A Byzantine Resilient Approach to Network Security[1]

Axel W. Krings
Computer Science Dept.
University of Idaho, Moscow ID, USA
krings@cs.uidaho.edu

Miles A. McQueen
INEEL
Idaho Falls, ID, USA
amm@inel.gov

This paper describes a joint project between the University of Idaho and the Idaho National Engineering and Environmental Laboratories (INEEL) which addresses solutions to the problem of malicious intrusions in networked systems. A new approach to network security is proposed that is based on technology transfer of methods and algorithms found in restrictive distributed fault-tolerant systems. The principles of hardware and software redundancy are applied to general network security. Whereas much research focuses on intrusion avoidance and intrusion detection, this research addresses survivability issues of systems in the presence of malicious, possibly coordinated, attacks. Specifically, multiple distributed copies of critical agents or functionalities are used that combine fault-tolerant agreement to mask malicious network attacks. Thus the focus is shifted from intrusion avoidance and detection, to guaranteeing that critical functions are secure even in the presence of intrusion. Now malicious acts can be masked, while intrusion detection methods are applied. A byproduct of intrusion masking is intrusion detection based on the critical functionalities.

There are many commonalities between the area of fault-tolerance and network security. The main goal of fault-tolerant system design is to mask hardware faults, such that failure of components does not affect the ability of the system to perform to its specifications. Several fault models exist in which faults are classified by their behavior and relative probability of occurrence. For example, the likelihood of self-evident faults, i.e., benign faults, is much greater than that of malicious faults [Tha88], i.e., Byzantine or asymmetric faults. Fault-tolerance is mainly achieved by means of redundancy. One key issue in redundant systems is the implementation of agreement algorithms in order to guarantee that all non-faulty processors agree on the same data or action. A multitude of agreement algorithms exist to achieve agreement under varying assumptions and cost in terms of system overhead.

The fault-tolerant concept just described matches very closely issues of system survivability and network security in distributed systems, where mechanisms like authentication protocols are used to prevent unrightful access or malicious tampering. However, in network security the efforts after an attack focus on the detection of infiltration and identification of the attacker rather than fault masking. Furthermore, whereas in fault-tolerant systems malicious faults are the least likely, in network security the attacker is expected to behave maliciously.

Our research considers an approach to system survivability and network security that takes the philosophy of fault-tolerant multiprocessor systems implementing Byzantine Agreement and transfers it to general distributed systems. The problem with any non-replicated approach to survivability is that any mechanism that empowers can be used against you. Once a hacker has “superuser” (or “root”) status, little stands in his/her way to compromise the system. Consequently, power has to be decentralized, requiring redundancy and the overhead that results from redundancy management. To reduce overhead redundancy should be limited to specific functionalities or services and should be used sparsely. The principle of redundancy based on critical functionalities can be described best using an example. Domain Name Servers (DNS), which resolve network addresses, are one common subject of system attack. Rather than the usual single DNS executing on one specific server, assume that DNS is replicated on several network nodes. Upon receiving a DNS request, the DNS servers initiate a voting process to reach agreement on proper DNS resolving, thus masking and possibly identifying any compromised DNS server. This approach of course assumes that enough servers have been unaffected by the attack.

We have derived a framework that is based on critical functionalities. These functionalities, ranging from DNS to *telnet* and *passwd*, need to be identified as required by specific applications domains. Once identified, instrumentation of the operating system for specific functionalities is required. The operating system is minimally modified with a call to and return from an isolated module, the so-called Byzantine Agreement Module (BAM). The BAM is the central agent on each computer that deals with management and specifics of defined critical functionalities as well as all communication to BAMs on different systems participating in the replication. The concept is explained

[1] This project has been supported by the INEEL and DoD TSWG

using Figure 1. Here the operating system is executing functionalities f_i on processor p. In

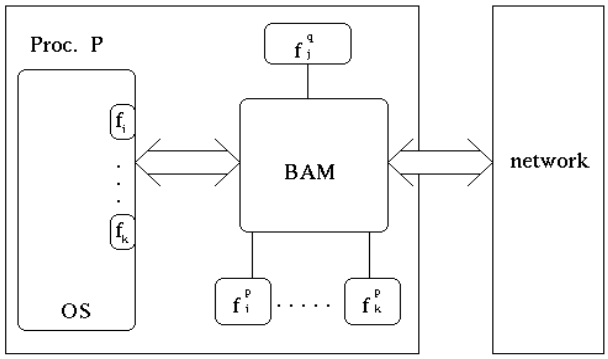

Figure 1: BAM Integration

the unmodified operating system, f_i simply executes and terminates. However, here f_i is modified to call the BAM module. The BAM on processor p is interfaced with a service module which implements any code specific to the modification of f_i. Service modules f_i^p are in one-to-one correspondence with their critical functionalities, i.e., service module f_i^p maps to functionality f_i on processor p. As a result, the BAM on processor p may have a service module f_j^q which corresponds to functionality f_j on processor q. After receiving a call from f_i and under considerations of the specifics in f_i^p, the BAM initiates an agreement with all processors involved in the replication of the functionality. This set of processors is called the *survivability cluster* of f_i and need not be the same for all functionalities. The BAM forwards the request of f_i to all processors in the survivability cluster which in turn execute their corresponding service modules. At this point in time each BAM in the cluster derived a result. Next all BAMs in the survivability cluster initiate a voting algorithms. The voted upon result is returned, processed, and a return from the BAM to f_i finishes up the execution.

With respect to the DNS example above, a request to resolve the address for computer "snake" in Figure 2 results in a call to the local BAM. The BAM then requests each processor in the survivability cluster to resolve the same request using their respective service modules. If the DNS server has been compromised by an intruder and the DNS entry has been maliciously altered from 129.101.55.119 to 129.101.55.22, the BAM will encounter disagreement during the voting process and will overwrite the DNS resolution of the DNS server with the agreed upon value.

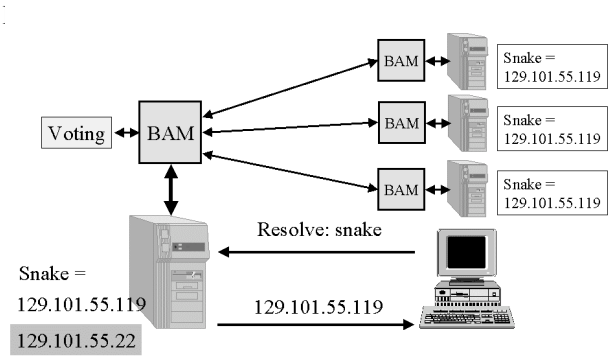

Figure 2: Survivable DNS

The principle of minimal redundancy is applied. Thus we have selected the least restrictive agreement algorithm for each functionality. For functionalities with no possibility for initial disagreement a simple majority vote is sufficient. Wherever Byzantine agreement is necessary we have selected early stopping algorithms [Kri99] in order to reduce overhead. Some functionalities have different redundancy requirement depending on the service provided, e.g., the DNS resolution of Figure 2 requires only results from 2 of the 3 redundant servers in order to achieve agreement in the primary DNS server. However, DNS updates require up to 4 processors in order to ensure consistency.

Since our approach requires alterations to the operating system and thus access to its source code, our implementation is based on Linux and is written in C. We are currently implementing libraries of functionalities and are analyzing overhead and synchronization issues. In the next project phase intrusion detection and fault isolation based on functionality-usage profiles are envisioned.

References

[Tha88] Thambidurai, P.M., Park Y.K., "Interactive Consistency with Multiple Failure Modes", *Proc. 7th Reliable Distributed Systems Symposium*, Columbus, OH, pp. 93-100, Oct. 1988.

[Kri99] Krings, A.W., Feyer, T., "The Byzantine Agreement Problem: Optimal Early Stopping", *Proc. 32nd Hawaii International Conference on System Sciences*, No. stdds03, pp. 1-12, January 5-8, 1999.