CHAPTER II

BACKGROUND AND RELATED WORK

This chapter is divided into 3 parts. The first part describes the background. The second parts describes other work related to name services and the third summarizes the features of variouse name services.

2.1 Background

2.1.1 TCP/IP Protocol Suite

The TCP/IP protocol suite [17–19], an open protocol standards, allows computers of all sites, from many different computer vendors, running on different operating systems, to communicate with each other. It is truly an open system which forms the basis for what is called Internet. The TCP/IP protocol suite is the combination of different protocols at various layers. Normally, it is considered to be a four-layer system as shown in Table 2.1.

Table 2.1: The four layers of the TCP/IP protocol suite

Application	Telnet, FTP, e-mail, etc.
Transport	TCP, UDP
Network	IP, ICMP, IGMP
Link	Device driver and interface card

2.1.2 Internet VS internets

The Internet is the collection of over one million hosts around the world that can communicate with each other using TCP/IP. An internet is any network made up of multiple networks using a common protocol suite. It is not necessarily connected to the Internet, nor does it necessarily use TCP/IP [20].

2.1.3 Names and Objects

The Internet Protocol document [21] defines names and addresses as follows: "A name indicates what we seek." and "An address indicates where it is". Therefore hosts are assigned the identical names because they are easier to remember and type correctly. Every network interface attached to a TCP/IP network is identified by a unique 32-bit address (IPv4 address) and normally is written as 4-decimal numbers called dotted-decimal notation. The Internet Network Information Center (InterNIC) is a central authority for allocating these addresses for networks connected to the world wide Internet. There are 3 types of IP address:

1. Unicast is destined for a single host.

2. Broadcast is destined for all hosts on a given network.

3. Multicast is destined for a set of hosts that belong to a multicast group.

Currently, version 6 of the Internet protocol (IPv6 address) [22–24] also supports to map names to addresses.

2.1.4 Services and Name Services

When a user requests a service - a distinct part of a computer system that manages collection of related resources, a name is referred to operate on a named object or resource. In a distributed system, names are needed to refer to entities such as users, computers, and services themselves. A name service stores a collection of one or more naming contexts - sets of bindings between textual names and attributes for objects such as users, computers, services, and remote objects [25].

2.2 Related Work

2.2.1 Grapevine

In the early 80's, a multicomputer system on the Xerox research internet called Grapevine [2] was developed to provide message delivery, resource location, authentication, and access control services. A registration database is used to map names to information about users, machine services, distribution hosts, and access control lists. The naming structure is a two-level hierarchy. The two-level naming hierarchy worked well for delivering computer mail within the Xerox Cooperation Research and Development community.

2.2.2 Clearinghouse

In 1983, Clearinghouse, an extension of Grapevine, was developed [3]. Clearinghouse is a decentralized agent for locating named objects. The objects in Clearinghouse are of many types including workstations, file servers, print servers, and human users. The naming structure is three-level hierarchy for mapping names to objects across organizations.

2.2.3 Domain Name System (DNS)

The Internet is a world-wide computer network. It is a network that interconnects millions of computing devices throughout the world. Most of computing devices such as PCs, Unix-based workstations and servers, which are called hosts, need identification. In the Internet, these hosts are identified via IP addresses as shown in Figure 2.1.

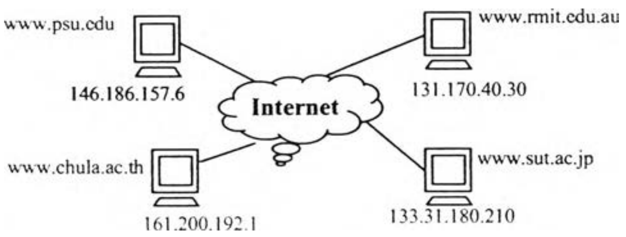

Figure 2.1: Internet and IP addresses.

Domain Name System (DNS) was designed in 1984 and specified in [4, 6]. DNS is a distributed database based on the TCP/IP protocol. The DNS maps human-readable host names to numerical IP-addresses. The database structure is strictly hierarchical. A name such as sc.chula.ac.th has a structure as shown in Figure 2.2.

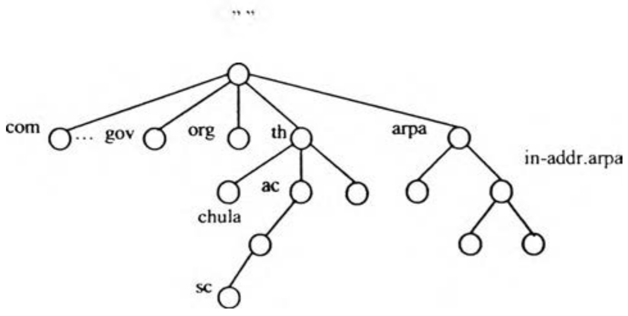

Figure 2.2: The organization of DNS

The domain names are restricted to a subset of ASCII known as the "LDH" rules for "letter - digit - hyphen" [26]. Each domain name is a path in an inverted tree, called the domain name space. The tree has a single root at the top called "the root". Every node has a label of up to 63 characters, except the root, and must have a unique domain name. A domain name that ends with a period is called an absolute domain name or a fully qualified domain name (FQDN) for example: sc.chula.ac.th.

The information about the domain name space is stored in the name servers. There are 2 important processes related to DNS:

1. The resolver: Typically, resolvers are library routines called by programs that need to look up names.

2. The name resolution or simply resolution: A resolver performs name resolution or asks a name server to do for it by querying name servers to determine a mapping from a name to an address. Queries can be either recursive or non-recursive. The result from recursive queries is the direct address of the required destination. whilst the non-recursive queries return the IP address of the next hop of the related link.

A generic resolution example is shown in Figure 2.3 with a brief explanation

1. Name server A receives a query from the resolver.

2. A queries B.

3. B refers A to other name servers, including C.

4. A queries C.

5. C refers A to other name servers, including D.

6. A queries D.

7. D answers.

8. A returns answer to resolver.

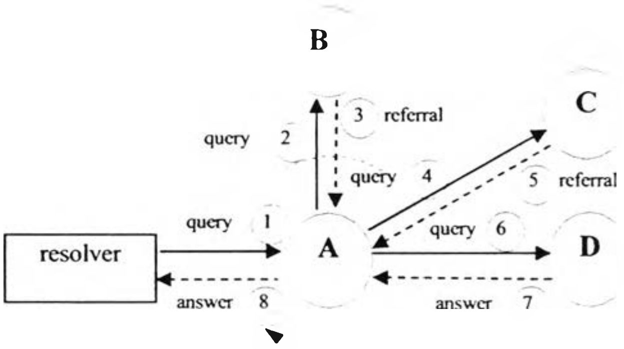

Figure 2.3: The resolution process

As the growth of using applications through Internet is increasing rapidly, the demand of using a non-ASCII character domain name is required. Therfore, efforts to add non-ASCII character names, deriving from languages or character sets based on other simple ASCII and English-like names, have been arosen. Consequently, the Internationalized Domain Names Working Group (IDN-WG) was initiated.

The IDN-WG had developed a technique called the ASCII-Compatible Encoding (ACE) to support the required non-ASCII character domain name. The ACE preserves the LDH conventions in the DNS itself. However, the ACE systems, requires much of the matching mechanism. Hoffman proposed Nameprep [28] to allow users to enter internationalized domain names (IDNs). Nameprep is a Stringprep profile used by the

IDNA protocol [27] for preparing domain name. Then, it is implemented to process domain name labels, not domain names. Furthermore, punycode [29] converts domain name labels which are non-ASCII characters into ASCII characters.

2.2.4 Global Name Service (GNS)

A global name service(GNS) was developed by Lampson and colleagues at the DEC Systems Research Center [7]. GNS provides facilities for resource location, mail addressing and authentication and was designed on the basis of Grapevine [2] and Xerox Clearinghouse [3] including these requirements such as large size, long life, high availability, fault isolation, and tolerance of mistrust.

GNS provides a name service for use in an internetwork which supports a naming database that may extend to include the names of millions of computers and eventually e-mail addressing for billions of users.

The name service is divided into 2 levels: client level and administration level. At the client level, client sees a structure as a tree of directories (see Figure 2.4)

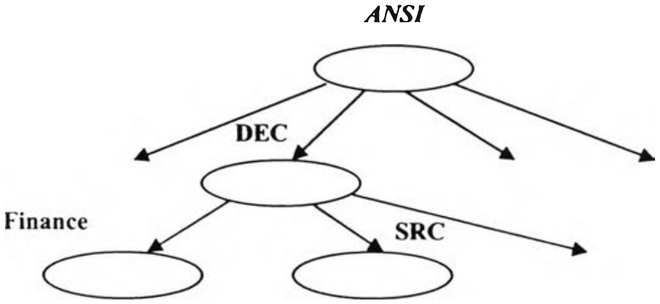

Figure 2.4: The GNS directory tree

In GNS, names have 2 parts: <directory name, value name>. The first part identifies a directory; the second refers to a value tree or some portion of a value tree. The name space is hierarchical. There are operations for reading and updating names and their values. Furthermore, GNS provides the facilities for authentication and protection or authorization. The authentication is based on the use of encryption to provide a set of secure channel between 2 principals using the key to establish communication. Each directory has an authentication function to map between keys and principals. Principals are the entity which are identified by a full name or a relative name similar to Unix file system. For authorization, GNS provides the access control function which is defined by a set of triples (principal pattern, path pattern, rights) such as (ANSI/DEC/*, Lampson/*, {read}). At this level, the database which is distributed and replicated is invisible. At the administration level, naming database which is a tree of directories holding name and values can be partitioned and stored in a number of servers and replicated it for reliability. The copies of the databases are visible. Table 2.2 summarize some features of GNS.

Table 2.2: Features of GNS

Feature	Global Name Service
Name space	Hierarchical
Creation Date	1986
Supports	Small to large network
Large Number of Objects	Not scalable: in case of merging and moving directory tree for large network
Resource Location	Transparent to user
Security	Authentication: secret key to provide a secure channel Authorization: access control function
High availability	A set of directory copies stored in servers

GNS successfully addresses the need for scalability and reconfigurability with the exception of the solution adopted for merging and moving directory trees [25, 30]. For example, two previously separated GNS services may be integrated with the introduction of a new root above the two existing roots. Well-known directory tables are used to store the previous directory identifiers and remap to the current real root directory of the naming database. Whenever the real root of the naming database changes, the new location of the real root will be informed to all GNS servers. In a large-scale network, reconfigurations may occur at any level, and this makes the table grow rapidly, conflicting with the scalability goal.

2.2.5 Network Information Service Plus (NIS+)

The Network Information Service Plus(NIS+) [8] was developed by the Sunsoft engineering team. It stores information about workstation addresses, security information, mail information. Ethernet interfaces, and network services in central locations where all workstations on a network can access it.

The NIS+ name space is hierarchical like DNS. The NIS+ name space can be divided into multiple domains, each of which can be administered autonomously and consists of 3 structural components: directories, tables, and groups. These components are called NIS+ objects. A domain of NIS+ is a collection of objects. A DNS domain, on the other hand, stores names and IP addresses of all the workstations in each domain. Figure 2.5 shows the structure of NIS+.

To store and access to the information contained in an NIS+ name space, NIS+ uses a client-server model and enhances reliability by supporting each domain with master

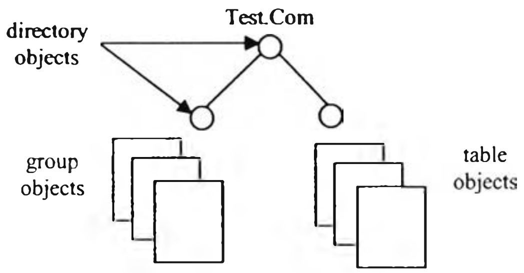

Figure 2.5: The structure of NIS+

server and replicas for backing up which is similar to DNS. Even though each domain is supported by a set of servers, a single set of servers can support more than one domain. In addition, NIS+ stores information in tables instead of the zone files used in DNS. These tables are not ASCII files, but are tables in the NIS+ database which are viewed and edited using NIS+ commands. NIS+ clients can obtain their network information from one or more of these sources: NIS+ tables, NIS maps, the DNS hosts table, and local path (/etc) files using Name Service Switch (NSS).

Furthermore, NIS+ provides security using authentication and authorization to protect the information in the name space and the structure of the name space itself from an unauthorized access. The security is an integral part of the NIS+ name space. Figure 2.6 describes NIS+ security process.

1. Client sends server a request for access to the name space.

2. Server authenticates client's request by examining principal's credentials.

3. Server examines object's definition to determine access rights granted to principal.

4. Server determines the class of principal: Owner, Group, World, or Nobody.

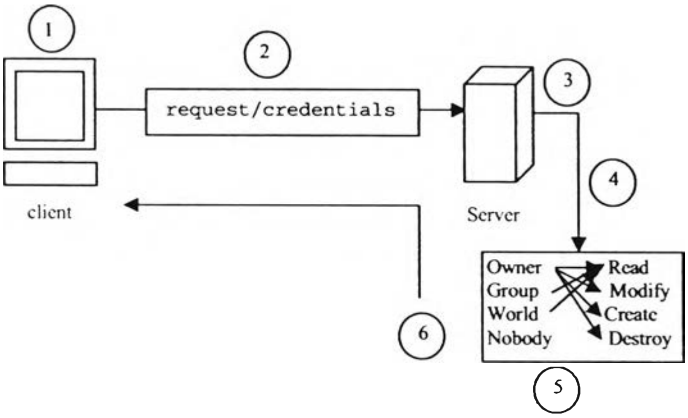

Figure 2.6: Summary of NIS+ Security Process

5. Server determines access rights granted to principal's class: Read, Modify, Create, or Destroy.

6. If the access rights granted to principal's class match the types of operations, the operation is performed.

NIS+ uses credential to authenticate the identity of principal and uses authorization to specify access rights. A principal is an entity that submits a request for NIS+ service from an NIS+ client. There are two types of principal, a client user which is a general user and a client workstation which a user logs in as a superuser. Principals have 2 types of credential: LOCAL and DES (Data Encryption Standard) [31, 32]. To determine the type of credential principals, NIS+ provides security levels. Currently there are 3 security levels: 0, 1, and 2 which are provide by the NIS+ server. Normally, level 0 is provided for testing and setup the initial NIS+ name space. Level 1 is provided for testing and debugging and level 2 is the most secure level and used in general. A client user can have both LOCAL and DES, but a client workstation can only have a DES credential. For secure environment, NIS+ uses DES authentication. DES authentication uses DES and public key cryptography - Diffie-Hellman scheme [33]. Table 2.3 describes the characteristics of NIS+. NIS+ focuses on making network administration more manageable over a variety of network information and uses under Solaris 2.x which is proprietary, whereas DNS focuses on making communication simpler using workstation names instead of addresses and supports a variety of hosts running on various operating systems. Additionally, DNS is a key infrastructure for Internet that nobody has a monopoly on access to or use of it. Therefore the more Internet community is being growth, the more DNS is being important.

Table 2.3: NIS+ characteristics

Feature	Network Information Service Plus (NIS+)
Supports	Small to large network
Large Number of Objects	Scalability
Domain Store	Collection of information such as workstations, users, and network services
Data Storage	Table (NIS+ database)
Data Update	Propagation
Security	Authentication: DES + Diffie-Hellman Authorization: Access rights(read, modify, create and destroy) depend on authorization classes: Owner, Group, World, and Nobody.
High availability	Master server and backup servers called replicas
Connect to Internet	Use Name Service Switch
Target users	In organization

2.2.6 Novell Directory Service (NDS)

Novell Directory Service (NDS) [9–11] was introduced in 1993 as a method of managing Netware networks. NDS is originally designed to manage distributed Netware networks.

It now supports management of basic network resources including applications, services, and other information concerning the network.

NDS is a full-function directory service based on the 1988 X.500 standard and provides a single, logical tree-structured view of all resources on the network. Table 2.4 provides a comparison of NDS terminology to the terms used in X.500 [11,34].

Table 2.4: A comparison of NDS terminology to the terms of used in X.500

Functionality	NDS Term	Analogous X.500 Term
Directory entries	Attributes	Attributes
Definition of directory contents	Schema	Schema
Logical representation of directory	Directory tree	Directory Information Tree (DIT)
Data Storage	Directory database	Directory Information Base (DIB)
Subdivision of directory	Partition	Naming Context
Data Update	Synchronization	Replication
Server agent	NDS server	Directory Service Agent (DSA)
Client agent	Client	Directory User Agent (DUA)
Query resolution	Tree-walking and referral	Chanining and Referral

Although NDS is based on X.500 in design and operations, the protocols used are primarily proprietary. Portions of the NDS database are distributed on volume storage devices at strategic locations on the network. These elements are called partitions. NDS partitions are copied or replicated across the network as necessary to enhance reliability. In addition, NDS uses shared secret authentication methods as well as the Public Key Infrastructure(PKI) [35] certificates for establishing the user identity. Both private and public key technologies are used for NDS authentication. NDS uses RSA public-key technology [36] to provide for a single login and encrypted authentication. With this process, users can access applications through a single sign-on (i.e. One-password access to any authorized resources on the network). Users can sign on to a multiserver network and view the entire network as a single information system.

NDS stores information in a multiple file structure instead of the zone files used in DNS. Table 2.5 describes the characteristics of NDS.

Table 2.5: NDS characteristics

	Novell Directory Service (NDS)
Name space	Hierarchical with a collection of network resources and services
Support	Small to large network
Large number of objects	Scalibility
Data Storage	Directory database (NDS file structure)
Data update	Synchronization
Security	Authentication:- RSA: public key Authorization:- Access rights use a combination of settings: trustee assignment, security equivalence, inheritance and inherited rights filter (IRF)
High availability	Partitions and replicas
Target users	In organization

NDS purpose supports to easily manage distributed Netware networks and resources. The core protocols using in NDS are proprietary, whereas DNS uses worldwide for the Internet which is truly an open system.

2.2.7 Handle System

A Handle System [12,13] developed by CNRI (http://www.cnri.reston.va.us) is a general-purpose global name service that allows secured name resolution and administration over the Internet. The Handle System includes an open protocol, a name space, and a reference implementation of the protocol. The protocol enables a distributed computer system to store names or handles, of digital resources and resolve those handles into information necessary to locate, access, and make use of resources. It has been designed to serve as naming system for very large number of entities and to allow administration at the name level while DNS names are managed by the network administrator(s) at the zone level. In Handle System, name or handle consists of 2 parts: naming authority and unique local name. The naming authority and local name are seperated by the ASCII character "/" as shown below :

<Handle> ::= <Handle Naming Authority> "/" <Handle Local Name>

The naming authority is globally unique within the Handle System. Moreover, it is hierarchical. Unlike DNS, the naming authorities are constructed left to right, concatenating the labels from the root of the tree to the node that represents the naming authority and each label is seperated by ".". For example, a naming authority for the National Digital Library Program ("ndlp") at the Library of Congress ("loc") is defined as "loc.ndlp". The handle name space can be considered as a superset of many local name spaces, with each existing local name space can join a global handle name space by obtaining its own unique naming authority.

The Handle System defines a hierarchical service model which is categorized into 2 types: handle resolution service and handle administration service. Clients use the handle resolution service to resolve handles into their values. To manage the handles, including adding and deleting handles, or updating their values, the handle administration service deals with client requests. It also deals with authority administration via naming authority handles. The top level of Handle System is a single global service, or the Global Handle Registry and the lower level consists of all other handle services, which are known as local handle services. The Global Handle Registry provides a handle service for resolution and used to manage any handle name spaces. The local handle service layer provides resolution and administration service for the local names.

The Handle System provides authentication depending on client's request. In the normal situation, the handle resolution service does not require client authentication. On the other hand. if there is any confidential data assigned by an administrator, the authentication will be performed. Handle clients use either a secret key or a public key cryptography for authentication. Table 2.6 describes the features of Handle System. In addition. the Handle System allows the name to persist over changes of locations, ownerships, and other state conditions. For example, when a name resource moves from one location to another, the handle updates its value in the Handle System to reflect the new location. The disadvantage [37] of this system is the effective use requires users install the special browser software.

Table 2.6: Handle System Features

Feature	Handle System
Name space	Hierarchical
Creation Date	1998
Supports	Internet
Large Number of Objects	Scalability
Domain Store	Digital objects or resources
Data Update	Use handle administration service
Security	Authentication: secret key/public key cryptography
High availability	Group of handle servers into one or more handle sites

2.2.8 CORBA Naming Service

In addition to object-based middleware, the CORBA naming service is a generic service for the CORBA architecture [14–16]. It allows names to be bound to remote objects. A name binding is always defined relative to a naming context [38]. The names are structured in a hierarchical fashion. In the CORBA naming service, different names can be bound to an object in the same or different contexts at the same time.

2.2.9 Other Naming Systems

The Intentional Naming System or INS [39] was proposed in 1999. The INS attempt to use a naming system to achieve various transparencies. INS has a great capability of transparently locating various objects.

The Interface-based Naming System or IFNS [40, 41] is designed to support the two stated requirements: locating and adaptation. The IFNS is capable of locating objects transparently in a ubiquitous Internet. The objects called functional objects can be multiple named. For example, a functional object such as physical location, can have different names. However, each functional object should have at least one interface name.

The Federated Naming Service (FNS) [42] is a system for uniting various name services under a single interface for basic naming operations. It is produced by Sun Microsystems and was included in the Solaris Operating Environment versions 2.5 to 9. FNS is an extended implementation of X/Open's 1994 XFN (X/Open Federated Naming) specification. The purpose of XFN and FNS is to enable applications to use widely heterogeneous naming services (such as NIS, NIS+, DNS, etc.) via a single interface. in order to avoid duplication of programming effort. Neither XFN nor FNS was popular or widely used. Nevertheless, FNS was last included in Solaris 9.

2.3 Features of Various Name Services

Name service is a fundamental service that provides a mechanism to maintain status and accesses information about network resources. Regarding to the growth of computer technology, the interconnection of networks and other issues related to names are needed to be considered. Most researches and products have been designed with the basic requirements such as a long life time, high availability. fault tolerance including scalability. The issues that influence the structure and semantics of a name space must be taken into account [43].

Table 2.7: Features of current name services.

Name Service	Human-readable name	Hierarchical name space	Sharing unique name	Character Support	Anonymity
Grapevine	Yes	Yes	No	ASCII	No
Clearinghouse	Yes	Yes	No	ASCII	No
DNS	Yes	Yes	Yes[a]	{a-z, A-Z, 0-9, -}[b]	No
GNS	Yes	Yes	No	ASCII	No
NIS+	Yes	Yes	No	ASCII	No
NDS	Yes	Yes	No	ASCII	No
Handle System	Yes	Yes	No	Unicode	No
CORBA Naming Service	Yes	Yes	No	Unicode	No

[a] DNS allows a domain name mapping to multiple addresses for load balancing.

[b] any character (octet value) can be in any DNS label, but other applications (e.g. e-mail. www, etc) only handle {a-z, A-Z, 0-9, and hyphen (-)}, and no encoding label such as UTF-8 is in DNS.

The characteristics of each name service are summarized in Table 2.7. According to Table 2.7, it shows that human-readable name, hierarchical name space, and anonymity are the basic features that contain in every name service. In addition, most of them cannot share a single name to refer various objects. Though, DNS supports sharing unique name, this feature is intended only to help load balancing. The evaluation of name service is developed for decades. Grapevine, Clearinghouse, DNS, GNS, NIS+, and NDS were originally supported only ASCII-based characters, even if some name services have an extension for dealing with non-English languages. For example, there are a number of DNS extensions for using non-ASCII characters domain names. The iDNS, proposed by J.K.Tan et al. [44, 45], employs an iDNS-compatible server to transform multilingual string names compliant with RFC1035 [5]. Moreover, Internationalized domain names in Applications (IDNA) [27] is defined for handling internationalized domain name. For preparing domain names, Nameprep [28] is used, and the ACE labels [27] and Punycode [29] represent non-ASCII based label.

Therefore, this thesis aims to justify a system that eliminates drawbacks and overcomes some difficulties brought by a strictly hierarchical naming system.