uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Jun Chen

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.A.Sc. (Electrical Engineering)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

A Distributed Network Management Systems for User-Controlled Lightpath Provisioning and its Security Requirements

TITRE DE LA THÈSE / TITLE OF THESIS

G. Bochmann

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

R. Liscano

C.H. Lung

Gary W. Slater

LE DOYEN DE LA FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTDOCTORALES /
DEAN OF THE FACULTY OF GRADUATE AND POSTDOCORAL STUDIES

A Distributed Network Management Systems for User-Controlled Lightpath Provisioning and its Security Requirements

Jun Chen

Ottawa-Carleton Institute of Electrical and Computer Engineering
School of Information Technology and Engineering
University of Ottawa
Ontario, Canada

March 2005

Your file Votre référence
ISBN: 0-494-11234-4
Our file Notre référence
ISBN: 0-494-11234-4

Abstract

There is a growing trend for many large enterprises and other users to acquire their own optical networks. These customer-owned and managed networks bring cost saving benefits as well as management challenges, especially in the security aspect. In this thesis, a distributed network management system based on Service Oriented Architectures for customer-owned optical networks is presented. The system realizes User-Controlled Lightpath Provisioning and provides users the ability to make end-to-end connections across multiple administrative domains.

However, the current version of the system does not meet the security requirements of UCLP. This thesis also presents the research work on how to overcome the security challenge in customer-owned optical network management. The security requirements of the system are analyzed and the major difficulties are identified. Different access control models and security technologies are investigated. An access control mechanism based on Attribute Certificate is chosen for the future implementation of the UCLP system.

Acknowledgements

I would like to thank my supervisor, Dr. Gregor von Bochmann, whose direction and comments have been a great help. I also would like to thank my colleagues from the Communications Research Centre for their cooperation and support. They are Dr. Jing Wu, Dr. Michel Savoie, Hanxi Zhang and Scott Campbell.

Table of Content

List of Figures

Part 1 Introduction

Customer-owned optical networks have become common in the past few years. The most important benefit brought by customer-owned optical networks is that it provides users the ability to create dedicated high bandwidth end-to-end connections. In this kind of networks, some applications which used to be impractical become possible, such as a Distributed File System over a Wide Area Network [3], large file transfer over long-distance networks [50] and so on. It can also improve existing network applications, such as Video-On-Demand Service in HDTV quality [47] and Virtual Private Network service [51].

However, the traditional management system for carrier-owned networks is inadequate [43]. It can not deal with customer-owned optical networks, which consist of multiple administrative domains and heterogeneous network equipments. Each domain has its own management authority and may use different optical technologies in their network. How to coordinate these domains to achieve User-Controlled Lightpath Provisioning (UCLP) is a very challenging task (see Section 1.3.3). It demands a distributed network management architecture for multiple domains (see Section 1.2.3) and abstracted management functions for heterogeneous network equipments. Also, the management system for customer-owned optical networks should not be an isolated system. It must be able to work as a basis for all kinds of value-added services and applications (see Section 7.1.3). Therefore, interoperability is also a major concern, which makes it more difficult to realize, especially on the security aspect.

In this thesis, we present a distributed network management system for customer-owned optical networks. The system follows the distributed network management architecture and the implementation is based on both Web Services and Jini technology. It realizes User-Controlled Lightpath Provisioning and provides users the ability to make end-to-end connections across multiple administrative domains. For the security aspect, the current version of the system does not implement any sophisticated authentication and authorization. The second half of this thesis

continues on this topic and presents the research work on how to overcome the security challenge in customer-owned optical network management. We investigated different access control models and security technologies and proposed three different approaches. Based on the result of the security requirement analysis, we compared the three approaches and came up with a choice for the future implementation.

This thesis is divided into four parts. Part 1 (Chapter 1) introduces the fundamentals of network management and customer-owned optical networks. Part 2 presents the architecture of the UCLP system and related considerations. In Chapter 2, the Service Oriented Architectures which is used in the UCLP system, including Jini, Web Service and Grid Service, are introduced and compared. Chapter 3 presents the UCLP system architecture from both intra-domain and inter-domain perspective. The physical architecture of CA*net 4, where the UCLP system runs, is also given as a background. Chapter 4 briefly introduces the definition of users and functions in the UCLP system. Part 3 presents the research work on UCLP security. Chapter 5 reviews several security technologies. Chapter 6 focuses on authentication, integrity and confidentiality. Requirement analysis and evaluation on the current system are presented. Possible improvements are proposed at the end of this chapter. Chapter 7 focuses on access control. It starts with identifying difficulties. The review on access control models follows. We then analyze the requirement and propose solutions for access control on different resources in the UCLP system. For the Light Path Object, three different ways to implement Discretionary Access Control model are given. The comparison shows that the one based on Attribute Certificate is the best. Finally, Part 4 concludes the thesis.

Chapter 1 Network Management and Customer-Owned Optical Networks

From the perspective of network management, the functions provided by network devices or the whole network are separated into three logical planes: Data Plane, Control Plane and Management Plane (Figure 1). Theoretically, the interconnections of these planes should have strict definitions so that it allows for the development of systems where one or more layers can be implemented independently.

Data Plane is also referred as User Plane or Application Plane. This plane enables the rapid introduction of new services via standardized open interfaces to the service logic and data. The goal is for a wide variety of applications (voice service, data service and more) to be capable of being supported within this architecture.

The Control Plane manages the routing of the traffic in the Data Plane, allocating adaptation and switching resources, facilitating set up and tear down of connections. Signaling is a major component of this plane and support for multiple protocols is essential. Therefore, this plane is sometimes also called Signaling Plane.

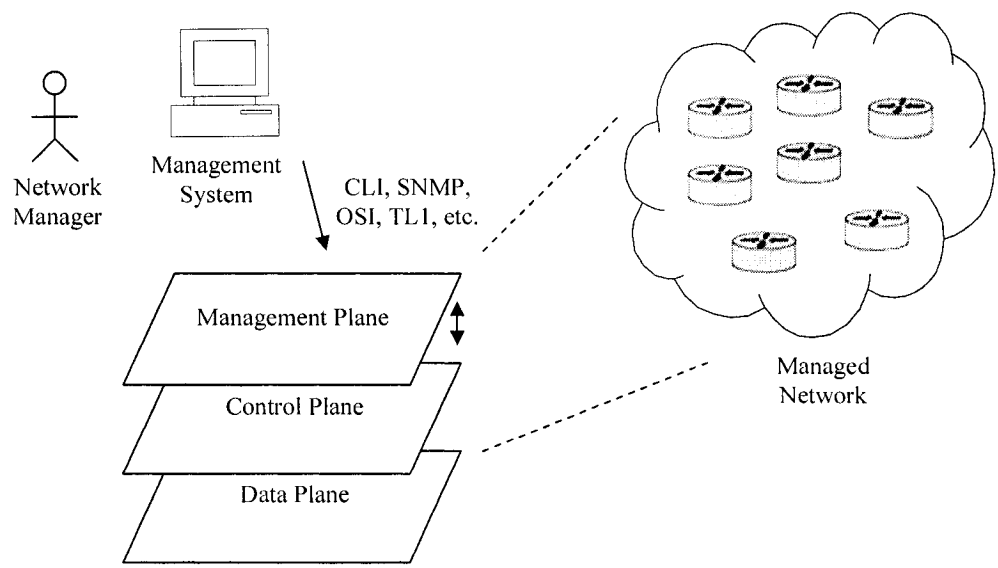


Figure 1. Management system and conceptual device planes [38]

The Management Plane performs management functions for an overall system and provides coordination among all the planes. Under the direction of the network manager, the management system interacts with the Management Plane to achieve all kinds of management functions.

This chapter is organized as follows. In Section 1.1, the basic functions of typical network management systems are briefly introduced. Section 1.2 focuses on the architecture of network management systems. Several prevalent architectures are presented. Section 1.3 introduces the characteristics of customer-owned optical networks and reviews the requirements and challenges of the management system for such kind of network.

1.1 Network Management Functions

Normally the network management functions are categorized into five areas: configuration management, fault management, performance management, security management and accounting management. This definition was originally proposed by the International Standards Organization (ISO) in their Open Systems Inter-connection (OSI) network management standard [21]. It has been adopted by many organizations as a basis for their network management system.

1.1.1 Configuration Management

The key concept in configuration management is the Managed Object. A managed object is a conceptual view of a resource that can be managed, such as switches, routers or wavelengths in optical Wavelength Division Multiplexing (WDM) networks or time-division channels in Synchronous Optical NETworks (SONET). The mapping between a resource and a managed object does not have to be a one-to-one relation. A resource can be modeled by more than one managed object. In this case, each managed object represents a different abstract view of the resource. On the other

hand, a managed object may also represent a relationship between different resources, e.g. a cross connection between two ports on a switch.

Configuration management is used to identify and control managed objects. By initializing, operating, reconfiguring and closing down the managed objects, the configuration management facility can make the logical representation of the network reflect the status of the real network when something happens, e.g. new devices join in or hardware failures occur on existing devices. More specifically, configuration management is responsible for the following activities:

- To identify any managed object and the assignment of names to the object,
- To introduce any new managed object,
- To set the initial values for the attributes of objects,
- To manage the relationships between managed objects,
- To change the operational characteristics of managed objects and reports on any changes in the state of the objects,
- To delete managed objects.

1.1.2 Fault Management

Fault management includes the following three sub-functions:

- Fault detection: Faults can be detected by monitoring or through error reports generated by the affected object
- Fault diagnosis: By implementing diagnostics functions on managed objects, faults can be diagnosed. Diagnostics functions include error reproduction, error analysis and error report collector.
- Fault correction: By utilizing Artificial Intelligence (AI) technology, such as expert systems, fault correction can sometimes be automated. Normally the decision made by the AI module is executed by using other facilities, such as the configuration management facility.

Log control is an important part of the fault management facility. Most of fault management functions rely on log functions, such as error tracing and time stamping of messages.

1.1.3 Performance Management

The purpose of the performance management is to keep the performance of the network at a satisfactory level. Performance management functions can be grouped into three categories: monitoring, analysis and tuning.

Performance monitoring includes the collection of basic performance data monitoring and workload monitoring. Basic performance data includes throughput, response time, propagation delay and so on. Workload monitoring takes these data and put them into a workload model to evaluate the workload level of the managed objects or the whole network. Workload model could be as simple as a bunch of threshold values. For example, once the response time becomes larger than a certain threshold, the network will be considered to be overloaded.

The analysis function uses statistical calculations to produce some more meaningful performance data, such as utilization, availability, error rate, failure probability and so on.

Based on the performance data collected by monitoring functions and produced by analysis functions, performance tuning functions can make the network working more efficiently. One example of network performance tuning is traffic engineering. What it does is basically choosing the best path for network traffic according to the historical network usage statistics so that the overall performance of the network can be improved.

1.1.4 Security Management and Accounting Management

Security management is concerned with protecting the managed objects against unauthorized and malicious access. It handles authentication procedures and enforces access control rules by using security technologies, such as encipherment and digital signatures.

Accounting management defines how network usage, costs and charges are to be identified. It allows managers and users to place limits on usage and to negotiate additional resources when required.

1.2 Network Management Architectures

This section introduces four network management architectures: centralized, hierarchical, distributed [22] and networked.

1.2.1 Centralized Architecture

In a centralized network management system, there is only one single machine which takes care of the entire network. As shown in Figure 2, each managed network device is associated with a network management agent. The network management station collects the information and controls the managed devices by communicating with agents using some network management protocol, such as Simple Network Management Protocol (SNMP) or Common Management Information Protocol (CMIP) [21]. The collected information is stored in a centralized database.

The disadvantages of this simple approach are obvious. Firstly, it is not reliable. The management station is a single point of failure. The whole network relies on it. Even if it does not fail, part of the network could be out of management in the case of network failure, for example, when a router between the management station and certain parts of the network is incorrectly configured. Secondly, it is not scalable. When the size or complexity of the network increase, this approach becomes infeasible.

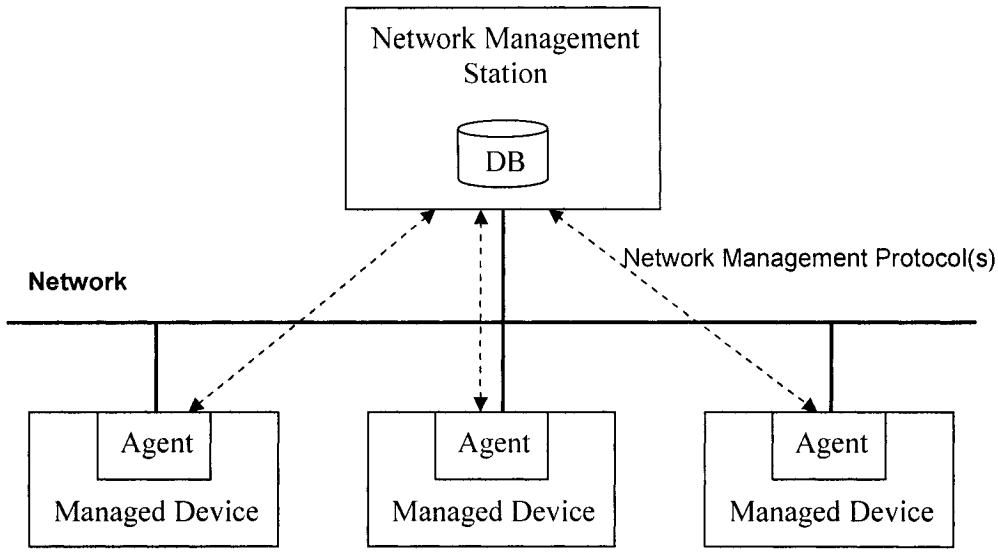


Figure 2. Centralized Architecture

A variation of the centralized approach is the platform-based approach. It divides the manager into two parts: the management platform and the management application. The manager platform collects the information and does some simple calculations. The management application handles higher level management functions by using the services provided by the management platform. In this way, applications do not need to worry about (network management) protocol complexity and heterogeneity. However, this approach still inherits the limited salability from the centralized architecture.

1.2.2 Hierarchical Architecture

The hierarchical architecture is associated with the management domain concept which is defined in the OSI standards [39]. Basically a management domain is a collection of managed objects and administered by a single organization. In the hierarchical architecture, multiple managers exist in the network and each of them is solely responsible for one management domain. They are unaware of each other and only talk to the manager of managers (MOM) which operates on a higher hierarchical level. It retrieves information from the domain managers and coordinates them.

The most significant advantage of this approach is scalability. By creating more domains and adding a respective number of domain managers, the system can be easily scaled up without losing performance. If needed, a multiple level hierarchy can be achieved by adding another level of MOM.

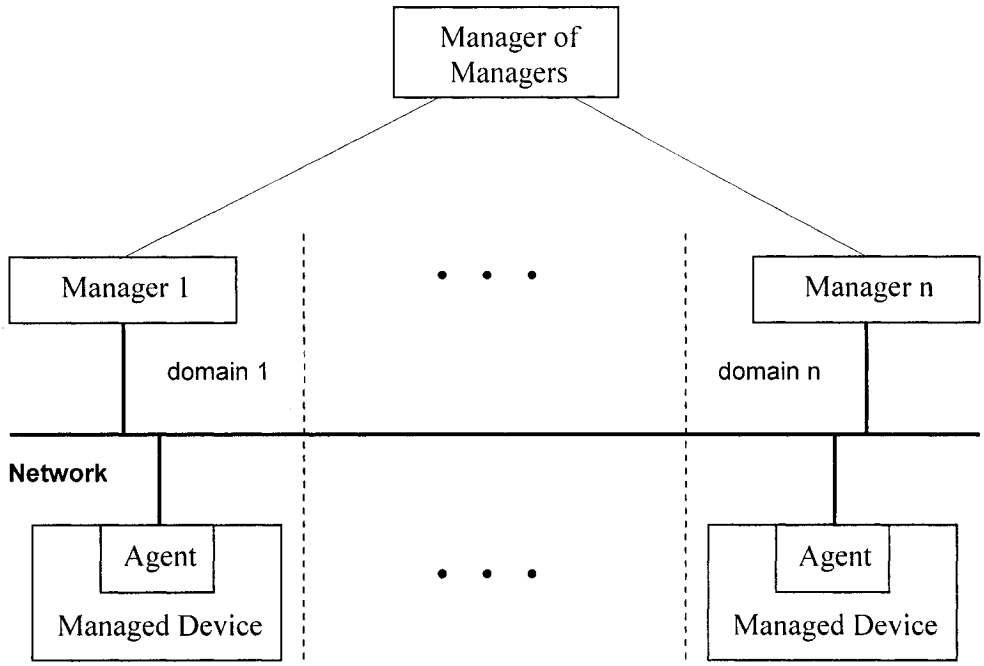


Figure 3. Hierarchical Architecture

1.2.3 Distributed Architecture

Similar to the hierarchical architecture, the distributed architecture also uses the management domain concept. The difference is that there is no manager of managers. Each domain manager communicates with the other domain managers in a peer-to-peer manner. When information from another domain is needed, the corresponding manager is contacted and the information is retrieved.

Compared with the hierarchical architecture, this approach keeps domains independent while offering the same scalability. In some cases, a MOM which coordinates all domain managers is not acceptable [20]. This approach has also been

adopted by ISO standards [21] and the Telecommunication Management Network (TMN) architecture [23].

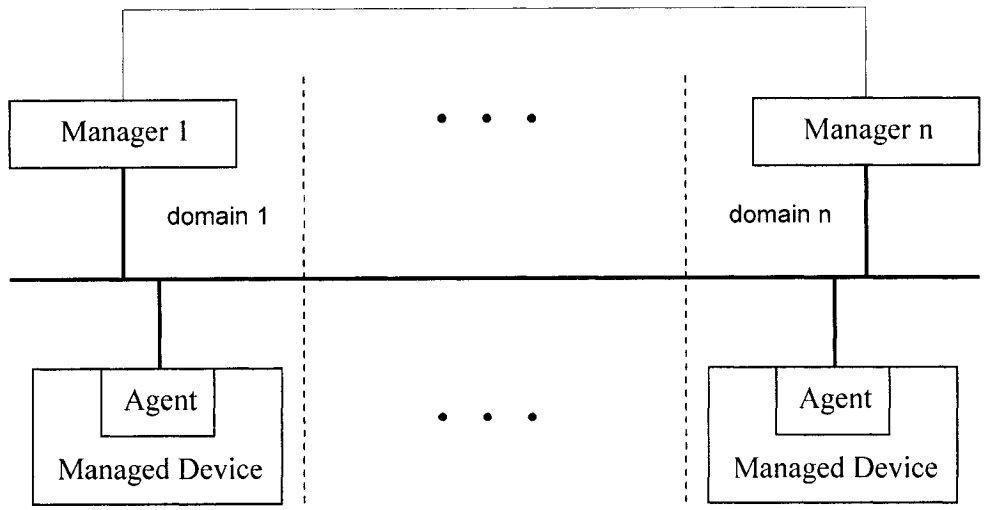


Figure 4. Distributed Architecture

1.2.4 Network Architecture

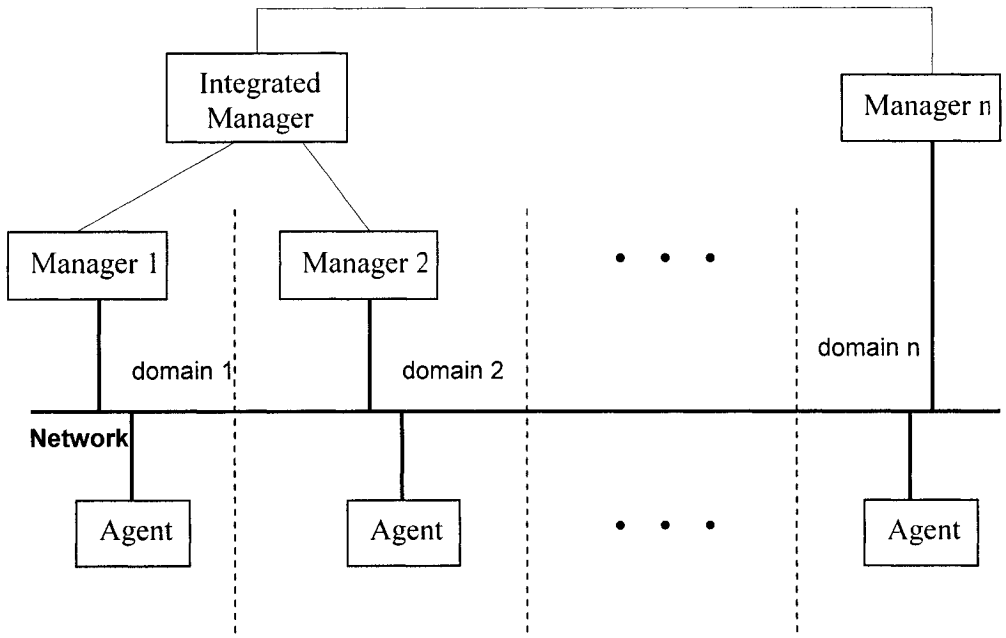


Figure 5. Network Architecture

The network architecture is a combination of the distributed and hierarchical architecture. It uses both the manager-per-domain and manager of managers concepts. It is appropriate for diverse, non-canonical environments, e.g. a network which includes both multi-level and peer-to-peer structures.

1.3 Customer-Owned Optical Network Management

This section will introduce two basic types of customer-owned optical network and their advantages as well as management challenges.

1.3.1 Dark Fiber and Long-haul Wavelength Networks

Dark fiber [40] refers to unused fiber-optic cable. Usually companies lay more lines than they need. These dark lines can be leased to individuals or other organizations (universities, hospitals and government departments) who want to establish optical connections among their own sites. In this case, the fiber is not under the control of the company who laid it. The one who leases it provides the necessary equipments to make it functional. This procedure is so-called "lighting up". Compared with traditional bandwidth leasing, this solution gives customers more control on the network. For example, customers can decide what optical protocol runs over the line and what capacity will be achieved. Hence dark fiber networks are categorized as a kind of customer-owned network.

Dark fiber networks are generally only available in high-population-density areas where plenty of fibers have been deployed already. In order to extend it, long-haul connections are needed. However, the capital costs of deploying long-haul optical networks are too high to afford for most of customers. One solution is that a number of customers share the deployment cost and the wavelength channels in the network [40]. Since the customers actually pay for building the network, they own their channels as an asset rather than a telecom service. Therefore, this is also a kind of customer-owned network.

1.3.2 Advantages of Customer-Owned Optical Networks

The advantages of customer-owned optical networks can be seen from the following four aspects.

Firstly, the cost of bandwidth is considerably reduced since it becomes a one-time investment rather than a monthly charge. Furthermore, customers can save extra costs on expanding network capacity. On customer-owned network, this can be done by applying new optical technologies or simply adding more wavelengths. It is much cheaper than renegotiation with service providers especially when the requirement of bandwidth increases dramatically.

Secondly, the customer is able to optimize the overall resource consumption. Customers obtain dark fibers or wavelength channels from different providers and use them to construct their own network. Therefore, they get more flexibility when negotiating with different providers.

Thirdly, Internet costs are reduced by remote peering and transit over customer-owned network. The Local Area Networks of customers can be connected without going through the public Internet service providers.

Lastly, since the network is owned and managed by the customer, the bandwidth and Quality of Service are determined by the customer.

To see how these advantages can be utilized by applications, refer to [41].

1.3.3 Management Challenges

The first challenge is the management of the network which consists of resources from different providers. As explained before, each customer leases or purchases dark fibers or wavelength channels to construct their own network.

Therefore, it is the customer (not the providers) who has total visibility of the whole network. From the provider's point of view, traditional network management technologies can not be applied since they use centralized or hierarchical approach and assume the provider controls the whole network. From the customer's point of view, how to manage the heterogeneous resources from different providers is really a challenging issue.

The second challenge is how to coordinate collaborations among independent customers without central management. A typical collaboration scenario is making end-to-end connection across networks belonging to different customers. To realize such a scenario, the following questions should be addressed: (1) how to search and take control of resources in other customer's networks, (2) how to authenticate each other, (3) how are authorization and access control applied and (4) what kind of trust relationships are required. For further discussion on the security issues, see Part 3 of this thesis.

The third challenge is how to make customers able to do the dynamic partitioning of the provider's resources. It is a common requirement that customers want to partition the provider's resources to optimize the traffic of their own network. This means providers have to give customers "changing" levels of access rights. As mentioned in Section 1.3.1, the provider's resources are normally shared by customers in a condominium fashion (also called condominium fiber [40]). How to help providers to manage the access control on shared resources is really challenging.

Part 2 A management system for User Controlled Lightpath Provisioning

In this part of the thesis, we introduce the User Controlled Lightpath Provisioning (UCLP) system, which is a network management system prototype for customer-owned optical networks. This prototype was built in collaboration by the University of Ottawa and Communications Research Centre[1] of the Government of Canada with partial funding from CANARIE [2], Canada's advanced Internet development organization. The purpose of this project was to verify the new concepts related to Customer-Owned Optical network, including both benefits and challenges (see Section 1.3.3).

As the key developer of the University of Ottawa team, I contributed to the design of the high level system architecture, especially the Grid Service part. For the implementation, I completed the Grid Service part and helped other developers on the GUI client and one of the Jini services. Furthermore, I made an evaluation on the major security aspects of the system and analyzed its security requirements. Possible improvements based on existing security technologies are also proposed. These are presented in the next part of this thesis.

This part is organized as follows. Chapter 2 introduces service oriented architectures which have been used in the UCLP system. The architecture of the UCLP system is given in Chapter 3. Chapter 4 summarizes the user types and functions of the UCLP system.

[1] Communications Research Centre Canada, http://www.crc.ca
[2] CANARIE Inc., http://www.canarie.ca

Chapter 2 Service Oriented Architectures

A Service Oriented Architecture (SOA) is an approach to build a distributed system that delivers application functionality as services to end-user applications or for building other services. In this context, all functions are defined as services, and each component is either a service provider or a service requester or both.

A service in SOA is normally implemented by a self-contained reusable software component. It has a well-defined interface which is invocable in a platform-neutral manner. Each service is fairly independent of other services and works as a black box. It does not depend on the state of other services, which means the internal states of other services will not affect how a service works. A composite service may request data from other services. However, how it processes the data totally depends on the service itself.

Since the service has the above characteristics, the relations between service providers and service requesters are relatively loosely coupled. The service requestor neither knows nor cares how the service provider performs its function. They can be running on totally different platforms. Compared to the traditional client-server model (for building distributed systems), SOA can make system integration much easier.

SOA is not a newly-invented term. The concept was developed over more than one decade. The Distributed Component Object Model (DCOM) [24] (by Microsoft) and the Common Object Request Broker Architecture (CORBA) [25] (by the Object Management Group) can both integrate applications on disparate heterogeneous platforms. More recently, Jini (by Sun) and Web Services (by the World Wide Web Consortium) have been promoted along with Java and XML. This chapter will briefly introduce Jini and the Web/Grid Service architecture (because they are used in the UCLP system) and make a simple comparison.

2.1 Jini Architecture

Jini is a SOA based on Java. As mentioned in the "Jini Architecture Specification" [26], a Jini system should not be thought of as sets of clients and servers, users and programs, or even programs and files. Instead, a Jini system consists of services that can be collected together for the performance of a particular task. Each Jini service has an interface defined in the Java programming language. It can be called through the Java Remote Method Invocation (RMI) which uses Java Remote Method Protocol (JRMP) as the wire-level protocol. The service implementation is also written in Java.

Unlike DCOM, CORBA or Web Services, Jini is a pure Java architecture. However, since Jini inherits hardware- and operating system- independence from the Java programming language, it does provide most of the SOA benefits. Besides, Jini also provides some useful facilities for building distributed systems, such as a service lookup service, a transaction manager and JavaSpaces.

2.1.1 The Jini Lookup Service

The Jini Lookup Service is a registry for Jini services and a primary means for finding services within a Jini system. It interacts with the Jini services or clients using three protocols: discovery, join and lookup.

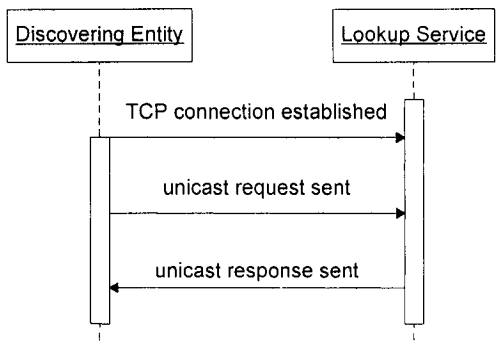


Figure 6. Unicast Discovery

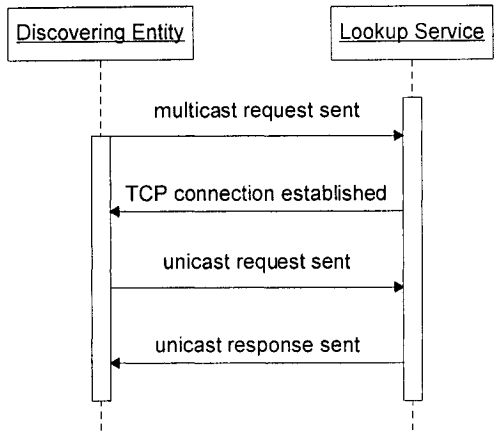


Figure 7. Multicast Discovery Scenario 1

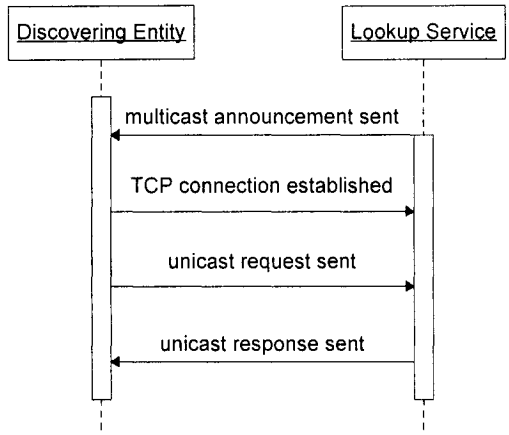


Figure 8. Multicast Discovery Scenario 2

Discovery occurs when a Jini service provider or user is looking for a lookup service. The Jini discovery protocol is very flexible and supports both unicast and multicast. Figure 6, Figure 7 and Figure 8 show the three typical discovery scenarios. If the discovering entity knows where the lookup service is, it can use the unicast discovery (Figure 6). If it does not, it can use the multicast discovery (Figure 7). Even if the lookup service is not started in advance, the discovery can still be triggered by the multicast announcement sent by the lookup service when it is starting (Figure 8). In order to be distinguished by the discovering entity, each lookup service is associated with a group name. The discovery entity can specify a group name in its discovery request.

Join occurs when a service has located a lookup service and wishes to register its service in it. The service object which contains the service interface and other necessary information for making invocation (e.g. a RMI stub object which implements the service interface) will be loaded into the lookup service along with any descriptive service attributes.

Lookup occurs when a client or user needs to locate and invoke a service described by its interface type (written in the Java programming language) and possibly other attributes. The RMI stub object (also called proxy) of the matched service will be returned to the client by the lookup service. As mentioned before, the proxy implements the service interface. The major task of the proxy is to translate local invocations to the Java RMI invocations, forward them to the real service, and recuperate the result of the invocation. All this is transparent to the client. The proxy is actually a local service provider to the client.

Since the Jini Lookup Service is also a Jini service, it can join other lookup services too. A service registry hierarchy can be constructed in this way.

2.1.2 Jini Transaction

Transactions are an important tool for distributed systems. A transaction can group a set of operations together in such a way that they either all succeed or all fail. Jini follows the traditional two-phase commit protocol [27] and provides a reliable transaction manager implementation.

The Jini transaction manager is also a Jini service. It can be invoked through the common procedure: discovery, lookup and invoke. It allows clients to create transactions and/or to join in a particular transaction. Any Jini service or client can become a transaction participant by implementing the Jini transaction participant interface. As a participant, the service or client can force the transaction to abort or commit by sending a request to the transaction manager. Then all the participants of the transaction will be triggered by the invocation from the transaction manager and

execute their "abort" or "commit" operation which has been defined in the transaction participant interface.

2.1.3 JavaSpaces

JavaSpaces is a Jini service which provides distributed persistence for Java objects. The basic functions are writing an object into a JavaSpace and looking up an existing object. When doing look-up, a template should be given to the JavaSpace. The template must be of same type as the target object. Lookup criteria can be specified by setting the values of the attributes in the template. If an attribute is left as "null", the JavaSpace will ignore it when looking for matching objects.

All operations that modify a JavaSpace are performed in a transactionally secure manner. The JavaSpaces service has already implemented the Jini transaction participant interface. The only thing left to do is simply passing a transaction as a parameter to the JavaSpace operation.

The most important advantage of the JavaSpaces service is its simplicity. The JavaSpace API is much easier to use than JDBC, the Java database connector API. Also it will take considerable efforts to transform a Java object to an entry in a SQL database. As a trade-off, JavaSpaces does have some functional limitations. It only supports exact matching and only returns the first matched object instead of a set of objects that match the template.

2.2 Web Service

Web Service [28] is a SOA using service specifications and protocols based on XML. The definition of Web Services has three cornerstones: SOAP, Web Service Description Language (WSDL) and "Universal Description, Discovery and Integration" (UDDI).

SOAP [29] was originally an acronym for Simple Object Access Protocol. The name has been abandoned since it does not reflect what SOAP really is any more. Basically, SOAP is a Remote Procedure Call (RPC) protocol based on XML. It is used to carry messages for Web Services. The body of a SOAP message normally includes a RPC request with input parameters or a RPC response with the returned values. It is not specified what protocol should be used to transport SOAP messages. In practice, however, the HyperText Transfer Protocol (HTTP) has become the most popular choice because it is a protocol of choice for Web applications.

WSDL [30] is used to define Web Service interfaces. A typical Web Service interface definition in WSDL may include 6 parts: types, messages, port types, bindings, ports and service. In the "types" section, the data types that are used in the interface (including parameters and return values) are defined. It is suggested to use an XML Schema [31] to define data types in WSDL. The "messages" section defines the abstract messages which are used in the operations, including requests and responses. It specifies the type and number of the parameters and their order. The "port type" section defines abstract operations and associates them with request and response messages defined in the "messages" section. In the "binding" section, the concrete protocols for the operations and messages are specified, e.g. SOAP over HTTP. The "ports" section specifies an URL address for each binding and the last section "service" aggregates a set of ports used by the service.

UDDI [32] is the major means to lookup Web Services. At first, it aimed at a simple model and proposed a public, global Web service registry like a phone book. This turns out to be an unrealistic approach. The author of the new version of UDDI has put more efforts on the interactions between registries. Several registry types are proposed, such as public registry, private registry and semi-private registry.

Web Service is a truly platform-neutral and programming language-neutral architecture because all specifications about Web Service are based on XML and XML has been well accepted as a standard for the exchange of data across different platforms. This is also the most important advantage of Web Service.

2.3 Grid Service and Globus Toolkit

Grid Service is a concept defined in the Open Grid Service Architecture (OGSA) [33]. It is an extension of the Web Service architecture intended for Grid computing [34]. Grid computing is a hardware and software infrastructure that clusters and integrates high-end computers, networks, databases and scientific instruments from multiple sources to form a virtual supercomputer on which users can work collaboratively.

Typical Web Services are stateless and transient. In grid computing, services are usually required to be persistent or have states. The existing Web Service standards does not address about the service life-cycle and state management. The OGSA standard extends the Web Service and provides more choices on these service characteristics. For Grid Services with states, it follows the "factory" pattern [35] to handle the creation of multiple instances of the same service. Each instance holds a set of service data (states) which is also defined in the WSDL service description.

The Globus Toolkit [36] is an open source software toolkit for building grid application and service. Starting in version 3, it follows the OGSA. The OGSA implementation provided by the toolkit (called GT3) is written in Java. By extending the skeleton class that comes with GT3, Java programmers can easily develop their own OGSA-compliant Web/Grid Services. GT3 also includes a bunch of WSDL-to-Java and Java-to-WSDL utilities. Even programmers that have no knowledge about WSDL can develop Web/Grid Services with GT3. However, manually editing WSDL files may be needed when complex types are used, such as arrays and customized types. The GT3 utility might not be able to convert them appropriately.

2.4 Comparison between Jini and Web Service

Generally speaking, Web Service is appropriate for more static environments over wide area networks and Jini is appropriate for a more dynamic environment over local area networks. They are both qualified as a SOA and provide the benefits which comes with an SOA.

Compared to Jini, the most significant advantage of Web Service is that it is programming-language-neutral. This is an important feature, especially when the ability of integration with legacy systems is a serious concern. Although Java is able to run everywhere, it is still a fairly new technology. There are plenty of legacy applications written in C, COBOL, Fortran and other languages.

Compared to Web Service, Jini has a few advantages under certain circumstance. First, the Jini service discovery is more flexible than Web Service when it is being used within the multicast boundary. Second, Jini has much better transaction support than Web Services. The Web Service Transaction specification [37] is still being defined. Only alpha versions of implementations are available on some business software platforms, such as the WebSphere Application Server from IBM. The Jini transaction specification has been defined for sometime and Sun provides a reliable implementation for free under their Sun Community Source License (SCSL). Thirdly, Java RMI has better performance than SOAP. When computing power is insufficient, e.g. on handheld device, Jini has a much better chance to be applied.

Chapter 3 The UCLP System Architecture

The UCLP system consists of a set of Jini services and Grid services. This chapter introduces the UCLP system architecture and explains how it takes advantages of both Jini and Web Service architectures.

3.1 CA*net 4 and UCLP

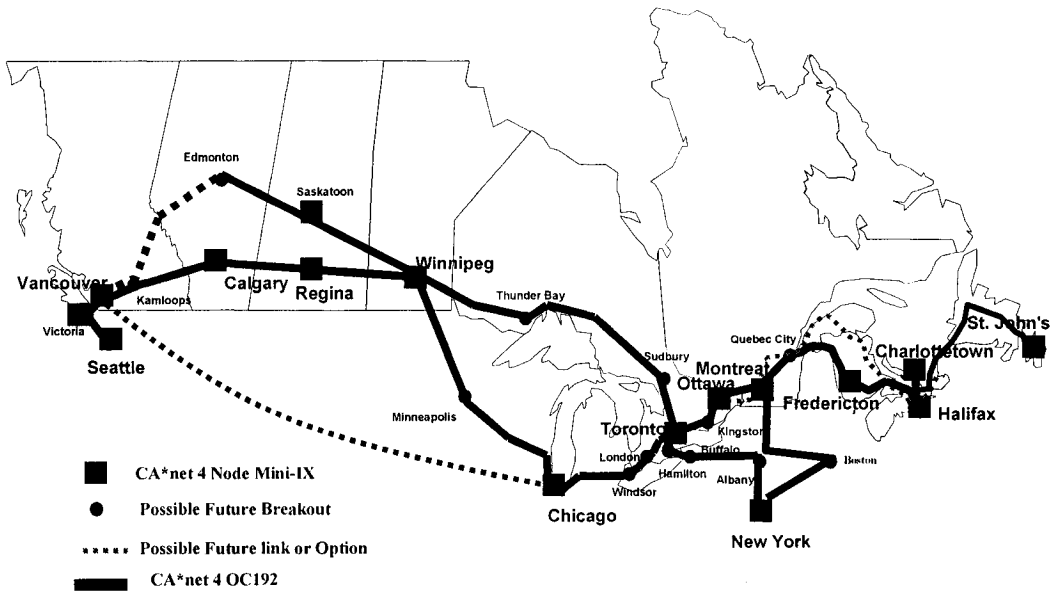


Figure 9. CA*net 4 Topology

CA*net 4 is Canada's national optical Internet research and education network. It interconnects the provincial research networks, universities, research centres and government research laboratories. As shown in Figure 9, the main body of CA*net 4 consists of 16 optical switches. Most of the connections between these switches are provisioned at OC-192 (10Gbps) speeds.

To prove the concept of customer-owned optical network on CA*net 4 using the UCLP system, we defined 14 federations (Figure 10), which are considered as independent administrative domains. The following sections will introduce the UCLP system architecture from both intra-domain and inter-domain aspects.

3.2 Intra-Domain Architecture

It is mentioned in Chapter 3 that each customer-owned optical network is an independent administrative domain (also called a federation in the UCLP system). Within each domain, the UCLP system includes four layers (Figure 11).

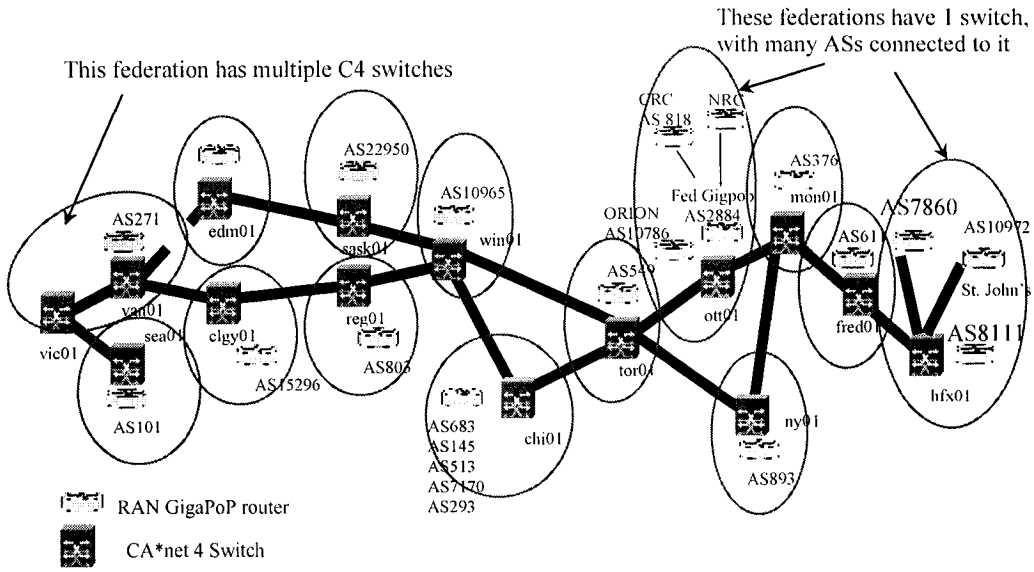


Figure 10. Federations in CA*net 4

3.2.1 Layered Service Structure

The bottom layer is the **Device Management Layer** which talks directly to network devices, such as optical switches. It controls these devices according to the requests from the upper layers. It is also responsible for listening to the events returned from devices and forwarding them up when necessary, e.g. network failures.

The **Resource Management Layer** provides all functions about lightpaths. It can create, delete, partition and concatenate managed lightpaths according to the request from the upper layers.

The **Customer's Management Layer** provides functions that utilize lightpaths. For example, it can create different kinds of connections for customers in different ways (lightpath searching criteria) using the available lightpaths.

Since the **Customer's Management Layer** is implemented as a Jini service, the UCLP system can only talk to clients written in Java. To make it able to interact with applications written in other programming language, a Web/Grid service is implemented as a programming-language-independent interface on top of the Customer's Management Layer. Since grid applications usually demand private network channels with high bandwidth, Grid users are the most foreseeable customer of the UCLP system. They must be happy to have a UCLP Grid interface, which can make it easier to integrate the UCLP system with grid applications.

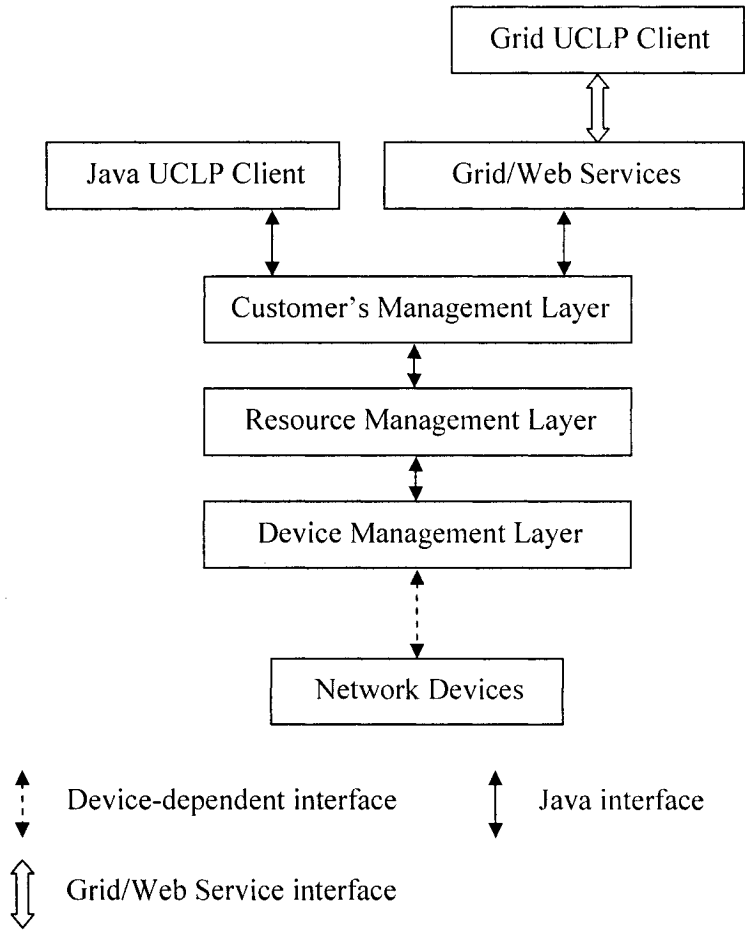


Figure 11. Layered Service Structure

3.2.2 Resource Hierarchy

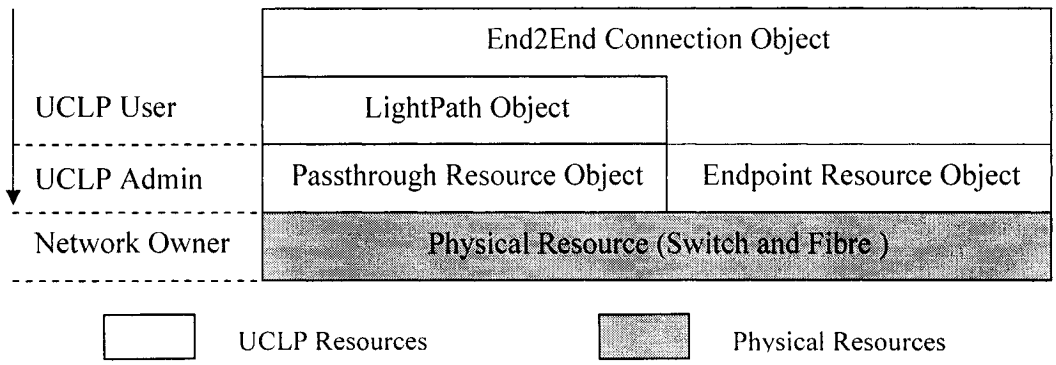


Figure 12. UCLP resource security model

Along with the layered service structure, a resource hierarchy is defined (Figure 12). At the bottom of the hierarchy there are physical resources. Physical resources are those optical network facilities which physically exist, such as fibers, add-drop multiplexers, ports, channels and slots in switches. Above the physical resources there are different kinds of UCLP resources. UCLP resources are logical objects which represent physical resources or their combinations. Physical resources are under the control of the traditional optical network management and can not be utilized by the UCLP system before being allocated into the UCLP system as UCLP resources.

The following UCLP resources are defined:

- **Passthrough Resource Objects/Endpoint Resource Objects**
  These two kinds of resource objects represent a port on a switch and the resources associated with that port. Passthrough ROs are associated with those ports for the East/West portion of the switch and Endpoint ROs are associated with some endpoints or facility where data is added into or dropped off the network. The ROs are created and managed by the **Device Management Layer**.

- **Light Path Object (LPO)**

  An LPO is an abstraction of one or more lightpaths with a set of attributes that represent a connection between two switches. It is always associated with two passthrough RO which represent the ports that are occupied by the lightpath on the source and destination switch. LPOs are created and managed by the **Resource Management Layer**.

- **E2E Connection Object**

  An E2E connection object represents the end-to-end connection which consists of one LPO and two Endpoint ROs (could be one if it is a connection for peering). E2E Connection Objects are created and managed by the **Customer's Management Layer**.

To store these UCLP resources, a persistent object repository is required. Naturally, the JavaSpace service which comes with the Jini framework becomes a candidate. And the simplicity of the JavaSpace service makes it perfect for such a proof-of-concept system. If the system is evolving into a real production system, the JavaSpace service may be replaced by some commercial database product without too much effort.

3.2.3 Detailed Service Architecture

Figure 13 shows the detailed UCLP system architecture which includes following components:

A **Grid Service Access Point (GSAP)** is the main access point of the UCLP system. It provides UCLP functions to users through two Grid Services, the User Function Service and the Admin Function Service. A GSAP implements the same service interface as the **Customer's Management Layer**.

The **Jini Service Access Point (JSAP)** implements the functionality of the **Customer's Management Layer**. It serves Java clients and GSAP through its Java application interface.

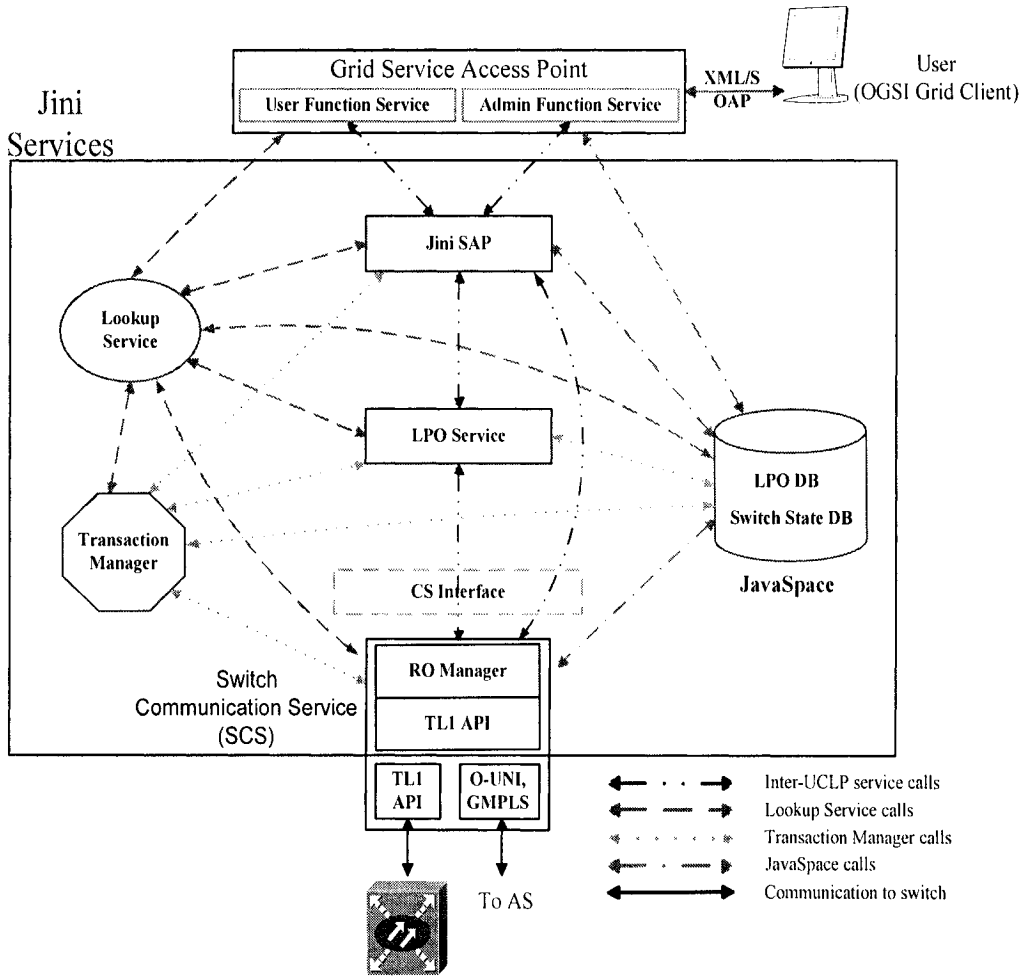


Figure 13. UCLP System Intra-Domain Architecture

The **Lightpath Object Service (LPOS)** performs functions on LPOs. It also makes the appropriate calls to the SCS to set up or tear down E2E connections for the JSAP. LPOS implements the **Resource Management Layer**.

The **Switch Communication Service (SCS)** communicates with switches and makes cross-connections within switches. The LPOS talks to the SCS through the **Communication Service (CS) interface** which defines abstract operations for controlling a switch. For different types of switches, different SCS implementations are required. SCS belongs to the **Device Management Layer**.

The **JavaSpace (JS)** service provides persistent storage for UCLP resources.

The **Transaction Manager** manages the Jini transactions that are used by the UCLP Jini services.

The **Jini Lookup Service (JLS)** is the service registry that helps the services above to find each other. Within one domain, multicast discovery should be sufficient for locating an appropriate JLS for service registration and look-up.

3.3 Inter-Domain Architecture

As mentioned in Section 1.3.3, centralized management is not feasible for customer-owned optical networks. Therefore, the distributed architecture (Figure 4) was selected for the UCLP system. Each domain/federation communicates with each other in a peer-to-peer manner. Since the multicast discovery probably does not work across federations, a separated facility called "federation manager" is added to help federations find each other using unicast discovery.

The federation manager is also a JLS and runs on a fixed network address as a Lookup Service where federation JLSs are registered. Any newly started federation JLS registers to the federation manager and retrieves information about other federation JLSs. Also the other federation JLSs will receive a notification from the federation manager and download the information about the newly-joined federation JLS. Thus every federation knows where the other federations are and a peer-to-peer system is maintained.

3.4 Service Interactions

This section introduces how the services from different federations work together and provide functionalities to end users. Creating E2E connections is taken as an example. Suppose User A wants to create a connection between Federation A

and C. Federation B connects to both A and C. The following is the service interaction sequence:

Step 1. User A accesses the GSAP(A)* and is authenticated by a user name and password.

*Note: GSAP (A) means the GSAP in Federation A.

Step 2. User A sends a request to set up an E2E connection between Host A in Federation A and Host C in Federation C. The GSAP(A) passes the request to the JSAP(A).

Step 3. The JSAP(A) finds the route (A-B-C) and the LPOs required to get from Federation A to Federation C by querying JavaSpaces in these federations. Once finished, the JSAP(A) will have all the LPOs required to make the connection, e.g. a LPO1 between Federation A and B and a LPO2 between Federation B and C.

Step 4. The JSAP(A) asks LPOS(A) to make a connection using LPO1 and LPO2.

Step 5. The LPOS(A) uses the SCS(B) which talks to the physical switch in Federation B to make the cross connect between LPO1 and LPO2.

Step 6. Once the concatenation between LPO1 and LPO2 is done, a new LPO is created (LPO3 (A-C)).

Step 7. The LPOS(A) calls the SCS(A) to make the cross connect between Host A and LPO3 at the Switch in Federation A.

Step 8. The LPOS(A) calls the SCS(C) to make the cross connect between Host C and LPO3 at the Switch in Federation A.

3.5 Summary

The UCLP system architecture takes advantages of both Jini and Web Services. By using the Jini architecture, the UCLP system gains the following benefits:

- Flexibility of service discovery within federations
- Support for reliable transaction processing
- Persistent Java object storage

By using Web Services in the GSAP, the UCLP system gains interoperability with applications written in programming languages other than Java.

The well defined resource hierarchy and generic Communication Service (CS) interface make the UCLP system able to manage heterogeneous resources on different networks (the first challenge mentioned in Section 1.3.3). The resource hierarchy is quite flexible to match resource models of different optical networks. The CS interface abstracts control operations of different equipments and network management interfaces (O-UNI, GMPLS). For the other two challenges, see Part 3 of this thesis.

Chapter 4 UCLP User Types and Functions

4.1 UCLP User Types

Within each federation there are two kinds of UCLP users, "admin" users and "normal" users. An admin user is a super-user who is responsible for user and resource management. Resource management includes two major tasks. One is creating Passthrough/Endpoint Resource Objects and LPOs based on physical resources obtained from network owners. The other is distributing created UCLP resources among normal users within federations.

A UCLP normal user could be an end-user or a value-added service provider. What they mostly do is creating E2E connections using LPOs and Endpoint ROs obtained from admin users. Besides that, they need to manage their free resources by leasing and concatenating/partitioning their LPOs. For example, if a user needs a connection from Toronto to Montreal and does not have such a LPO, he may try to find a LPO between Toronto and Ottawa and concatenate it with a LPO between Ottawa and Montreal. Another possible approach is to find an available longer LPO which goes through X-Toronto-Ottawa-Montreal-Y. The user can cut it into three LPOs (X-Toronto, Toronto-Montreal and Montreal-Y) and take the middle one.

The following diagram shows the UCLP resource hierarchy. The resources which normal users can manage are LPOs and E2E Connections. Admin users can go deep down to Passthrough ROs and Endpoint ROs. As the resource provider to the UCLP system, network owners only deal with physical resources.

4.2 UCLP Functions

The UCLP system offers the following functions to all users:

- Create/Delete/Query/Sublease E2E Connections

- Query LPOs
- Query/Modify user's own profile

The content of a user profile includes the user's basic contact information as well as a userID/password pair. The subleasing function is used to make the LPOs associated with a temporary unused E2E connection available to other users. The connection will still logically exist in JavaSpace although it has been physically torn down. Once the subleasing period passes, the connection will be restored for the original user.

The following functions are only available for admin users:

- Create/Delete LPOs
- Create/Delete ROs
- Query switch
- Create/Delete/Query/Modify any user profile belonging to the same federation

The switch query function retrieves information about the physical resources.

As a proof-of-concept system, the UCLP system does not implement all aspects of the OSI management functions.

- Configuration Management

  Configuration Management is the major aspect of the UCLP functions. All functions related to UCLP resources belong to this category.

- Fault Management

  Fault Management is handled by the Device Management Layer (LPOS) and the Resource Management Layer (SCS). The SCS keeps listening to the switch and forwards fault messages up to the LPOS. When the LPOS gets a fault message, it will investigate who

owns the UCLP resources related to the failed physical resource. Then it will notify the related users according to the contact information contained in their user profiles.

- Security Management

  See Part 3.

- Performance Management and Accounting Management

  These two aspects are not addressed by the UCLP system.

Part 3 UCLP Security

The security problems are one of the challenges discussed in Section 1.3.3. In this part, the security aspects of the UCLP system are analyzed in relation with existing security technologies. Furthermore, three approaches dealing with the challenges relating to access control are proposed.

This part is organized as follows. Chapter 5 introduces existing security technologies which could be useful for the UCLP system. Chapter 6 focuses on authentication, integrity and confidentiality and proposes improvements to the UCLP system in these three fields. The last chapter is about authorization and access control, the hardest part of the UCLP security.

Chapter 5 Existing Security Technologies

5.1 Kerberos

Kerberos [9] [10] is an authentication protocol for distributed systems. It is designed to provide strong authentication for client/server applications by using secret-key cryptography. Kerberos is based on a key distribution model developed by Needham and Schroeder [42]. It does not demonstrate possession of secret key by simply divulging it. Therefore, keys can not be stolen just by eavesdropping.

To apply Kerberos, the following requirements should be satisfied:

- A trusted third party authentication service should be always alive.
- Users and services who want to authenticate each other must share some secret key.

5.2 Secure Socket Layer (SSL) / Transport Layer Security (TLS)

The SSL/TLS [11] protocol is located between the Application Layer (HTTP, FTP) and the Network Layer (TCP/IP) and provides encrypted connections between authenticated parties. The authentication of SSL is based on public key cryptography. To be authenticated by some server(client), the client(server) needs to present a valid X. 509 certificate issued by a certificate authority (CA) that is included in the server's(client's) list of trusted CAs.

The two major advantages of public key authentication are:

- It does not need a third party service.
- Authentication can be done between two parties who do not share any secrets.

5.3 Jini Security Framework

Jini 2.0 comes with a pluggable implementation of the Java RMI programming model, Jini Extensible Remote Invocation (JERI) [12], which can perform RMI calls through secure channels provided by different security technologies, such as SSL/TLS, Kerberos, etc. It supports mutual authentication and allows developers to configure different protection levels for encrypted connections.

5.4 Grid Security Infrastructure (GSI)

The Globus Toolkit uses the Grid Security Infrastructure (GSI) [13] for enabling secure authentication and communication. GSI is based on public key encryption, X.509 certificates and the SSL protocol.

The Globus Tookit version 3 implements both transport layer and message layer security. The transport layer security is realized by the HTTP protocol running over a secure connection provided by SSL. The message layer security is based on WS-Security [14], XML Encryption [15] and XML Signature [16] standards.

Both the transport layer security and the message layer security of GSI provide the same security features, including mutual authentication, communication integrity and confidentiality.

5.5 Attribute Certificates (AC)

The standard X.509 **name certificate** binds a public key to a Distinguished Name. An **attribute certificate** binds an attribute to a Distinguished Name. The attribute could specify group membership, roles, security clearance or other authorization information. To be used by an application, an attribute certificate needs to be used with a name certificate in order to complete the mapping from the authorization to the key (Figure 14). Apparently, both attribute certificate issuer and

name certificate issuer should be trusted by the application or service. In some case they are the same CA.

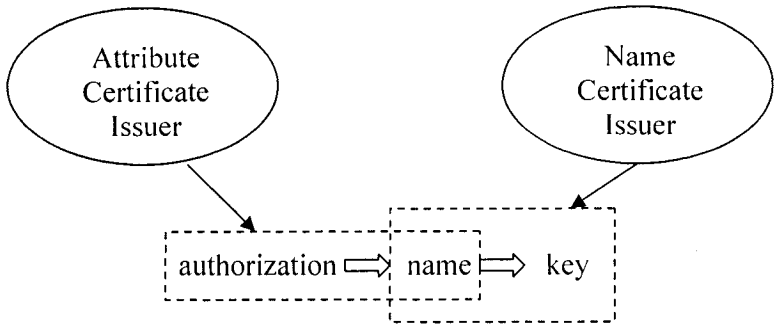


Figure 14. Name certificate and attribute certificate

Attribute certificates can be used to realize distributed access control [1]. Since the authorization information in attribute certificates is tamperproof, they do not need to be stored in a protected database. Therefore the access control becomes much easier to implement. The user who needs access to some service simply presents a set of attribute certificates to the service provider. The service provider will grant the access if there is enough authorization in those certificates. No complicated communication between service provider and the authorization service is needed. For a distributed system which has multiple administrative domains, using attribute certificates can make each domain more independent because the different domains need not share any more a centralized authorization service or database.

5.6 The Authorization, Authentication and Accounting (AAA) Architecture

The Authentication, Authorization, Accounting (AAA) architecture [4] is a generic framework which tries to cover the common AAA requirements of all Internet services. It is still in the requirement analysis and high-level design phases. However, the authorization part of this standard is already fairly well defined [5] and implemented.

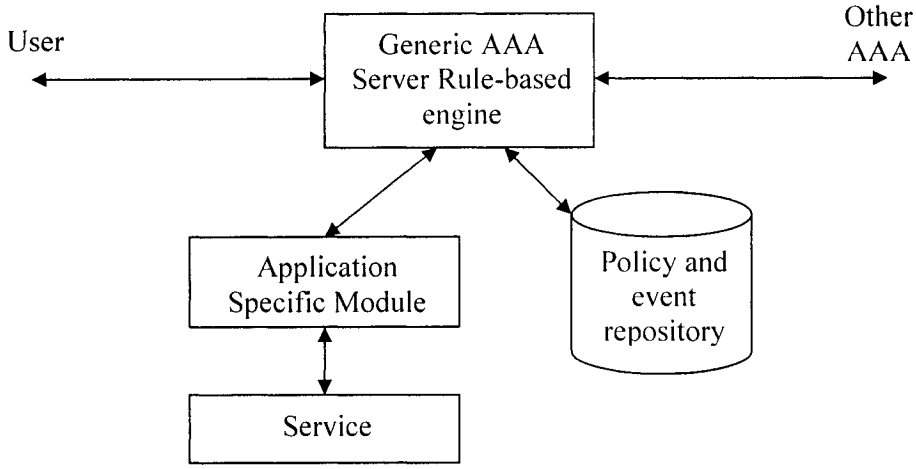


Figure 15. Generic AAA Server Interactions

The AAA authorization framework is based on AAA servers. As shown in , the AAA server accepts requests from users or other AAA servers. It needs to access a repository to get the authorization information related to the request. It interacts with the service through the application specific module. All these interactions should go through standardized protocols or interfaces. Apparently, to define such generic protocols and interfaces is a very challenging work. However, the authorization across different domains will be as easy as the one in a single domain once these standards are defined and adopted by most of Internet services. The following figures (taken from [6]) show how AAA handles inter-domain authorizations. Figure 16 shows a "pull" model. The service provider pulls authorization information from AAA servers after receiving requests from users. Figure 17 shows a "push" model. The user gets authorization information from the AAA server in advance and pushes it to the service provider when sending service requests. Figure 18 shows an "agent" model. The AAA server in the user's home domain works as an agent. It passes requests sent by the user to the service provider and negotiates with the provider on the user's behalf.

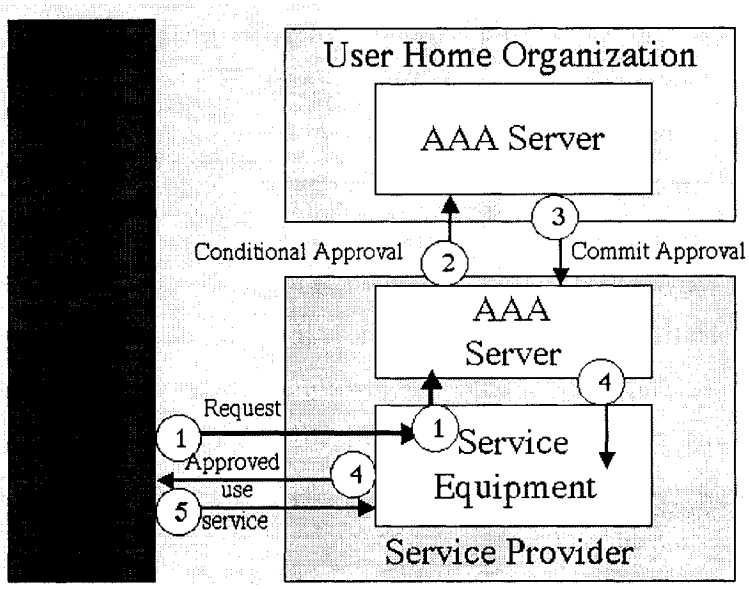


Figure 16. AAA example application 1: Mobile IP, PPP dial in

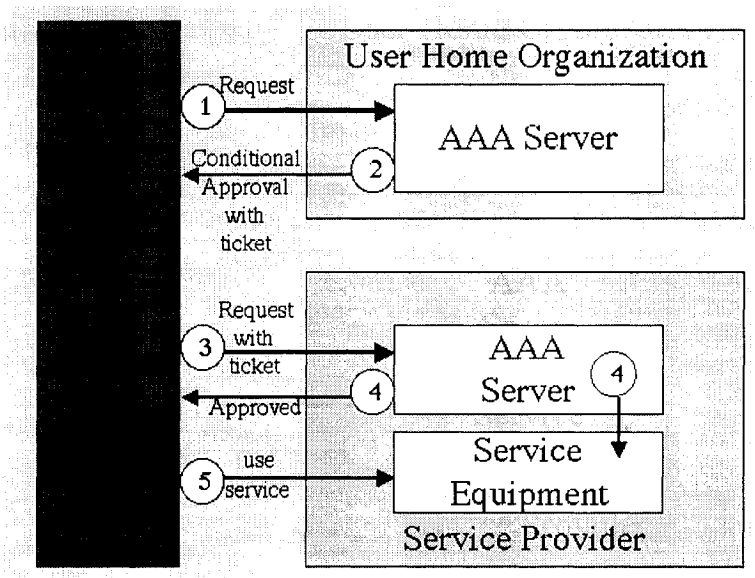


Figure 17. AAA example application 2: Internet Printing

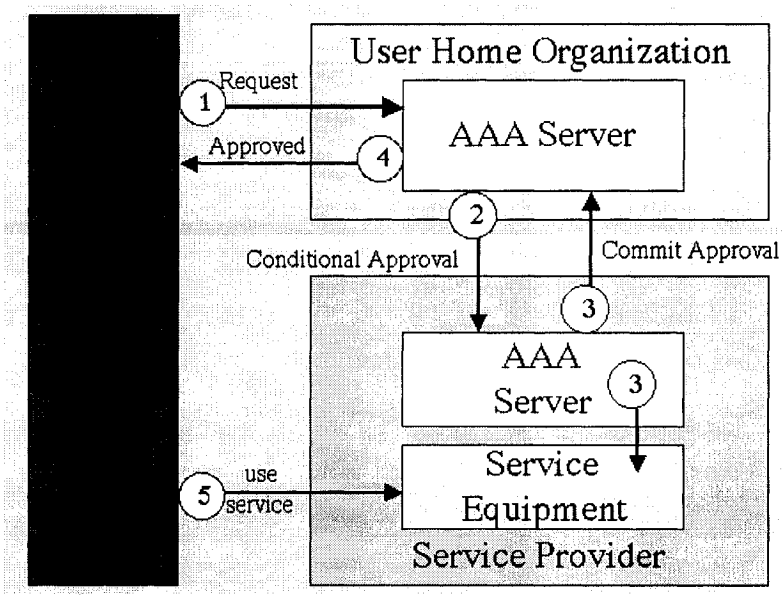


Figure 18. AAA example application 3: bandwidth brokerage at Service Provider boundary

Chapter 6 Authentication, Integrity and Confidentiality in the UCLP System

6.1 Requirements Analysis

This section gives a requirement analysis on authentication, integrity and confidentiality.

6.1.1 Authentication

Authentication is the process of determining if a user or entity is who he/she claims to be. For distributed systems, mutual authentication is a basic security requirement. As shown in Figure 13, all UCLP services (including Jini and Grid Services) and clients need to authenticate each other before they start working together. The following requirements should be satisfied if password based authentication is used:

- Passwords should not be transferred over unsecured network connections;
- Password maintenance schemes should be applied, e.g. restrictions on the length, complexity and lifetime of passwords.

Existing mechanisms based on encryption algorithms and keys are recommended, such as Kerberos, PKI and so on. These mechanisms often come with a sophisticated authentication protocol, which avoids transferring the user's password over the unsecured network.

6.1.2 Integrity and Confidentiality

There are two levels of protection on message flows. Integrity-level protection can only prevent tampering. Confidentiality-level protection can prevent both tampering and eavesdropping. Considering confidentiality-level protection needs

much more computing power, it would be nice to have it as an option. Then the actual protection level of the UCLP system can be configured according to the encryption capability of the system and the requirements for confidentiality.

6.2 Evaluation of the Current Implementation

This section gives a security evaluation on the current implementation at two different levels.

6.2.1 Basic-level Security Evaluation

The basic-level security evaluation is against those malicious users who have no sophisticated programming skills. In this context, password based authentication and the integrity protection provided by TCP are secure enough. The only loopholes in the current UCLP system (version 1.0) are:

- Client does not authenticate GSAP.
- GSAP and UCLP Jini services do not authenticate each other.

This causes the following risks:

- Anyone who has got the UCLP software can start a fake UCLP service. By doing this, they can mess up the legitimate UCLP service discovery and affect the whole UCLP system. They can also steal user passwords by starting a fake GSAP, which takes user name and password as input parameters.
- Anyone who has the interfaces of the UCLP Jini services can call them directly. They can pretend to be any user, because these calls do not go through the GSAP, which checks the user identity.

There is no neat solution for this problem without using the Jini and Grid security facilities. However, some simpler preconditions can be introduced:

- Restrict the access to servers that hosts UCLP Jini services. Only the access from registered IP addresses is allowed.
- Publish the list of the legitimate UCLP entry services on the official UCLP website.

6.2.2 High-level Security Evaluation

The high-level security evaluation is against those experienced hackers who have the ability to eavesdrop or tamper TCP packets. They may be also good at guessing passwords. Therefore, the following facts about the current implementation become loopholes at this security level.

- Password based authentication is used when the GSAP authenticates clients.
- The password is transferred over an unsecured channel during the authentication process.
- No password maintenance scheme is applied.
- No sophisticated integrity and confidentiality protection is provided.

As a result, the current implementation has the following security risks,

- Password based authentication is easy to be cracked, especially when it is used in the Grid Service or Web Service. The Grid/Web Service uses SOAP, a XML-based RPC protocol. All SOAP messages are in plain text. To pick up a password from the parameters in a SOAP call is very easy for eavesdroppers.
- The messages between services and clients could be tampered because of the lack of integrity protection.
- The private information of users might be stolen by eavesdroppers because no confidentiality protection is provided.

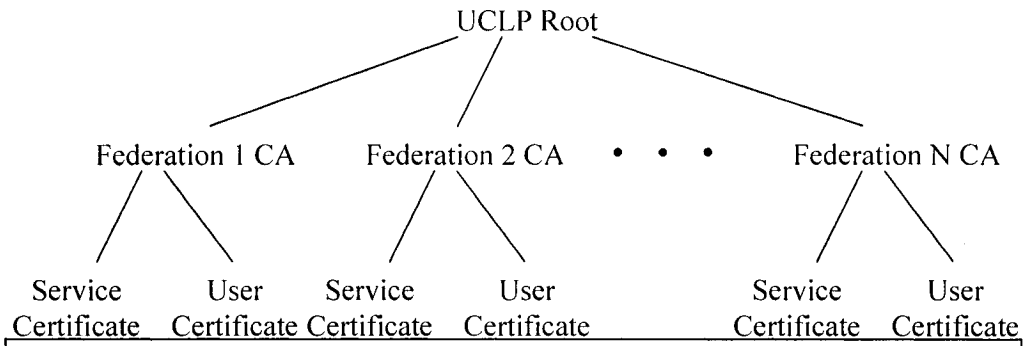
6.3 Proposed Improvements

This section gives a SSL-based approach which meets the security requirements on authentication, integrity and confidentiality.

6.3.1 Authentication

SSL is chosen to be the authentication mechanism for two reasons:

- The UCLP system is supposed to be used across multiple administrative domains. The SSL authentication protocol is based on public key cryptography. It does not rely on a centralized authentication service. It can authenticate two parties who do not share any secret. Therefore, using SSL can make each domain more independent.
- Mutual authentication using SSL is supported by both the Jini security framework and the Grid Security Infrastructure.



UCLP Root CA: C=CA, O=UCLP, CN=UCLP Certificate Authority
Federation's CA: C=CA, O=UCLP, OU=federation name, CN=Federation Certificate Authority
Service Certificate:
C=CA, O=UCLP, OU=federation name, CAT=Service, CN=service name (LPOS, for example)
User's Certificate:
C=CA, O=UCLP, OU=federation name, CAT=User, CN=username

Figure 19. UCLP certificate hierarchy and examples

Figure 19 shows a proposed UCLP certificate hierarchy. Each federation has its own Certificate Authority (CA). It issues two kinds of certificates, Service Certificates for all Jini and Grid services in the federation and User Certificates for all users of the federation. The following trust relationship should be built:

- It is necessary that each service trusts all service certificates, including those issued by CAs from other federations, since most of the UCLP operations require inter-federation calls.
- User certificates need to be recognized only by the UCLP entry service (GSAP), because users are not allowed to call UCLP Jini services directly. If the GSAP is shared by multiple federations, it should trust all user certificates of these federations.

To achieve the above trust relationships, each UCLP service has to trust all federation CAs. To facilitate their trust relationships, a UCLP root CA is introduced. This CA is only responsible for the establishment of the trust relationship between federations. It does not issue any service or user certificates. The UCLP root CA can be managed by an organization which can represent all the UCLP participants, e.g. a UCLP Alliance or Consortium.

The certificate hierarchy shown in Figure 19 reflects the current structure of the physical network, CA*net 4. In the future, other networks may join in the UCLP system at different levels. Provincial networks in Canada may join in as children federations of current federations. Networks from other country may join in as an managed element equivalent to the whole CA*net 4. Therefore, a multi level certificate hierarchy will be required. In the Astrolabe system [48], a similar structure is used for the same purpose.

6.3.2 Integrity and Confidentiality

SSL provides integrity and confidentiality protection. The Jini Security Framework and Grid Security Infrastructure (GSI) inherit these features and make them configurable in their service deployment configuration files.

Chapter 7 Authorization and Access Control for UCLP

The access control in the current implementation can be described by the following two simple rules:

- All physical resources are assumed to be free to use for all federations. Network owners can not specify which federation is allowed to access their resources.
- All free LPOs are assumed to be available to all users. There are no access control lists on LPOs.

Apparently, the current UCLP system is inadequate to overcome the second challenge mentioned in Section 1.3.3. Without access control on physical resources and LPOs, there is no way to achieve collaborations between users and federations when making end-to-end connections. In the beginning of this chapter, the major difficulties of access control in the UCLP system are identified. Section 7.2 briefly introduces several access control models. Detailed requirement analysis of both UCLP services and resources are given in Section 7.3. Finally, three different approaches to realize access control for UCLP are proposed in Section 7.4.

7.1 Difficulties

7.1.1 Network owners and users

In traditional network management systems, network owners manage the whole network and provide services to users. Users do not need to invoke management functions directly. In the UCLP system, users need to manage the resources assigned to them. Therefore, the requirements of users and network owners may sometimes conflict. Generally speaking, network owners would like to keep more management functions away from users so that they can make sure the network is secure and working properly. In the other hand, users would like to have access to more management functions so that they can utilize their resource in a more flexible

way. How many functions should be open to users is a question that deserves serious consideration.

7.1.2 Multiple administrative domains

The UCLP system is supposed to be used on the network that consists of heterogeneous network elements and multiple administrative domains. Each domain needs its own access control authority which has full control of the network resources within the domain. To access resources in other domains, it needs inter-domain authorizations from the authorities of other domains.

The inter-domain authorization is very important for the UCLP system. Most users' motivation to use the UCLP system is to create end to end connections across multiple domains and they can not do that before they get enough authorizations from all domains along the connection path. However, it is difficult to realize inter-domain authorization based on traditional access control frameworks. The ideal framework for the UCLP system must be simple, flexible and extensible.

7.1.3 Composition with value-added services

As a kind of low-level network service, the UCLP service can be the basis of other value-added services that need private connections with high bandwidth, such as a Distributed File System (DFS) over Wide Area Network (WAN) [3], high-end multimedia service (internet HDTV, remote education, ...) [47], Virtual Private Network (VPN) service and so on.

Figure 20 shows an example of a UCLP-based service hierarchy. The VPN service takes advantages of the UCLP service directly and makes a distributed file system service available in a wide-area context; the video-on demand (VOD) service uses the DFS service as a media library and the UCLP service for stream transportation; a UCLP resource broker service helps people to sell their free resources and provides more choices to UCLP users, e.g. Remote Education Service

and VOD service. The relation between UCLP services and other value-added services is a kind of multi-level lattice structure.

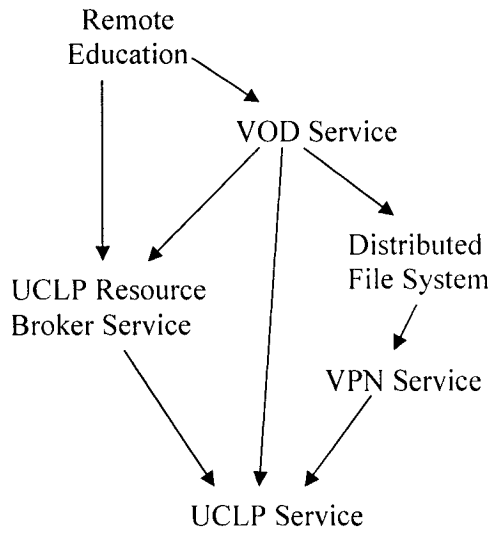


Figure 20. A UCLP-based service hierarchy

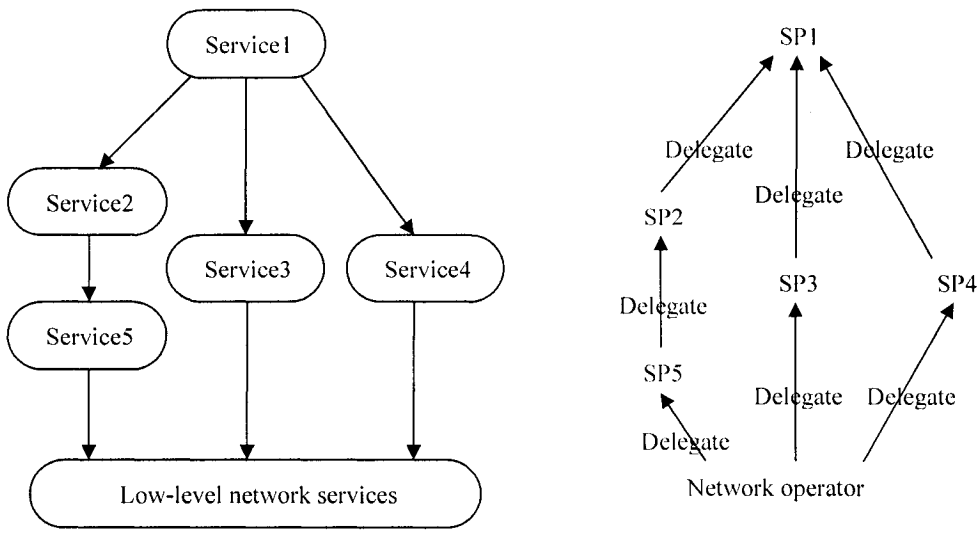


Figure 21. Service composition and delegation in AIN [2]

As another example, Figure 21 shows the service composition hierarchy in an Intelligent Network (IN) [2]. In this model, the network operator who owns the infrastructure offers network resources to service providers (SP). Service providers

can sell the right to use their services to end-users or other SPs for building more sophisticated services. The relations between the SPs form a lattice structure.

For services in this kind of structures, the question of how to enhance their security without losing interoperability is a challenging problem. The access control model should be suitable to apply to all these services. Some interesting work based on distributed Discretionary Access Control (DAC) model has been done in the context of IN [2]. It would be natural to apply the same approach to UCLP since the UCLP service hierarchy is quite similar to IN (see Section 7.4.5.3).

7.2 Access Control Models

Access control models (also called security models or protection models) provide a formalism and framework for implementing security policies in multi-user systems. Before the requirement analysis, it is necessary to briefly introduce these common access control models.

7.2.1 Mandatory Access Control (MAC)

MAC [17] mechanisms assign a security level to all information, assign a security clearance to each user and ensure that all users only have access to that data for which they have a clearance. MAC models have the following common characteristics:

- All data or resources are assigned a security level that reflects its relative sensitivity, confidentiality and protection value.
- Only administrators, not data or resource owners, make changes to the security level of a resource;

MAC provides strong protection on data and resources in a simple way while it is not flexible for users. Policies that MAC enforces are built into the design of the system and can not be altered.

7.2.2 Discretionary Access Control (DAC)

DAC [18] models make access decisions based on user identity and membership in certain groups. The owner of information can change its permission at his discretion. DAC models have the following common characteristics,

- Data Owners can transfer ownership of information to other users.
- Data Owners can determine the type of access given to other users (read, write, copy, etc.).
- Access control lists are based on user identity and group membership.

Whether DAC models are secure enough relies on how users enforce the security policies on their information. Compared to MAC, DAC is easier to be compromised because there are always some careless users who do not protect their information properly. The most typical application of DAC is in Distributed File Systems (e.g. Andrew, Coda ...).

7.2.3 Role-Based Access Control (RBAC)

In the RBAC model [19], access decisions are based on an individual's roles and responsibilities within the organization or user base. The process of defining roles is the key procedure of developing a RBAC system.

- Roles are assigned based on the organizational structure with emphasis on the organizational security policy.
- Access rights are only granted to roles, not users.
- Normally access rights of roles do not overlap each other. Role hierarchies can be established in case that some roles need to share some access rights.
- Roles can be activated and deactivated by certain events.
- Roles should not be transferred easily between users.

- The assignment of roles to users is managed centrally by an administrator.

The most significant advantage of RBAC is that it makes the administrator's work much easier. All access rights are grouped into roles, which are often associated with people's titles or positions. It makes roles much easier to memorize and understand than access right assignments based on user identity.

7.3 Requirement Analysis

A well designed access control mechanism is normally a combination of these three models. For the UCLP system, MAC and DAC are more appropriate than RBAC. RBAC is mostly used for single-administrative systems which have a complex role hierarchy, e.g. a project management system of a consulting company. In opposition, the UCLP system is spanning across multiple administrative domains.

In the UCLP system, both UCLP operations and UCLP resources need access control. For UCLP operations, MAC is appropriate. Some UCLP operations are labeled "Admin Functions" which are only allowed to be used by admin users. The others can be called by all users. For UCLP resources, the requirements are varying. Fixed rules are enough for E2E Connection Objects and Resource Objects while LPOs need Discretionary Access Control.

7.3.1 E2E Connection Object

An E2E Connection Object can not be shared since it is only useful for the user who controls the two endpoint hosts and creates the connection. Therefore, the access rule for an E2E Connection Object is quite simple:

- Only the creator can perform operations (query, deleting and subleasing) on an E2E Connection Object. Note: subleasing E2E connection does not really mean sharing the connection itself. The connection is actually torn down after being subleased and the resources are available as LPOs.

- The admin users can perform all operations on E2E Connection Objects which belong to users that are associated with the same federation as the admin user.

7.3.2 Resource Object

Resource Objects represent the physical resources which network owners distribute to different federations. The Endpoint Resource Object can not be shared for the same reason as E2E Connection Objects can not be shared. It is created by admin users and owned by the user who physically controls the endpoint. The Passthrough Resource Object is of no use before it becomes part of a LPO, so it does not need to be shared either. The admin user creates and owns all passthrough ROs. The access control rule for a Resource Object therefore is:

- Only the admin users in the same federation as the owner of the RO can perform operations (query, deleting and creating fundamental LPO) on it.

7.3.3 Lightpath Object

A Lightpath Object represents a connection between switches. It can be useful for users other than its owner. To allow users to manage the access to their LPOs, DAC should be applied.

The owner of the LPO (the grantor) can grant the access to other users (the grantee) in two ways, **Sharing** and **Leasing**. By sharing, the grantor will not lose the access. By leasing, the grantor will lose the access to the LPO during the period of the lease. Sharing normally comes with a long-term agreement between organizations. For example, university A is going to have a joint project with research centre B and they agree on sharing their LPOs to fulfill the network requirements of the project. On the contrary, leasing is normally a short-term business behavior. For example, an ISP needs a high-bandwidth connection between two particular places to solve some network congestion problem.

When granting sharing or leasing, the grantor should also specify a list of allowed operations and a validity period. The default operation is using the LPO to create an E2E connection. In addition to that, the owner may grant more LPO operations, such as concatenating/partitioning, splitting/bonding and so on.

7.4 Proposed Solutions

This section gives access control solutions for different UCLP resources.

7.4.1 Physical Resource

Access control on physical resources is required by network owners when they distribute their resources between federations. Since the UCLP system is not supposed to be the replacement of the traditional switch management software, this part of the function should not be implemented within the UCLP system. One possible solution is adding a middle layer between the UCLP system and the switches. The TL1 proxy, which was developed for CANARIE [43] for the acceptance testing of the UCLP system could be used for this purpose. When allocating physical resources into the UCLP system ("create RO" operation), the SCS should login to the TL1 proxy using the federation name. Thus the TL1 proxy can distinguish resource allocation requests from different federations.

7.4.2 UCLP Operations

The MAC on UCLP operations has been realized in GSAP. The "Admin Functions" and "User Functions" are defined in separated services (see Section 4.2). Normal users have no authorization to the "Admin Function Service".

7.4.3 E2E Connection Objects

Since the access control rules for E2E Connection Objects are fixed, they can be realized in the implementation of the operations (query, deleting, subleasing). Only if the user identity matches the owner attribute in the E2E Connection Object, the operation can be performed. This access control is already realized in the current UCLP system.

7.4.4 Resource Objects

The access control on Resource Objects can be realized in the same way as for E2E Connection Objects. However, the Resource Objects in the current UCLP system do not have an "owner" attribute. To distinguish the ROs that belong to different federations, such an attribute must be added. According to the requirements defined in Section 7.3.2, the operations such as "query", "deleting" and "creating fundamental LPO" should check whether the user identity is an admin user from the same federation as the owner of the ROs.

The transfer of RO ownership between federations needs not be supported in the UCLP system because it needs the network owner's cooperation on the physical resource access control.

7.4.5 Lightpath Objects

The Lightpath Objects need Discretionary Access Control which is much more complicated than the access control for the other UCLP resources. Three alternative approaches are proposed to realize DAC on LPOs.

7.4.5.1 Traditional DAC

The traditional way to implement DAC is quite straightforward.

- The authorization information is written in Access Control Lists (ACL) and stored with the resources in databases or file systems. The owner of the resource can edit the ACL at his or her discretion.
- An Access Control Decision Facility (ADF), also called Policy Decision Point (PDP), is made to evaluate access requests. It understands the ACL of the specific system and makes decisions according to the requester's identity, the requested access type (operation) and the ACL of the requested resource.
- The Access Control Enforcement Facility (AEF), also called Policy Enforcement Point (PEP), requests authorization from the ADF before the operation is really executed.

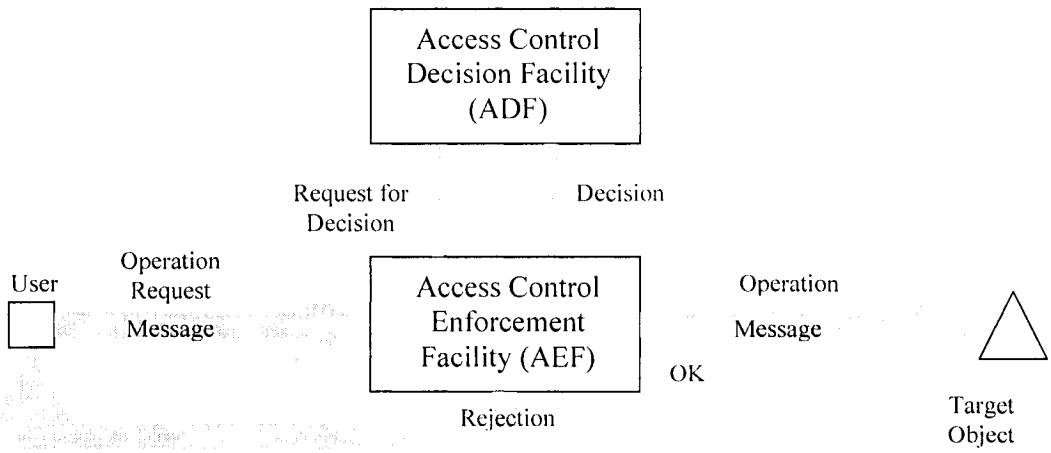


Figure 22. Traditional DAC Implementation [7]

To realize DAC on LPOs in this way, the following work needs to be done for the UCLP system,

- The format of the LPO ACL should be defined. The format should be able to describe the basic requirements mentioned in Section 7.3.3. To make it extendable, the format should be defined in a meta language.
- An LPO authorization (ADF) module needs to be implemented and integrated with the LPO Service (AEF).
- Extra operations should be implemented to allow users to modify the ACLs of their LPOs.

If XML is used to encode the ACL information, the structure of this information may be described by an XML Schema. The following is an example of a simple XML Schema for that pupose:

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
            targetNamespace=" http://www.uclp.ca/acl"
            xmlns="http://www.uclp.ca/acl"
            elementFormDefault="qualified">
   <xsd:element name="lpoACLs">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="lpoID" type="xsd:string"/>
            <xsd:element name="share" type="shareType"
                  maxOccurs="unbounded"/>
            <xsd:element name="lease" type="leaseType"
                  maxOccurs="unbounded"/>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
   <xsd:complexType name="shareType">
      <xsd:sequence>
         <xsd:element name="userID" type="xsd:token"
                     maxOccurs="unbounded"/>
         <xsd:element name="permissions" type="permList"/>
         <xsd:element name="startTime" type="xsd:time"/>
         <xsd:element name="endTime" type="xsd:time"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="leaseType">
      <xsd:sequence>
         <xsd:element name="userID" type="xsd:token"/>
         <xsd:element name="permissions" type="permList"/>
         <xsd:element name="startTime" type="xsd:time"/>
         <xsd:element name="endTime" type="xsd:time"/>
      </xsd:sequence>
   </xsd:complexType>
   <xsd:complexType name="permList">
      <xsd:sequence>
```

```
            <xsd:element name="createConnection" type="xsd:Boolean"/>
            <xsd:element name="partition" type="xsd:Boolean"/>
            <xsd:element name="concatinate" type="xsd:Boolean"/>
            <xsd:element name="split" type="xsd:Boolean"/>
            <xsd:element name="bond" type="xsd:Boolean"/>
            <xsd:element name="share" type="xsd:Boolean"/>
            <xsd:element name="lease" type="xsd:Boolean"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

The following is an example of a XML ACL which conforms to the above schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<lpoACLs xmlns=http://www.uclp.ca/acl/LPO
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://www.uclp.ca/acl/LPO.xsd">
    <lpoID>ottawa-20040125203600-15368843</lpoID>
    <share>
        <userID>crc@ottawa</userID>
        <userID>UofO@ottawa</userID>
        <permissions>
            <createConnection>true</createConnection>
            <patition>true</patition>
            <concatinate>true</concatinate>
            <split>false</split>
            <bond>false</bond>
            <share>false</share>
            <lease>false</lease>
        </permissions>
        <startTime>200402151600</startTime>
        <endTime>200403151600</endTime>
    </share>
    <lease>
        ......
    </lease>
</lpoACLs>
```

The following is an example Java interface for the authorization module:

```
interface LPOAccessController {

   public boolean check(LPO[] lpoList, String userID,
                        String[]operationList);

}
```

The traditional DAC framework is pretty simple and easy to implement. However, it is difficult to apply in a multi-domain context. Most DAC implementations assume there is only one centralized authority. As mentioned in Section 7.1.2, the UCLP system must have multiple access control authorities for different administrative domains. It is very difficult to realize inter-domain authorization if each domain authority follows the simple architecture shown in Figure 22. For example, an authorization protocol needs to be designed and implemented. Considering that inter-domain authorization happens a lot, the protocol must be simple and efficient. Even if the inter-domain authorization has been realized within the UCLP system, there is still a problem – how to make the UCLP system interoperable with value-added services (see Section 7.1.3). The access control of those value-added services could be implemented in various ways. Making all of them support the UCLP authorization protocol is almost impossible.

7.4.5.2 Distributed DAC based on AAA

To apply the traditional DAC in a multi-domain environment, a distributed architecture is required. The AAA architecture introduced in Section 5.6 is such a distributed authorization framework. As shown in Figure 16, Figure 17 and Figure 18, the inter-domain authorization problem can be solved by introducing generic AAA servers which can talk to each other in a peer-to-peer manner.

More specifically, the following work needs to be done,

- The interfaces and protocols used by the generic AAA server should be defined. The AAA Architecture work group [45] is working on these definitions.
- The ADF (PDP) module in the UCLP system should be implemented as an AAA server ().
- The format of the LPO ACLs needs to be defined in the way specified by AAA Policy Framework (work in progress).

The benefit of applying AAA architecture is not just about inter-domain authorization. It also helps on solving the service composition problem (Section 7.1.3). The Application Specific Module (ASM, see) wraps the service-specific interface so that one implementation of the policy evaluation engine can be used with all kinds of services (theoretically). This design makes it possible that value-added service providers accept the AAA architecture. For more details about ASM and AAA protocol/interface, see reference [4] [5] [46].

7.4.5.3 Distributed DAC based on Attribute Certificate

Another way to realize a distributed DAC is using Attribute Certificates (AC). Normally, ACs are signed by access control authorities. In this case, normal users can sign ACs too, so that they can control the access to their resources (as defined in DAC). The AC signer doesn't have to be the owner of the resource. As long as the signer has the access to the resource, the AC is valid. In this way, access rights can be passed through multiple users and the ACs become a chain (Figure 23). In the chain, the identity of each AC signer matches the subject of the previous AC. In each AC, the signer can specify how much access is passed along and whether re-authorization is allowed. As mentioned in Section 5.5, AC can only be validated if the signer's name certificate is available. Therefore, the AC chain shown below also includes the name certificates of all AC signers.

To gain the access to a piece of resource, the user should provide an AC chain. The user's identity (assuming authentication has been done) should match the subject of the last AC in the chain (S_n). To tell if the signer of the first AC (S_0) in the chain

really has the access right, a bootstrap mechanism needs to be implemented. For example, a trusted AC authority can be used to do all initial authorizations within a domain and all AC chains must begin with an AC signed by the authority.

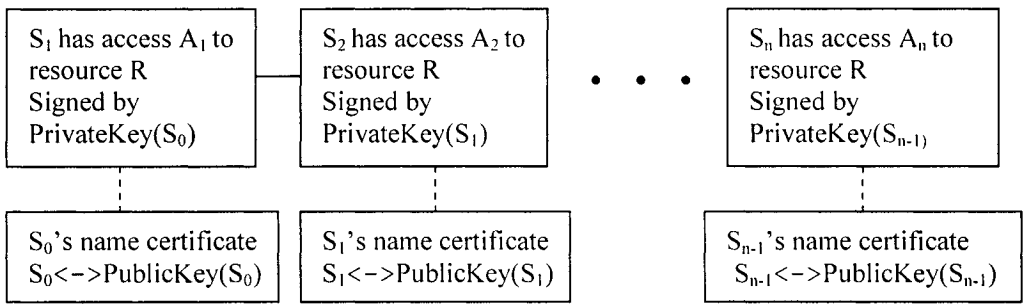


Figure 23. An AC Chain

It is a burden to carry many name certificates with an AC chain. One alternative way is using the AC that binds authorization information to a key directly. The structure of this kind of AC is shown in Figure 24. Key1 is the private key of the grantor and Key2 is the public key of the grantee. The user (grantor) who signs the AC must make sure the validity of Key2, i.e. if it really represents the party (grantee) who should get the access right.

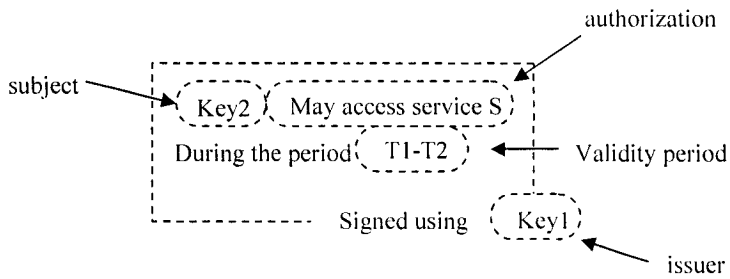


Figure 24. An Attribute Certificate which binds authorization to a key directly

This approach is extremely suitable for a highly distributed system. No centralized access control authority is needed. Each administrative domain has its own authority. One important characteristic of certificates is that the information in them is tamperproof. They do not have to be stored in a centralized and secured database. This characteristic brings two benefits. First, most aspects of authorization can be

done in a peer-to-peer manner between user agents, whether it is intra-domain or inter-domain authorization. Even the authorization between UCLP users and value-added service users can be done in the same way. The UCLP domain authority only needs to participate in the initial authorizations within the domain. Second, it makes this approach easy to be adopted by value-added service providers. It does not have any architectural implication on how and where information is stored.

Besides the above advantages for distributed systems, the implementation complexity of this approach is fairly low. The public key infrastructure has been well accepted and there are APIs and libraries available for most programming language. However, this approach has the following disadvantages which might increase the complexity.

- AC revocation is not easy to implement. There are two methods to revoke a certificate. One is submitting the revocation request to a centralized revocation service which is always accessible on-line. The other is propagating the revocation to everyone who might verify the certificate. Both methods add significant complexity to the system.
- Access rights transfer needs extra effort to realize. The access rights granted in ACs are actually shared by the grantee and the grantor.

To apply this approach on the UCLP system, each federation must start an AC authority and issue initial ACs. The Astrolabe system [48] uses a similar approach. The major difference is that in the Astrolabe system only zone administrators can issue ACs. In the UCLP system, not only federation administrators but also normal users can sign ACs.

Figure 25 shows how to publish a LPO so that it can be leased by others. The two new concepts in this figure are the User's AC Library and the LPO Broker Service. The user's AC library contains all ACs the user has. It could be as simple as a plain text file (the default format of X.509 certificates). A more sophisticated format (e.g. exported relational database tables) can be used to optimize the query process if the number of ACs is very large. The portability of the user's AC library can be

achieved in various ways, such as network storage (e.g. ftp) or personal portable storage (e.g. flash drive).

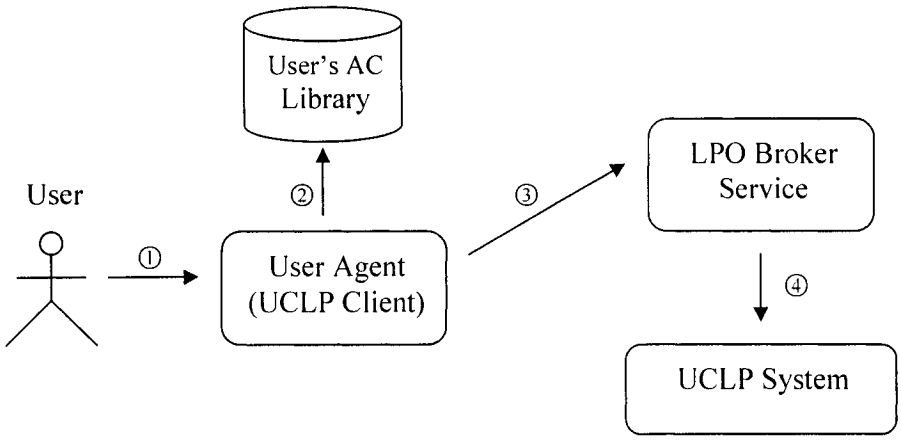


Figure 25 Typical scenario for publishing LPOs

The LPO broker service maintains a LPO repository. It could be either a part of the UCLP system or a third-party service. Users can publish their free LPOs through the UCLP client ①. The client gets the information of available LPOs which the user has access to (from the User's AC Library) ②. Once the user decides which LPO he wants to publish, the client will sign a new AC, which grants access (to the LPO) to the broker. The new AC and the original AC chain associated with the LPO become a new AC chain. Then the new AC chain will be submitted to the broker ③. The broker needs to validate the AC chain from the user by checking with the UCLP system and download the information of related LPOs ④. The broker's certificate has limited access to the UCLP system. Only query is allowed although it has all kinds of access rights to published LPOs.

Figure 26 shows how a user leases LPOs from the LPO Broker Service before creating an E2E connection. Once a user initiates a connection creation request ①, the user agent will query the user's AC library first ② (assuming the user agent knows what kind of LPOs are required [49]). If the user does not have sufficient ACs, the agent will try to lease some from the broker ③. The broker can do some negotiation (price, lease term, and so on) with the leaser on behalf of the LPO publisher. After the negotiation is done, the broker will sign a new AC for the user and pass it back along

with the original AC chain. If the agent obtains enough resources from the broker, it will send the request to the UCLP system along with all required AC chain(s) ④.

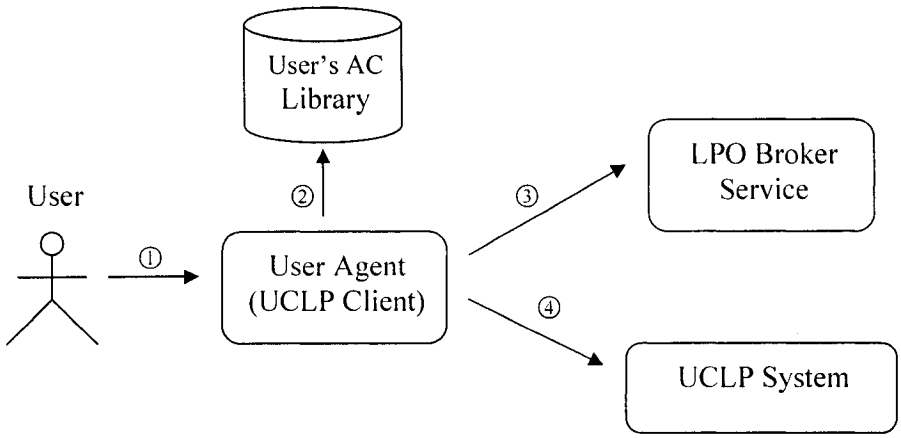


Figure 26. Typical scenario for creating E2E connections

The LPO broker service is a helper for LPO sharing among UCLP users. It is not the only way to borrow LPOs. Users can always get ACs by other means and import them into their own library. They can even interact manually and transfer ACs by email. The advantage of having a broker service is that the UCLP system becomes a pure network management system. The broker service takes care of realizing complex LPO query, negotiation with user agents and integration with the payment and billing system. Also the users who have common interest in LPOs can start their own private broker service. For example, small ISPs may have their own networks within the country and have much more interest on sharing inter-country LPOs. The AC-based access control makes such a third-party resource pool very easy to realize. All they need to do is applying for a broker certificate from the UCLP CA.

7.4.5.4 Summary and comparison

The first approach is the simplest solution. It satisfies most of the requirements on the access control and needs less implementation effort than the other two approaches. However, it is difficult to be used in a multi-domain environment.

The second one realizes distributed DAC using AAA architecture which is designed for multi-domain systems. It fits all requirements very well. However, the AAA standard, as a generic solution for all internet applications that need access control, is too complicated to be implemented or adopted by other service providers.

The third approach realizes a distributed DAC using Attribute Certificates. Compared to the second approach, the simplicity of this approach makes it easy to be accepted by value-added service providers and the UCLP-based service hierarchy becomes more scalable. The example scenario of Section 7.4.5.3 shows the flexibility of this approach. Authorizations can be made in various ways without going through the management system. Although this approach has some disadvantages, it is definitely the best out of the three.

Part 4 Conclusion

This thesis introduces the current version of the UCLP system and presents my research on UCLP security. My major contributions are:

- **Design and Implementation**

  I contributed to the design of the high level system architecture, especially the Grid interface of the UCLP system. I implemented the Grid Service and the Grid GUI client. I also helped other developers on the JSAP.

- **Security Analysis**

  I made the requirement analysis for the UCLP security and evaluated the current system on major security aspects.

- **Proposal of Security and Access Control**

  I proposed solutions to enhance the security of the current system. I also investigated different approaches to realizing sophisticated access control for UCLP. The AC-based distributed DAC approach is chosen to be the candidate design for the next implementation. Example access control scenarios are given to demonstrate the benefits of the chosen approach.

The current version of the UCLP system is running on CA*net 4. It also has been introduced to South Korea, Spain and Taiwan. Several versions of SCS service was developed for different optical equipments on their networks, such as Cisco ONS 15454, Nortel OpteraMetro 5200 and so on. The UCLP systems deployed at these places can work together as expected and a tri-continental optical connection (South Korea - CA*net 4 – Spain) was created successfully [47]. A HDTV demo was performed to show the strength of the dedicated long haul connection. All these facts proved that the design of the UCLP system is successful:

- The system architecture based on Web Services and Jini works well across different networks spanning on three continents.

- The definition of the resource hierarchy and the communication service interface is generic and flexible. The system is able to handle different equipments with limited extra implementation work.

As a continuation of the work presented in this thesis, the implementation of the improved security and access control would be part of the future work. Other interesting topics are:

- New implementation compliant to the latest standard.

  The Grid Service standard has been converged with the Web Services standard. The new version of the Globus Toolkit which is compliant to the latest standard will be released on Jan 31, 2005. It is necessary to implement the UCLP Grid Service based on the new Globus Toolkit.

- Sophisticated routing algorithm and support for more network equipments

  As the UCLP system is deployed on more and more networks, the physical topology will become much more complicated than CA*net 4. Therefore, more sophisticated routing algorithm will be required as well as more network device support.

Reference

[1] W. Johnston, S. Mudumbai and M. Thompson, "Authorization and Attribute Certificates for Widely Distributed Access Control", In *Proceedings of Seventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '98)* (1998), IEEE Press, pp. 340–345.

[2] T. Aura, P. Koponen and J. Rasanen, "Delegation-based Access Control for Intelligent Network Services". In *Proc. ECOOP Workshop on Distributed Object Security*, Brussels, Belgium, July 1998.

[3] M. Bourque, "CXFS Shared File System Trial". Presentation, *Réseau d'informations scientifiques du Québec (RISQ)*, Montreal, Quebec, October 2003.

[4] C. de Laat, G.Gross, L. Gommans, J. Vollbrecht and D. Spence, "Generic AAA Architecture", RFC 2903, August 2000.

[5] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege and D. Spence, "AAA Authorization Framework", RFC 2904, August 2000.

[6] C. de Laat and G.Gross, "Generic AAA Server architecture", presentation for the AAA-WG, 46th IETF, Washington DC, November 1999, http://www.aaaarch.org/genaaa/index.htm, last visited in Mar 2005

[7] J. Moffett, M. Sloman and K. Twidle, "Specifying Discretionary Access Control Policy for Distributed Systems", *Computer Communications,* Vol. 13 no 9, pp 571-580, November 1990.

[8] T. Aura, "Distributed Access-Rights Managements with Delegations Certificates", In J. Vitek and C. Jensen, editors, *Secure Internet Programming: Security Issues with Distributed Mobile Objects*, Vol.1603 of *Lecture Notes in Comp. Sci.*, pp 211–235. Springer-Verlag, 1999.

[9] B. Clifford Neuman and Theodore Ts'o, "Kerberos: An Authentication Service for Computer Networks", *IEEE Communications*, 32(9):33-38. September 1994.

[10] John T. Kohl, B. Clifford Neuman, and Theodore Y. T'so, "The Evolution of the Kerberos Authentication System". In *Distributed Open Systems*, pages 78-94. IEEE Computer Society Press, 1994.

[11] "Introduction to SSL", Netscape DevEdge Online Documentation, Copyright 1998 Netscape Communications Corporation, http://developer.netscape.com/docs/manuals/security/sslin/, last visited in Dec 2004

[12] F. Sommers, "Call on Extensible RMI - An Introduction to JERI", Technical Article, Column *Jiniology* in *JavaWorld*, Dec. 2003, http://www.javaworld.com/javaworld/jw-12-2003/jw-1219-jiniology.html, last visited in Mar 2005

[13] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke, "A Security Architecture for Computational Grids", *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pp. 83-92, 1998.

[14] Web Services Security, OASIS Technical Committee, http://www.oasis-open.org/committees/wss/, last visited in Mar 2005

[15] T. Imamura, B. Dillaway and E. Simon, "XML Encryption Syntax and Processing", W3C Recommendation, Dec. 2002.

[16] M. Bartel, J. Boyer, B. Fox, B. LaMacchia and E. Simon, "XML-Signature Syntax and Processing", W3C Recommendation, Feb. 2002.

[17] D. Denning, "A lattice model of secure information flow", *Communications of the ACM 19*, 5, 236-243, 1976.

[18] B. Lampson, "Protection", In *5th Princeton Symposium on Information Science and Systems (1971)*, pp. 437-443, 1971.

[19] R. Sandhu, E. J. Coyne, H. L. Feinstein and C. E. Youman, "Role-based access control models", *IEEE Computer 29*, 2 (February), 38-47, 1996.

[20] J. Wu, S. Campbell, J. M. Savoie, H. Zhang, G. v. Bochmann and B. St.Arnaud, "User-managed end-to-end lightpath provisioning over CA*net 4", *Proceedings of the National Fiber Optic Engineers Conference (NFOEC)*, pp. 275-282 Orlando, FL, USA, Sep. 2003.

[21] U. Black, "Network Management Standards", 2nd Edition, Ch. 8, McGraw-Hill, Inc., 1995.

[22] A. Leinwand and K. Fang, "Network Management: A Practical Perspective", Addison Wesley, 1993.

[23] D. K. Udupa, "Telecommunications Management Network", McGraw-Hill, 1999.

[24] M. Horstmann and M. Kirtland, "DCOM Architecture", Technical Article, MSDN Library, July 1997.

[25] Common Object Request Broker Architecture, http://www.omg.org/corba/, last visited in Mar 2005

[26] Jini Architecture Specification v2.0, http://java.sun.com/products/jini/2.0/doc/specs/html/jini-spec.html, last visited in Mar 2005

[27] G. F. Coulouris, J. Dollimore and T. Kindberg, "Distributed Systems: Concept and Design", 3rd Edition, Ch. 12, Addison-Wesley, 2001.

[28] Web Services Activity, World Wide Web Consortium (W3C), http://www.w3.org/2002/ws/, last visited in Mar 2005

[29] SOAP, XML Protocol Working Group – W3C, http://www.w3.org/2000/xp/Group/, last visited in Mar 2005

[30] WSDL, Web Services Description Working Group - W3C, http://www.w3.org/2002/ws/desc/, last visited in Mar 2005

[31] XML Schema, XML Schema Working Group – W3C, http://www.w3.org/XML/Schema, last visited in Mar 2005

[32] UDDI, OASIS (Organization for the Advancement of Structured Information Standards) standards consortium, http://www.uddi.org/, last visited in Mar 2005

[33] I. Foster, C. Kesselman, J. M. Nick and S. Tuecke1, "The Physiology of the Grid - An Open Grid Services Architecture for Distributed Systems Integration", *Open Grid Service Infrastructure WG, Global Grid Forum*, June 2002.

[34] I. Foster, C. Kesselman and S. Tuecke, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations", *International J. Supercomputer Applications*, 15(3), 2001.

[35] E. Gamma, R. Helm, R. Johnson and J. Vlissides, "Design Patterns", Ch. 3, Addison-Wesley, 1995.

[36] Globus Toolkit, Globus Alliance, http://www-unix.globus.org/toolkit/, last visited in Mar 2005

[37] F. Cabrera, G. Copeland, B. Cox, T. Freund, J. Klein, T. Storey and S. Thatte, "Web Service Transaction Specification", August 2002.

[38] S. Morris, "Security and the Management Plane, Part 1", Professional Technical Reference, Prentice Hall, June 2004

[39] Information processing systems - Open Systems Interconnection – Systems Management Overview - International Organization for Standardization – International Standard 10040 - November 1992.

[40] B. St. Arnaud, "Frequently Asked Questions about Customer Owned Dark Fiber, Condominium Fiber, Community and Municipal Fiber Networks", Customer Owned Fibre Networks - CA*net 4 Online Library, March 2002, http://www.canarie.ca/canet4/library/customer.html, last visited in Mar 2005

[41] B. St. Arnaud, A. Bjerring, O. Cherkaoui, R. Boutaba, M. Potts and W. Hong, "Web Services Architecture for User Control and Management of Optical Internet Networks", *Proceedings of the IEEE*, Vol. 92 No. 9, pp1490-1500, August 2004

[42] R. Needham and M.Schroeder, "Using encryption for authentication in large networks of computers", *Communications of the ACM*, 21(12):993--999, 1978.

[43] B. St. Arnaud, J. Wu and B. Kalali, "Customer Controlled and Managed Networks", CA*net 4 Library, January 2003, http://www.canarie.ca/canet4/library/canet4design.html, last visited in Mar 2005

[44] R. Boutaba, W. Ng and A. Leon-Garcia, "Web-Based Customer Management of VPNs", CA*net 4 Library, http://www.canarie.ca/canet4/library/customer.html, last visited in Mar 2005

[45] Authorization, Authentication and Accounting Architecture research group, http://www.aaaarch.org/, last visited in Mar 2005

[46] S. Farrell, J. Vollbrecht, P. Calhoun, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege and D. Spence, "AAA Authorization Requirements", RFC 2906, August 2000

[47] S. Figuerola, E. Grasa, M. Hurtado, J. Alcober, J. Recio, G. Junynet, M. Savoie, S. Campbell, H. Zhang and J. Wu, "Transmission of HDTV over IP on an Optical Connection Managed with UCLP", *'Jornadas Técnicas de RedIris*, Toledo, Spain, October 2004

[48] R. van Renesse, K. Birman, and W. Vogels, "Astrolabe: A Robust and Scalable Technology For Distributed System Monitoring, Management, and Data Mining", *ACM Transactions on Computer Systems* (TOCS), 21(2), May 2003.

[49] G. v. Bochmann, "Hierarchical Inter-Domain Management for Networks with Condo-Switches", November 2004, in preparation.

[50] B. St. Arnaud, W. Hong, G. Hayward, C. Cost, B. Caron and S. MacDonald, "Customer Controlled Lightpaths for Large File Transfer", presentation on Protocols for Fast Long-Distance Networks (PFLDnet) Workshop, Geneva, January 2003, http://www.canarie.ca/canet4/library/canet4design.html, last visited in Mar 2005

Table of Acronyms

AAA	Authentication, Authorization, Accounting
AC	Attribute Certificate
ACL	Access Control List
ADF	Access Control Decision Facility
AEF	Access Control Enforcement Facility
ASM	Application Specific Module
CA	Certificate Authority
CMIP	Common Management Information Protocol
CORBA	Common Object Request Broker Architecture
CS	Communication Service
DAC	Discretionary Access Control
DCOM	Distributed Component Object Model
DFS	Distributed File System
GSAP	Grid Service Access Point
GSI	Grid Security Infrastructure
HTTP	HyperText Transfer Protocol
IN	Intelligent Network
ISO	International Standards Organization
JDBC	Java DataBase Connector
JERI	Jini Extensible Remote Invocation
JLS	Jini Lookup Service
JRMP	Java Remote Method Protocol
JS	JavaSpace
JSAP	Jini Service Access Point
LPO	LightPath Object
LPOS	LightPath Object Service
MAC	Mandatory Access Control
OGSA	Open Grid Service Architecture
OSI	Open Systems Inter-connection
PDP	Policy Decision Point
PEP	Policy Enforcement Point

PKI	Public Key Infrastructure
RBAC	Role-Based Access Control
RMI	Remote Method Invocation
RO	Resource Object
RPC	Remote Procedure Call
SCS	Switch Communication Service
SCSL	Sun Community Source License
SNMP	Simple Network Management Protocol
SOA	Service Oriented Architecture
SONET	Synchronous Optical NETwork
SP	Service Provider
SSL	Secure Socket Layer
TLS	Transport Layer Security
UCLP	User-Controlled Lightpath Provisioning
UDDI	Universal Description, Discovery and Integration
VOD	Video-On-Demand
VPN	Virtual Private Network
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing
WSDL	Web Service Description Language
XML	eXtensible Markup Language