CHAPTER 4

DEMONSTRATION AND ANALYSIS

This chapter analyzes the performance and operation of the NetCAD system. The system has been implemented to provide secure remote access to a .STL repair program that has been developed by Bøhn [Bøhn93]. This test case is discussed in Section 4.1. Section 4.2 discusses the strengths of the NetCAD system, while Section 4.3 discusses some of the limitations of the system.

4.1 CASE STUDY: .STL FILE REPAIR USING NetCAD

The .STL file format [Burns93] consists of an unordered list of triangular facets that approximate the actual surface of an original CAD model. It is a boundary representation of three-dimensional geometry, but contains only minimal topological information. Specifically, it contains a list of possibly unconnected triangular facets, with no guarantee that the a .STL model is closed or properly oriented. A model that is not closed and properly oriented cannot be fabricated because it does not describe a rigid solid.

Bøhn [Bøhn93] has developed a software program that takes a .STL file as input and repairs it as necessary to produce a .STL output file which describes one or more solids. This repair program lends itself well to integration with the NetCAD system, because it fits the general requirement of the system, wherein an input file is processed to produce an output file. Therefore, to demonstrate the NetCAD system, a sub-server was developed to provide .STL file repair service using the above program.

Figures 4.1 (a) and (b) illustrate the successful use of the NetCAD system to provide a .STL repair service. Here, the .STL representation of a cube with a missing facet was used as the input file (Figure 4.1 (a)). This .STL file was placed on a Pentium™ 166 MHz client computer running Windows 95™. The NetCAD server ran on an SGI Octane™ 195 MHz R10000 workstation running IRIX 6.4. The SGI Octane also hosted the NetCAD web site where the NetCAD applet code resided. Table 4.1 presents the configuration of the client and the server computers. The NetCAD applet, running on the client, sent the original .STL file to the server. The server repaired the file and returned it to the applet on the client computer, where it was saved in the user specified location. Figure 4.1 (b) shows the repaired .STL file.

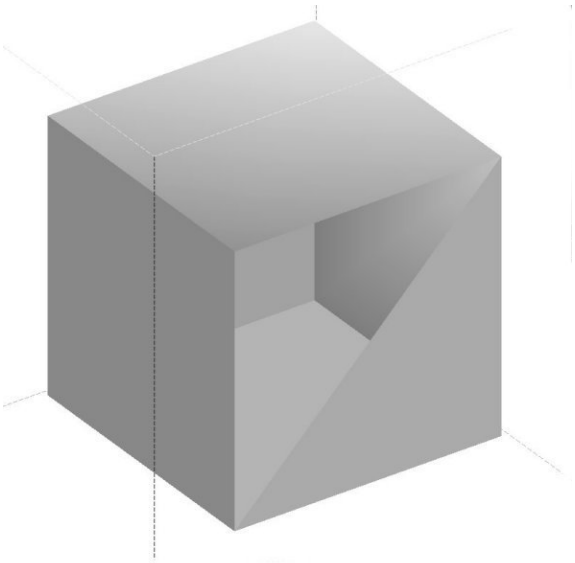

Figure 4.1(a) A cube described in the .STL file format with a triangular facet missing.

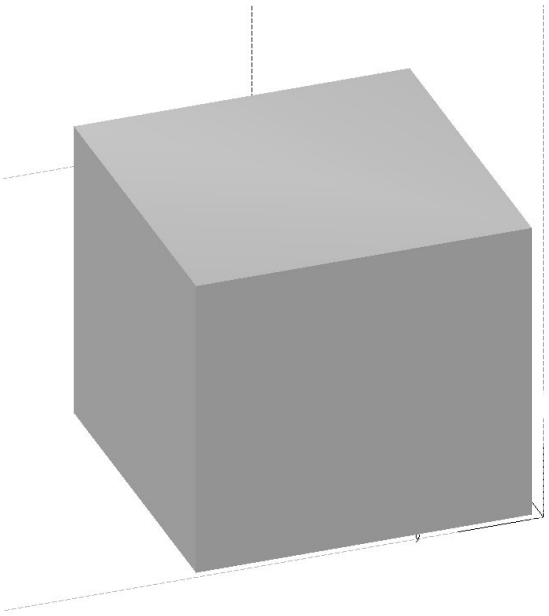

Figure 4.1(b) The same cube, described in the .STL file format, after being repaired by a .STL repair software made available by the NetCAD system.

Table 4.1. Configuration of the client and the server computers used for sample sessions.

Computer	Architecture	Operating System	Memory	Internet Connection
Client	Pentium 166MHz	Windows 95	32 MB	10 Base T
Server	SGI Octane 195 MHz, R10000	IRIX 6.4	128 MB	10 Base T

Time performance data was gathered from sample sessions with (1) the applet running on the Pentium computer and the server running on the SGI Octane, and (2) both the applet and the server running on the SGI Octane. Table 2 presents the performance data for these sessions. Table 4.2 shows that the bulk of the time is spent in the method that performs encryption and file transfer. The difference between the time taken to encrypt and transfer a file and the time taken to transfer the file by FTP gives an estimate

of the time used for encryption. This is a rough estimate as there are other factors involved in the file encryption method, for example, reading from the buffer and the relatively slow speed of Java executables compared to fully compiled executables. Using this estimate, it appears that encryption alone accounts for more than 99% of the processing time, and it is obvious that the transfer process is CPU-bound.

Table 4.2. Time required for various operations on the client-side on sample sessions.

Client Computer	File size	Session key generation	Key exchange	File transfer & encryption (T1)	File Transfer (ftp) (T2)	Encryption estimate (%) 100*(T1-T2)/T1
Pentium	612 KB	17,250 ms	330 ms	183,610 ms	800 ms	99.56
SGI Octane	612 KB	5,465 ms	159 ms	24,182 ms	70 ms	99.71
Pentium	1045 KB	18,361 ms	367 ms	275,614 ms	1,400 ms	99.49
SGI Octane	1045 KB	5,738 ms	162 ms	40,513 ms	110 ms	99.73
Pentium	3211 KB	18,029 ms	441 ms	902,257 ms	3,500 ms	99.61
SGI Octane	3211 KB	6,075 ms	149 ms	99,560 ms	310 ms	99.69

4.2 STRENGTHS OF THE NetCAD SYSTEM

The main strength of the NetCAD system lies in the extensibility of its server structure, the portability of its client, easy maintenance and its provision for secure access to remote engineering services. This is discussed further in the following sub-sections.

4.2.1 Extensibility

The NetCAD system has been designed with the extensibility of its server structure as an important feature. It is easy to create a new sub-server to offer a new service by reusing the existing code. As discussed in Section 3.2, to create a new sub-server, a new subclass

of *ClientHandler* and its corresponding *ClientHandlerFactory* subclass need to be created, while the rest of the structure remains unchanged. The only other changes that are required are (1) to assign a new unique network port, and (2) to add a few lines of code to create and launch the new sub-server with its *ClientHandlerFactory*. Thus, the NetCAD server structure can easily be extended to offer new services.

4.2.2 Portability

The NetCAD clients are Java applets which are inherently portable software modules. The applets have successfully been tested on computers running the Windows 95, AIX 4.2 and IRIX 6.4 operating systems. The applets only use the standard user-interface classes that are defined in Java Abstract Window Toolkit (AWT) package (*java.awt* package). The standard Java AWT is well-supported by all the major Java-enabled Web browsers, including the Netscape Communicator and the Microsoft's Internet Explorer. However, the applets have been designed to work with Netscape Communicator 4.5 browsers in that the applets use Netscape's Capability classes which are not supported by Internet Explorer. Furthermore, these two major browsers use different authentication mechanisms for signed applets. The applet's code was digitally signed with the author's VeriSign Digital ID for the Netscape Communicator, and this signing certificate is not valid for authentication with Internet Explorer. Hence, the applets currently only work with Netscape Communicator 4.5 browsers.

4.2.3 Easy maintenance and distribution

Since the NetCAD's client is a Java applet, every time a user accesses the NetCAD's website, the most up-to-date applet is downloaded and executed within the browser automatically, without requiring any installation steps. This obviates the difficulties associated with distribution of client software to the users. Furthermore, an upgrade of the server software is automatically accessible to the users through the applet's interface.

4.2.4 Security

Network security is an important feature of the system. The system only assumes end-point security; that is, that the executables and the JVMs at the end-points of the communicating entities are secure against tampering. Hence, the NetCAD security system includes the following features:

1. Hybrid cryptography is used to secure the network traffic by using an RSA key-pair to exchange a session key based on the DES algorithm for securing the rest of the data exchange.
2. Authentication of the applet is achieved by digitally signing the applet using the developer's digital signing certificate.
3. All Cryptix's classes loaded from locally installed Crypto_Support.jar file are checked for integrity by the applet at run-time. If any of the class files that are used has been tampered with, then the applet informs the user of the subversion and stops functioning.
4. The key-distribution of the public-key is made automatic and secure by archiving the key with the applet's signed JAR file, and by checking its integrity from within the

applet. Furthermore, the signing of the applet code prevents a man-in-the-middle attack on the applet.

5. A new session key is used for each service session. This prevents block-replay attacks. Furthermore, the cipher feedback (CFB) mode of encryption [Schneier96] links the plaintext characters together so that the ciphertext depends on all preceding plaintext. This conceals the stereotype beginnings and endings found in some files and protects against known-plaintext attacks [Schneier96].
6. Access control to local resources on the user's machine is achieved by using a capabilities-based model for managing access privileges to the resources. The user has to explicitly give permission to the applet for it to access a particular resource.
7. Since the encrypted files transferred to the server and the session keys can have persistence beyond the connection (unlike in the SSL protocol), encrypted files and the associated session keys can be saved for auditing.

4.3 LIMITATIONS OF THE NetCAD SYSTEM

There are two basic limitations of the NetCAD system: (1) it constrains the applet's code and files archived in the Crypto_Support.jar file to be signed by the same signing certificate, and (2) it is hampered by its slow encryption.

The main limitation of the NetCAD system is that it constrains the applet's class files archived in the applet's JAR and the files archived in the Crypto_Support.jar file to be signed by the same signing certificate. In effect, this means that if another developer creates a new server system which follows the same security scheme, then the user will need to replace the existing Crypto_Support.jar file in his or her browser with the one

signed by the new developer. This is because the security capability provided to the applet by the Crypto_Support.jar file is linked to its archived files' being signed by a particular signing certificate. This property is rooted in the modifications made in the *Security* class in the *java.security* package that is archived in the Crypto_Support.jar file. This linkage is by design, and can be disabled in the *Security* class. However, disabling it would open the download of Crypto_Support.jar file to a man-in-the-middle attack.

This situation can be remedied if the browsers' JVM come with the support of Java Cryptography Architecture (JCA) classes and have the ability to incorporate security capabilities by plugging in a JCA-compliant security provider library, such as Cryptix 3.0.3, into the browsers' directory structure. However, at this time, such advanced security support is not built into the JVM of any commercial browser. This problem is due, in part, to the laws governing export of cryptographic algorithms from the U.S.A, which currently only permit 40-bit encryption algorithms to be exported. The lowest acceptable RSA key is 512 bit, while the standard size of a DES key is 64 bits. Since the Internet is a global medium, such laws prevent the distribution of browsers with a JVM that has advanced cryptographic support. In addition, many of the encryption algorithms are patented and cannot be incorporated in the browsers' JVM free of cost.

The other limitation of the NetCAD system is its slow encryption. The NetCAD system requires transmission of whole encrypted files, which can be an extremely slow process if the file being transferred is large. Unfortunately, this problem is not easily solved. Encryption hardware can alleviate this problem to a certain extent, since encryption algorithms run significantly faster in the hardware. However, this would require all the client and server computers to be equipped with encryption hardware.