M.I.T. Laboratory for Computer Science
Computer Systems Structures

Request for Comments No. 221
April 2, 1982

Ensuring The Satisfaction of Requests to Remote Servers in Decentralized Computer Systems

by Claudio Topolcic

The following is my Master's thesis proposal.

1. Introduction

In almost any multi-user computer system, a client may write his program so it will invoke servers over which he has no direct control because he cannot or prefers not to provide the services himself. It goes without saying that the user would want his program to behave as he intends under any circumstances. More realistically, since hardware breaks and systems crash, he may desire that his program either behaves as desired or does not do anything at all and he is notified of that failure. In fact, he usually must settle for some probability that the program will exibit this behavior, and some way to affect this probability. In order to ensure this kind of behavior, certain requirements must be placed not only on the facilities under the client's direct control, but also on services his program might use but over which he has no direct control. If the user invokes resources from remote sites in a network, the situation is much more complicated than if the resources were provided locally due to his lack of control over the serving site's hardware or software. However, making use of servers is a potentially more interesting issue in networks of personal computers because here a greater variety of software, hardware and peripherals can be made available to the user. This thesis will explore what kind of behavior can be required of remote servers in order to produce the desired behavior of a user's program, and useful ways of ensuring this behavior within a realistic model of a network of personal computers.

Mechanisms such as type safe languages and program certification are intended to ensure that programs behave as their programmer intends. They have been the objects of research for many years. Such mechanisms are very different from those which ensure the behavior of servers invoked by the program, but provided by different programmers. The difference comes mainly from the lack of direct control of the service's programmer by the client. Ensuring that a server behaves exactly as its programmer intends does not ensure that his intentions are not malicious. This thesis cannot be broad enough to cover all these issues. It will therefore ignore local program failures by assuming that any facility under the client's direct control will behave exactly as he desires.

2. Using servers

Even if the client's programs are assumed to be perfect, there are still good reasons why he might use services provided by others. Three popular ones might be the following. First, the client might not have the ability to provide certain services for himself. In a centralized system, he would probably not have direct access to a secondary storage device, so would have to invoke a system procedure to do so. In a decentralized system, the client might not have the desired secondary storage device at his local site, so he would have to access a remote file server. Second, he might have the ability to provide the service, but a server might be able to provide it at a lower cost. A site with a floating point unit could provide mathematical functions to less sophisticated sites in a network. Third, a client may not wish to duplicate effort already spent on building a server if it is willing to provide the fruits of that effort. A user might invoke a compiler even though he could create one himself.

Part of this thesis will concern determining what kind of behavior should be required of a server in order to ensure the proper operation of a client's programs. A reasonable starting point is to assume that in order to ensure the acceptable overall behavior of his program, the user must be sure that any services invoked by his program either operate correctly or if they don't, do not have any harmful effects and his program discovers the failure. The program should know of the server's failure so that it can take remedial actions, such as using another server, or notifying the user if no remedial actions are possible within the system. A server might fail to deliver the desired service because of the malicious intentions of the server's programmer, unintentional bugs in the server, or malicious or unintentional interference by a third party. The client should not suffer any failures because of these events.

3. Servers in a centralized system

In a centralized system, a user program invokes a server when it calls a shared procedure which is maintained by some other programmer. The client might see a problem because he has no direct control over the service, so cannot know its behavior. The server might provide the desired service, it may do nothing at all, or it may attempt to do something malicious. In a centralized system, certain mechanisms exist which simplify the problem. The system's architecture is known publicly, so its strengths and weaknesses are known by the client. For example, a system which has good confinement mechanisms could limit the extent to which the client's information could be leaked by a malicious server. Furthermore, there is the inherent concept of a trusted central authority in such a system, both within the computer itself and in the administrative organization. Within the computer, many features can be mediated by the operating system, which serves as a trusted authority. For example, inter-process communication can be an operating system function, so that third party interference can be eliminated or at least detected. In the organization, since there is a single computer system, there is a single administrative authority, which for better or worse will enforce policies concerning the clients' and servers' behavior. The administrator might exercise legal, social or economic pressure on a service provider who is the cause of too many client complaints.

4. Servers in a decentralized system

In a decentralized system, the problems are much worse than in a centralized one because many of the mechanisms which are available in the centralized system are not available here. In such a system, the client may have control of a single computer (assumed infallible for simplicity), which is connected by a network to some indeterminate number of other computers, including the server. Its communications with the server is by network messages which may in fact be generated or intercepted by any site. The client has less control over what occurs in this system than in a centralized system. Specifically, he has no direct control or knowledge of the hardware or software at any remote site. Since the client has no knowledge of the architecture of the server's system, he can make no assumptions about what the server can or cannot do. There is no inherent concept of a central authority; the network itself does not represent one. In fact, the client has no direct, built-in control over the sites with which he communicates. Mechanisms do exist which help alleviate this problem. Known encryption systems solve most communication problems, but they don't completely ensure desired behavior from remote servers, and they are often based on the existence of some trusted central authority.

5. Scope of this thesis

As noted above, this thesis will not be concerned with proving local program correctness. Furthermore, it is concerned only with the problems of decentralized systems. This problem can be further divided into a problem of ensuring that servers actually provide the service they are supposed to and a problem of ensuring that servers cannot leak any of the client's information. This is equivalent to stating that a server should do what it is supposed to and it should do nothing else. Since controlling the escape of information is a topic of intense research, and ensuring satisfaction of requests is not, this thesis will be primarily concerned with ensuring satisfaction of requests.

6. Relation of this thesis to actual situations

As local networks and sets of networks grow out of individual buildings or organizations in the near future, they will increase the opportunity for clients to make use of untrusted remote servers. As networks grow larger, more varied computers will be connected to them, and some will make their special abilities available to others for profit. This will be an attractive situation because other sites, especially personal computers, can be made less expensive if they are designed to make use of services from the network. Furthermore, the low cost and high bandwidth of local networks will make the required communication acceptably inexpensive. However, along with this increased opportunity, there is a decreased control and accountability over remote sites. To take advantage of the possibilities, clients of such networks will require mechanisms which allow them to use untrusted remote servers with reasonable assurance of success. Research to provide such security within realistic models of the real world will be of practical importance there in the near future.

7. Related work

Many problems of secure communication over insecure networks are almost completely solved by known encryption techniques as described by Needham and Schroeder[1], Kent[2], and Rivest, Shamir, and Adleman[3]. This thesis will use these mechanisms as foundations on which to build other mechanisms, and will make no attempt to study their mathematical basis. Known encryption techniques will be viewed as black boxes with predictable characteristics.

The concept of ensuring desired behavior from untrusted agents has been alluded to in the mechanisms of digital signatures as described by Needham and Schroeder[1], Rivest, Shamir, and Adleman[3], and Popek[4], as well as many others. The purpose of these mechanisms is to prove the origin of a message to some third party. These systems, however, do not include a sufficiently complete description of how the digital signature is used to ensure proper service, some are based on the unrealistic assumption that a trusted authority exists, and they all lack robustness under the assumption that secret keys can in fact be discovered. Although based on many of the concepts and mechanisms of such digital signatures, this thesis will attempt to provide a more complete, realistic, and robust system.

8. Schedule

April 15
Complete development of a useful system of guarantees, including a realistic model of a network of personal computers, a key distribution system, and a description of how failures are handled.

April 30
Address issues which were not included in the above discussion, such as ensuring the correct operation of transactions which cannot be repeated, and protecting information from being leaked by servers.

May 15
Determine the implications of this system on the client's operating system and programming language, and define primitives for use by the client to control the system from his programming language.

May 31
Determine what can be said about the overall security of the client's programs using this system.

August 31
Complete writing the thesis report.

9. References

[1] Needham, R. M. and Schroeder, M. D. Using Encryption for Authentication in Large Networks of Computers. Comm ACM 21, 12 Dec. 1978.

[2] Kent, S. T. Encryption-Based Protection Protocols for Interactive User-Computer Communication. MIT/LCS/TR-162, May 76

[3] Rivest, R. L., Shamir, A., and Adleman, L. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Comm ACM 21, 2 Feb. 78

[4] Popek, G. J. and Kline, C. S. Encryption and Secure Computer Networks. Computing Surveys 11, 4 Dec. 79