Security in a Client-Server Environment

Gerald Bernbom
Assistant Director, Data Administration and Access
Indiana University
1000 E. 17th Street
Bloomington, IN 47405
Phone: 812-855-4624
Fax: 812-855-7868
bernbom@indiana.edu

Mark Bruhn
Manager, Security Administration
Indiana University
1000 E. 17th Steet
Bloomington, IN 47405
Phone: 812-855-7450
Fax: 812-855-7326
mbruhn@indiana.edu

Dennis Cromwell
Manager, Information Technology and Standards
Indiana University
1000 E. 17th Street
Bloomington, IN 47405
Phone: 812-855-7326
Fax: 812-855-7868
dcromwel@indiana.edu

Indiana University
Eight Campuses
94,000 Students
State University

Originally presented at CAUSE93, December 9, 1993 in San Diego, California, and published in the proceedings, this paper describes the strategy adopted by Indiana University in the design and implementation of a security model for the client/server environment. The strategic initiatives which were the catalyst for this effort are presented: deployment of a high speed network, an orientation to workstation-centered computing, and a growing commitment to open-systems solutions. Details of the IU security architecture are discussed, focusing on the three components of security -- identification, authentication and authorization -- and why they are critical success factors to implementing a client/server information system. The paper identifies security problems of an open-systems and client-server environment, the technology components of a security solution, and the work done by Indiana University to supplement the immature technology in this area. The paper also makes a case for clear analysis of security exposures, and the importance evaluating security solutions in light of evolving industry standards.

Security in a Client-Server Environment

"Securing a client/server environment is like throwing mud at the invisible man -- it may be messy, but pretty soon you get an outline of what you're up against."

(James Daly, *ComputerWorld Client/Server Journal*, August 11, 1993)

Security: the basic message

There are three basic messages that we want to communicate in this paper. First, that the purpose of security is to enable access, not to impede access. Our approach to the design of security solutions is focused on delivering access to computing and information resources at levels of risk that are known and accepted. The more access we want to deliver, the more attention we pay to security. Client-server computing opens new paths of access, and these require new security solutions.

Second, that there is no single solution to information security. The metaphor of a locked door -- that if we can just put the right lock on the door we'll be protected against intruders -- no longer applies. A complex computing environment, especially one that includes a client-server computing component, presents multiple points of entry: the workstation, the network, and local and central servers. The challenge is to recognize these points of entry, to understand and assess the risk that each presents, and to choose protections that are prudent and cost-effective.

Third, that even if specific security solutions are tactical responses to risk assessment (and they most frequently are), the process of designing and implementing security solutions is based on principles, objectives, and an overall strategy.

Information Systems at Indiana University: a two-minute tour

Security solutions are always in response to specific problems of access and risk. A brief overview of Indiana University's computing environment for enterprisewide information systems will help form the basis for understanding the design of our security responses.

The network is the integrating force in our computing environment; it ties together an array of computing and information resources, and is the bridge between central computing facilities and individual workstations or departmental networks. IU operates a multi-protocol network, carrying the TCP/IP, IPX, DECnet, and Appletalk protocols. User workstations at Indiana University are a mix of Intel-based DOS and Windows personal computers, Macintosh computers, and a small but growing number of Unix workstations.

The primary host computer for enterprise information systems has been a large MVS mainframe. To this we have recently added Hewlett-Packard application hosts running Unix (HP/UX) for client-server systems. We also run a large VMS cluster, primarily for instructional and research computing and as a host for IU Bloomington's campus-wide information system (though our CWIS is rapidly migrating to client-server technology with Gopher and World-Wide-Web).

The database management systems we use for enterprise information systems are DB2 on the MVS host,

and Sybase on Unix application hosts. We also run Ingres, again primarily for instructional and research computing.

Our application development and CASE tools are Uniface and Bachman. Uniface is an application development tool for creating client-server and host based applications. Bachman is used for data modeling, process modeling, and database design.

The computing environment that is the target for our first major administrative information system using client-server technology consists of: a Hewlett-Packard (HP/UX) host, the Sybase database management system, applications written in the Uniface development environment, TCP/IP connectivity, and client software running on Windows or Macintosh workstations.

The application that is the target for our first major client-server system, and thus the first iteration of a client-server security design, is a university-wide financial information system. We have explored issues of client-server security with two smaller systems that we have used to pilot new technology, but the security risks of a distributed financial information system -- entry of financial update transactions at their source, routing and approval of transactions by multiple users, and eventual posting of transactions to the general ledger -- required a more comprehensive security response.

Security: principles, objectives, and strategy

Security principles apply to all stakeholders in an information systems implementation effort: application developers, users, security managers, and university administrators. There is virtually no such thing as the elimination of risk in a computing environment. If there is a resource, and there is access to that resource, then there is risk to the resource. The principles of security that we apply are risk analysis and risk reduction, with the intent to manage risk at an acceptable level. What everyone involved in an information system implementation must understand is that the final design will entail risk. The responsibility of these stakeholders is to understand the risks, understand the steps taken to reduce risk, and accept the results.

Security objectives provide users, developers and security managers with a way to focus their analysis and attention on broad, general areas of risk. The primary objectives of a security analysis and response are:

- User identification. Knowing who the user is; assuring that each user can be uniquely identified.
- User authentication. Knowing that the user is who s/he says s/he is.
- User authorization. Knowing what is permitted or prohibited to each user, and enforcing these permissions and prohibitions.
- User accountability. Knowing and keeping record, for each access or other significant event in a system, the identity of the user responsible for that event.

A *security strategy* provides the framework for the development of an overall security design. It is the strategy that brings continuity and helps assure forward progress to the security efforts of an organization. The information systems security strategy that we use at IU has four key components.

Security design is *iterative*. We engage in a cycle of identifying security exposures, assessing the relative risk of each exposure, designing an intervention to reduce the highest risk exposures, evaluating the effect of the intervention, and identifying the remaining exposures.

Security design is *collaborative*. There are multiple stakeholders, both within the computing organization and among the user population, who have an interest in the design of security solutions. Because understanding and acceptance of risk is the basis for a security solution, these stakeholders must participate in design

process. There are also multiple areas of expertise needed to identify exposures, assess risk, and design interventions. In increasingly complex computing environments, the need for collaboration across several areas of technology specialization becomes greater: network designers, network operations staff, workstation software specialists, database administrators, in addition to application developers, user support staff, and security management.

Security design is *responsive*. Fundamental technology components are changing on an annual basis, if not more frequently. Changes in technology may open new exposures that did not previously exist. Or new technology may create opportunities to respond to exposures that had been previously left unaddressed. Security management and its collaborators from other technology areas need to monitor change in the industry, to assess the effect on risk and on the available measures of protection.

Security design is *accumulative and evolutionary*. Each security action is a response to a specific exposure or set of exposures; it is an intervention designed to reduce some known risk. As such, security actions are components of an overall security solution, but no total solution is ever implemented. One of the greatest challenges in security design is to choose components that work together, and that minimize the constraints placed on future choices of security actions. Equally challenging is to choose security components that fit with, or anticipate, the direction of the industry on providing information systems security solutions.

Securing the mainframe in an open environment

These principles and strategies were initially developed and refined in the design of security solutions for our mainframe computing environment, especially as we expanded access to this traditionally closed environment to a wider audience of university users. A brief overview of the migration of our mainframe connectivity from a relatively closed SNA network to a more open TCP/IP network will set the stage for the more radical transformation we are responding to in the area of client-server security.

At most institutions, security of the mainframe environment is relatively mature, and there is abundant experience in implementation and administration of the various host access control products available for these computers. The Indiana University situation in this area is typical: mainframe security has been the focus of our attention for some time. CA-Top Secret and Terminal Productivity Executive (TPX) have been installed for several years, and are interfaced to provide user login and menu services, password authentication and management, scripted application logins, and access authorization. CA-Top Secret is also integrated with many other program products to limit multiple application user data bases, which helps reduce administrative overhead.

Outside of these standard host access and authorization requirements, we also had some specific objectives to consider in providing access to the mainframe over a more open network:

- We had to offer "guest" access to an otherwise secure computer (e.g., to provide anonymous and unlimited access to the library's on-line catalog).
- We had to cope with unpredictable connections from diverse users on the same "open" network.
- We wanted to protect passwords on the network as much as possible.
- We needed to improve on password as the sole user authentication method.

In order to satisfy these objectives, computing staff members from data administration, security administration, and network operations spent many meeting hours analyzing the network topology for possible exposures. In the end, we addressed these concerns with four modifications:

- We established two "access areas" on the mainframe, each with its own network interface. One area

permits access to only "guest" services, such as the library on-line catalog and some student-oriented applications; the other area permits access to all defined applications. We used CA-Top Secret and TPX to enforce identification (login) and password management policies, and to limit the applications that could be accessed in the "guest" region of the computer.

- In conjunction with these separate access areas, we installed router filters that permit access to the secure access area only from a select set of networks within the IU domain, and that deny access from specific high-risk networks (e.g., campus public computing facilities).

- We implemented password token cards as an additional method of user authentication for users accessing the secure area. Each card is keyed to an individual user. It is used in a challenge/response dialogue during the system login sequence and must be in the possession of the user at that time. The combination of these two authentication methods -- something the user knows (password) and something the user possesses (password token card) -- is generally accepted in the industry as adequate for all but the most sensitive systems.

(Figure 1 gives an overview of this mainframe security configuration.)

We feel comfortable that our efforts in these areas have resulted in adequate protections for the mainframe environment. However, we still must contend with the constantly changing software set and various network topologies in order to ensure that changes to the mainframe and network environment do not adversely affect security mechanisms.

Client-server security

The client-server environment is new to almost everyone. It is a new way to provide access to the same data, stored in a new place, in a possibly new format. But there are still the same security requirements that were encountered in the mainframe environment. Security administrators, application developers, and system managers must still have the same comfort level in user identification, authentication, authorization, and accountability.

As designers of a client-server security architecture, starting basically from zero, we agreed on some basic understandings:

- All of our solutions may very well be interim ones.

- We must always plan for enhancements or replacement based on new software, changes in application requirements, changes in server configuration, etc.

- The mechanisms that we deploy should be able to protect against what we perceived as the highest risk exposures, both in terms of the degree of damage that *might* be done and the probability that the damage would actually occur.

Given these basic thoughts, we developed five core objectives for our client-server security design:

- protect host passwords;

- reduce exposure to network intruders;

- require the same challenge/response password tokens used for mainframe access;

- protect database server passwords; and

- restrict database server access to authorized connections.

There are several components that comprise the security architecture we have developed to satisfy these objectives: client application, network filtering, host security, Security Server, Gateway Server, and Telnet Server.

The client application performs two main security functions:
- it interacts with the host security process and provides the user interface to the challenge/response authentication dialogue; and
- it encrypts the user's host password during identification and authentication so that it never passes on the network in clear text.

A standard client module has been developed to execute these functions. This module can be called from any Uniface client application running on a desktop computer.

The network filtering element of the architecture is comprised of subnet router filtering and a subnet bridge. The router filtering is modeled on our use of network control of access to the secure mainframe region; the router filtering denies all connections to the host server from non-IU addresses and from high-risk addresses within the IU domain. The subnet bridge is placed directly in front of the database server on the host computer; it denies ANY network connections to the host's database server port.

The host security component involved the conversion of our HP/UX operating system to a "trusted system". This irreversible conversion (provided with the operating system by HP) involves the use of a shadow password file and the installation of an audit server, which permits full auditing of users or events.

The Security Server is the heart of the security architecture. This program interacts with all other components, and is the "authorizing agent" for access to the database server. For client application sessions, this host-based server receives, decrypts, and validates the username and password from the client application. If the supplied password does not match, a negative return code is passed to the client application. For both client and telnet application sessions, the server obtains a unique session challenge from the authentication software, and passes it back to the application for presentation to the user. The application then returns the user-supplied response, which the Security Server validates with the authentication software. Given that the challenge/response validation is successful, the Security Server generates a one-time database server password, accesses the database server and changes the user's password to the new one-time password, and writes the new one-time password and a session ticket to a database.

The Gateway Server manages access to the database server. All access to the database server MUST come through this program (the database server port is blocked!), and only with the "permission" of the Security Server. This Gateway Server intercepts connection requests to the database server, and searches a ticket database for a valid access ticket issued by the Security Server. Given that a current ticket is found, the Gateway connects the user to the database server with the one-time password that was issued by the Security Server and stored (encrypted) with the access ticket. Subsequent traffic for the user session are passed by the Gateway Server directly to the database server.

Although the applications developed for the client-server environment are primarily meant to be accessed from a client workstation, we also had to provide for a host-based version of the application for users without adequate devices to handle the client code. The Telnet Server uses the standard telnet service of HP/UX, and is invoked when users telnet directly to the host for host-based applications or other database access tools. The telnet service has been bundled with an interface to the Security Server as well as a menu structure. Following standard host login validation, the interaction with the Security Server provides the same challenge/response dialogue that the client user undergoes, and issues a database server access ticket and one-time password for the user session. The menu structure serves to limit user access to the HP/UX system prompt, and adds convenience for the user when choosing applications. The options on this menu vary with the user: some have only user application choices, others have DBA-oriented tools, such as Interactive SQL. In any case, applications on this menu which access the database server must go through

the Gateway Server, which will first check for a valid ticket in the ticket database before connecting the user to that database server.

By way of review and comparison of the security architecture components with our stated objectives we see that:

- o we have protected passwords on the application host by encrypting passwords at the client and by using the host's shadow password file facility;
- o we have reduced network exposure by using network router filtering to limit the source of connections to the application host;
- o we require the use of challenge/response password tokens for all accesses to the application host;
- o we are protecting database server passwords by issuing one-time passwords -- which are never known by the users -- for database server access; and
- o we are restricting database server access to authorized connections by denying direct access to the server port, and by requiring all other access via the Gateway Server.

(Figure 2 gives an overview of this client-server security design.)

It's worth noting that our client-server security design is a evolutionary development of our mainframe security design. The network filtering of traffic to the application host is borrowed directly from our mainframe security design, as is the use of challenge/response password token cards. In fact, our choice of vendor for password token cards was based on the requirement that a user be able to use the same physical card for authenticating his/her identity on multiple host computers.

Security: the state of the industry.

> "No significant headway has been achieved in any of the competing visions of enterprisewide security..."
>
> "It is left...to the user to build together the available technologies with sound business practices to guarantee the integrity of business information."
>
> (Gartner Group; "Client/Server Security"; *Third Annual Symposium on the Future of Information Technology*; October 4-8, 1993; Orlando, FL)

Our experiences in designing security solutions for a client-server computing environment are consistent with this view of the industry that the Gartner Group offers. There is, among the vendors we have worked with, no shared vision of a heterogeneous client-server security solution.

The database and software tool vendors we have reviewed and worked with offer basic security services, with much attention focused on the problems of authorization services: increasing the functionality of roles and groups, for instance, as a means of more easily managing the grant and revoke of database permissions. By contrast, the database and tool vendors have spent less effort on authentication services, which are often incomplete and need to be supplemented with outside help: either third-party or home-grown add-ons. Unfortunately, vendor emphasis is weighted toward proprietary security solutions: looking for answers within the constraints of their product offerings, rather than helping build solutions that cross these lines. Although their products are "open" in many respects, they are slow to adopt emerging security standards and are surprisingly closed when it comes to enabling software integration with products from other vendors or with user-written code. This mix of minimal solutions for user authentication and an unaccommodating attitude toward external software has made development of high-quality authentication services a particularly difficult challenge in this multi-vendor client-server environment.

Our experience is that the hardware and operating system vendors are doing a somewhat better job on security. They seem to have a good awareness of security issues, and are improving their solutions to problems of auditing, accountability, and system integrity. It is the hardware vendors, too, who have put the strongest support behind OSF/DCE, which presents the best potential as a standard for supporting a heterogeneous client-server security environment.

Responding to the industry

There are three ways in which computing organizations can respond to the state of the industry:

- Assemble its own client-server security solution.
- Design with the future in mind.
- Respond directly through collaboration and market pressure.

Since no vendor or group of vendors -- whether of hardware or software, of mainstream or specialty products -- offers a solution to heterogeneous client-server security that can be purchased and used, the only viable answer today is to assemble a security solution from a mix of purchased and locally-developed components.

In our first iteration of a client-server security design this consisted of:

- Selecting *specialty products that fulfill specific needs* in the computing environment and for the target application. (In our case we chose a specialty product, Unix-Safe, to offer challenge/response authentication on a Unix host computer.)

- Using *features available in primary products*. (In our case, we used the "open server" and "open client" features of our database product to develop a Gateway Server that validated user connections against a valid-ticket database.)

- Using *home-developed code to tie the pieces together.* (Our Security Server is a locally-written piece of code that interfaces to our Unix-Safe authentication product, interacts with stored procedures in our database product to set passwords, and writes the valid-ticket entries that our Gateway Server uses to permit database access.)

Given the developing state of the industry for distributed computing, any client-server security solution should be designed with the future in mind, acknowledging that the security design will be undergoing change for some time to come. One area to anticipate change is in the future features (announced, promised, or merely rumored) of existing software products. A second area to anticipate change is the potential adoption of standards-based features, such as those in DCE Security Services, by hardware and software vendors.

The future availability of these features should be considered in the initial security design -- postponing inclusion of the feature altogether or, if the feature must be locally-developed, isolating it in the design so that a commercial product or standards-based design may be more easily substituted in its place. For example, in our first design of client-server security we include a ticket-database which has interactions with the Security Server (the source of tickets) and with the Gateway Server (the user of tickets). If an industry standard for authentication tickets is adopted by any of our vendors, our design would permit us to replace this initial ticket management system with one that is standards-compliant.

A final course of action available to every computing organization is to create market pressure on vendors to adopt security standards and address client-server security needs in their products. They can make their case to vendors, arguing the need for security standards, and they can take their business to vendors who are willing to work with customers on security solutions. Organizations may do this individually or, more effectively, in collaboration with others.

Toward this end, the Big Ten computing directors have collectively endorsed the OSF/DCE standard for distributed computing and are focusing their attention on influencing a key group of hardware and software

vendors. One important way to influence vendors is to place security requirements prominently in all RFPs for client-server hardware and software. Compliance with standards or a commitment to work on an integrated security solution should be a heavily-weighted factor in the evaluation of any vendor's product. Indiana University used OSF/DCE compliance as a major criteria in its RFP and evaluation of host/server hardware for the client-server financial information system.

Conclusion

Indiana University is in the very early stages of implementing a security design for client-server computing that can be applied to enterprisewide information systems. The design we have today is a package of individual security actions in response to known exposures. Our view is that all client-server security designs will, for the foreseeable future, be a collection of such tactical security responses, and that our design will change and evolve in significant ways over the coming months and years. Our confidence in this initial security design is based on our strategy of iterative risk reduction and evolutionary growth; because we are addressing exposures in a planned way and are at the same time planning for change, we feel our first step is a step in the right direction.

Note: The work of two University Computing Services staff members needs to be acknowledged in this paper. Charles McClary (Information Technology Analyst) and Tom Davis (Security Analyst) have done significant research, detail design, and code development on our first iteration of a client-server security solution. Their initiative and individual efforts were essential to the overall success of this project.

Figure 1: Mainframe Security Design

Figure 2: Client-Server Security Design

Presenter's Biographic Sheet

Author/Presenter Information

Gerald Bernbom
Assistant Director, Data Administration and Access
Indiana University
1000 E. 17th Street
Bloomington, IN 47405
Phone: 812-855-4624
Fax: 812-855-7868
bernbom@indiana.edu

Gerald Bernbom is Assistant Director, Data Administration and Access at Indiana University. As part of University Computing Services, his unit is responsible for data administration, data dictionary management, campus-wide information systems and the information center.

Mark Bruhn
Manager, Security Administration
Indiana University
1000 E. 17th Street
Bloomington, IN 47405
Phone: 812-855-7450
Fax: 812-855-7326
mbruhn@indiana.edu

Mark Bruhn is Manager, Security Administration in the University Computing Services department of Indiana University. He has a Bachelor of Science in Computer Science (1986) from Park College in Parkville, Missouri. Mark has been with IU for 9 years, first in applications development, then in security administration. His unit is responsible for analyzing security exposures, and recommending, designing, and implementing security mechanisms required for the protection of administrative institutional data on all platforms where this data is maintained within the University. Currently, he is participating in a large team of computing and financial support staff involved with the development and deployment of a large client-server-based financial information system at IU. His group was specifically tasked with the design and implementation of a security system for this, and subsequent, client-server applications.

Dennis Cromwell
Manager, Information Technology and Standards
Indiana University 1000 E. 17th Street
Bloomington, IN 47405
Phone: 812-855-7326
Fax: 812-855-7868
dcromwel@indiana.edu

Dennis Cromwell is Manager, Information Technology and Standards at Indiana University. Dennis has been with IU for 3 years as part of the University Computing organization and his team is responsible for technology planning, technical architecture, information system standards and development methodology. Dennis holds a Bachelor of Science from Indiana University School of Education (1980) with a major in Mathematics and prior to coming to IU was a product specialist for a large software company.

Institution Details:
Indiana University is a mulit-campus

Title of Paper:
Security in a Client-Server Environment

Abstract:

Originally presented at CAUSE93, December 9, 1993 in San Diego, California, and published in the proceedings, this paper describes the strategy adopted by Indiana University in the design and implementation of a security model for the client/server environment. The strategic initiatives which were the catalyst for this effort are presented: deployment of a high speed network, an orientation to workstation-centered computing, and a growing commitment to open-systems solutions. Details of the IU security architecture are discussed, focusing on the three components of security -- identification, authentication and authorization -- and why they are critical success factors to implementing a client/server information system. The paper identifies security problems of an open-systems and client-server environment, the technology components of a security solution, and the work done by Indiana University to supplement the immature technology in this area. The paper also makes a case for clear analysis of security exposures, and the importance evaluating security solutions in light of evolving industry standards.