Secure Compartmented Data Access over an Untrusted Network Using a COTS-based Architecture

Dr. Paul C. Clark
Marion C. Meissner
Karen O. Vance
SecureMethods, Inc.
8460 Tyco Road, Suite A
Vienna, Virginia 22182

Abstract

In this paper, we present an approach to secure compartmented data access over an untrusted network using a secure network computing architecture. We describe the architecture and show how application-level firewalls and other commercial-off-the-shelf (COTS) products may be used to implement compartmentalized access to sensitive information and to provide access control over an untrusted network and in a variety of environments. Security-related issues and assumptions are discussed. We compare our architecture to other models of controlling access to sensitive data and draw conclusions about the requirements for high-security solutions for electronic business as well as DoD applications.

1. Introduction

Users of commercial applications as well as more traditional users of high-security applications, such as the DoD and intelligence community, increasingly require access to sensitive information over an untrusted wide area network using high-assurance security mechanisms. Although many systems have been designed for this purpose [3][5], often they are highly specialized, non-scalable, expensive and cumbersome to use. Such systems can be difficult and costly to maintain and upgrade, since changes must be implemented and tested in a customized and highly specialized environment. Furthermore, operating system changes often require modifications and re-testing of system applications to ensure the entire system continues to conform to security requirements.

Historically, the intelligence community has been a forerunner in requiring and developing high-security network applications. By analyzing their groundbreaking work, other researchers and developers have arrived at requirements for the next generation of high-security applications for electronic business, healthcare data management, and a plethora of other commercial and non-commercial applications. These requirements include the following and form the starting point for the secure network computing architecture described in this paper.

- *High-assurance security*: Applications should provide high confidence through strong security services for confidentiality, data integrity, user-based authentication, and non-repudiation. System assurance should be limited only by the underlying COTS platform and applications.
- *Secure submission and retrieval*: Data must be protected while allowing authorized users to access and update information.
- *High usability*: Uniform user interfaces, such as those provided by e-mail and the World Wide Web, should be used to increase user acceptance and efficiency while reducing the training costs of high-security solutions.
- *Scalability*: A secure solution must be extensible to meet ever-increasing throughput and storage requirements.
- *Commercial-off-the-shelf (COTS) products*: Secure solutions should employ COTS-based hardware and software where possible to leverage vendor and industry development, testing, and validation and to minimize redevelopment and support efforts. COTS clients, for example

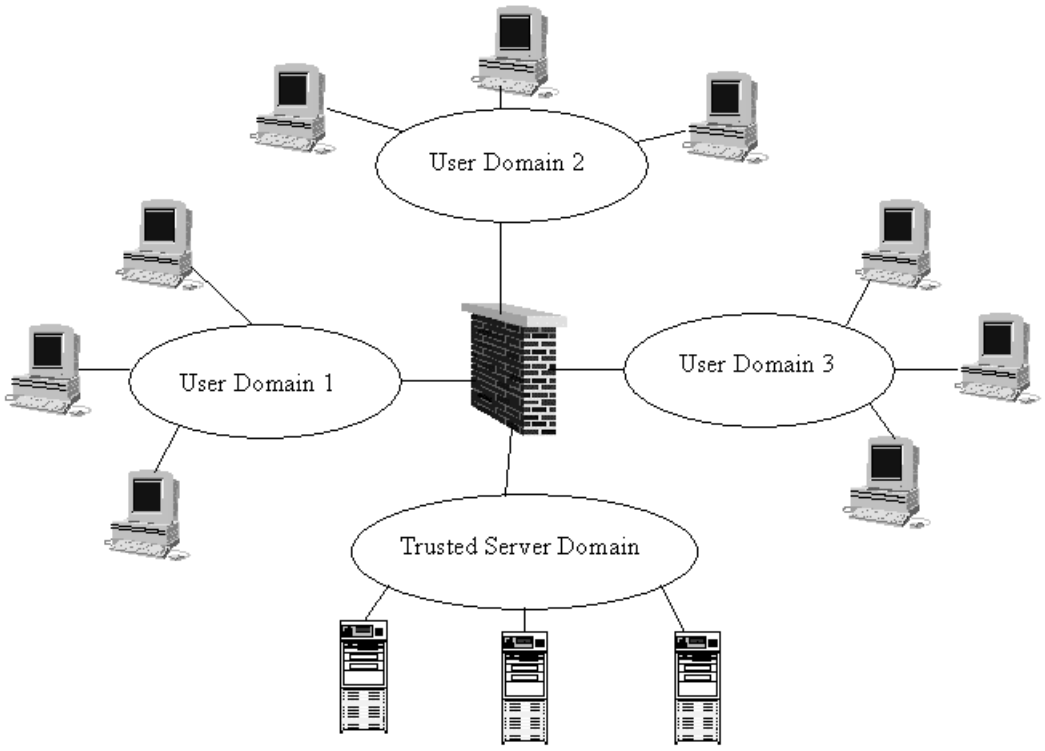

Figure 1. High-level secure network architecture

personal computers running Windows operating systems, are already prevalent but servers also benefit from the use of COTS products while maintaining the appropriate level of security assurance for each application.

Current firewall technologies and powerful, low-cost computing platforms lend themselves to a solution based on COTS hardware and software and standards-based cryptographic algorithms and mechanisms. In this paper we present such an approach to providing secure user-based access to protected data over an untrusted Wide Area Network (WAN). The deployed secure network architecture comprised of COTS components provides secure data access over trusted or untrusted networks and high-assurance access control in a highly usable, flexible, and scalable manner.

In our approach, transaction-oriented access to sensitive data is controlled on a per-user basis. Individual users are authenticated for each transaction using cryptographic techniques and authorizations are verified before data submissions or retrievals are permitted. The data being protected and securely accessed is compartmented in a manner that reflects the organization of data in the real world and/or in the application environment. This compartmentalized approach has been used successfully on standalone systems to restrict access to information on a "need to know" or project-specific basis. It also allows access to be controlled by specific resources and access rights, as well as on a per-project or per-compartment basis. Trusted servers on a protected network domain can store the data for different compartments and process requests for data accesses to that compartment whether requests are generated on the local network or remotely over the WAN.

This paper describes our approach and how the architecture satisfies the requirements of convenient, secure and tightly controlled access to sensitive data over a network. Then, we address security issues inherent in the architecture, compare it to related research work in the area, and, finally, discuss future directions for the architecture and its implementations.

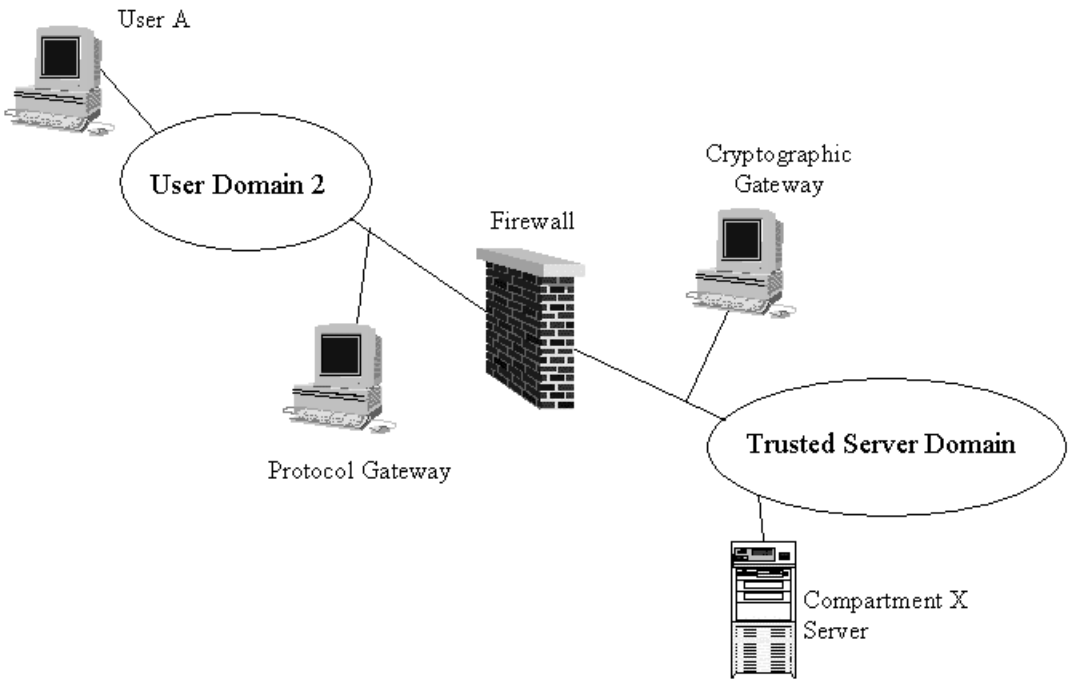

Figure 2. Configuration for a Sample Access Request

2. Security Architecture

The architecture presented here uses a commercially available application-level firewall and other COTS machines running standard Unix or Windows NT/2000 operating systems to act as gateways between the users and the servers. The firewall mediates access between multiple network domains, with trusted servers and potentially several different levels of users in two or more separate domains. In the trusted server domain, a different server stores data and processes access requests for each of several compartments per transaction. These servers run unmodified COTS operating systems and software. By policy, only administrative user logins are permitted on this trusted server domain. The architecture therefore provides strong protection for sensitive data by physically separating servers containing compartmented information from other users and servers on the network. This configuration is illustrated in figure 1.

As shown in figure 1, users can use client workstations located in any of the user domains. They access compartmented information on the trusted servers via a user interface, such as a Web-based form. A protocol gateway, such as a Web server, located on an untrusted network domain, presents the user interface on the client workstation. Using a Web-based form, the user completes a secure data submission or retrieval request, and encrypts and digitally signs the request using cryptographic algorithms and optional hardware tokens, as appropriate for the application and compartment to be accessed. In addition to or instead of a digital signature, a user may utilize a biometric to provide authentication for the request. For higher usability, the enhancement process (encryption, signature, encoding, etc.) can occur automatically, for example, through a Web plugin. Depending on the application requirements, users optionally have to initiate or acknowledge enhancement of requests. The firewall relays the data submission or retrieval request to the trusted server on the appropriate network domain, via a cryptographic gateway.

Once the request reaches the trusted domain, the cryptographic gateway verifies the digital signature and/or biometric template and decrypts the transaction. The architecture supports Public Key Infrastructure (PKI) mechanisms and directories for managing and retrieving user certificates and cryptographic keys in a scalable manner, e.g., LDAP. After successful authentication, the cryptographic gateway checks the requested action and the originator against its access control list (ACL), and determines whether the user is authorized to perform the requested action. If the user's request is not authorized or the authentication fails, the cryptographic gateway will reject the request immediately. Authenticated requests from authorized users are delivered to the appropriate server to be processed. The server sends the result of the request to the cryptographic gateway, where it is encrypted for the originating user and digitally signed, and relayed back to the originator. Figure 2 shows the configuration necessary for an authenticated and authorized access request from User A on User Domain 2 to Compartment X.

Network accesses to the trusted server domain are strictly transaction-oriented and must be digitally signed and/or biometrically authenticated by the individual user making the request. The security policy for this architecture permits no direct network logins to trusted servers, except for administrative access. This policy is enforced by the firewall, which mediates access between the different network domains, allowing only transaction-oriented communications addressed to the cryptographic gateway to pass through to the trusted server domain. Administrative logins are protected using strong authentication. Thus the firewall is used to control the perimeter of the trusted server domain.

As mentioned previously, authorizations are performed through the use of ACLs maintained on the cryptographic gateway located on the trusted server domain. In our approach, ACLs are human-readable files containing information on protected resources and the individual users that may access them. To be accessible from the WAN, each resource must be explicitly listed in one of the ACL files, along with authorized users. Resources are represented by Uniform Resource Locators (URLs), making them both specific and finely granular. The protocol used to access the resource, the hostname on which the resource is located, and the location and name of the resource can all be specified with this format. Since the firewall prevents direct connections to machines on the trusted domain, the ACLs are inaccessible to regular users and cannot be modified. Furthermore, the firewall routes all transactions to the cryptographic gateway to be verified and authorized against the ACL before being processed by a server. In this way, the architecture enforces access controls, equivalent to mandatory access control (MAC), for any compartmentalized data stored on the trusted servers.

As described above, the firewall, the cryptographic gateway, and the trusted servers on the server domain together enforce a unified security policy over the system. Thus, according to TCSEC and TNI criteria [6][7], the firewall and the trusted network domain constitute a distributed Trusted Computing Base (TCB) for the architecture – without the need for specialized operating systems. Together with an integrity protected, properly functioning client and the cryptographic mechanisms used to encrypt and digitally sign each transaction between the user and the cryptographic gateway, this TCB provides a trusted path between the user and the compartmentalized sensitive data storage at a level of assurance determined by the cryptographic algorithm, key length, and key management scheme employed.

The architecture described here can be scaled easily to meet the capacity and performance needs of different applications. As system loads increase, additional protocol gateways, cryptographic gateways and trusted compartmented data servers may be installed to provide increased capacity and fault tolerance through redundancy and load balancing. Also, individual hardware components may be upgraded to faster and more robust equipment as technology advances. In current computing environments, data throughput is limited by the speed of the network infrastructure, and network links can be upgraded as needed.

3. Security Issues and Assumptions

The security of this architecture depends on the proper functionality and object reuse characteristics, as well as the security and continued integrity of the individual client workstation's operating system and application software used to initiate data access requests and receive data from different compartments. For example, once data has been retrieved to a shared workstation, the client application and platform must clean up after itself, which includes emptying history and cache files and wiping the application memory and file system, or the next user can obtain access to the data via the temporary files and storage. In a more malicious example, programs running on a client workstation must not capture sensitive data retrieved from a trusted server and forward it to an unauthorized user. Another potential danger of a compromised client is a denial of service attack, where software on the workstation could garble the digital signature on an outgoing request, causing the signature to be unverifiable on a trusted server and legitimate access

to be denied. Controls to prevent such accidental or malicious threats to the client workstation exist and are described below.

Therefore, to offer a reasonable level of assurance in this architecture, we must be able to make the following assumptions about client workstations:

- A client is functioning properly, with no accidental or malicious disclosure of sensitive data, data corruption, or denial of service. In addition, a client will not save or transmit protected data without the user's knowledge. Also, an authorized user presented with a sensitivity level will not deliberately capture or forward sensitive data.
- A client runs only as a client, i.e., there are no server services such as an FTP server, file sharing or a telnet server running. This implies that a client is not remotely accessible and users wishing to use a client workstation to access data must have physical access to that machine.

The level of trust required in the client's functioning properly depends on the application and the sensitivity of data being accessed. For many applications, it is sufficient to use a standard commercial operating system because it has been tested extensively by the vendor and through routine use in the field. However, for an application requiring further assurance of the operating system and client workstation functionality, the clients to be used for the application should be certified by a trusted certification agency.

Once we are confident that the client is performing properly, there are several mechanisms available to ensure that the client is not corrupted and that no malicious code can be run, either deliberately or inadvertently. Individual workstations can be booted from a smartcard using the patented Boot Integrity Token System (BITS) [4]. BITS stores a computer's boot sector on a smartcard and requires the smartcard and a password to boot the machine. Once booted from BITS, the computer is assumed to be operating correctly and can run anti-virus software and other integrity checks to assure continuing trust in client applications. Strict configuration control can be used to manage legitimate changes to the client.

In addition to reliable software and hardware controls on the client, users must be responsible for protecting data retrieved onto a client workstation. Any data saved to the local client by the user must be removed or protected (e.g., by encryption) before another person, who may not have the same authorizations, uses the workstation. If the client is a mobile workstation such as a laptop computer, the machine's owner must take care to physically protect it.

Enforceable policies must also be in place to prevent users from deliberately or inadvertently circumventing the security inherent in the network configuration. For example, users must not connect workstations directly to the trusted server network domain and bypass the firewall. Some of these threats can be managed by placing tight controls on configuration management or by attaching network sniffers to trusted server domains to monitor for inappropriate network traffic and alert administrators if necessary. Nevertheless, user compliance is vital in this and other compartmented data access models. It can be encouraged through education and administrative or legal remedies and is assumed for the purposes of this discussion.

In these assumptions, the architecture described here is no different from any other implementation of secure compartmented data access. For example, in a paper-based system, high-security controls may be in place to ensure that only authorized users see certain classified documents. However, if an authorized user leaves sensitive material lying around in the open, all security measures may be for naught.

4. Comparison with related work

The work presented in [5] uses the Secure Shell (SSH) remote login protocol to access data on Compartmented Mode Workstation (CMW) hosts from COTS clients over an untrusted network. Although this system requires modifications to SSH, in addition to a specialized operating system on the data 'server,' it illustrates the power of data accesses from widely available commercial clients such as Windows NT. In this model, servers and clients authenticate to each other before any data or even requests for data are transmitted, whereas in our solution clients generally send digitally signed and encrypted requests through cryptographic gateways before having any assurance of their identity. Receipt notifications and query results returned from the trusted servers are also digitally signed and encrypted. While our architecture can support a mode of operation where all components authenticate themselves to each other before communicating, this requires additional steps, software, and safety measures.

Domain and Type Enforcement (DTE) [1][2] also provides user-based access to data. DTE enforces mandatory access control through operating system modifications to associate types with objects on a system, for example files, and domains with processes on a

system. A user chooses a role upon login, which is associated with a domain. A security specification file specifying each domain's access rights to different types is read at system boot into data structures. During system operation, kernel mechanisms check the data structures to verify authorizations for accesses to types from domains. This is analogous to our ACL file described earlier. DTE protection is extended across networks [8] in a manner that protects connection-oriented and transaction-oriented (connectionless) protocols.

Unlike our approach, the DTE research prototype requires significant modifications be made to the operating system and to some standard applications (e.g. login) to provide the mandatory access control mechanisms. Each new operating system release entails porting these changes to the new release and retesting the system. Since our approach uses unmodified COTS products, these updates are unnecessary. The distributed nature of our architecture makes it scalable to greater numbers of users, since redundant components can be added to provide load balancing and fault tolerance. Currently, our approach assumes that objects are usually not created, but instead updates are made to existing databases, etc. Under this assumption, requiring administrative intervention to create and label new objects is acceptable. In cases where new objects are frequently created, our approach could benefit from the scalability provided by DTE's implicit attributes, in which type labels are implicitly inherited from parent objects until overridden by an explicit label specified in the security specification.

Also, DTE performs user authentication only once rather than for each transaction. At login, DTE can achieve any desired level of assurance in user authentication, from simply requiring a password to using token-based or biometric authentication. However, once the user has logged in, he or she is authenticated and trusted by the operating system for all further actions performed during a session. Our approach authenticates and protects each data request and submission.

5. Future Directions

Our secure compartmented data access model works well for submissions to and retrievals from files and databases that are already in place. However, creating or modifying data structures such as database tables or subdirectories requires administrator intervention and is not very scalable. Planned enhancements include revisions to the way access to resources is controlled through ACLs. We anticipate that making access control inherited from parent to child data structures will make data resource creation less cumbersome while retaining a high level of security. For example, a newly created subdirectory would inherit its authorized users from its parent directory. In conjunction with this modification, we will maintain the capability to override inheritance, similar to the DTE model discussed earlier.

Besides making the system more powerful, inheritance of access rights increases its usability for system administrators. Other improvements in usability that are being investigated for future work include tools for secure remote user registration and high entropy key generation with a secure web interface and remotely existing issuer credentials.

6. Conclusion

Other systems exist that provide controlled access to sensitive, compartmented data. However, the architecture described here enables electronic business on a broad scale by using scalable and maintainable COTS-based server solutions with low or no-cost clients. Our approach implements a distributed TCB using COTS hardware and software components. Compartmentalized sensitive data is remotely accessible over an untrusted wide area network but cannot be directly accessed by users, thereby providing high-assurance protection of the data. We provide a trusted path for the data from its protected storage to authorized users. By allowing cryptographic enhancements to be transparently applied to individual transactions, usability of the system is increased in comparison to other secure systems and even relatively insecure systems – for example, those that merely require and rely on the use of user passwords. Due to its transaction-based nature, the presented solution also supports a secure audit capability and resource based authorization model. Finally, this architecture has been deployed and tested in multiple production environments with different assurance levels, where it has successfully met the challenges of providing secure, controlled access to sensitive compartmented data in a low-cost and usable manner.

Secure compartmentalized access in a networked environment is not only possible but it can be accomplished using standard, relatively inexpensive equipment and software. The use of COTS products and standard algorithms make for a flexible and scalable solution. The described architecture provides embedded strong user-based authentication and encryption for all accesses to trusted server resources over the wide area network. Although the architecture is subject to the same physical and administrative requirements as non-networked systems and special purpose secure data access

systems, it provides efficient technological mechanisms to help monitor and enforce diverse security policies.

References

[1] Lee Badger, Daniel F. Sterne, David L. Sherman, Kenneth M. Walker, and Sheila A. Haghighat, "A Domain and Type Enforcement Unix Prototype," USENIX Computing Systems Volume 9, Cambridge, MA, 1996.

[2] Lee Badger, Daniel F. Sterne, David L. Sherman, Kenneth M. Walker, and Sheila A. Haghighat, "Practical Domain and Type Enforcement for Unix," *Proceedings of the 1995 IEEE Symposium on Security and Privacy*, Oakland, CA, May 1995.

[3] Tse-Huong Choo, "Vaulted VPN: Compartmented Virtual Private Networks on Trusted Operating Systems," HPL-1999-44, Extended Enterprise Laboratory, HP Laboratories Bristol, March 1999.

[4] Paul C. Clark, Lance J. Hoffman, "BITS: A Smartcard Protected Operating System," Communications of the ACM, Volume 37, Number 11, New York, NY, November 1994.

[5] Chris I. Dalton, "Strongly Authenticated and Encrtyped Multi-level Access to CMW Systems over Insecure Networks using the SSH Protocol," HPL-99-98 (R.1), Extended Enterprise Laboratory, HP Laboratories Bristol, February 1999.

[6] National Computer Security Center, "Department of Defense Trusted Computer System Evaluation Criteria," DoD 5200.28-STD, December 1985.

[7] National Computer Security Center, "Trusted Network Interpretation," NCSC-tg-005 (Rainbow Series), November 1993.

[8] Karen Oostendorp, Lee Badger, Christopher Vance, Wayne Morrison, David Sherman, and Daniel Sterne, "Domain and Type Enforcement Firewalls," *Proceedings of the 13th Annual Computer Security Applications Conference*, San Diego, CA, December 1997.