Click here to view PowerPoint presentation; Press Esc to exit

Achieving Cross-Domain Collaboration in Heterogeneous Environments

Mr. Thomas Macklin and Ms. Phyllis Jenket
Naval Research Laboratory
4555 Overlook Ave, S.W.
Washington, D.C. 20375
USA

E-mail: Thomas.Macklin@navy.mil phyllis.jenket@navy.mil

ABSTRACT

The military community relies on chat and Instant Message (IM) technologies for planning operations and near real-time collaboration. For all of the strengths of chat/IM technologies, they do not lend themselves well to use within conventional cross-domain communications architectures. The United States Naval Research Laboratory (NRL) has developed a portable, hybrid architecture that introduces multilevel security (MLS) technologies into environments comprised of multiple security levels (MSL). This architecture was then used as the framework for integrating various software systems into an enabling capability for cross-domain chat. The resultant multilevel chat system utilizes various trusted mechanisms to maintain strong process separation, privilege management, and communications interface control. This multilevel chat system was then used in a limited operational experiment (LOE) to enable users in disparate security domains to collaborate with each other based on a pre-defined, tested, and approved system security policy. Efforts are currently underway to develop a certification profile for this system, as well as for the system's hybrid multilevel architecture. We hope to determine the scalability of this architecture through future operational test scenarios. We are also investigating the scope of the solution set to which this architecture may apply, including multilevel web services.

1.0 INTRODUCTION

As NATO and its member nations move forward in transforming from platform-centric to network-centric force structures, a shift in the nature of the technological problems has occurred. Increasingly, military and civilian organizations implementing information systems are discovering a greater need for secure, reliable interconnections between existing systems than for systems that provide new capabilities [1]. Information systems that were once partially or completely autonomous now must support interconnections with all sorts of other networked systems; many of which are similarly ill suited for supporting interconnections. As organizations struggle with accommodating net-centricity, they inevitably find themselves in a dilemma: while they find that in order to fulfill their mission they must support interconnections to other information systems with differing security policies and of varying levels of trust, they also find that many times some subset of the systems in question lack sufficient robustness in their security mechanisms to safely implement the required interconnections.

Paper presented at the RTO IST Symposium on "Coalition C4ISR Architectures and Information Exchange Capabilities", held in The Hague, The Netherlands, 27-28 September 2004, and published in RTO-MP-IST-042.

In most of these cases, organizations turn to security devices to mediate interconnections between systems that exist in different security domains. Systems such as network encryption devices, high assurance guards, or separated multilevel security (MLS) enclaves are often used to enable secure, cross-domain communications in a manner that meets at least some operational requirements. While these systems often succeed in accommodating limited cross-domain communications, the levels of functionality, reliability, and performance are far below the levels usually experienced in traditional, single level systems. Furthermore, as most of these security systems are third-party developed, they often do not evolve with the systems they are designed to support, so as the commercial-of-the-shelf (COTS) and government-off-the-shelf (GOTS) systems they connect change, the Guards are unable to continue to work within the new system architectures [2][3]. In order to realize net-centricity and operate at the level required to support today's high-paced battle rhythms, information systems must be able to support interconnections between disparate security domains, and they must be able to operate at assurance levels comparable to those expected of military command and control systems.

This paper explains how the Multilevel Chat system was developed to meet the US Navy's requirements for near-real-time, cross-domain collaboration, and how the multilevel chat system's architecture may be used to solve other cross-domain communications problems. The paper is made up of five different sections. The first section is this introduction. The second section briefly describes the US Navy's cross-domain chat requirements, and then maps the extents to which traditional cross-domain architectures would satisfy the requirements. The third section describes how we analysed the properties of these cross-domain architectures against the requirements for a cross-domain chat system, and in turn developed the design for the Multilevel Chat system. The third section also describes the Multilevel Chat system implementation, and how the system supports cross-domain collaboration in heterogeneous environments. The fourth section describes how the Multilevel Chat system has been tested, the results of those tests, and how the results compare against testing for other cross-domain solutions. The fifth section describes ongoing efforts that have come out of the Multilevel Chat project.

2.0 TRADITIONAL CROSS-DOMAIN COMMUNICATION MODELS

During the initial planning phases of the cross-domain chat development effort, we evaluated several possible solution architectures. During this evaluation, we found that all of the traditional cross-domain architectures had certain inherent properties that would preclude them from adequately addressing the US Navy's requirements for cross-domain chat. It is worthwhile then to explain these findings, and how they were in turn used to deduce a hybrid, multilevel architecture that could satisfy the US Navy's cross-domain chat requirements.

Traditionally, there have been three different models for accommodating multiple security domains within a single operating environment: Multilevel Security (MLS), Multiple Security Levels (MSL) with a MLS local area network (LAN), and guarding solution-based interconnections. First, a brief explanation of the requirements that these models are meant to satisfy will be provided. Some of the requirements stated are general cross-domain system requirements, while others are requirements that are more specific to chat/IM services. Then, each of these three cross-domain communication models will be evaluated against these requirement statements. Finally, any discrepancies between the stated requirements and the systems' capability to satisfy those requirements will be briefly explained.

2.1 Requirements for Cross-Domain Communications Systems

Requirements for cross-domain communication systems may be separated into four different categories. Any requirement to protect data objects processed by the system will be classified as data protection requirements. Any requirement to protect computing assets, such as servers, clients, or networks, will be classified as resource protection requirements. Any requirement that relates to the system's ability to process data in a manner that complies with the system's functional requirements will be classified as functional performance requirements. For the purposes of this paper, functional performance requirements will be stated as they apply to chat and/or IM-based collaboration. Finally, any requirements that are outside the technical scope of the system, but are nonetheless levied on the system, will be classified as non-technical requirements.

2.1.1 Data Protection Requirements

a. *Enforce non-discretionary data protection policies*- Most organizations have basic rules that dictate how data is to be shared between subjects within their information systems. Any system that connects an organization's sensitive information systems to one or more external information systems must have provisions for ensuring that the organization's data protection policies are enforced at all times.

b. *Provide discretionary access control mechanisms*- In addition to organizational data protection policies, often times data owners within an organization will have additional rules governing data access rights. As such, any system that connects an organization's sensitive information systems to one or more external information systems must have provisions for ensuring that data owners may control access to data objects based on the identities of user subjects within the system.

c. *Transfer policy enforcement*- Most systems can be used for purposes other than those for which they are meant to be used. Any system that connects an organization's sensitive information systems to one or more external information systems must have provisions for ensuring that the data processing services provided by the system may only be used for their intended purposes. For example, an email transfer system that is meant to provide plain text only email connectivity between security domains should prevent users from embedding Uuencoded file attachments within email messages.

2.1.2 Resource Protection Requirements

a. *System communications protection*- Communications between subjects within the information system subjects must be protected from unauthorized modifications, surveillance, and injections.

b. *System self-protection*- Data processing components within the information systems, along with the whole information system itself, must have security mechanisms that have sufficient strength of function to ensure that it is adequately resistant to the categories of attack [4] to which it is likely to be subjected.

c. *Detection of Malicious or Dangerous Content*- Any system that connects an organization's sensitive information systems to one or more external information systems must have provisions for detecting potentially malicious data, and must protect connected systems from these sorts of content in the manner defined by the connected systems' respective information assurance policies.

d. *Strong network separation*- Any system that connects an organization's sensitive information systems to one or more external information systems must ensure that the connected networks are completely separated, except for the approved data transfer mechanisms defined in the system's security policy. Furthermore, any data transfer mechanisms that span enclaves must be completely mediated by trusted security mechanisms.

2.1.3 Functional Chat/IM Performance Requirements

a. *Service functionality*- The system must enable a basic group chat and/or IM function to be extended to users residing in different security domains.

b. *Timeliness of service*- As chat/IM functions are near-real-time in nature, data transfers must be perceived by users as close to immediate; i.e., at most a few seconds.

c. *System usability*- Regardless of the system's security posture, the system must provide a user experience that is sufficiently comfortable and familiar as to gain user acceptance [5].

d. *System scalability*- The system must provision for scalability with respect to traffic load, the number of connected enclaves, and the types of connected networks.

2.1.4 Non-Technical Requirements

a. *Economically Feasible*- Any cross-domain solution must be deployable in a manner that is comparably cost-effective to other potential solutions. Cost considerations include development, procurement, installation and configuration, training, accreditation, and lifecycle support.

b. *Accreditable*- Most organizations, especially in military and government arenas, have strict accreditation requirements for systems that connect to their sensitive networks. As such, any potential system must provide sufficient assurance (and evidence thereof) to have a reasonable chance at obtaining a system security certification/verification, but it must also provide evidence that there is a reasonable probably that it can obtain accreditation for use *in a specific target environment* [6].

c. *Manageable*- The system must have adequate provisions for system administrators to manage the system. Specifically, system management mechanisms should minimize, if not eliminate, any possibility of an administrator misconfiguring an accredited system such that it no longer meets the system's approved security posture.

d. *Interoperable*- Ideally, the system should be able to interoperate with existing information systems.

Figure 2-1 summarizes these requirements into 8 high-level requirement statements. These requirement statements are used throughout this section as a reference for describing the fitness of traditional cross-domain architectures with respect to facilitating cross-domain chat.

Requirement Statement	Source
FACILITATE NEAR-REAL-TIME COLLABORATION	2.1.3.a 2.1.3.b 2.1.3.c 2.1.3.d
SUPPORT DYNAMIC COALITIONS	2.1.3.d 2.1.4.d
ENFORCE DATA TRANSFER POLICY	2.1.1.a 2.1.1.c
PROVIDE STRONG ENCLAVE SEPARATION	2.1.1.b 2.1.2.d
LOW RISK FOR ACCREDITATION	2.1.3.b
INTEROPERABLE WITH CURRENT INFRASTRUCTURE	2.1.4.c 2.1.4.d
USE PROVEN SYSTEMS	2.1.4.a
PROTECT DATA SECURITY	2.1.2.a 2.1.2.b 2.1.2.c

FIGURE 2-1: CHART FOR REQUIREMENTS DEFINITIONS

2.2 Multilevel Secure (MLS) Systems

2.2.1 Meeting Cross-domain Chat Requirements with MLS

Many properties of MLS systems lend themselves well to providing chat services to users operating in disparate security domains. The chart in figure 2-2 depicts the extent to which we determined that an MLS system would address the requirements described in section 2.1.

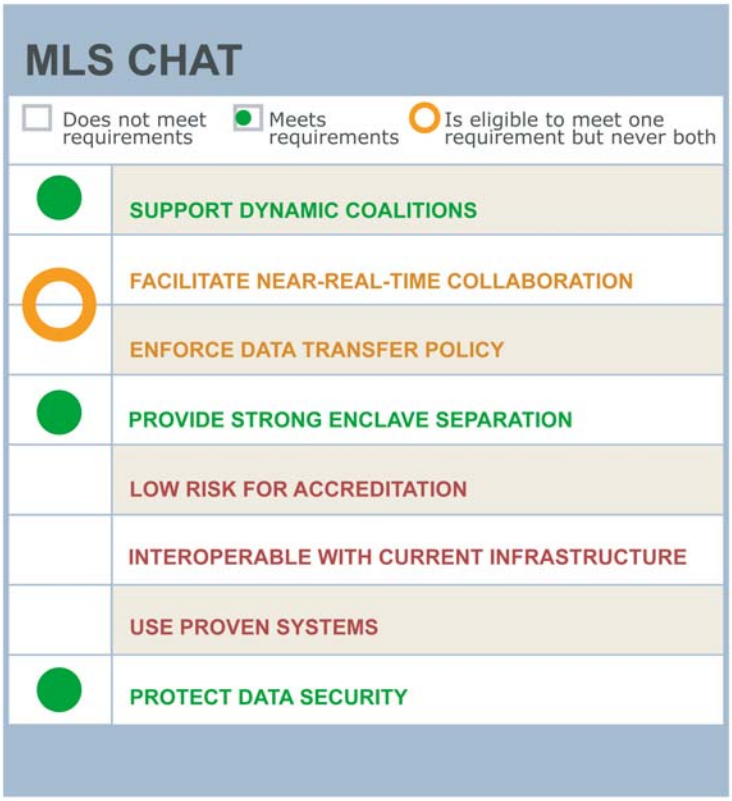

FIGURE 2-2: CROSS-DOMAIN CHAT REQUIREMENTS MAPPED TO A GENERIC MLS SOLUTION

2.2.2 Disadvantages of Using MLS to Enable Cross-Domain Chat

Most of the problems associated with using MLS information systems stem from the fact that most data currently exists in systems that do not support trusted labels. For example, if an information system that connects to an MLS system does support trusted labelling, the MLS system will have to apply a single label to all data that comes from that information system. Similarly, any connected information system that does not support trusted labels may only operate at a single level within the MLS environment, which would be the level that the MLS system associates with the connection. Furthermore, most client components do not support trusted authentication paths across security domains, and as such would require a trusted path extension in order to securely interface into an MLS system. To make matters worse, many trusted path extension systems require hardware modifications to client systems [1][7][8]. Such modifications would impose significant initial fielding and management costs that would grow at least linearly as the use of the information system expanded. Finally, most COTS systems do not support MLS schemes. Thus within an MLS domain, all applications would have to be modified to support MLS functionality in order to function [9]. As a result, a suite of applications would have to be modified to support a particular MLS scheme before that scheme will be able to support any operational requirements, and getting military or commercial organizations to invest in such modifications without any promise that the MLS scheme will be widely accepted would be a difficult endeavour.

While MLS could be used to enable cross-domain chat/IM services, several difficult problems would be left either partially or completely unaddressed. Most significantly, any MLS-based chat/IM solution would require

a new client infrastructure to support trusted path authentication. Furthermore, the MLS environment would need an MLS-enabled chat/IM system that would support the underlying MLS labelling scheme. Even if these challenges are overcome, there remains another, more serious problem: MLS does not enable communication between subjects with differing, non-hierarchical security and integrity policies at all. That is to say, there is no way for a user operating at a security level to chat with a user if that user has a lower security label and a lower integrity label without violating either Bell Lapadula or Biba system policies. To illustrate this problem, assume a user wishes to send a chat message from a high security, high integrity enclave to a user in a low security, low integrity domain. To do so, the user must violate Bell Lapadula and write data down [10]. Similarly, a user would have to violate Biba to send a message from a low integrity domain to a high integrity domain, regardless of the domains' sensitivity levels [11]. As most inter-domain communication systems that have non-hierarchical data security policies also have non-hierarchical data integrity policies, this problem greatly limits the viability of MLS systems in satisfying cross-domain collaboration requirements, even in cost-insensitive situations [12].

2.3 Ultra Thin Client-based MSL Systems

2.3.1 Meeting Cross-Domain Chat Requirements With Ultra Thin Client-based MSL

Current world events have demonstrated that today's cross-domain communication requirements cannot be satisfied solely by maintaining multiple, separate information systems for processing each distinct security level. As new bilateral and multilateral relationships continuously form, evolve, and dissolve, organizations simply cannot roll out new infrastructure in a manner that keeps pace with demand. Accordingly, the information technology community has experienced widespread growth in the understanding that traditionally separate information systems must be interconnected somehow, even if the systems have widely differing security policies. One popular solution to this problem is to use an MLS-enabled front-end local area network (LAN) to interface users into a distributed, heterogeneous MSL-operating environment.

Many different architectures fit this description of MLS-fronted MSL [13]. Perhaps the most widely accepted architecture of this type today uses a trusted, multilevel session server to service ultra thin clients (UTCs) on one interface, while connecting into multiple, single-level security domains with the system's subsequent interfaces. Examples of this system include Trusted Computer Solution's Secure Office suite and the US Navy's Multilevel Thin Client system.

The chart in figure 2-3 depicts the extent to which we determined that a UTC-based MSL system would address the requirements described in section 2.1.

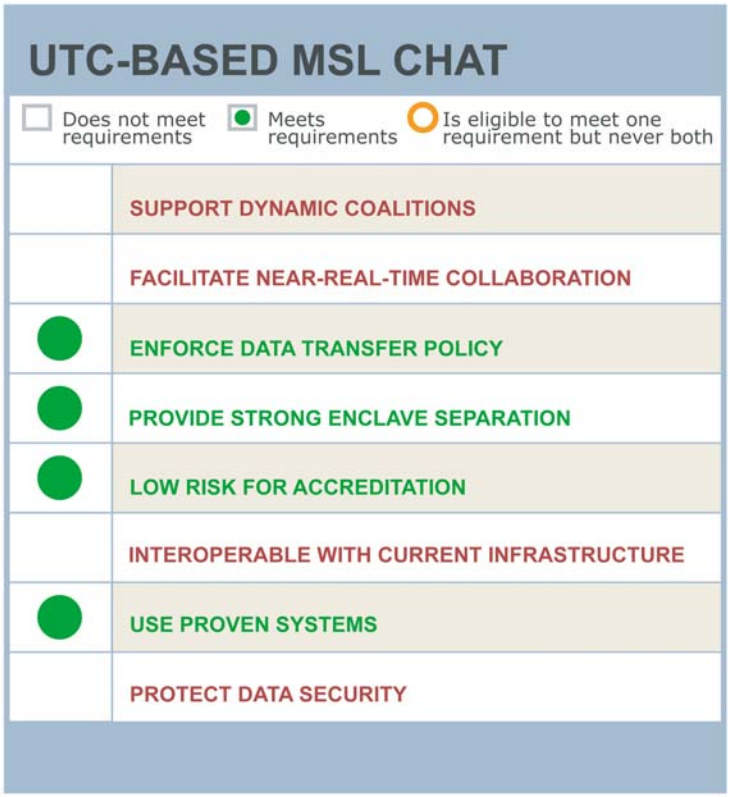

FIGURE 2-3: CROSS-DOMAIN CHAT REQUIREMENTS MAPPED TO A GENERIC UTC-BASED MSL SOLUTION

2.3.2 Disadvantages of Using UTC-based MSL to Enable Cross-Domain Chat

Not all applications would work properly in a UTC-based MSL environment, as many applications do not work properly with server-based computing. Chat applications are generally not computationally demanding though, and as such chat applications can be expected to function properly when used in conjunction with UTCs. However, UTC-based collaboration systems would not scale well beyond two or three different security domains, as it is essentially an MSL system in which it is difficult to securely pass data in between domains. Thus, any collaboration that is to occur within the operating environment must occur within a single domain, which in turn requires that the number of individual domains increases linearly with the number of inter-domain relationships defined in the mandatory access policy. Similarly, users may be able to simultaneously chat with other users in disparate security domains, but users from more than two domains will never be able to engage in a single group chat session without the creation of a new enclave. As a result, an increase in the number of enclaves interconnected in a UTC-based MSL system increases degradation in a user's ability to effectively and simultaneously chat with users in other connected domain.

2.4 Guard-Based MSL Systems

2.4.1 Meeting Cross-domain Chat Requirements with Guard Systems

Guarding technologies have been used for several decades to mediate cross-domain communications. The terms "guard" and "high assurance guard" are often used to refer to many different types of systems. In this

paper, these terms refer to any system that is used to mediate one or more data flows between otherwise completely separated information systems. Guards provide strong network separation for connected systems, and if well implemented and used in the appropriate context, can provide a high level of assurance that a security policy is being enforced for a given data flow. Most importantly, guarding mechanisms are the most widely accepted way to securely move data between security domains. The chart in figure 2-4 depicts the extent to which we determined that a guard-based solution would address the requirements described in section 2.1.

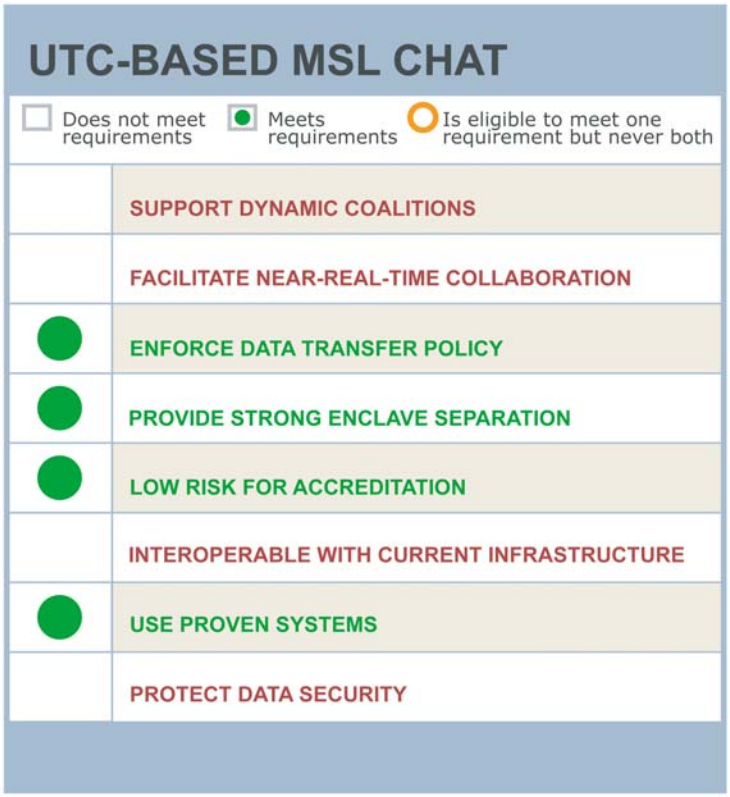

FIGURE 2-3: CROSS-DOMAIN CHAT REQUIREMENTS MAPPED TO A GENERIC GUARD-BASED SOLUTION

2.4.2 Disadvantages of Using Guards to Enable Cross-Domain Chat

Guards are usually third party platforms that mediate communications between systems developed by different commercial or government organizations [14][15]. Accordingly, one problem with guards is that it is difficult to guard for a communications path in a manner that is both high assurance and functionally robust, as they must impose strict content type enforcement mechanisms against data objects with fluid formats. As chat/IM systems lack a commonly used standard protocol, this problem would make the development of a viable Chat/IM guard difficult.

Guards are designed to give externally connected systems minimal information regarding content rejections. When using guards, users often experience what seems to be sporadic functionality when in fact the guard is

operating properly and is rejecting their traffic. As chat/IM systems require immediate feedback, handling rejections would be problematic with a guarding system.

The most significant problem with using guard systems to enable cross-domain chat in a coalition environment has to do with the very purpose of a Guard: to maintain a hierarchical security policy. Accreditation bodies that require and approve the usage of guarding solutions usually view separate security domains as "higher" or "lower" than each other. For example, if domain A is a more sensitive domain than domains B and C, and C is also more sensitive then D, then A is the "high" side, B, C, and D are the "low" sides, and C is also above D. This architecture may satisfy the requirements of the "A" domain, but it does not readily accommodate communications between the non-hierarchical flow paths of B-C and B-D. Thus, while guarded architectures may be sufficient for intra-organizational cross-domain models, they do not address scenarios such as NATO's, where two or more sovereign organizations must interconnect their information systems. In many such cases, all involved organizations would consider their connection as the high side enclave, and all other connections as low side enclaves. The solution is usually for the organizations to implement fronting guards, where each organization implements their own guarding policy. Figure 2-4 illustrates the difficulties that arise in fronting guard architectures.

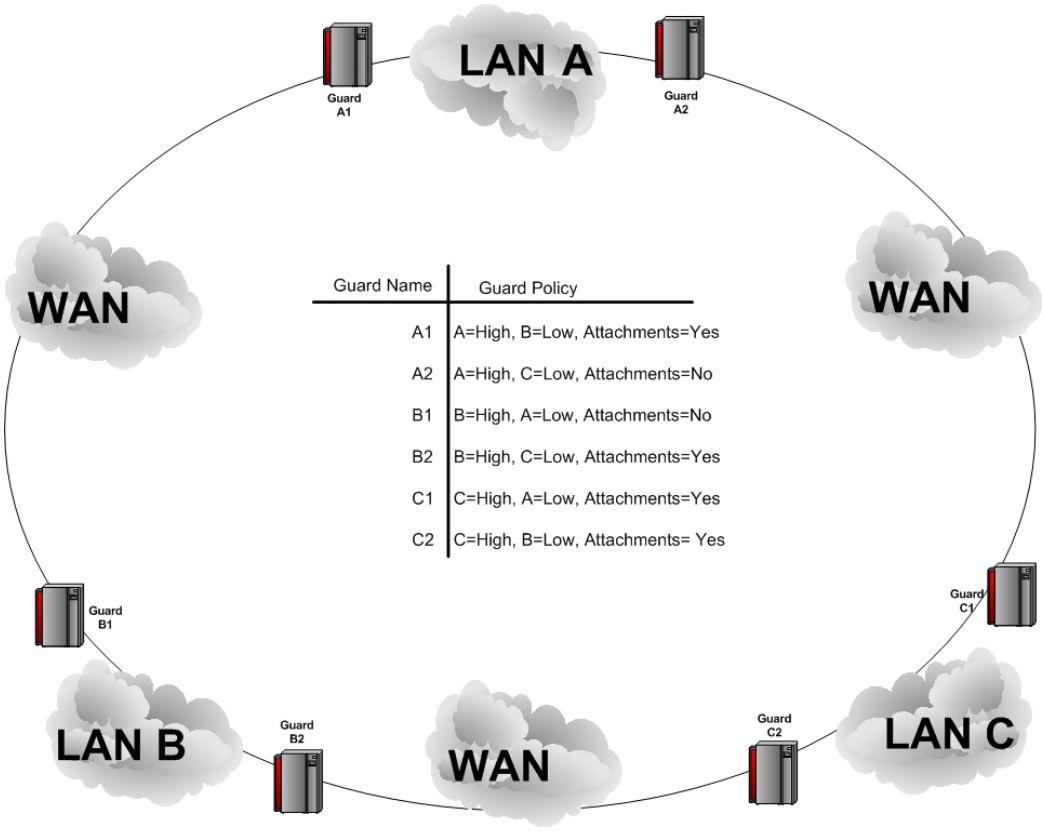

FIGURE 2-4: A FRONTING GUARD SCENARIO

3.0 THE MULTILEVEL CHAT SYSTEM

3.1 The Multilevel Chat System's Hybrid Architecture

Analysis of the properties of the traditional cross-domain architectures indicated that no single one of them was adequate for satisfying the US Navy's requirements for a cross-domain chat system. However, evidence also indicated that it would be possible to develop a satisfactory solution with a hybrid architecture that exhibited some characteristics of each model. Hybrid multilevel security architectures are by no means new [16]. However, many within the information security community remain skeptical about the viability of hybrid cross-domain security systems in target environments that have medium or high assurance data integrity requirements [17]. Obviously then, any viable hybrid security architecture must provide assurances that it accommodates adequate data integrity protections. This section describes a composite MLS, MSL, and Guarding hybrid security architecture that that can be used to securely enable cross-domain chat.

3.1.1 Domain Separation

Trusted MLS operating systems provide several mechanisms for enforcing non-hierarchical mandatory access control (MAC) policies. Multilevel chat would require a candidate trusted OS platform to provide mechanisms for strong separation between processes running in different security domains. In order to maintain strong domain separation, multilevel chat would use a security and integrity compartmentalization scheme that had the separate security domains in mutually dominant (e.g., "high" side) spaces. In this scheme, no security domain is hierarchically above any other domain, and as such, all data flows between security domains would require mediation by a trusted process. Thus, if there were no mediating trusted process, then the level of assurance of domain/network separation in this architecture can be considered at the level of the trusted operating system, but users would not be able to share data across domains. Otherwise, the assurance level of domain/network separation should be considered at the level to which the trusted process maintains domain separation while processing chat data flows between domains.

3.1.2 Chat Data Protection

The architecture described in section 3.1.1 constitutes MSL within an MLS system. In order to provide users access to cross-domain chat resources, the system provides each security domain with controlled access to chat resources by protecting chat data with a hierarchical MAC compartmentalization scheme [18]. In this scheme, chat resources are stored in integrity compartments dominated by only enclaves that are authorized to access those resources. For example for enclaves A, B, and C in integrity compartments 1, 2, and 3 respectively, a chat room in compartments 1 and 2 would be accessible by A and B, but not C, while a chat room in compartments 1 and 3 would be accessible by A and C, but not B. Since within this architecture all chat data can be protected according to the system's security and integrity policies, the chat data security assurance level can be considered at the level of the trusted operating system.

3.1.3 Chat Data Flows

The architecture described in section 3.1.2 is a simple MLS scheme where mutually dominant security domains may be able to read any chat data in integrity compartments below that of their domain. However, in order to allow users within a security domain to write chat messages in this scheme, a trusted mediating process is required to upgrade the integrity of chat messages. This trusted process amounts to a guarding mechanism to service all security domains within the system. Notionally, when users within an enclave submit chat messages to the chat server interface in their domain, the server in turn submits the messages to a trusted chat monitor for analysis, reclassification, and posting. Thus, the assurance of data integrity within

this architecture for a given data flow can then be considered at the assurance level of the system's trusted chat monitor.

3.1.4 Identity Management

One of the challenges faced by both MLS and UTC-based MSL was ensuring that users could not gain unauthorized access to data by masquerading as another user with a different clearance. In order to alleviate the risks associated with these schemes, the target architecture uses a single, high integrity database of user clearance registrations, which is then used by the chat application servers in each enclave to authorize users. This scheme allows for a centralized, high assurance user registration process, but prevents users from using their credentials outside their approved security domain. For example, if a user is registered in the system with a clearance to access security domain A, then their credential is only valid in enclave A. Thus, they may potentially violate discretionary access control policies by masquerading as another user within their domain, but they will never be able to violate the system's mandatory access control policy without both gaining access to the credential of a user in another domain and gaining access to a client system within that security domain. As most organizations use cryptographic systems to encapsulate their secure enclaves, this MLS-MSL hybrid identity management system binds the assurance level that the authentication mechanism cannot be used to violate the MAC policy to the assurance level of the environment's network separation mechanism. Furthermore, this identity management scheme ensures that the authentication mechanism cannot be used to violate the system's MAC policy, regardless of the strength of the system's authentication mechanism itself. Finally, this scheme also allows for a non-enclave specific object, the user clearance registration, to be used to enforce discretionary access controls across the multiple mandatory access levels at which the chat rooms exist.

3.2 The Multilevel Chat System Prototype

Using the hybrid cross-domain architecture described in section 3.1, the NRL cross-domain chat development team was able to implement, field, and test a multilevel chat system prototype in about 12 months. This prototype system, commonly referred to as "ML Chat," provides basic chat services in accordance to the requirements laid out in section 2.1. This section briefly describes the basic elements of the ML Chat system.

3.2.1 ML Chat's Operating Environment

The ML Chat system is currently developed to run on DigitalNet's XTS-400 platform, which currently runs the STOP 6.1 operating system. Although ML Chat's architecture does not preclude the possibility of porting to another trusted operating system, STOP provides several useful security features that are currently unavailable on any other platforms. [INSERT GOOD COMMENT???] Currently, STOP 6.1 is limited to accepting a maximum of 256 simultaneous network connections, and only supports a maximum of 768 MB physical memory. All trusted ML Chat modules are built using the STOP OS trusted software developer's kit (SDK).

3.2.2 Chat Application Services

ML Chat maintains a separate chat server instance in each operational security domain, and each chat server instance services user connections for their respective enclaves. The server application itself is a version of a single-level COTS collaboration server that has been modified to work within the ML Chat framework. Users connect to the server through a slightly modified version of the chat application's web-based client suite, and thus do not require any special knowledge of the inner workings of multilevel systems. These client-server connections can be either plain text or SSL-protected, and allow for the optional use of client-side certificates.

3.2.3 ML Chat Application Service Multilevel Extensions

The chat application itself is unaware of STOP's data labelling system. Rather, each chat server processes a high integrity MAC policy definition file created by the system administrator. The purpose of this file is to provide an abstract representation of the trusted operating system's implemented MAC policy that the application can read but not modify. Each instance of the application server then uses the policy defined in this file to enhance the system's security functionality. For example, this file defines the human readable labels that correspond to platform-specific trusted labels, it defines the valid cross-domain chat data flows, and it defines which user clearances may be used in which security domains. Similarly, access control lists for chat rooms are protected at the same MAC level of the chat rooms they control. As such, server instances may read the access control lists to mediate user-level access, but may never modify the access control lists. This protection scheme allows the system's discretionary access control (DAC) scheme in turn to be protected by higher assurance MAC mechanisms.

3.2.4 Chat Application Trusted Mediation Extensions

As was described in section 3.1.3, all chat data flows must be mediated by a trusted process. As such, all chat write operations are redirected to a trusted process by way of interprocess communication (IPC) pipes. Enabling this functionality required not only significant modification of the original COTS collaboration server source code, but it also involved the implementation of a new protocol for communications between chat server instances and the trusted chat monitor.

3.2.5 The ML Chat Trusted Chat Monitor

The trusted chat monitor has two main responsibilities. First, the monitor is responsible for initialization and shutdown; that is, it stops/starts the chat server instances and creates/deletes the IPC pipes used for communications between the monitor and the chat server instances. The second responsibility of the chat monitor is to mediate all chat data flows, which entails performing content checking on all traffic, ensuring that a request action request complies with the system's MAC policy, and executing all privileged operations.

3.2.6 Application Self-Protection Mechanisms

Several small modifications were made to the COTS collaboration server in order to increase the overall security posture of the ML Chat system. For example, an ASCII character filter was implemented that removes all non-allowed characters from chat data, and all chat messages are limited to a pre-defined size. This filter allows a system administrator to ensure that only valid chat content is being processed, and that potentially dangerous and/or malicious content (i.e., executable code) cannot be passed through the system. Most other security enhancements involved disabling system functions that were either not required, exceeded the scope of the prototype development, or degraded the security posture of the system. Most administrative functions were disabled, and most user features except group chat were disabled. While most of these modifications do not merit special attention, some rationale should be provided for disabling some traditionally "core" services in near-real-time collaboration tools. Specifically, a discussion of our rationale for disabling Point of Presence (PoP) and IM services is appropriate.

3.2.6.1 Rationale for Disabling PoP Services

Most chat systems have some sort of PoP feature (e.g., a "buddy list") that informs them of the status of other system users that interest them. While this feature is popular among users, it presents great challenges within the ML Chat system's architecture. From a policy perspective, it is unclear as to the extent to which

organizations would want users in external security domains to have a channel for surveillance as to its users' working patterns. From a security perspective, it would be difficult to implement a practical access control mechanism for determining which users could be on which other users' buddy lists. Finally, from a performance perspective, PoP systems generate a significant amount of cross-domain network traffic, and all of this traffic would most likely impact the performance of the system's trusted monitor.

3.2.6.2 Rationale for Disabling IM Services

One main reason IM services were disabled is IM services generally depend on PoP services. Also, IM services are difficult to centrally manage, as the usage practices associated with instant messaging are usually even more ephemeral in nature than those of group chat. Finally, there is the difficult problem of implementing cross-domain IM services in environments that require multi-enclave data flows. If a user attempts to send a message to multiple users in separate domains, it becomes quite difficult for the system to handle partial rejections; that is, scenarios in which a message can be transmitted to some of the intended recipients, but not to all of them. Secure solutions to this problem generally frustrate users, and user-friendly solutions often have security problems. Rather than attacking this dilemma, we avoided it by using a group chat-based scheme for enabling collaboration. One area of interest will be to determine the extent to which multilevel chat can and cannot be used to satisfy collaboration requirements traditionally satisfied by instant messaging systems.

4.0 MULTILEVEL CHAT TESTING AND RESULTS

The ML Chat system was exposed to three different testing scenarios: stress testing, penetration testing, and an operational assessment. Each of these tests will be briefly described in this section.

4.1 Stress Testing

The initial prototype of the ML Chat system underwent stress testing in the development laboratory. Thirty clients were configured to access the ML Chat system from two different security domains, with 10 clients in the first domain and 20 in the second. Using the Ktest Journal Macro automated testing tool, each client was configured to send one 50 character chat message each second to one of three chat rooms for 6000 seconds. The results showed that the ML Chat server could process over 100,000 messages an hour with no performance degradation on a two-interface implementation. While server load metrics were unavailable, client performance did not degrade throughout the testing period. While not exhaustive, this testing was interpreted as sufficient for assuming that the system could handle any load of chat traffic that could be imposed by the system's maximum limit of 256 simultaneously connected users.

4.2 Penetration Testing

While the exact results of the initial penetration testing efforts are classified, the testing did yield sufficiently favorable results to obtain approval to use the system in a limited operational experiment (LOE) that connected a classified US network to a classified multilateral network. Additionally, the US Naval communications program executive office was sufficiently encouraged by the results of the penetration testing to move forward in transitioning the ML Chat system into a US Naval Program of Record (POR).

4.3 Operational Assessment in a Limited Operational Experiment

The ML Chat system was used in a combined joint task force exercise. In this experiment, users in disparate security enclaves and a heterogeneous mixture of operational environments used the ML Chat system to conduct various mission functions. Specifically, US and coalition forces used the system to coordinate joint task force communications coordination. Users from shore sites, large and small deck coalition naval platforms, and mobile command centers were able to successfully conduct near-real-time collaborative planning for coordinating communications across multiple security enclaves. Users were able to access the ML Chat server from a variety of workstation platforms, including Windows NT, Windows 2000, and the US Navy's Multilevel Thin Client (MLTC) MSL system. While the system usage was limited to a maximum of 150 users, the system functioned properly throughout the exercise. There were no system failures during the exercise, although one user did experience a malfunction with their user account. This error was resolved within 30 minutes of notification, and affected no other users during the experiment.

5.0 CONCLUSIONS AND FUTURE EXTENSIONS

The initial goal of the effort described in this paper was simply to develop an enabling capability for cross-domain near-real-time collaboration. From an operational perspective, this development effort has yielded evidence indicating that traditional cross-domain system architectures are less adequate for enabling cross-domain collaboration in today's non-hierarchical, net-centric information technology environment than certain heterogeneous cross-domain security architectures. From a programmatic perspective, the fact that this development effort successfully integrated commercial-of-the-shelf (COTS) products using government-developed security extensions shows that military organizations can enjoy the time and money savings afforded by using COTS products, while also providing a solution that is tailored to meet their exact needs by using GOTS extensions. From a systems perspective, the hybrid security architecture described in this paper proved adequate for enabling the ML Chat prototype system to meet the US Navy's cross-domain, near-real-time collaboration requirements in the highly heterogeneous environment of the US Joint Forces Command's limited operational experiment, JTFEX04-02. Furthermore, based on favourable performance, security, and operational testing results, the Naval program executive office is planning to transition the ML Chat system into a program of record to support multinational coalition collaboration in heterogeneous operational environments. Currently, future plans for enhancing the ML Chat system include:

- Integrating the ML Chat system into the Horizontal Fusion Portfolio
- Enhancing the authentication mechanism for ML Chat
- Enhancing system management functions to accommodate simplified system administration
- Implement system longevity enhancements and perform extended system longevity testing
- Performance enhancements for implementations with more than 3 enclaves

In addition to improving the ML Chat system, work has begun on a project to build a multilevel web server based on the ML Chat system's hybrid security architecture. This system will provide for more granular policies, a greater variety of data types, stronger authentication, and better data mining tools. Ideally, the ML Chat system and this proposed multilevel web system could then be integrated into a comprehensive cross-domain collaboration system. The end result of these efforts should improve security, performance, and international interoperability for coalition forces working in today's network-centric operational environment.

6.0 ACKNOWLEDGEMENTS

The work described in this paper was funded by the United States Navy's Office of Naval Research. Their support is gratefully acknowledged. Also, we would like to thank the staff of US Commander, Second Fleet and NATO Commander, Striking Fleet Atlantic for their invaluable support and feedback during development and prototyping.

REFERENCES

[1] Robert Vietmeyer. "Net-centric Enterprise Services", from *Boundaryless Information Flow Conference*, Boston, MA. Available at http://www.opengroup.org/public/member/proceedings/q304/Presentations/ge_nces.pdf, July 2004.

[2] Cristina Gacek, Barry Boehm. "*Composing Components: How Does One Detect Potential Architectural Mismatches?"* Center for Software Engineering. Available at http://sunset.usc.edu/TechRpts/Papers/usccse98-505.html, May 1998.

[3] Maurizio Morisio, Nancy Sunderhaft. "*Commercial-Off-The-Shelf (COTS): A Survey"*, Air Force Research Laboratory Data & Analysis Center for Software. Available at http://www.dacs.dtic.mil/techs/cots/cots.pdf, December 2000.

[4] Cynthia E. Irvine, Timothy E. Levin, et. Al. "Overview of a High Assurance Architecture for Distributed Multilevel Security", in *Proceedings of the 5th IEEE Systems, Man and Cybernetics Information Assurance Workshop*, United States Military Academy, West Point, NY, 10–11 June 2004, pp 38-45.

[5] Cynthia E. Irvine, Timothy Levin, "A Cautionary Note Regarding the Data Integrity Capacity of Certain Secure Systems," *Integrity, Internal Control and Security in Information Systems*, ed. M. Gertz, E. Guldentops, L. Strous, Kluwer Academic Publishers, Norwell, MA, pp 3-25, 2002.

[6] D. E. Stevenson. "From DEVS to Formal Methods: A Categorical Approach", in *Proceedings of SCSC 03, Summer Simulation Multiconference*, Montreal, Canada, July 20-24, 2003.

[7] Kristina C. Rogers. *ITT Industries Dragonfly Guard Security Target Version 2.0*, ITT Industries. Available at http://niap.nist.gov/cc-scheme/st/ttap/TTAP-ST-0001.pdf, October 1998.

[8] Allan McClure, "A US Joint Force Command Solution to Coalition Interoperability, " *The Edge*, vol. 5, July 2001.

[9] J. Hackerson, "Design of a Trusted Computing Base Extension for Commercial Off-The-Shelf Workstations (TCBE)," Master's thesis, Naval Postgraduate School, Monterey, CA, September 1997.

[10] D. E. Bell, L. LaPadula, "Secure Computer System: Unified Exposition and Multics Interpretation," Tech. Rep. ESD-TR-75-306, MITRE Corp., Hanscom AFB, MA, 1975.

[11] K. J. Biba, "Integrity Considerations for Secure Computer Systems," Tech. Rep. ESD-TR-76-372, MITRE Corp., 1977.

[12] W. Boebert and R. Kain. “A practical alternative to hierarchical integrity policies.” In *Proceedings 8th DoD/NBS Computer Security Conference*, pp. 18-27, Gaithersburg, MD, September 1985.

[13] Steven R. Balmer, Cynthia E. Irvine, "Analysis of Terminal Server Architectures for Thin Clients in a High Assurance Network," *Proceedings of the National Information Systems Security Conference*, Baltimore, MD, pp 192-202, October 2000.

[14] Wang Government Services, Inc., McLean, VA, *DII Guard Concept of Operations, Document ID: FS98-122-01*, February 1999.

[15] M. H. Kang, I. Moskowitz, “Design and Assurance Strategy For the NRL Pump,” *IEEE Computer*, vol. 31, pp. 56–64, April 1998.

[16] Ravi Sandhu, “On Some Research Issues In Multilevel Database Security”, in *Proceedings From the 5th RADC Workshop on Multilevel Database Security*, October 1992.

[17] Cynthia E. Irvine, Timothy Levin, "Data Integrity Limitations in Highly Secure Systems," in *Proceedings of the International Systems Security Engineering Conference*, Orlando, FL, February 2001.

[18] James Rome, “Compartmented Mode Workstations,” from 18th Department of Energy Computer Security Group Training Conference, Seattle, WA. Available at: http://jamesrome.home.comcast.net/security/fileroom/doecmw.pdf, April 24, 1996.