J. N. Froscher, D. M. Goldschlag, M. H. Kang, C. E. Landwehr, A. Moore, I. S. Moskowitz, C. N. Payne. "Improving Inter-Enclave Information Flow for a Secure Strike Planning Application," Proceedings of the 11'th Annual Computer Security Applications Conference, New Orleans, Louisiana, December, 1995, pages 89-98.

Improving Inter-Enclave Information Flow for a Secure Strike Planning Application

Judith N. Froscher, David M. Goldschlag, Myong H. Kang,
Carl E. Landwehr, Andrew P. Moore, Ira S. Moskowitz,
Charles N. Payne*
Center For High Assurance Computer Systems
Naval Research Laboratory
Washington, D.C. 20375-5337
+1 202.767.2389†

Abstract

DoD operates many system high enclaves with limited information flow between enclaves at different security levels. Too often, the result is duplication of operations and inconsistent and untimely data at different sites, which reduces the effectiveness of DoD decision support systems. This paper describes our solution to this problem as it arises in installations of the Joint Maritime Command Information System (JMCIS), an integrated C4I system. Our approach views databases in more classified enclaves as potential replica sites for data from less classified enclaves. Replicated data flows from lower enclaves to higher ones via one-way connections, yielding a high assurance MLS (multi-level secure) distributed system. The one-way connections are the only trusted components. This approach is based on our work on SINTRA (Secure Information Through Replicated Architecture), and applies generally to any collection of systems each running a database at system high. It complements and exploits modern system design methods, which separate data management from data processing, and enables effective, low-cost MLS operation within that paradigm. In addition to describing current JMCIS installations and our architectural approach, the paper presents our approach for justifying a system's security and our use of formal methods to increase assurance that security requirements are met.

Keywords: Accreditation, confidentiality, formal methods, high assurance, replication, security.

1 Introduction

DoD operates many system high enclaves with limited information flow between enclaves at different security levels. Too often, the result is duplication of operations and inconsistent and untimely data at different sites, which reduces the effectiveness of DoD decision support systems. This paper describes our solution to an instance of this problem that arises in operations of the Joint Maritime Command Information System (JMCIS), an integrated Navy C4I system used for tracking ships and planning missions.

NRL's Center for High Assurance Computer Systems is leading a one-year project to improve information flow in JMCIS. Most installations have two JMCIS systems: a less classified (LOW) JMCIS system and a more classified (HIGH) JMCIS system. The LOW and HIGH systems each includes its own copy of the Central Data Base Server (CDBS) that serves many applications. Users at each level update their local CDBSs independently, although updates to the LOW level system are provided periodically to the HIGH level system via tape. The current mode of operation does not permit HIGH users to exploit LOW CDBS updates promptly or consistently.

A conventional approach to this problem might call for installing a bi-directional "guard" processor between the two systems. Such guard systems, though increasingly prevalent, typically require a human reviewer to monitor traffic they pass, because they have the capability for sending traffic in either direction.

Our experience developing SINTRA (Secure Information Through Replicated Architecture)[5] provided us with a different framework in which to consider possible solutions. From the SINTRA perspective, the HIGH CDBS simply needs to include replicas of data residing in the LOW CDBS. Instead of a potentially bi-directional guard processor, the SINTRA approach calls for a device that permits one-way flow of information upward. Because the flow is upward and consists of structured database records, not programs to be executed, human review should not be required.

We need not alter the basic operation or the com-

*Author's current address: Secure Computing Corporation, 2675 Long Lake Road, Roseville, MN 55113, +1 612.628.2700, cpayne@sctc.com

†e-mail addresses: {froscher, goldschlag, mkang, landwehr, moore, moskowitz}@itd.nrl.navy.mil

puting platforms of the systems; instead, we add a small set of untrusted software and a single trusted device which combine to automatically replicate selected data from low databases to high databases in real time, while maintaining the confidentiality of the high data. In this way, users of a more highly classified system can use data produced by a less highly classified system without degrading the security posture of the existing systems.

The SINTRA approach replicates some of the tables on the LOW CDBS as read-only tables on the HIGH CDBS. As LOW users update their primary copies, the updates are forwarded to tables in the HIGH database. HIGH users can read, but not modify, the replicated tables, so they remain consistent with the LOW tables. Replication is accomplished by a commercial database replication product, integrated with a high assurance one-way communication device. The technical challenge is to find a way to use the replication product in an environment where confidentiality requirements dictate that there be little or no communication, even indirectly, from the HIGH CDBS to the LOW CDBS, and hence no way to pass error messages or other control information from HIGH to LOW. The solution must satisfy both the security requirements and the functional requirements of reliable data replication.

Before a system can be used to process classified information, it must be accredited for that purpose. We must provide rigorous evidence that the system is secure. The SINTRA approach provides a framework that isolates critical functions, making the analysis tractable and amenable to mathematical techniques, including the application of formal methods.

The rest of this paper is organized as follows: Section 2 describes the fielded JMCIS and discusses the security issues that constrain our design. Section 3 describes SINTRA and how the SINTRA approach can be applied to improve information flow between JMCIS enclaves. Section 4 describes a replicated architecture for JMCIS based on commercial products, but lacking needed security controls. Section 5 describes how to use the commercial replication architecture, together with a high assurance one-way communication device and suitable "wrappers," to build a practical system that meets the security requirements. Section 6 describes our approach for constructing an assurance argument that includes the use of formal methods to provide evidence to accreditors that the architecture is secure. Section 7 describes our plans and conclusions based on our progress to date.

2 JMCIS Context

2.1 JMCIS Operation

JMCIS is a collection of application programs integrated with a shared database (CDBS). It tracks ships and ground forces in support of the development of both defensive and offensive plans, which may include air-strikes, refinement of forces in consideration of the local environment, and actions to obtain more information. JMCIS applications analyze military message traffic, satellite imagery, and data from other sources to develop a coherent picture of some part of the world, presented to the user as an annotated map. Users can click on map objects to get more information. Friendly platforms are colored *blue,* enemy *red,* and neutral *white.* Incoming information cannot be processed completely automatically, since data are often ambiguous: human intelligence may be needed to recognize that two messages refer to the same ship by different names or to recognize a meaningful pattern in a combination of sensor reports. The resolved information and successfully parsed messages are stored in CDBS.

Much of this information is available at the LOW level. It makes sense to correct the data at that level if possible, because there are many LOW customers for the information.

Concurrently, analysts on another JMCIS system running at the HIGH level are also annotating their maps, using information from HIGH sources. For example, HIGH sources may confirm that one track in fact represents two ships, or they may suggest that a *white* merchant ship is not neutral after all. This information can change the map or the interpretation of its objects. The LOW analysis helps the HIGH analysts, but to reduce the chance of leaking HIGH information, it is primarily conveyed by physically moving data by tape. This approach limits the timeliness and consistency of data between the two systems. The same track may be assigned a different number in each system, making coordination difficult and potentially yielding incorrect HIGH plans. The existing constraints on data flow thus either reduce the quality of the data or require duplication of effort, wasting scarce resources.

2.2 JMCIS Architecture

A typical JMCIS configuration, shipboard or ashore, is a collection of workstations, all operating at the same classification level, running JMCIS applications that communicate over a local area network (LAN) (figure 1). One distinguished workstation processes external message traffic. Another provides the central database service, CDBS, using a commercial relational database. Oracle is used on land and Sybase aboard ships. CDBS currently stores both national resources, including maps, image catalogs, and reference information, and tactical resources, including messages, plans, and tracking data. It will continue to grow in importance as JMCIS applications share data better.

An installation with both LOW and HIGH JMCIS systems has two LANs (figure 2), each running JMCIS workstations, CDBS, and a server that processes

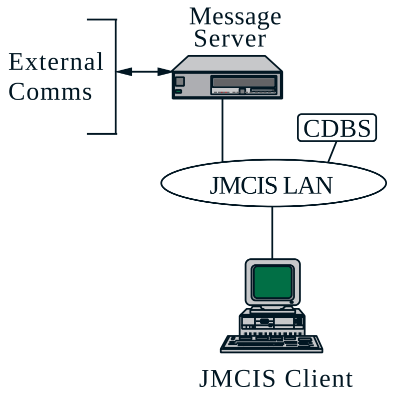

Figure 1: A single JMCIS LAN.

external messages. A standard serial cable with a pin cut to eliminate physically the return data path connects the LOW message server to the HIGH message server, replicating incoming LOW messages.

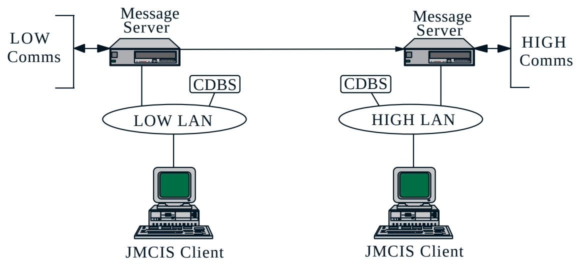

Figure 2: Partially linked LOW and HIGH JMCIS LANs.

Other information from the LOW LAN can be forwarded to the HIGH LAN by disk or tape, or it can be routed manually through the LOW message server. Although this architecture preserves the confidentiality of HIGH information, a HIGH analyst's access to LOW results is neither automatic nor efficient.

2.3 Security Requirements

Our goal is to provide a direct, reliable one-way connection from the LOW CDBS to the HIGH CDBS that permits timely use of LOW results while limiting the possibility that HIGH information can leak to the LOW system. Leaks could occur directly through the transmission of sensitive information, or indirectly through the transmission of sensitive information that has been encoded in covert channels (e.g., [18]). Information can be encoded by manipulating legitimate communication, for example, changing message formats slightly (storage channels) or by manipulating the timing of legitimate operations (timing channels).

The fundamental security requirement is to quantify the information leakage that is possible and identify the mechanisms that are needed to exploit these leaks. This analysis can then be used to evaluate the risk to the confidentiality of the HIGH information.

The requirements for protection of classified information are usually specified by the agency that "owns" the information. In JMCIS, HIGH information is owned by the Defense Intelligence Agency (DIA). DIA's requirements (DODIIS[4]) are the source of our concern about leakage (both overt and covert). This document also requires a careful analysis of the system to justify its security. Our approach to this assurance argument is presented in section 6 and is intended to satisfy the DODIIS requirements, but is more rigorous than is typically applied. We use information theory to quantify the capacity of the covert channels, and we use other formal methods to analyze both the system design and the security of critical components, and how they compose. The system is designed so critical operation is localized, to make this formal analysis tractable.

3 How the SINTRA Approach Fits JMCIS

The SINTRA approach to providing MLS database service is based on physical separation and data replication. A database exists for each hierarchical security level and contains all data at that level and below. Since users of each database are cleared for all data it contains, commercial, untrusted database products can be used. The databases reside on physically separate machines, so access to each database is controlled in the same way as access to its host machine.

The only information flow required by this approach is from low to high, an inherently secure flow with respect to confidentiality. In this way, SINTRA limits opportunities for malicious code in untrusted applications to exploit system vulnerabilities. The technical difficulty is to ensure the consistency of the replicas without introducing downward information flow. The SINTRA approach, solutions to technical problems, and prototype SINTRA implementations are documented in numerous papers and reports[5, 9, 3, 10].

A SINTRA database can be realized in centralized or distributed configurations. In the centralized case, users at different levels connect to the appropriate databases via a Trusted Front End, which also acts as a replica controller, propagating upwards changes to lower level databases. In a distributed SINTRA configuration, users at a given security level connect directly to their database, and changes are propagated upwards to other databases via a trusted backend.

The notion of security enclaves meshes particularly well with the distributed SINTRA approach: users in each enclave connect directly to their local database, and the databases in different enclaves are connected by replica controllers. Thus, viewed from the SINTRA perspective, two JMCIS systems, one LOW and the other HIGH, each with its own CDBS, is an instance of the distributed SINTRA architecture lacking only the trusted backend to provide the replica controller function.

4 A Non-secure, Commercial Approach

The commercial marketplace is also finding many applications for replicated databases, typically to ensure data availability, reliability, autonomy, and fault tolerance. Consequently, commercial replication servers are becoming available that can provide one part of a trusted replica controller; adding the high assurance guarantee of restricted downward communication is a technical challenge that is solved in our effort.

If confidentiality were not a concern, we could simply use Sybase's Replication Server (RS) to replicate LOW CDBS tables to the HIGH CDBS (figure 3). In Sybase's terminology, there is a single *primary* database and possibly many *replicates* of that database.

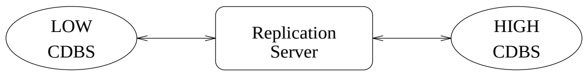

Figure 3: Non-secure replication architecture (abstract view).

RS continually monitors the primary database, the LOW CDBS, identifying the transactions that are to be replicated. Each such transaction is queued in RS's stable storage and forwarded to the replicate database, the HIGH CDBS. It is only removed from the queue when the replicate database acknowledges that the transaction committed.

This configuration requires the downward communication of acknowledgements and other error messages from RS to the primary database and from the replicate database to RS. In our application, however, there must not be significant downward communication from HIGH to LOW, so we must be able to divide this architecture into a LOW and a HIGH part and guarantee that no HIGH information passes directly, and very little indirectly, to the LOW system. We must also guarantee that updates are not lost during replication.

5 A Practical, Secure Architecture for JMCIS

Despite the difficulties that security requirements raise for the use of commercial replication servers, we have been able to identify a replication-based architecture for improving the integration of a pair of JMCIS CDBSs in LOW and HIGH enclaves. Figure 4 depicts the architecture simply but abstractly. The one-way store and forward device blocks any information except for flow control (conveyed by the timing of simple ACKs) from being transmitted from HIGH to LOW. In addition, the amount of information leakage due to flow control can be made arbitrarily small (see section 5.2).

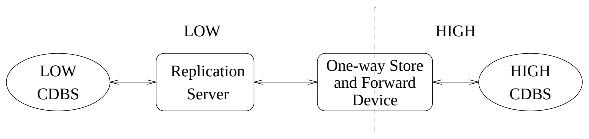

Figure 4: Secure replication architecture (abstract view).

RS and its stable buffers on the LOW side are separated from the HIGH side by a secure one-way store and forward device. This device accepts data from the LOW side and guarantees that the data will be buffered until the HIGH side processes them. More precisely, RS, running on the LOW LAN, forwards data to the secure store and forward device. That device's acknowledgment is a commitment on its part to forward that information to the HIGH LAN. Therefore, RS can interpret the acknowledgment as a *transaction committed* response from the HIGH CDBS.

Figure 4 leaves many details unspecified. The interfaces presented by the commercial products and the feasibility of implementing the one-way device could make this approach either easy or impossible. In particular, this approach requires a *wrapper* on each side of the one-way device, which

- makes it appear to the RS that it is communicating directly with an ordinary replicated CDBS, when it is in fact communicating with the one-way device, and
- makes it appear to the HIGH CDBS that it is communicating directly with a typical client which is providing a stream of SQL commands, when it, too, is in fact communicating with the one-way device.

These wrappers must be designed so replication is reliable and efficient and the system as a whole is still manageable. They must also interface effectively with the one-way device.

The balance of this section first addresses the wrappers. Since our primary realization of the store and forward device is the Pump[11, 13], we will describe these wrappers with reference to the Pump, but will assume nothing about the Pump beyond the already described behavior of the store and forward device (i.e., that device's acknowledgment of a message from LOW is a commitment to forward the information to HIGH). The Pump itself is described next. Finally, we describe an interim solution[8] that limits reliability, but also downward information flow, more than the Pump and may provide a simpler accreditation path.

5.1 Locating and Specifying the Wrappers

A more detailed look at the simplest replication architecture using Sybase products[24] (figure 5) will help us locate the wrappers and define their requirements.

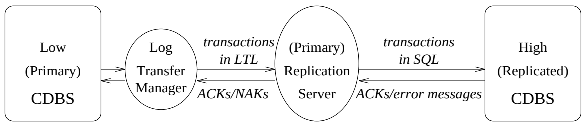

Figure 5: Non-secure replication architecture (detailed view).

Replication is accomplished by two major processes: the Log Transfer Manager (LTM) and RS. The responsibilities of the LTM include reading the transactions from the log of the primary database and sending them to RS using Log Transfer Language (LTL). The responsibilities of RS include storing the update transactions in stable storage for recovery and sending them to appropriate destinations (other RSs or replicate databases).

We could have put the Pump in either of two places: (1) between LTM and RS or (2) between RS and the High CDBS. We chose the second option (figure 6) because the protocol between RS and the High CDBS is an open protocol (SQL). Even though LTL is "officially" open, we could not find sufficient documentation.

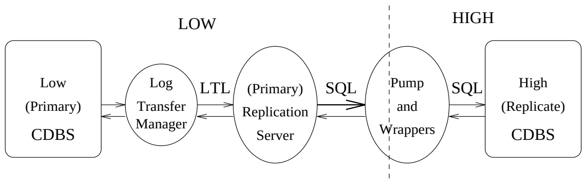

Figure 6: Secure replication architecture (detailed view).

When the Pump is inserted between RS and HIGH CDBS, it will block all acknowledgments and error messages from HIGH CDBS to RS. When RS sends a transaction to HIGH CDBS, it expects either an ACK or an error message. Also, when HIGH CDBS receives a transaction, it assumes the transaction is from its client and sends a proper response (e.g., ACK). To fulfill the application-specific expectations of RS and HIGH CDBS, the Pump is accompanied by two external processes: *LOW wrapper* and *HIGH wrapper* (figure 7).

The LOW and HIGH wrappers are application programs based on the Sybase Open Server and Open Client products. All Sybase server and client processes have uniform interfaces. The Open Server and Open Client products are sets of library functions that assist application programmers to write their own server and client programs that can communicate with other Sybase servers and clients.

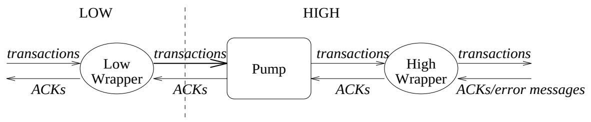

Figure 7: The Pump with wrappers.

The LOW wrapper is an application program based on the Sybase Open Server product. Its main functions are

- to receive transactions from RS, pass them to the Pump, and send an ACK back to RS after receiving an ACK from the Pump. No error messages are sent to RS since errors should not occur: The replicated portions in HIGH CDBS should have the same table structures and integrity constraints as the (primary) tables in LOW CDBS. Also, the replicated tables are read-only tables. Hence in most cases, the transaction that was successfully executed in the LOW CDBS should execute in the HIGH CDBS without error. If an error does occur in the HIGH (replicate) CDBS, it should be handled by the HIGH database administrator or the HIGH wrapper; sending error messages from HIGH to LOW would open up a covert channel. In any case, error codes do not usually convey enough information, by themselves, to allow LOW to fix HIGH's problems.

- to maintain the last replicated transaction ID and to return it to RS when requested.

As far as HIGH CDBS is concerned, RS is a client that submits SQL transactions. The HIGH wrapper is an application program based on both the Sybase Open Client and Open Server products; its main functions are

- to receive transactions from the Pump and submit them to HIGH CDBS. The HIGH wrapper sends an ACK to the Pump after the HIGH CDBS sends an ACK. If it receives an error message, it performs an appropriate action and then sends an ACK.[1]

- to perform appropriate actions in response to an error message. The HIGH wrapper performs

[1]Actually, RS sends transactions as sequences of packets. This requires additional bookkeeping by both the LOW and HIGH wrappers.

three classes of actions, *retry, ignore,* and *error-log*. The HIGH database administrator specifies error codes for each action class. For example, the error code 1205 from the Sybase SQL server is a deadlock error. Hence, if the HIGH database administrator specifies error code 1205 in *retry* class then the HIGH wrapper will retry the aborted transaction. The HIGH wrapper's default action to an error is *error-log* which will write the error message to a log file, notify the database administrator, and then stop.

These wrappers mimic the responses that the secure store and forward device masks. Implementations of this device are described next.

5.2 The Pump

The one-way store and forward device must satisfy two equally important classes of requirements: security requirements, such as highly restricted information flow from HIGH to LOW, and database replication requirements, such as reliability, recoverability, and performance. Even though communication without acknowledgments, such as blind write-up and similar read-down methods may satisfy the security requirements, they do not satisfy the database replication requirements[11, 13].

The (NRL) Pump is a device that balances these requirements[11, 13], which are fundamentally in conflict [18]. An abstract view of the Pump is as follows (figure 8):

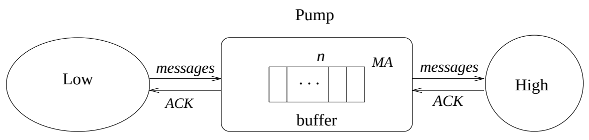

Figure 8: An Abstract view of the Pump.

The Pump places a non-volatile buffer (size n) between LOW and HIGH and sends ACKs to LOW at probabilistic times, based upon a moving average of the past m HIGH ACK times[11, 13]. A HIGH ACK time is the time from when the buffer sends a message to HIGH to the time when HIGH sends an ACK back. By passing ACKs to LOW at a rate related to HIGH's historical response rate, the Pump provides flow control and reliable delivery without unduly penalizing performance. We emphasize that ACKs are not passed through the Pump from HIGH to LOW. In fact, the Pump can acknowledge receipt of messages from LOW before HIGH receives them (otherwise a buffer would not be necessary). Each ACK sent to LOW is generated internally by the Pump only in response to a message from LOW. The average rate at which these ACKs are sent from the Pump to LOW reflects the average rate at which HIGH acknowledges messages from the Pump.

The rate of the ACKs from the Pump to LOW does represent a downward flow of information. However, the algorithm controlling the rate at which acknowledgments are returned is parameterized to allow the capacity of this timing channel to be made as small as accreditors may require. Prototype Pump implementations exist in our laboratory[17]. Recently, the concepts behind the Pump have been expanded to address the complications of fairness and denial of service in the network environment[12].

As with any other trusted device, we must provide evidence to accreditors that the Pump will not undermine a system's security. This means that the Pump must be analyzed on two levels: Does its specification satisfy security requirements? Does the implementation realize the specification? The Pump's specification has been modeled mathematically and permits us to quantify the theoretical leakage rate of an implementation.

There is often a world of difference between abstract formal models (e.g., [14]) and engineering solutions. Experience with an operational system will show us how robust our formal models are and, if necessary, suggest how to modify these formal models to deal with issues that are not apparent at the higher theoretical level[16, 18].

5.3 The Interim Solution

Because the accreditation process can be lengthy, we have designed an interim approach[8] to implementing a one-way device that is straightforward to accredit, but has unreliable communication because no ACKs are returned (figure 9). This means that HIGH cannot inform LOW that data was corrupted or lost during transmission. It is secure by inspection and has no trusted components.

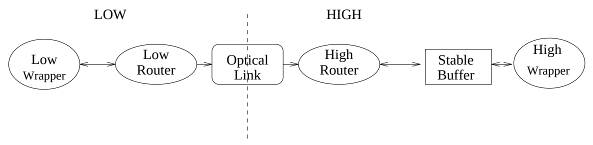

Figure 9: The interim solution.

The OPTICAL LINK is a commercially available, high speed, physically reliable one-way communication medium, which is already in use in many secure applications. The operation of this implementation is as follows:

(1) The LOW wrapper sends data to the LOW Router.

(2) The LOW Router packages data and sends the packets over the OPTICAL LINK.

(3) The HIGH Router receives the packets and forwards them to the STABLE BUFFER for storage until the HIGH wrapper is both ready to receive them and acknowledges their receipt to the STABLE BUFFER.

Both the LOW Router and the OPTICAL LINK must be fast enough to process all data on the LOW LAN. The HIGH Router and STABLE BUFFER must be fast enough never to lose data from the OPTICAL LINK and still be able to forward updates to HIGH.

How large must the STABLE BUFFER be? On average, the replicate database must be able to process all updates from the primary database, so it must be possible to identify a buffer that can handle the periodic bursts of updates from the primary. In fact, this assumption is reflected in RS itself, which includes a stable buffer whose size must be carefully chosen. The interim solution's STABLE BUFFER need not be any bigger than RS's stable buffer would be in a non-secure use of RS. Therefore, if the STABLE BUFFER fills, RS's stable buffer would have filled in a non-secure replication architecture. In fact, since the LOW Router and the OPTICAL LINK operate as fast as the LOW LAN, RS's stable buffer may be made much smaller, because RS will never need to wait on the LOW Router when forwarding an update.

Data could be corrupted or lost in transmission over the LANs or over the OPTICAL LINK. The STABLE BUFFER could fail or overflow, especially if HIGH fails. Sending each packet multiple times might reduce some of these losses, and any remaining losses could in any case be detected and flagged when uncorrupted data finally enters the STABLE BUFFER. Human intervention will be necessary to recover.

The lack of automatic recovery is a consequence of the complete isolation of HIGH data from LOW in this design. Because the physical design of the interim solution excludes all downward channels, the accreditation process is greatly simplified. The interim solution is very much like the Big Buffer[15], except it has no trusted components.

6 Developing an Assurance Argument

The system architect needs a clear and convincing argument to persuade the accreditor that the risk of compromise is small enough to justify operating the system. We call this argument the *assurance argument*. In order to have such an argument at the end of a project, one needs an *assurance strategy* during the development to integrate security engineering and system engineering. Initially, the assurance strategy records the set of assumptions and assertions[21] derived from the requirements. It is elaborated and refined throughout the development, yielding the assurance argument, delivered with the system.

Assertions are statements about the security that a particular INFOSEC discipline (computer security, communication security, administrative security, personnel security, physical security, and emanations security) is required to provide. *Assumptions* document requirements that one discipline places on others (for purposes of its own effectiveness). For example, the computer security assertions might include (for the Pump): Every ACK sent to LOW is delayed by the current value of the moving average. Similarly, there might be an assumption (again for the Pump) that the low input/output ports are correctly connected to the LOW LAN; this would be a requirement placed on physical and administrative security disciplines by the computer security discipline. Each assumption about some security discipline should match an assertion for another discipline; a gap in this mapping indicates a vulnerability.

To assure that we can answer accreditors' questions about the risks of our proposed modifications to JMCIS installations, we are developing an assurance strategy for a range of JMCIS installations. This generic assurance strategy defines a confidentiality policy and identifies an architecture that minimizes its vulnerability to existing threats. We argue the effectiveness of this architecture and rigorously define its critical requirements. The assurance strategy forms the core of the assurance argument that will need to be produced for each JMCIS site.

The assumptions and assertions approach has a formal basis in the theory of composition, but instead of composing components, we are composing specifications[1], and in this case, specifications about security properties[16]. Our approach is to use the logical language and tools for analysis that are appropriate to the level of design: the more critical a property or a component is, the more rigorous we can be in our analysis[22].

The assurance strategy includes three sections. The first models the confidentiality policy in terms of users and the data they can access, identifying the threats to mission assets and defining the critical requirements that must be satisfied to counter those threats. The second section decomposes those critical requirements onto the primary INFOSEC disciplines. It compares candidate architectures, assessing their potential to satisfy the confidentiality policy and other operational requirements, such as performance and availability. This permits identifying the risk that remains due to vulnerabilities of the architecture. The third and final section defines the implementation plan, which when carried out refines the assurance strategy into an assurance argument.

We have begun to define the overall functional requirements and operation formally in the languages of Statemate[6]. The Statemate tool, based on the formal theory of Statecharts[7], allows modeling system behavior and graphically executing this model to test its validity. The Statemate specification provides a formal structure for the assurance strategy and argument. Since Statemate has a formal semantics, we can reason formally about the description, both within Statemate's logic, and using other support tools and methodologies (e.g., mechanical proof checkers[2]). Security assumptions and assertions are defined in terms of the Statemate primitives and refined according to the Statemate decomposition. By modeling systems from both a logical and a physical perspective, Statemate permits specifying critical requirements independently of the physical architecture as a basis for comparing alternative physical partitionings for the security they afford. This is particularly important for this application because it relies on physical distribution for security assurance.

Our approach to arguing the effectiveness of this architecture promotes a slightly different structure for the certification documentation than that advocated by other certification approaches:

- It defines the security requirements and operations independently of the system architecture providing a basis for explicitly comparing alternative architectures (countermeasure partitionings). The architecture-independent statement of policy is defined in the Confidentiality Policy.

- It integrates security and system engineering to permit explicitly trading off security requirements with other critical system requirements. This reduces the redundancy (and documentation maintenance problems) that accompanies separate security and development documents.

- It traces the residual risk through the (possibly many) levels of system refinement. At any stage of refinement of the system and assurance argument it is possible to assess the risk of using the system.

- It is more rigorous. In addition to testing, we use formal methods as a static analysis technique to evaluate both designs and critical components.

We expect that this approach will simplify the accreditation of the system; we will report our experiences with it and with the accreditation process in subsequent papers.

7 Concluding Discussion

This paper reports work in progress, so we cannot draw strong conclusions yet about the effect of our work on actual JMCIS operations. Nevertheless, we believe that the approach reported here will benefit JMCIS substantially and that the same approach will prove useful to others facing the problem of improving the flow and coordination of information among secure enclaves.

Our design reflects the confluence of several efforts to improve information flow in DoD systems in a practical and cost-effective way without reducing security. C4I systems like JMCIS have developed enclave-based architectures to support both security and function, but these architectures restrict our ability to maintain timely and consistent data at the highest security levels. The SINTRA approach, though initially focused on providing a centralized MLS database capability, presents interesting possibilities for distributed databases. For example, the MLS data flow needed in JMCIS, which might have seemed an appropriate application for a guard device, turns out to be a natural application for a simpler one-way device (i.e., the Pump) combined with commercial database replication products. This isolation of critical function to small components also makes rigorous analysis, including formal and mathematical modeling, tractable and significantly lowers the extra cost of high assurance.

Now let us withdraw a little from the details of this effort to consider briefly the larger picture of tactical information processing in the Navy and DoD and how the ideas that have proven effective here might be applied more widely.

The fundamental concepts behind our design are

- to provide confidentiality through physical separation,

- to provide data flow from less-protected to more-protected environments through simple and high assurance one-way devices,

- to apply database replication concepts to coordinate copies of data stored at different security levels,

- to organize the system to avoid downgrading (flows from more-protected to less-protected environments), and

- to use commercially available technology wherever possible.

We think that the application of these concepts in a broader context of DoD tactical information processing could be of great benefit. They imply doing as much processing as possible at the lowest legitimate level of classification, so data can be made available by replication to the largest set of customers. This will reduce the need for downgrading, which is the most expensive path between enclaves. Also, these concepts can be applied to heterogeneous databases or file-based storage just as well as to the homogeneous database environment we found in JMCIS.

But gaining the maximum benefit from these concepts depends on having well-structured data. Currently, many military databases simply store military messages as text, wasting much of a database management system's power. Military messages provide a rich source of information for populating databases; they are carefully formatted, so much parsing can be done automatically. Furthermore, messages could be broken apart to separate *sources* from *content,* permitting the content to be analyzed at a lower security level.

We look forward to efforts to improve the structure of DoD databases, separating the functions of data processing from data management to improve information flow and simplify application development. Within this context, data replication will enable low-cost, yet high-assurance MLS data management.

8 Acknowledgments

This project required significant effort by many people. As project leaders, Eather Chapman and Maria Voreh of NRL provided technical coordination between the NRaD JMCIS development team, Sybase, and the NRL project team, and developed the demonstration scenario. Kathie Jesionowski and Michelle Pagan of NRL are analyzing tradeoffs in system engineering. George Kamis of NRL and George Thomas of Sierra Cybernetics provided important background information on JMCIS and other Navy MLS systems. John McDermott of NRL developed and analyzed the big buffer approach to blind write-up and is always a good source of general knowledge about Navy operations. Bruce Montrose of NRL designed and implemented the event driven Pump on the XTS-300. Rodney Peyton of Edge Computer Systems, Inc. designed and implemented the wrappers between the Pump and the Sybase database products. Mary Rock of Kaman Sciences focuses on system engineering with a special concern about accreditation issues. Finally, the insight, encouragement, and cooperation of the JMCIS CDBS developers at NRaD, including Nate Schatz, Fran Wright, Tom Miller, Diana Snow, Rich Snow, Jack Sommer, and Craig Tutz, and of Stan Davis of Delfin Systems is greatly appreciated.

References

[1] M. Abadi and L. Lamport. "Composing Specifications." ACM Transactions on Programming Languages and Systems, 15(1):73-132, January, 1993.

[2] R. S. Boyer and J S. Moore. "A Computational Logic Handbook." Academic Press, Boston, 1988.

[3] O. Costich, and M. Kang, "Maintaining Multilevel Transaction Atomicity in MLS Database Systems with Replicated Architecture," in *Database Security VII: Status and Prospects*, eds. T. F. Keefe and C. Landwehr, IFIP Transactions A-47, Elsevier Science B.V., Amsterdam, ISBN: 0 444 81833 2, pp.329-356.

[4] Defense Intelligence Agency (DIA). *DODIIS Developer's Guide for Automated Information Systems Security in DOD Intelligence Information Systems*, 1993.

[5] J. N. Froscher, M. Kang, J. McDermott, O. Costich, and C. E. Landwehr. "A Practical Approach to High Assurance Multilevel Secure Computing Service," Proc. 10th Computer Security Applications Conference, Orlando, Florida, Dec. 1994.

[6] D. Harel, H. Lachover, A. Naamad, A. Pnueli, M. Politi, R. Sherman, A. Shtull-Trauring, and M. Trakhtenbrot. Statemate: A Working Environment for the Development of Complex Reactive System, *IEEE Transactions on Software Engineering*, 16(4):403-414, Apr 1990.

[7] D. Harel, J. Schmidt, R. Sherman. On the Formal Semantics of Statecharts, In *Proceedings of the 2nd IEEE Symposium on Logic in Computer Science*, pages 54-64, Ithaca, NY, 1987.

[8] D. M. Goldschlag. "Several Secure Store and Forward Devices," To appear in Proceedings of the 3rd ACM Conference on Computer & Communications Security, New Delhi, March 1996.

[9] M. Kang, O. Costich, and J.N. Froscher. "A Practical Transaction Model and Untrusted Transaction Manager for a Multilevel-Secure Database system" in *Database Security VI: Status and Prospects*, eds. B. Thuraisingham and C. Landwehr, IFIP Trans. A-21, ISBN 0 444 89889 1, Elsevier, Amsterdam, 1993, pp. 285-300.

[10] M. H. Kang, J. N. Froscher, J. P. McDermott, O. Costich, and R. Peyton. "Achieving Database Security through Data Replication: The SINTRA Prototype," *Proc. 17th National Computer Security Conference*, Baltimore, MD, Oct., 1994, pp. 77-87.

[11] M. H. Kang and I. S. Moskowitz. "A Pump for rapid, reliable, secure communication," Proceedings ACM Conf. Computer & Commun. Security '93, pp. 119 - 129, Fairfax, VA, 1993.

[12] M. H. Kang, I. S. Moskowitz and D. Lee."A network version of the Pump," Proc. of the IEEE Symposium on Research in Security and Privacy, pp. 144 - 154, Oakland, CA, May 1995.

[13] M. H. Kang and I. S. Moskowitz. "A data Pump for communication," NRL Memo Report 5540-95-7771.

[14] Encyclopedia of Software Engineering, ed. John Marciniak, chapter on "Security Models" by John McLean, Wiley & Sons, 1994.

[15] J. McDermott, "The b^2/c^3 problem: how big buffers overcome covert channel cynicism in trusted database systems," in *Database Security*

VIII, Status and Prospects, J.Biskup, M. Morgenstern, C. Landwehr, eds., IFIP Transactions A-60, Elsevier Science B.V., Amsterdam, ISBN 0 444 81972 2, pp.111-122.

[16] J. D. McLean. "A general theory of composition for trace sets closed under selective interleaving functions." Proceedings 1994 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 79 - 93, Oakland, CA, 1994.

[17] B. E. Montrose and M. H. Kang. "An Implementation of the Pump: Event Driven Pump," NRL Memo. Report 5540-95-7782, 1995.

[18] I. S. Moskowitz and M. H. Kang. "Covert channels — Here to stay?," Proceedings COMPASS '94, pp. 235 - 243, Gaithersburg, MD, 1994.

[19] I. S. Moskowitz and M. H. Kang. "The modulated input modulated output model," To appear: *Database Security 9, Status and Prospects*, IFIP Transactions, Elsevier Science B.V., Amsterdam. In the interim is available in the preliminary conference proceedings of the 9th annual IFIP WG 11.3 Working Conference on Database Security, August 1995.

[20] National Computer Security Center. *DOD-5200.28-STD, Department of Defense Trusted Computer System Evaluation Criteria*, (Orange Book), Dec. 1985.

[21] C. N. Payne, J. N. Froscher, and C. E. Landwehr. Toward a comprehensive INFOSEC certification methodology. In *Proceedings of the 16th National Computer Security Conference*, pages 165–172, Baltimore, MD, September 1993. NIST/NSA.

[22] C. Payne, A. Moore, D. Mihelcic. "An Experience Modeling Critical Requirements." In Proceedings of the 9th Annual Conference on Computer Assurance, pages 245-255, Gaithersburg, MD, 1994.

[23] N. Shankar, S. Owre, and J. M. Rushby. "The PVS Proof Checker: A Reference Manual." SRI International, 1993.

[24] Sybase Inc. "Replication server administration," Sybase Inc. 1994.