NEW DESIGN CONCEPTS FOR AN INTELLIGENT INTERNET

Geng-Sheng Kuo and Jing-Pei Lin

Addressing security requirements, performance efficiency, and operational simplicity in a high-speed network environment.

Information service is the focal point of all networks. In recent years, Advanced Intelligent Network (AIN), a prominent network-based service-independent architecture for current circuit-switched telecommunications networks, has been widely accepted by the global telecommunications and computer industries [5]. Its merit is to introduce and develop network-based information services intelligently, rapidly, and economically. Although considerable achievements on the concept and architecture of AIN have already been made, progress on its real development and deployment aspects is much slower than what we expected due mainly to many network-based distributed computing (from now on, the term network computing will be used) technical issues, especially in a multivendor multiprovider wide-area network (WAN) environment.

Currently, the information services created on AIN are very few and simple, which can hardly reveal the true value of AIN. The recommended speed of Signaling System No. 7 (SS7), performing all signaling control functionalities on service creation for AIN, is 56Kbps for the ANSI standard, and 64Kbps for the CCITT international standard [5], which might cause traffic-related issues when new services will be created and deployed on the networks. In addition, the current SS7 is a dedicated non-open packet-switched data network with no extensions made from outsiders. Therefore, there is no need for security considerations on SS7 yet, which might face significant challenges derived from internetworking with mobile wireless communications. With the AIN platform, SS7, and cellular mobile wireless networks as a starting point, Personal Communications Service (PCS) has already been planned and initiated by integrating wired and wireless networks together to provide users with personal and terminal mobility. It is therefore necessary to perform more research on secure network computing from different viewpoints.

End-consumer PCS needs. It is frankly impossible to completely understand consumers' detailed telecommunications needs. Of course, we already know the basic needs of consumers: voice communications, email, file transfer, remote login, for example. However, there are many specific service features

THE IMPORTANCE OF ONE POPULAR NETWORK COMPUTING TECHNOLOGY, REMOTE PROCEDURE CALL (RPC), WILL BE REFLECTED AND INCREASED CONSIDERABLY IN THE FUTURE HIGH-SPEED BISDN-BASED INTERNET.

needed by consumers in different places at different times under different circumstances, which can change very often. It is our expectation that PCS can provide many end-user-controlled capabilities for service customization, association, and differentiation in addition to location mobility and calling number uniqueness. It needs both network-based and end-user-based intelligences with signaling networks to achieve this flexibly and easily. The following crucial technical issues have been addressed:

- Internetworking of different networks—internetworking wireless networks with AIN, Internet, for example, will be one of the most important issues for PCS success.
- Network control—SS7 networks will emphasize network control due to service feature customization, association, and differentiation. The network control contains two levels: network level and end-user level.
- Network resource management—this can introduce and deploy new services easily and immediately, and can guarantee the quality of service (QOS) through negotiation. Here, the network resources include service-based databases, function-specific servers, bandwidth volumes, and so on.
- Integration and coordination—these are the key capabilities for creating versatile real-time multimedia information services for PCS in future high-speed ATM-based networks. The importance of signaling control functions will be increased considerably.
- Network security—there will be many service feature customizations and associations, and many service transmission interfaces and connections in WAN computing environment for PCS. In wireless communications with network access mobility, the challenges to network security will need to be faced seriously. Therefore, the network security will be one of the main concerns for the operations of service creation and transmission for future PCS.

Network technology trend. As is generally presumed, the current telecommunications networks will gradually evolve to ATM-based cell-switched Broadband ISDN (BISDN) [4]. At the same time, the packet-switched Internet is going to be updated to the high-speed ATM-based BISDN. It is quite reasonable to expect that both the future telecommunications networks and Internet will merge as the ATM-based BISDN in some degree, which will definitely include the AIN's intrinsic concepts. Due to the tremendous success of IPv4 internetworking technology and the considerable efforts on IPng, the emerging ATM-based Internet will play a driving role for shaping the future BISDN. Therefore, it might be better that the Internet will provide the signaling control role as a whole for future PCS in some sense. For reaching the requirements of intelligent, rapid, and economical network-based information service creation and deployment, the Internet is still going to emphasize the distributed client/server architecture as the base. It is obvious there will be many more clients and servers in the future ATM-based Internet than there are at present. Therefore, the importance of one popular network computing technology, remote procedure call (RPC), will be reflected and increased considerably in the future high-speed BISDN-based Internet.

Although both clients and servers in the Internet may have some access control mechanisms locally and independently to protect their own security, there will be inevitable threats and concerns to interprocess communications (IPC) and processing on distributed client/server architecture of network computing, and many challenges to the security of client/server-based RPC operations and processing in the high-speed ATM-based open-architectured WAN computing environment [3]. Hence, secure network computing [10] might be considered one of the most crucial technical issues impacting the success of future BISDN-based service creation and electronic commerce, and ensuring the correctness and legality of large-volume quick-response business-related electronic-media operations automatically. However, research results on this topic are not mature enough yet to achieve our expected complexity flexibly, and more efforts are needed to encourage new concepts and approaches or innovative technologies.

Our work. In this article, some new design concepts for secure RPC frameworks for information service creation on future high-speed ATM-based open-architectured Internet are proposed. The emphasis is on the WAN and open consideration. Obviously, the security requirement conflicts with performance efficiency. Our proposed new design concepts of secure RPC accomplish not only the security requirements but also the efficiency expectations. Our design concepts for a secure RPC for network computing, initiated in our 1993 work [6], are to shape a concrete RPC framework conceptually that achieves the security goal without hurting performance and retains or even improves the operational simplicity of distributed processing in a high-speed WAN computing environment.

Proposed Secure RPC Framework

We propose a new concept—that the client host and its corresponding server host can cooperatively accomplish secure RPC. The workload in our design is reduced due to the simplicity of our design and the balance of load distribution to each of these hosts.

The basis of our secure RPC framework is the client host and server host. A *client host* has users with many applications processes to run, the *clients.* It maintains crucial server information, such as service type and service instance, which is owned by all its clients. A client host also helps its clients choose appropriate service instances. In addition, the client host centralizes service binding and calling. The *server host* is a host of server processes, *servers*, who have many service instances. It maintains booked service instances, message encryption keys, and client host identifications. In addition, it manages service booking and client authentication.

There are several advantages of our secure RPC framework. A client host is responsible for service binding, including naming and calling of its clients, and server authentication. A server host is responsible for service booking and client authentication. There is no dedicated name server or authentication server existing in our proposed secure RPC activities. Therefore, the framework can avoid threats to IPC and need less time in booking or updating a service, and performing an authentication, especially in a WAN computing environment.

Since service booking is separated from service binding, the performance and flexibility of binding are improved considerably. If a server wants to book a service, its server host can book the service directly to a client host or a set of client hosts through a multicasting mechanism instead of through a name server. Therefore, the server has the capability to select the client hosts. In other words, the server can select a set of trusted client hosts to export its services via booking. Similarly, a client can choose one or more service instances to invoke one or more secure RPCs at a time. The conceptual diagram of a client host and its corresponding server host is illustrated in Figure 1. Any host in the future BISDN-oriented Internet might play these two roles—client host and server host—simultaneously, since clients and servers may coexist *within* the same host. It is definitely reasonable to perform our secure RPC within the same host. This can be considered as a local procedure call with reduced overhead of remote binding, argument copying, message passing, and so on.

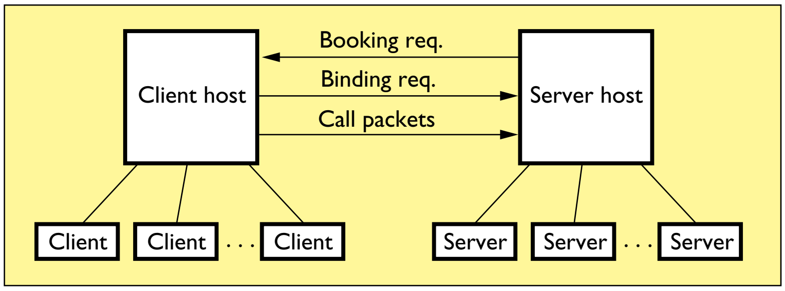

Figure 1. A client host and its corresponding server host. Clients and servers are managed by their client host and server host, respectively. All service booking, binding, and calling are carrried out at the host level; therefore, both security and performance requirements can be achieved.

Three main components of proposed secure RPC framework. Secure RPC is needed to guarantee any client securely communicates with its requested server in a network computing environment. Derived from the aspects of security in distributed systems [3], we propose a new secure RPC framework for high-speed global network computing consisting of three main components: *authentication protocols, service protection,* and *secure message transmission.* Authentication protocols are the procedures and formats for a server authenticating its access client and a client authenticating its requested server, respectively. Service protection is maintaining necessary client and server information, such as service instance identification. And secure message transmission means to transmit information securely between a corresponding client and server.

Since more service instances will be virtually stored on the high-speed BISDN-oriented Internet for resource sharing by all predefined or authorized clients, the authentication issue should be faced seriously. Authentication can be considered from two different aspects. First, the requesting client is checked by its corresponding server host to ensure it is genuine before it can perform the desired RPC on the requested server. An unauthorized client cannot perform any expected RPC on the server. Second, the requested server is checked by its corresponding client host to ensure that it too is genuine. The client can thereby be prevented from accessing unauthorized services. These two aspects are called client authentication and server authentication, respectively. When a client intends to perform an RPC on the requested server, the client itself and the server to which it wants to bind must first be authenticated in order to execute the RPC securely and correctly.

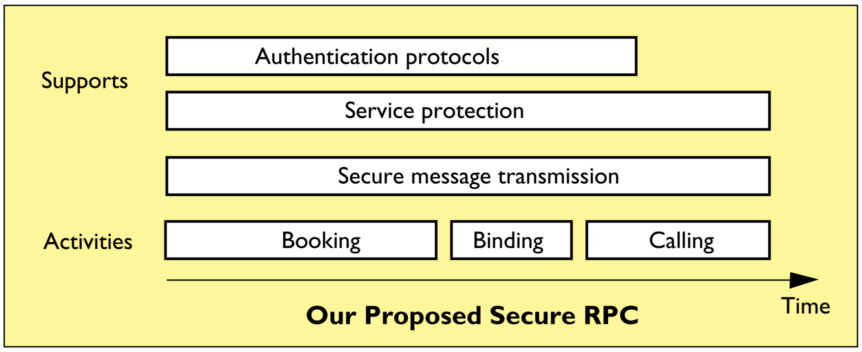

Figure 2. The three components of our proposed secure RPC are authentication protocols, service protection, and secure message transmission, which support the activities of our secure RPC: service booking, binding, and calling.

The validity of authentication is another security issue. To overcome this problem, service protection is provided that conducts the maintenance of additional important information, such as service instance identification, client host identification, and so on, before performing the authentication. It is also needed to transmit messages securely during the authentication. After the authentication is finished, the binding of the client with its requested server is made and considered to be successful. Since a logical connection between the client and its corresponding server has already been established, the only other concern is secure message transmission. The relationship among these three components: service booking, binding, and calling, is shown in Figure 2.

Encrypted digital signature. The encrypted digital signature has been used to perform the secure message transmission, authentication, and authorization. Messages are encrypted during transmission, and authentication is achieved by performing the necessary decryption and comparison of the signatures. This conceptual approach is borrowed from [12], which was originally used as an object (or resource) access control mechanism. In Amoeba [12], the digital signature was used for securely transmitting its capabilities.

In addition to the encrypted digital signature, the encrypted timestamp, depending on clock synchronization, has been used in the Kerberos authentication system of MIT Project Athena [1] and Sun ONC RPC [11]. However, there are some disadvantages to the timestamp. First, the timestamp is predictable because it relies on synchronized clocks. And the encryption keys for login of Kerberos and Andrew secure RPCs are easily guessed since they are derived from users' passwords, which are notoriously easy to guess [8]. Therefore, the timestamp encrypted by these keys is vulnerable to attack. Second, there are several significant flaws relating to the timestamp, such as message replay discussed in [1] and [2], and so on.

In 1989, Lomas, Gong, Saltzer, and Needham pointed out that a well-chosen encryption key should be selected at random from a large key space. Such keys can be used not only as digital signatures, but also as *challenges* [9] and *nonces* [1]. If the keys are changed frequently over time, the lifetimes of such keys become shorter, and the combination of such keys is more complex. Therefore, such messages are more secure than others against malicious attacks. And the encrypted digital signature is very useful for secure message transmission.

Service protection. The purpose of service protection is to protect service instances in a WAN computing environment from unauthorized access. We believe the identifications of clients and service instances should be maintained and that different users should have different access rights to the same service instance. The client host, server host, and the client handler in server stub maintain the related information. The client host maintains a service instance list to be used in service booking and binding for recording and referencing the service instance information, and a key table for secure message transmission. The server host also maintains an exporter list for service mapping in service binding, and a key table for secure message transmission. In addition, the client handler maintains a client host list for service booking, and a client list for service binding and calling.

The major aspect of service protection is authorization. The difference between authentication and authorization is that authentication is a process verifying the claimed identity of a client or a service instance, whereas authorization is a process allowing an authenticated client to use a specific service instance [7]. In other words, authorization is a process determining and verifying the access rights, or permissions, for a client to use a specified service instance.

In the client host list, there is one field known as *permission mask.* The permission mask determines the exact operation, or function, being performed on a service instance. The permission originating from the permission mask is also obtained in the service instance list at a client host during service booking. It is referenced by the client host in choosing an appropriate service instance at service binding. There is an option by which the client host can further determine the permission according to the client's identity. The function, specified by a client according to the permission, will be compared with the permission mask in the client host list at service calling. The service instance may have an option to store a new permission mask in the client list after each service binding in order to keep the *least privilege* [3] obtained by the client.

Authorization. Amoeba [12], Andrew [10] and Kerberos [7] use different approaches in achieving authorization. A comparison of those three systems with that of our proposed concept is shown in Table 1. The authorization models designed by Amoeba, Andrew, Kerberos, and ours are further compared on following three aspects—particular design, advantages, and disadvantages.

Amoeba [12] has sparse capabilities for naming and protecting services. Its capability consists of four components: *Server Port, Object ID, Rights,* and *Random.* The Server Port and Random are randomly chosen to make the capability sparse. This protects a capability from being used immediately by an intruder, because the intruder must know the service port name in order to make a correct service call and to break the Random encryption key to gain access to additional services.

The mechanism for generating subcapabilities in Amoeba is modified to make passing rights between processes easy. Each subcapability has a restricted set of rights, so that the owner of the subcapability has only restricted accesses to a specific service instance. Given an object, the revocation of privileges can be achieved easily since all capabilities of this object can be instantly invalidated by changing the content of Random. However, the service instance cannot know who should have its capability. Thus, message replay is possible if there is no client authentication. Additionally, the revocation of privileges is global. It is not very economical to have clients reproduce and pass capabilities all over again.

Andrew [10] uses the access control list for service protection. An access control list contains two sublists: *Positive Right* list and *Negative Right* list. Negative rights make the revocation of privileges efficient. Thus, the time-consuming process of determining all groups from which the user should be removed can be postponed. Both client and server hosts cache protection information for a small number of clients on each service instance in order to speed up permission checking. It is very hard to determine which service instances can be accessed, and which access rights a user can have immediately.

Kerberos [7] proposed that each service determine its own authorization mechanism to control who can use it and how to use it. And it provides a standard authorization model, so that each service can use shared public access control lists. Moreover, three authorization models of Kerberos—name server, authentication server, and public list server—can be integrated to provide integrity assurance.

The design goal of our proposed authorization model is to meet those three design principles [3]; the following have been achieved:

	Amoeba	Andrew	Kerberos	Our Proposed Secure RPC
Authorization Approach	There are no kernel-maintained tables or mechanisms for service protection. All service instances are named and protected by capabilities.	Access control lists.	A standard authorization model based on access control lists is provided, but each service instance can develop its own authorization model.	The service instance defines the permission of service functions to each client host. It maintains its own authorization information.

Table 1. The four different authorization approaches

- For any given service instance, it is easy and efficient to determine all client hosts which have access rights by using the client host list. After service binding, it is easy and efficient to determine which clients have access rights by using the client list.
- For any given client host, it is easy and efficient to determine all service instances that can be accessed and the associated access rights by using the service instance list. After binding, each client can know all service instances that can be accessed and the associated access rights by caching the binding information.
- Privilege revocation is easily achieved by a server host. The server host can delete an entry of the exporter list to revoke privileges of all clients. Or, it can delete an entry of the client host list to revoke privileges of all clients of a specific client host. Furthermore, it can revoke privileges of a client by deleting the entry in the client list.
- Both client host list and client list are stored and managed in the server stub. When the number of services increases, it scales up well. In contrast, the design of a public access list will suffer from maintaining a large list.

Improved performance. Since it is not possible to exactly know the time spent on each activity in our proposed secure RPC framework at the design phase, we do not compute the quantitative performance of our approach now. One of the most important design considerations is to use object-oriented technology on decomposing and designing our approach and to balance the distribution of total workload to a set of dispersed hosts in the Internet. Furthermore, the future high-speed networking and high-power computing will make the performance of any single piece of the workload insignificant to the total service performance. Therefore, our design concepts can minimize the waiting time spent during process synchronization, coordination, and creation, and improve the overall service performance considerably. The detailed study of activity frequency analysis and load index on our proposed secure RPC has already been made that supports our thoughts.

Conclusion

New design concepts concerning secure network computing for information service creation emphasizing both security and efficiency on a future high-speed ATM-based intelligent Internet have been proposed and discussed here. A concrete secure RPC framework has already been presented at a conceptual level. In that scheme, client and server authentications are performed by the corresponding server host and client host respectively, and all message transmissions and authentications are achieved through handshaking with encrypted digital signatures. Based on that, a detailed design of secure RPC without both dedicated name server and authentication server has further been proposed in our other work on the subject. In addition, the proofs of our designed protocols and the analyses of related performance considerations have been conducted and documented in our other work for reference. Furthermore, it is our strong belief that the improvements of secure RPC technology have positively affected the advancement and maturity of electronic commerce and business applications of information services on the future high-speed BISDN-oriented intelligent Internet. C

References

1. Burrows, M., Abadi, M. and Needham, R. A logic of authentication. *ACM Trans. Comput. Syst. 8*, 1 (Feb. 1990), 18–36.
2. Gong, L. A security risk of depending on synchronized clocks. *ACM Operating Systems Review 26*, 1 (Jan. 1992), 49–53.
3. Goscinski, A. *Distributed Operating Systems: The Logical Design.* Addison-Wesley, 1991.
4. Handel, R., Huber, M.N., and Schroder, S. *ATM Networks: Concepts, Protocols, Applications. 2d. ed.* Addison-Wesley, 1994.
5. Kuhn, P.J., Pack, C.D., and Skoog, R.A. Common channel signaling networks: Past, present, future. *IEEE J. Select. Areas Commun. 12*, 3 (Apr. 1994), 383–394.
6. Kuo, G.S. and Lin, J.P. New design considerations of secure RPC for future high-speed network-based distributed environments. In *Proceedings of GLOBECOM '93* (Houston, Tex., Nov. 29–Dec. 2 1993). IEEE, pp. 182–187.
7. Miller, S.P., Neuman, B.C., Schiller, J.I. and Saltzer, J.H. *Kerberos Authentication and Authorization System.* Project Athena Technical Plan Section E.2.1, Oct. 1988.
8. Morris, R. and Thompson, K. Password security: A case history. *Commun. ACM 22*, 11 (Nov. 1979), 594–597.
9. Otway, D. and Rees, O. Efficient and timely mutual authentication. *ACM Operating Systems Review 21*, 1 (Jan. 1987), 8–10.
10. Satyanarayanan, M. Integrating security in a large distributed system. *ACM Trans. Comput. Syst. 7*, 3 (Aug. 1989), 247–280.
11. Sun Microsystems, Inc. *Network Programming Guide.* Revision A of Mar. 27, 1990.
12. Tanenbaum, A.S. and Mullender, S.J. *The Design of a Capability-Based Distributed Operating System.* Rapport Nr. IR-88, Subfaculteit Wiskunde en Informatica, Vrije Universiteit, Amsterdam, 1984.

Geng-Sheng Kuo (gskuo@imrnet.mgt.ncu.edu.tw) is a professor in the Department of Information Management at National Central University in Taiwan.
Jing-Pei Lin (James_Lin@umc.com.tw) is a IC3 system automation engineer at United Microelectronics Corporation in Taiwan.

This work was supported by the National Science Council of R.O.C. under Contract NSC 83-0408-E-008-008.