Applications, 15*.

Garfield, S. (1994). *PGP: Pretty Good Privacy*. Sebastopol, CA: O'Reilly & Associates, Inc.

Garman, J. (2003). *Kerberos, The Definitive Guide*. Sebastopol, CA: O'Reilly & Associates, Inc.

Gemmill, J. (2004). The Mailing List Object Model. Internet2. Retrieved March 7, 2006 from http://middleware.internet2.edu/mlist/docs/draft-internet2-mace-mlist-object-model-02.pdf

Gemmill, J. & Robinson, J.-P. (2006). OpenIdP.org. Retrieved February 28, 2006 from http://www.openidp.org/

Gemmill, J., Robinson, J.-P., & Shealy, D. (2003). *NMI Enabled Open Source Collaboration Tools for Virtual Organizations* Birmingham, Alabama: University of Alabama at Birmingham.

Gemmill, J., Srinivasan, A., Lynn, J. L. W., Chatterjee, S., Tulu, B., & Abhichandani, T. (2004). Middleware for Scalable Real-time Multimedia Communications Cyberinfrastructure. *Journal of Internet Technology, 5,* 405-420.

Harrison, R. (2005). Internet-Draft: LDAP: Authentication Methods and Connection Level Security Mechanisms. The Internet Society. Retrieved July 8, 2005 from http://tools.ietf.org/tools/rfcmarkup/rfcmarkup.cgi?draft=draft-ietf-ldapbis-authmeth-14.txt

Hazelton, K. (2006). eduPerson Object Class. Retrieved March 4, 2006 from http://www.educause.edu/content.asp?PAGE_ID=949&bhcp=1

Henderson, S. & University of Southern California (2004). Shibboleth and Pubcookie at USC-Authentication and Authorization for All. NMI-EDIT. Retrieved April 10, 2006 from http://www1.sura.org/3000/USC-ShibPubc.pdf

Hepper, S. (2006). JSR 168: Portlet Specification. Retrieved April 10, 2006 from http://www.jcp.org/en/jsr/detail?id=168

Howard, L. (1998). RFC 2307: An Approach for Using LDAP as a Network Information Service. Retrieved January 4, 2006 from http://www.ietf.org/rfc/rfc2307.txt?number=2307

International Telecommunication Union (ITU) (2000). *Recommendation X.509 The Directory: Public-key and attribute certificate frameworks* International Telecommunication Union.

Internet2 & Gemmill, J. (2006). Middleware-Enabled Mailing List Working Group (MACE-MLIST). Retrieved February 16, 2006 from http://middleware.internet2.edu/mlist/

JASIG Open Standard (2005). uPortal. [Computer Software]. Retrieved April 14, 2005 from http://mis105.mis.udel.edu/ja-sig/uportal/index.html

Johnston, W., Mudumbai, S., & Thompson, M. R. (1998). Authorization and Attribute Certificates for Widely Distributed Access Control. In *Proceedings of IEEE 7th International Workshops on Enabling Technologies: Infrastructures for Collaborataive Enterprises*.

Jokl, J. (2005). SURA NMI Testbed Grid PKI Bridge Certification Authority. Retrieved December 21, 2005 from https://www.pki.virginia.edu/nmi-bridge/

Jokl, J., Basney, J., & Humphrey, M. (2004). Experiences using Bridge CAs for Grids. In *UK Workshop on Grid Security Experiences*.

Kent, S. & Atkinson, R. (1998). RFC 2401 Security Architecture for the Internet Protocol. http://www.ietf.org/rfc/rfc2401.txt?number=2401

Kessels, B. (2006). drupal. [Computer Software]. Retrieved February 20, 2006 from http://drupal.org/

Kirk, P. (2005). Gnutella Protocol. Source Forge. Retrieved April 17, 2006 from http://rfc-gnutella.sourceforge.net/index.html

Klapp, C., Dairiki, G. T., Urban, R., & Wainstead, S. (2006). phpwiki. [Computer Software]. SourceForge.net. Retrieved February 20, 2006 from http://sourceforge.net/projects/phpwiki/

Kohl, J. & Neuman, C. (1993). RFC 1510: The Kerberos Network Authentication Service (V5). Internet Engineering Task Force. Retrieved April 17, 2006 from http://www.faqs.org/rfcs/rfc1510.html

Landau, S., Venezuela, C. C., Ellison, G., Hodges, J., Kellomaki, S., Kemp, J. et al. (2005). Liberty ID-WSF Security and Privacy Overview Version 1.0. Retrieved April 17, 2006 from http://www.projectliberty.org/specs/liberty-idwsf-security-privacy-overview-v1.0.pdf

Lemelson-MIT Program (2001). Inventor of the Week Archive - Internet Browser Technology. Massachusetts Institute of Technology, MIT School of Engineering. Retrieved April 5, 2005 from http://web.mit.edu/invent/iow/andreesen=bina.html

Liberty Alliance Project (2001). Liberty Alliance Project. Retrieved April 14, 2005 from http://www.projectliberty.org/

Lindholm, T. Y. F. (1997). *The Java Virtual Machine Specification*. Addison-Wesley.

Mambo Foundation (2006). Mambo. [Computer Software]. Retrieved February 21, 2006 from http://www.mamboserver.com/

Martin, J., Basney, J., & Humphrey, M. (2005). Extending Existing Campus Trust Relationships to the Grid through the Integration of Pubcookie and MyProxy. In *2005 International Conference on Computational Science*. Atlanta, GA.

McRae, L. (2005). Signet. [Computer Software]. University of San Diego. Retrieved April 14, 2005 from http://middleware.internet2.edu/dir/groups/

McWilliams, B. (2005). Stealing MS Passport's Wallet. Wired News http://www.wired.com/. Retrieved June 20, 2005 from http://www.wired.com/news/technology/0,1282,48105,00.html

Nadalin, A., Kaler, C., Hallam-Baker, P., & Monzillo, R. (2002). Specification: Web Services Security (WS-Security) Version 1.0. Retrieved April 14, 2005 from http://www-106.ibm.com/developerworks/webservices/library/ws-secure/

National Cancer Institute (2005). Cancer Biomedical Informatics Grid (caBIG). Retrieved December 23, 2005 from https://cabig.nci.nih.gov/

National Science Foundation (2005). NSF Extensible Terascale Facility (TeraGrid). Retrieved April 14, 2005 from http://www.teragrid.org/

NMI (2005). National Science Foundation Middleware Initiative (NMI). Retrieved April 14, 2005 from http://www.nsf-middleware.org/

Novotny, J., Russell, M., & Wehrens, O. (2004). GridSphere: an advanced portal framework. In *Proceedings of the 30th EUROMICRO Conference (EUROMICRO'04)*, (pp. 412-419).

Novotny, J., Tuecke, S., & Welch, V. (2001). An Online Credential Repository for the Grid: MyProxy. In *10th IEEE International Symposium on High Performance Distributed Computing* IEEE Computer Society Press.

OASIS (2003). Security Assertion Markup Language (SAML) v1.1. Retrieved April 17, 2006 from http://www.oasis-open.org/specs/index.php#samlv1.1

OASIS (2004). Web Services Security (WS-Security). Retrieved April 14, 2005 from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss

OASIS (2005a). eXtensible Access Control Markup Language (XACML) version 2.0. Retrieved August 23, 2005a from http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

OASIS (2005b). Security Assertion Markup Language (SAML) version 2.0. Retrieved April 14, 2005b from http://www.oasis-open.org/specs/index.php#samlv2.0

Object Management Group (2000). *The Common Object Request Broker: Architecture and Specification Revision 2.4*.

OGCE Consortium (2006). Open Grid Computing Environments Collaboratory (OGCE). [Computer Software]. Retrieved February 16, 2006 from http://www.collab-ogce.org/nmi/index.jsp

Open Software Foundation (1993). *Introduction to OSFDCE*. Prentice Hall.

Open Source Initiative (2005). Lionshare. [Computer Software]. Penn State University. Retrieved April 14, 2005 from http://lionshare.its.psu.edu/main/

Phelps, J. (2004). MLIST Domain Model. Internet2. Retrieved February 19, 2006 from http://middleware.internet2.edu/mlist/docs/draft-internet2-mace-mlist-domain-model-08.pdf

President's Information Technology Advisory Committee (2004). *Revolutionizing Health Care Through Information Technology* Arlington, VA: National Coordination Office for Information Technology Research and Development, National Science and Technology Council, U.S. Government.

President's Information Technology Advisory Committee (2005). *Cyber Security: A Crisis of Prioritization* National Coordination Office for Information Technology Research and Development, National Science and Technology Council, U.S. Government.

Rigney, C., Willens, S., Rubens, A., & Simpson, W. (2000). RFC 2865 Remote Authentication Dial In User Service (RADIUS). Retrieved August 17, 2005 from http://www.ietf.org/rfc/rfc2865.txt?number=2865

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM, 21,* 120-126.

Roadcap, W. E. (2005). PHPki Digital Certificate Authority. [Computer Software]. Source Forge. Retrieved December 1, 2005

Robinson, J.-P., Gemmill, J., & Bangalore, P. (2005). Web-Enabled Grid Authentication in a Global Trust Fabric. *Manuscript submittted for Publication*.

Robinson, J.-P., Gemmill, J., Joshi, P., Bangalore, P., Chen, Y., Peechakara, S. et al. (2005). Web-Enabled Grid Authentication in a Non-Kerberos Environment. In *6th IEEE/ACM International Workshop on Grid Computing*. 6th IEEE/ACM International Workshop on Grid Computing.

Samar, V. & Lai, C. (1997). Pluggable Authentication Modules (PAM). Retrieved April 14, 2005 from http://www.freebsd.org/doc/en_US.ISO8859-1/articles/pam/

Schantz, R. E. & Schmidt, D. C. (2005). Middleware for Distributed Systems. In B.W.Wah (Ed.), *Wiley Encyclopedia of Computer Science and Engineering* (.

Siebenlist, F., Nagaratnam, N., Welch, V., & Neumann, B. C. (2004). Security for Virtual Organizations: Federating Trust and Policy Domains. In Ian Foster & Carl Kesselman (Eds.), *The Grid 2: Blueprint for a New Computing Infrastructure* (pp. 353-387). Elsevier.

Southeastern Universities Research Association (2005). SURAGrid. Southeastern Universities Research Association (SURA). Retrieved December 21, 2005 from http://www1.sura.org/SURAgrid.html

Stallings, W. (2003). *Network Security Essentials*. (2nd ed.) Upper Saddle River, New Jersey: Prentice Hall.

Tanenbaum, A. S. & van Steen, M. (2002). *Distributed Systems : Principles and Paradigms*. Prentice-Hall.

Telecommunication Standardization Sector of ITU (2004). *Recommendation X.500 Open Systems Interconnection - The Directory: Overview of concepts, models and services* International Telecommunication Union.

Thai, T. & Lam, H. (2001). *.NET Framework Essentials*. O'Reilly & Associates, Inc.

Thomas, A. (1998). Enterprise JavaBeans Technology; Server Component Model for the Java™ Platform. Sun Microsystems. http://www.cs.indiana.edu/classes/b649-gann/ejb-white-paper.pdf

Thompson, T. G. (2004). Remarks by Tommy G. Thompson, Secretary of Health and Human Services. United States, Department of Health and Human Services. Retrieved March 15, 2006 from http://www.hhs.gov/news/speech/2004/040506.html

Trauner, M., Finken, L., Hofer, E., & Krienke, J. (2004). ViDe Data Collaboration Survey. www.videnet.gatech.edu/datacollab/survey

U.S.Department of Education (2005). Family Educational Rights and Privacy Act (FERPA). Retrieved August 23, 2005 from http://www.ed.gov/policy/gen/guid/fpco/ferpa/index.html

U.S.Government (2000). Electronic Signatures in Global and National Commerce Act. Retrieved June 21, 2005 from http://www.ftc.gov/os/2001/06/esign7.htm

University Corporation for Advanced Internet Development, I. (2005). OpenSAML - an Open Source Security Assertion Markup Language implementation version 1.0.1. Retrieved April 14, 2005 from http://www.opensaml.org/

University of Washington (2005). Pubcookie: open-source software for intra-institutional web authentication. http://www.pubcookie.org/

Warsaw, B., Hylton, J., & Kikuchi, T. (2006). Mailman, the GNU Mailing List Manager. [Computer Software]. Retrieved February 21, 2006 from http://mailman.sourceforge.net/

Wason, T. & Cantor, S. (2005). Liberty ID-FF Architecture Overview. Retrieved April 14, 2005 from http://www.projectliberty.org/specs/draft-liberty-idff-arch-overview-1.2-errata-v1.0.pdf

Welch, V., Anaanthakrishnan, R., Meder, S., Pearlman, L., & Siebenlist, F. (2005). Use of SAML in the Community Authorization Service. Retrieved August, 2005 from http://xml.coverpages.org/WelchSAML20030819.pdf

Welch, V., Barton, T., Keahey, K., & Siebenlist, F. (2005). GridShib: Grid-Shibboleth Integration. http://www.globusworld.org/2005Slides/Session%201b(1).pdf

Westerinen, A., Schizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S. et al. (2001). RFC 3198: Terminology for Policy-Based Management. Internet Engineering Task Force. Retrieved August 20, 5 A.D. from http://www.ietf.org/rfc/rfc3198.txt

Wollrath, A., Riggs, R., & Waldo, J. (1996). A Distributed Object Model for the Java System. In *USENIX Computing Systems* (9 ed..

World Wide Web Consortium (W3C) (2005). Extensible Markup Language (XML). Retrieved April 14, 2005 from http://www.w3.org/XML/

World Wide Web Consortium, X. P. W. G. (2000). Simple Object Access Protocol (SOAP) 1.1. http://www.w3.org/TR/soap/

Yahoo (2005). Yahoo Groups. Retrieved August 26, 2005 from http://groups.yahoo.com/

Zerhouni, E. A. (2004). NIH Roadmap : Accelerating Medical Discovery to Improve Health. National Institutes of Health, Department of Health and Human Services, U.S. Government. Retrieved April 15, 2005 from http://nihroadmap.nih.gov/