OPEN SYSTEMS SECURITY

Horatiu Cirstea, Adrian Locusteanu, Florin Paunescu, Cristian Petculescu
Glosy S.A. - Global Services and Systems
Bd. Unirii 15, 74128 Bucharest
(paunescu@u1.ici.ro)

Abstract

The framework for Open Systems was set out by the International Standard Organisation through the definition of the OSI Reference Model. This identified the need for a set of internationally agreed standards, intended to be vendor independent, and to allow communications between IT systems from different suppliers. This paper refers to UNIX systems and his major characteristics: portability and interoperability, as well as the principles and the new technologies in internetworking. We also address the problem of dependability and his components - availability, reliability, security (integrity and confidentiality) and safety - relatively to a computing system.

1. Introduction

Open Systems is an approach to Information Technology (IT) planning, development and operations that uses internationally agreed standards to achieve a firm technical foundation, the IT infrastructure, on which flexible and responsive IT solutions to business needs can be built.

Open Systems standards define interfaces and basic infrastructure functions. Using these standards, IT users may develop, run and interconnect applications from a variety of sources on a variety of hardware.

The standards forming the foundation of the Open Systems approach must be in the public domain and cannot be owned or controlled by a single company or group of companies with commercial interests. They must embody neutrality and will result from a process of international cooperation and agreement. Standards must be fully documented and need to be sufficiently detailed to allow independent conformance testing. Standards ratified and controlled by the International Standards Organisation (ISO) satisfied these conditions.

Products already exist in the IT marketplace which exhibit characteristics of openness. The following are essential characteristics of openness:

- products which are implemented to internationally agreed standards
- standards which are non-exclusive, non-proprietary and vendor independent
- applications which can be moved as necessary between systems of different makes and sizes
- information which can be exchanged when required between different systems

When planning for and implementing systems within an Open Systems framework, users will need to consider which of the above characteristics are important for them. Pragmatic use of industry accepted standards can clearly play a vital role in the emerging Open Systems marketplace whilst migration occurs to definitive international standards.

Users can determine what steps are necessary to achieve their objectives of migrating to Open Systems and the speed with which this migration can occur.
2. UNIX - DOS/Windows - Novell NetWare Interoperability

UNIX provides interoperability with DOS/Windows and Novell NetWare operating systems either at workstation level and network level.

At workstation level we have two possibilities in order to run different applications for different operating systems: UNIX and DOS/Windows partitions on the same disk and the possibility to run DOS/Windows applications which reside on the UNIX partition or on other partitions or machines.

On the UNIX server can coexist UNIX, DOS/Windows, Novell NetWare (bootable) partitions. Each partition can be specified at operating system installation time or later by repartitioning the disk. Thus, for booting the operating system from one of the partitions that resides on the hard disk, we have to use the operating system utilities, such as fdisk (available for UNIX as well as for DOS), in order to set the active partition (the partition from which the operating system will be loaded). So, if we want to run DOS/Windows native application we can boot from DOS partition, if we want a Novell NetWare server or UNIX server we have to boot from the proper partition.

On the other hand, the Advanced Merge package delivered with UnixWare operating system (Novell) allow the coexistence of DOS and UNIX file systems on the same partition. The DOS file system from UNIX partition can be build at UNIX installation time or later. Thus, after UNIX installation we can install new DOS/Windows applications exactly in the same way as in a DOS/Windows environment (e.g., Windows, Word for Windows, etc.). After that, all DOS/Windows files and programs can be view and run exactly as the UNIX files and tasks.

We can install any version of MS-Windows with some restrictions:
- we cannot use the utilities for doubling the disk capacity specific for DOS operating system.
- we cannot run Windows applications in extended mode.

All DOS/Windows files and applications installed in the UNIX partition and run as UNIX tasks or use in the UNIX file hierarchy take full advantage of the access rights specific for a multi-user system. Thus, we can use familiar DOS/Windows applications for system’s users and preserve the investment in DOS/Windows applications. On the other hand, the users take advantage of the UNIX multitasking, so, we can run at the same time, more UNIX applications and more DOS/Windows applications.

With the Advanced Merge we can see and run any other DOS/Windows files or applications located on other partitions than the UNIX partition.

At the network level, UnixWare allows the coexistence of the IPX/SPX and TCP/IP network protocols, so we have some advantages:
- running UNIX (X-Windows) applications located on the UNIX server from a DOS station
- running DOS/Windows applications located on DOS stations or Novell NetWare servers on the UNIX server (or UNIX client stations) in the same time with UNIX applications
- running client-server applications with DOS/Windows or UNIX front-end.
- sharing devices (laser printers, network printers, etc.)

We can run UNIX applications from DOS or Apple Macintosh workstations by using some packages like LAN WorkPlace for DOS, for Windows, for Macintosh provided by Novell. This package allows a network communication with IPX/SPX and TCP/IP protocols, so, from a workstation we can run applications from NetWare servers, UNIX servers or local applications. This package provides the communication infrastructure for client/server applications with a DOS front-end.
These packages also provide the coexistence of network protocols and some standard network utilities such as:

- **Telnet** - virtual terminal emulation on a PC with DOS/Windows
- **ftp** - file transfer between DOS station and UNIX servers
- **ping** - network diagnosties

On the top of this package we can run X-servers emulators (e.g., XVision from VisionWare or PC-XView from NCD) which provide the possibility of running graphic applications specific to UNIX environments in a Windows workstation. In this way on the Windows workstation we have the appearance of a UNIX server, i.e. we can see the UNIX desktop and run specific X-Windows applications.

If we have some expensive devices, such as big capacity disks or fast laser printers, in the network, we can share these devices in a transparent way.

For disks on the UNIX servers are used network file sharing (NFS) facility, so, it’s provided a transparent and reliable multi-user access based on known standards.

As we have already seen, on the UNIX server we can see both UNIX and DOS files and programs, as well as files and programs located on the Novell NetWare servers available in the network. In order to see the NetWare servers on the network we have to start the IPX/SPX daemon and connect to the specified NetWare server. All these operations are done automatically and transparent when we specify a name for a NetWare server.

The file hierarchy from the Novell NetWare server is mapped in one directory of the UnixWare file hierarchy and, after that, all these files can be accessed and run exactly in the same way as the UNIX files. These preserve the investment by offering the possibility of running all the DOS/Windows applications residing on DOS/Windows stations or Novell NetWare servers.

Due to standard communications protocols and programming interfaces (sockets) is possible the design of distributed client/server applications with a DOS, Windows or UNIX front-end.

At this time is available the commercial version 1.1 of UnixWare (System V Release 4.2) and the Beta-version for UnixWare 2.0 is in tests.

ComputerLand has been appointed to participate in the beta testing of UnixWare 2.0, so, ComputerLand is a beta site for UnixWare 2.0. The areas designed for testing and evaluation include: Installation, DOS/Windows Terminal Emulation (NVT), DOS/Windows Application Compatibility, On-line Documentation, E-mail, Internationalisation, Backup/Restore - TSA, GUI Network Administration Utilities, GUI System Administration Utilities, Kernel Modules/Driver Compatibility, Multiprocessing, Netware Administration Utilities, Network Management (Netware environment), NIS, Printing, Release Upgrades.

### 3. Internetworking

#### 3.1. Overview

In the past, many organisations used to buy just specific hardware support and software applications in order to satisfy their needs. This approach often resulted in the acquisition of a great number of totally incompatible systems, all developed independently. As the automation of further business functions required access to information residing on different corporate systems, common transfer mechanisms were adopted. At this point, the organisations were faced with a great number of dissimilar computer environments and with even greater systems incompatibilities.
To counter the proliferation of incompatible environments, most of the companies involved in Information Technology are now taking a strategic approach to International Standards. Solutions for satisfying the overgrowing communication needs consist also in connecting different computer environments together.

And, if the 80’s were characterised by the interconnection of standalone computer systems in a network, the 90’s were and will continue to be defined by the challenge of putting the networks together, into collections of networks known under the name of “internetworks”. The term of “internetworking” is generally used to refer to the industry that has arisen around the problem of connecting networks together. The term can refer to products, procedures, and technologies. The internetworking devices will be divided into categories based on the OSI layer at which they operate, as following:

1. **Repeaters**, as Physical Layer devices, strip out and save the received digital data. They reconstruct and retransmit the signal, able to travel over a new network segment.

2. **Bridges**, as a Data Link layer device, can determine the physical addresses of the source and destination stations involved in the transfer and, based on those determinations, they can permit or deny access to other segments. Because they can filter by station address, bridges are usually used to divide a too-busy network into separate segments.

3. **Routers** send information through an internetwork using logical (this means specific to the protocol) rather than physical address information. Routers perform three logical functions: route calculation, interface input/output and packet switching. Routers also use one (or more) specific “routing algorithms” to calculate the best path through an internetwork. Paths may be calculated in real time (dynamically), so that they can constantly adjust to changing network conditions. Dynamic routing algorithms differ in the factors (metrics) they consider when calculating the best path. Modern routing algorithms consider a variety of different factors, each weighted differently.

4. **Gateways** are devices that make the translation between incompatible network protocol implementations. Gateways can be constructed either as standalone, box-level products or as board and software combinations that augment the functionality of already-installed computers. Box-level devices are usually more expensive, but may provide better performance. Board-and-software devices may be either dedicated or non-dedicated.

### 3.2. Case study for internetworking devices

So, routers and bridges sit at the junction points of the new, high-speed developing internetworks. Today’s high-end routers connect dozens of LANs and WANs. To maintain high applications availability, these routers must support the full bandwidth of each attached network without requiring changes to the existing network infrastructure.

One of the companies that provide a complete line of products to meet the customers’ increasing need for internetworking is **Cisco Systems**. Cisco’s distributed processing architecture, software, and specialised high-performance network interfaces permit the construction of complex networks that can link thousands of network subsegments and connected devices. Cisco supports a full line of routers, from the **modular** Cisco 7000, AGS+, and the **Cisco 4000** to the **fixed-configuration** Cisco 3000 and Cisco 2500. These high-performance, intelligent routers can interconnect networks that use different protocols and media. In addition, Cisco routers can act as bridges by forwarding data packets from one network to another. Cisco also provides other internetworking devices: LAN extenders, Access Servers, AccessPro PC Cards and ATM Switches.

Cisco routers support the following **hardware media and interface types**: Ethernet (IEEE 802.3 and Type II), *Token Ring* (IEEE 802.5), *Fiber Distributed Data Interfaces* (FDDI) for single and dual attachment stations, *synchronous serial* (includes EIA-613 V.35, EIA/TIA-232,
EIA/TIA-449, G.703,X.21 and UltraNet), ISDN Basic Rate Interface and Asynchronous Transfer Mode.

Cisco also offers a range of scaleable solutions, all embodied in the Cisco Internetwork Operating System - IOS. This system integrates technology into a single infrastructure to evolve both existing and new applications. Also, this system provides one integrated platform for all users/applications, provides redundancy, security, high availability, manages resources cost effectively, scales efficiently with size and geography, and expands to new applications and services. CISCO’s standard router software supports the following protocols: TCP/IP, ISO CLNS, CMNS, DECnet Phase IV and Phase V, Novell IPX, AppleTalk Phase 1 and Phase 2, Banyan VINES, XNS, Apollo Domain, HP probe, IBM and serial tunnel (STUN) and IBM SDLC Transport. Also, are included the following routing protocols: IGRP, OSPF, RIP, EGP, BGP, PIM, Enhanced IGRP.

As options, Cisco routers can include bridging software (transparent bridging, source-route bridging, source-route translational bridging), protocol translation support (between X.25 and TCP, X.25 and LAT, LAT and TCP, LAT and TN3270, X.25 and Xremote), packet switching software (for X.25, LAPB, Frame Relay, SMDS, CMNS).

IOS Release 10.0 and later support interfaces. ATM is a cell-switching and multiplexing technology designed to combine the benefit of circuit switching (constant transmission delay and guaranteed capacity) with those of packet switching (flexibility and efficiency for intermittent traffic).

3.3. ATM architectures in internetworking technology

Like router switching systems, ATM switches analyse address information, review tables, and queue small information units at appropriate outbound ports. The architecture of a typical ATM contains three primary components:

⇒ A general-purpose processor that supports the user interface in the ATM switch, handles the setup and tear-down of virtual circuits during signalling, and runs routing protocols to determine the best path through a network of ATM switches.

⇒ A cell-switching fabric which is the heart of the ATM switch. The cell-switching fabric makes switching decisions and uses one of several popular switching fabric designs to move fixed-size cells in hardware, allowing much greater performance.

⇒ Interface modules that provide the physical network interfaces. They perform physical media bit coding and format and buffer the data into and out of the switching fabric. ATM interface modules are also responsible for changing the input path identifier to the output path identifier and for buffering multiple packets destined for the same output interface.

Cisco’s new silicon switching technology is used to create special purpose, programmable packet-examination machines. Logically, silicon switching acts as a coprocessor in its interaction with the switch processor. When the switch processor receives a packet, the packet is passed to the silicon switch for examination. With packet examination functions in hardware, the silicon switch can quickly read layer two and three headers and identify the appropriate bridging or routing information. The switch processor then forwards the packet to the appropriate interface.

In addition to its speed advantage, silicon switching is programmable so it can quickly be adapted to reflect new or changed protocols. For example, new versions of IP or AppleTalk can be accommodated easily. This programmability makes silicon switching flexible enough to be useful in large multiprotocol environments.

Cisco Systems presents two products that support ATM: the Cisco HyperSwitch A100 ATM switch and the ATM Interface Processor (AIP) for the CISCO 7000 series router. In Cisco
7000, the Cisco IOS meets each distinct requirement of a router (as presented in the first section) with a separate subsystem: a route processor for route calculating, a silicon switch processor for packet-switching and interface processors for input and output. Route calculation is typically done in the background, separately from switching or interface activity. Because route calculation can occur simultaneously with the other two activities, a dedicated processor was introduced for it. For route-processing Cisco uses a general-purpose processor, processor which provides programming ease and strong real-time performance for the route calculation function. For interface processors, Cisco uses different processors for each type of interface. The performance is increased also by the Cisco’s TRI-Bus architecture, that prevents the system from internal bottleneck.

The Cisco 7000 architecture bears a close resemblance to many ATM switch architectures. The Cisco 7000 route processor handles background tasks, the Cisco 7000 switch processor makes switching decisions and switches information units at extremely high speeds, and the Cisco 7000 interface processors create the appropriate physical media-dependent format.

4. Dependability

4.1. Overview

Dependability is defined as the trustworthiness of a computer system such that reliance can justifiably be placed on the service it delivers. This paradigm is very important for open distributed computing systems. Dependability may be understood from the following points of view:

⇒ Availability
This means the capability of an informatic system to be ready to provide a requested service at a specific time T. The availability of an informatic system at a specific time T is mathematically expressed like the probability of the system to be operational exactly at the moment T. Availability does not request a continuous functioning in the [0;T] interval, because the system may follow the loop [fault, recovery] several times during this interval.

⇒ Reliability
This means the capability of an informatic system to provide a continuous service. The reliability of an informatic system at a specific time T is mathematically expressed like the probability of the system to be operational during the whole [0;T] interval, supposing that the system was operational at the beginning (0 moment).

⇒ Safety
This means the probability to protect an informatic system against natural disasters like earthquakes, water-flooding, etc..

⇒ Security
This means the capability of an informatic system to assure confidentiality and integrity for its data.

- Confidentiality is the capability to protect objects (data to be accessed) against any kind of unauthorised use and from release of information to unauthorised parties.
- Integrity is the property of being change only in an authorised manner.

Data security includes important functions like object and subject identification, subject authentication and access control.

- Identification of both subjects and objects is the system’s capability to refer them by unique identifiers (numerical or alphabetical).
- Authentication is the verification of the identity of a party who generated some data, and of the integrity of the data.
• Access control represents the system’s capability to establish policies that determine the ways in which subjects can access objects and to contain mechanisms that implement those policies.

A security policy is a set of rules stating what is permitted and what is not permitted in a system during normal operation. A security violation is as an action violating the rules stated in the security policy. Threats can be seen as potential violations of security and exist because of vulnerabilities which exist in a system. There are two basic types of threats: accidental threats and intentional threats. Intentional threats, known under the name of “attacks”, are of two different kinds: passive and active. Vulnerabilities can be hardware-related, architectural and administration-related.

4.2. Dependable systems

A system failure occurs when the delivered service no longer compiles with the specification, the latter being an agreed description of the system’s expected function and/or service. An error is that part of the system state which is liable to lead to subsequent failure: an error affecting the service is an indication that a failure occurs or had occurred. The adjudged or hypothesised cause of an error is a fault.

A failure occurs when an error “passes through” the system-user interface and affects the service delivered by the system. Thus the manifestation of failures, faults and errors follows a “fundamental chain”:

\[ \text{... -> failure -> fault -> error -> failure -> fault -> ...} \]

Faults and their sources are extremely diverse: they can be classified in various different ways: accidental and intentional, physical and human-made, internal and external, design and operational, permanent and temporary.

The methods involved in developing a dependable computing system can be classed into:

⇒ Fault prevention
   These methods refer to the manner in which fault occurrence can be prevented. One of the most neglected areas of computer science, requirements analysis and specifications, is considered to be the main solution in this field.

⇒ Fault tolerance
   These methods are concerned about how to provide a service complying with the specification in spite of faults. Researches are made in order to design reliable hardware and software architectures.

⇒ Fault removal
   These methods are concerned with the problem of reducing the presence (number, seriousness) of faults. The main techniques are focused on classical verification mechanisms (code review, formal proofs). The main problem is that proving a complete non-appearance of errors is impossible.

⇒ Fault forecasting
   These methods refer to the main manners of estimating the present number, the future incidence, and the consequence of faults. Are used methods like: security modelling, statistical testing of software, coverage evaluation by fault injections, etc..

4.3. Authentication servers - Kerberos
Modern computer systems provide services to multiple users and require the ability to accurately identify the user making a request.

The authentication by password, very used in computing systems, is not suitable enough for using in secured distributed systems since the password sent across the network can be intercepted by intruders and used to impersonate the user. Thus, it’s necessary to use better techniques for a proper authentication mechanism. An example is Kerberos authentication server which allows users working on not secured sites access remote secured servers. For this, the user must be authenticated by Kerberos for subsequent accesses to servers. The hypothesis for such a system is that the users and servers trust the Kerberos authentication server.

The Kerberos Authentication System uses a series of encrypted messages to prove a verifier that a client is running on behalf of a particular user, i.e., the client has knowledge of an encryption key that is known by only the user and the authentication server. The user’s encryption key can be thought as a password and each application server share with the authentication server an encryption key. The encryption in the present implementation of Kerberos uses the data encryption standard (DES).

When a client wishes to create an association with a particular verifier, the client uses the authentication request and response messages in order to obtain a ticket and a session key from the authentication server. In the request, the client sends its claimed identity, the name of the verifier, a requested expiration time for the ticket, and a random number that will be used to match the authentication response with the response.

The authentication server returns the session key, the assigned expiration time, the random number from the request, the name of the verifier, and other information from the ticket, all encrypted with the user’s identification key registered with the authentication server. This information is sent together with a ticket containing similar information and which is to be forwarded to the verifier as part of the application request.

The authentication request and response comprise the basic Kerberos authentication protocol.

After that the clients send a request consists off two parts: a ticket and an authenticator. The authenticator includes: the current time, a checksum, and an optional encryption key, all encrypted with the session key from the accompanying ticket.

Upon receipt of the application request, the verifier decrypts the ticket, extracts the session key, and uses the session key to decrypt the authenticator. If the same key is used to encrypt the authenticator as used to decrypt it, the checksum will match and the verifier can assume the authenticator was generated by the principal named in the ticket and to whom the session key is issued. It is also verified the timestamp to make sure that the authenticator is fresh.

At this point the identity of the client has been verified by the server. For some applications the client also wants to be sure of the server’s identity. If such mutual authentication is required, the server generates an application response by extracting the client's time from the authenticator, and returns it to the client together with other information, all encrypted using the session key.

Authentication is critical for the security of computer systems. Without the knowledge of the identity of a principal requesting an operation, it is difficult to decide whether the operation should be allowed. The use of strong authentication methods that do not disclose passwords is imperative. The Kerberos authentication server is well suited for authentication of users in such environments.

Researches are in progress to implement a Kerberos authentication server in a UnixWare/Oracle/Tuxedo environment (Part of CEDINF IT national Research Programme).

### 4.4 Access Control
The purpose of access control is to limit the actions or operations that a legitimate user of a computer system can perform. Access control constrains what a user can do directly, as well as what programs executing on behalf of the users are allowed to do. In this way access control seeks to prevent activity that could lead to a breach of security.

Access control relies on and coexists with other security services in a computer system (such as authentication - Kerberos -, auditing, and security administration). To reach its goal (limiting the activity of legitimate users), the authorization service is enforced by a reference monitor which mediates every attempted access by a user (or program executed on behalf of that user) to objects (resources) in the system. The reference monitor consults the an authorization database in order to determine if the user attempting to do an operation is actually authorized to perform that operation. Authorizations in this database are administered and maintained by a security administrator. The administrator sets these authorizations on the basis of the security policy of the organization. Users may also be able to modify some portions of the authorization database, for instance, to set permissions for their personal files. Auditing monitors and keeps a record of relevant activity in the system.

It is important to make a clear distinction between authentication and access control. Correctly establishing the identity of the user is the responsibility of the authentication service. Access control assumes that authentication of the user has been successfully verified prior to enforcement of access control via a reference monitor.

It is also important to understand that access control is not a complete solution for securing a system. It must be coupled with auditing. Audit controls concern a posteriori analysis of all the requests and activities of users in the system. Auditing requires the registration (logging) of all user requests and activities for their later analysis. Audit controls are useful both as deterrent (users may be discouraged from attempting violations if they know all their requests are being tracked) as well as a means to analyze the users’ behavior in using the system to find out about possible attempted or actual violations. Moreover, auditing can be useful for determining possible flaws in the security system. Finally, auditing is essential to ensure that authorized users do not misuse their privileges. Note that effective auditing requires that good authentication be in place.

In access control systems a distinction is generally made between policies and mechanisms. Policies are high-level guidelines that determine how accesses are controlled and access decisions determined. Mechanisms are low-level software and hardware functions that can be configured to implement a policy. Security researchers have sought to develop access control mechanisms that are largely independent of the policy for which they could be used. This is a desirable goal in order to allow reuse of mechanisms that serve a variety of security purposes. Often, the same mechanisms can be used in support of secrecy, integrity, or availability objectives. On the other hand, sometimes the policy alternatives are so many and diverse that system implementors feel compelled choose one in preference to the others.

In general, there do not exist policies that are “better” than others; rather, there exist policies that ensure more protection than others. However, not all systems have the same protection requirements. Policies suitable for a given system may not be suitable for another. For instance, very strict access control policies, which are crucial to some systems, may be inappropriate for environments where users require greater flexibility. The choice of access control policy depends on the particular characteristics of the environment to be protected.

(In the “MOSS Project-Researches and Experiments Related to Open Systems Security -1995-”, we implemented “from scratch” a security system whose authorization server was based on a modified ACL mechanism - combined with capability lists - which served to implement a discretionary, descentralized, and closed policy for access control.)

References


CISCO Systems - Products Catalogue, 1994


Novell - UnixWare System Guide, 1995

Paunescu, F. et al. - “MOSS Project - Researches and Experiments Related to Open Systems Dependability”, August 1994

