**CDSS Technical Security and Confidentiality Approach**

Security and confidentiality are vital issues for Web deployment of CDSS. The plan we have formulated aims to embody the state-of-the-art in this field: applying the National Research Council (NRC) guidelines (31) (as explained in the body of the protocol), and basing our general approach on lessons learned in implementing a framework for security and protection of confidential information with similar scope and technical requirements in other applications.

In this document, we will first describe our approach in terms of specific solutions proposed to address the NRC technical guidelines. Then we will give a detailed description of technical aspects of our approach.

**Technical Guidelines for Immediate Implementation**

1. *Individual Authentication of Users:* Users should have a unique user name/password, so that they can be held accountable of all actions taken under this user name. Moreover, frequently changing passwords are recommended.

*CDSS solution:* We have chosen an authentication approach based on public-key certificates (sometimes called “digital ID’s”). A certificate and corresponding private key known only to the person constitute the digital equivalent of an employee badge or driver’s license. CDSS certificates will be generated by our own Certificate Authority (CA) software. User certificates will expire within six months of issuance. One month before the expiration, we will contact physicians to determine whether they want us to issue a new certificate for them, and if so, for which patients. We will verify that these patients are indeed ones which they follow.

2. *Access Controls*: Access is granted to a user on a need-to-know basis. One common approach is to define groups of users, for instance by role (e.g., physician, nurse), Additionally, \*\*\*\* employees may have different access policies than external users.

*CDSS solution:* Properties associated with user certificates in conjunction with application-level access control will allow CDSS to restrict access to individuals functioning in particular roles. Individuals and role information are explicitly listed in security policy documents as described in the body of the protocol. We have defined several initial roles that will restrict access to non-public information:

• an external physician will have access to information about only the patients she or he follow.

• \*\*\*\* physicians or nurses will have access to all required information about any LTFU patient.

• Other \*\*\*\* employees will have access to only certain parts of the patients records.

• Security officers will have complete access authority to the information listed above, plus the possibility to issue, change, or revoke certificates.

3. *Audit Trails:* Retrievable and usable audit trails should be made available for each piece of clinical information. One assessment of this auditing system is to provide all accesses to the records for the employees who are also patients of the institution. These audit logs should be reviewed and commented also to prepare for the patients request of the audit trails about their own records.

*CDSS solution:* An audit trail database will record all accesses, actions, and results of user actions. Additional software will permit authorized users to retrieve from this vast database specific data such as a comprehensive listing of all accesses to a certain patient record or to particular fields of this record, with date and name of the person accessing and a justification of the access. This audit trail will serve several additional purposes. In particular, it will be the source of the evaluation data. Data transmitted via Internet will become part of the division LTFU database, as per standard policy for written and FAX transmitted clinical data.

4. *Physical Security and Disaster Recovery*: Unauthorized physical access to computers in the institution should be prohibited. A policy regarding paper printouts should be implemented. Provisions for disaster recovery should be undertaken, through practices such as regular backups.

*CDSS solution:* Computers used for client access to CDSS will be automatically disconnected after one hour of non-use. The agreement form covers the use of paper printouts. We will schedule daily backups of the different databases and programs.

5. *Protection of Remote Access Points:* Institutions should protect the electronic records from all external communications points, such as the Internet and dial-in telephone lines. A firewall should be immediately installed to protect a centralized Internet connection. For multiple entry points, other devices should be used, such as TCP wrappers. For users accessing the system through these points, encrypted or single-session passwords should be chosen.

*CDSS solution*: The CDSS system access is protected by a \*\*\*\* firewall. Firewalls are a kind of software designed to protect sensitive data from exposure on the Internet. All external access to this system will have to go through this firewall. No dial-in telephone line will be installed on the server where the CDSS system is physically installed.

6. *Protection of External Electronic Communications*: Institutions needing to protect information transmitted on open networks such as the Internet (including email) should encrypt all patient-identifiable information.

*CDSS solution:* All data transmitted on the Internet will be encrypted using a Secured Sockets Layer (SSL) protocol that employs an encrypted communication channel.

7. *Software Discipline*: Institutions should adopt a systematic virus-checking policy and limit as much as possible downloads to protect against malicious software.

*CDSS solution*: We check all new software received with an antivirus software and perform regular scans of the complete system. Java applets downloaded to users of the system will be authenticated as digitally signed code. Digital signing is a technique that is used to certify that software is downloaded from a trusted source.

8*. System Assessment*: Organizations should formally assess the security and vulnerabilities of the system on an ongoing basis, so that they can demonstrate it to external audits.

*CDSS solution*: An internal assessment of the security system will be provided every six months, using the latest ‘hackers scripts’ and ‘crackers’ programs. Reports from these assessments will be kept in order to be able to respond to external audits. We will engage the voluntary services of external computing professionals to review and routinely audit implementations, procedures, and policies in light of new technologies; lessons learnt in the deployment of CDSS, and the continuing evolution of the center-wide security and confidentiality approach.

**Technical Guidelines for Future Implementation**

1. *Strong Authentication*: Authentication systems making available single-session or encrypted authentication protocols should replace as soon as possible the user name/password of users. Token-based systems requiring some sort of card or badge may well become mandatory in health organizations.

*CDSS solution*: The authentication by public-key certificate is the technical equivalent to a token-based system. Moreover, any passwords required will not be entered directly on the keyboard, but via a graphic keyboard on the screen.

*2. Enterprise-wide Authentication*: Institutions are invited to develop a single authentication procedure that then permits the users to access all the data they are authorized to in the institution.

*CDSS solution*: All the data and programs accessible to CDSS users will require only one authentication procedure.

3. *Access Validation*: Institutions should ensure that users effectively get only the information that they are entitled to. The role or group of users access is often not sufficient for that purpose. Information needs often to be filtered more selectively.

*CDSS solution*: A security level is associated with each field of each database record. This is the finest granularity level possible. User certificates allow us not only to restrict information based on roles but also on individual identity.

*4. Expanded Audit Trails*: Expanded audit trails should allow the patients to trace the flow of information in their records through different institutions, for example in cryptographic envelopes and electronic watermarking technologies.

*CDSS solution*: We do not currently plan for CDSS to send any part of the record outside of the \*\*\*\*. Only limited access will be provided.

5. *Electronic Authentication of Records*: The logon identifier of the user who modifies each element of information in an electronic medical record needs to be recorded with the modifications.

*CDSS solution*: For each user, the audit trail mechanism keeps a record of each field of each database accessed, and the modifications made.

**Detailed Technical Description**

Because of the complexity of security and confidentiality issues, effective implementation requires a multi-faceted approach (figure 1). Though we do not intend to provide a technical tutorial on these topics in this section, we will attempt to briefly explain in more detail the key concepts and technologies involved in our approach.

**Firewall and reverse proxy server.** Figure 2 proposes a possible future high-level view of the overall \*\*\*\* security architecture, as we recommend it. This architecture is based on the assumption that \*\*\*\* will purchase, install, configure, and maintain a commercial firewall product and a reverse proxy server sitting outside the firewall. Reverse proxying is used to protect web server and databases behind the firewall from unauthorized access or tampering. A client connects to a proxy server (with a secure sockets layer (SSL) session if necessary). The proxy server initiates a second connection from it to the web server, from which it can retrieve data. All of this is transparent to the end user. Confidential data can remain behind the firewall and yet be accessible to the public as necessary. To provide additional security, the proxy server can be configured to speak only to the web server’s IP address and vice versa, and the firewall can be configured to allow HTTP traffic only between those two IP addresses.

For the present time, CDSS does not profit from the proxy server architecture, but the CDSS Web server is protected by the \*\*\*\* Web server and firewall, that play a role similar to that of the proxy server.



**Figure 1.** Multi-faceted CDSS approach to counter security and confidentiality risks.

**Authentication (proof of identity).** User authentication will be based on public-key certificate (digital ID) technology consistent with an International Telecommunication Union (ITU) standard called X.509. In addition to identifying information, certificates may contain special properties (in the form of X.509v3 extensions) such as “clearance type=A345” that may control the appearance of Web pages, what parts of the site or attributes of the database the user may have access to, or some other customization. Each extension has a criticality identifier, which indicates whether or not an implementation may ignore the extension.

Because the validity of digital ID’s has never been challenged in court, their legal status is not yet well defined. However Verisign (a leading supplier of certification services) has stated that preliminary legal research has resulted in the opinion that digital signatures based on this technology would meet the requirements of legally binding signatures for most purposes, including commercial use as defined in the Uniform Commercial Code (UCC). Similarly, they report that a GAO (Government Accounting Office) decision requested by the National Institute of Standards and Technology (NIST) has given their opinion that digital signatures will meet the legal standards of handwritten signatures.

Public-key technology is an advance over conventional forms of password protection—because certificates are public information, no sensitive data (e.g., passwords, private keys) need ever flow over the network where it could be intercepted. Certificates provide a high level of security because they are based both on something the user *has* (i.e., their certificate) and something they *know* (the password that protects the private key corresponding to the certificate).



**Figure 2.** High-level view of the overall CDSS security architecture.

At login time, a “personal certificate” establishes the identity of the user to a particular server (see “obtaining a personal certificate” below). Depending on the administrative policy, this “single sign-on” may grant access to additional servers provided that they also recognize the authority that signed the certificate. Access rights to particular information and services are selectively granted based on properties of personal certificates. Other types of certificates are used to verify the identity of servers (see “setting up a site certificate” below), of message-senders (e.g., S/MIME encrypted email), and of running code (see “signed code” and “agent authentication” below).

**Certificate authorities.** Certificates will be issued by our own private certificate authority server application (CA). Using a commercial CA product we will create a web-based interface allowing users to request certificates and allowing us to administer certificate requests. The CA will be hosted on the proxy server where it can be accessible to the public, yet isolated from applications and data behind the firewall. At some future time when certificate technology becomes pervasive within \*\*\*\*, CDSS certificates can be linked in a chain up to root certificates at the Center level.

**Setting up site certificates.** Site certificates allow clients to verify the identity of the server to which they need to gain access. To obtain a site certificate, the web administrator will generate a private and public key pair using a utility built-into the server. This generated pair is fed to the CA server, which issues the site certificate. This certificate is then installed in the web server. If, in the future, the CDSS application spans multiple servers, the same process is used to create site certificates for each server involved.

**Obtaining personal certificates.** Participants in the study requiring access to the CDSS will request a personal certificate from our CA server. Participants will install a signer certificate from the CA server in their browser to make it aware of their trust in the issuer of the personal certificate. Then they will fill out a form on a web page to supply the required information to verify their identity and the kinds of information to which they are requesting access.[[1]](#footnote-1) The administrator of the CA server will login to an administration page and process any pending requests. A policy will be established for the administrator to follow in verifying that the request was indeed sent by the person it said it was, and that the level of access privileges requested are appropriate to that person. Following successful processing of requests, the administrator will phone or send email to requestors telling them to log back in to the CA server to retrieve their personal certificates. The personal certificate will be automatically uploaded along with the required site certificates to the web browser.

**Certificate management.** Normally, personal certificates, like most other kinds of certificates, are set by the CA server administrator to expire after the period of time specified in our security policy description. The CA server administrator can access information about the certificate such as the date and time of issuance, who issued it, when it was revoked or due to expire, and so forth. In addition clients can also query the CA server to check the certificate’s status.

**Secure Sockets Layer (SSL).** SSL is an industry-standard security protocol for clients and servers that make use of the certificate-based public key technology described above (figure 3). It is supported by virtually all-leading vendors including Netscape and Microsoft. Normally, the SSL handshake between clients and servers should be transparent to the user. The client and server first exchange certificates to prove their identities. The exchange, which may include an entire chain of certificates, is followed by verification of validity dates and signatures by a trusted CA. Then the client randomly generates and securely communicates a set of four keys (two for client-to-server and two for server-to-client) for encryption and calculating message authentication codes. Finally, a message encryption algorithm (for encryption, see below) and hash function (for integrity, see below) are chosen. After successful completion of the handshake, a locked padlock appears in the lower left-hand corner of the main browser window. Requests for information initiated from client web browsers are processed in the following fashion: 1) the client secures and sends the message; 2) the encrypted message travels over the Internet; 3) the reverse proxy checks the message and passes the request through the firewall; 4) the received message passes security and is decrypted (both the possession of a valid certificate and the user’s access rights must be verified); 5) the server retrieves the requested information; 6) the requested information is secured and returned over the Internet; and 7) the received message is decrypted by the client application.



**Figure 3.** Secure Sockets Layer (SSL) overview.

**Code signing.** Since we expect Java applets to be extensively used as part of the CDSS client, the following security measures will be required. CDSS application developers will use their personal certificate to digitally sign Java class files and save them in a JAR (Java Archive). This allows users to be confident that the code was created by someone the CA has verified as trustworthy. When a signed object is accessed as an applet from a browser, an authentication protocol is executed between the client machine, where the applet will run, and the server that presents the applet. If the applet presents a certificate that was signed by a CA that the user has accepted into their web browser then the applet may be allowed certain access privileges (e.g., writing to disk, certain kinds of network access) that the user selectively grants upon being prompted by via a Grant/Deny dialog box. If the applet is signed, then the dialog box clearly shows this; otherwise if the applet is not signed or is signed but not by a CA from which the user has a signer certificate, then the dialog box will clearly present this as “untrusted” while still giving the user Grant/Deny opportunities. After the initial interchange, the user may indicate whether or not to display prompts from trusted applets in subsequent sessions.

**Agent authentication.** We are currently developing versions of KAoS agent technology for secure web applications. In addition to requiring agent code downloaded to the client to be digitally signed (like any other Java code), any use of agents in CDSS applications will also require that communicating agents mutually authenticate each other and encrypt sensitive information using SSL protocols before being allowed to engage in conversations over public network connections.

**Privacy (protection from disclosure).** Encryption is used as a preventive measure against unauthorized “eavesdropping.” As described above, before transmission over the network, RSA-based public-key and symmetric key cryptography techniques will be used to automatically encrypt the information or Java code. Decryption of the information can be accomplished with the user’s private key. To minimize the chances of unintended disclosure of patient identity, we will employ a well-proven algorithm for generating a globally unique identifier (GUID) for each Unique Patient Number (UPN). Access to patient identity will be governed by strict adherence to the policies described in the protocol.

**Non-repudiation (proof of transaction) and auditing.** In addition to session-level authentication and proof-of-transaction mechanisms based on digital certificate exchange described above, we intend to install Web-site auditing and analysis software that will permit us to do detailed tracking of site usage by particular individuals. Site usage logs will be routinely audited by CDSS personnel and will be available to users (e.g., patients, caregivers) upon request in accordance with the policies described in the protocol.

**Integrity (protection from alteration).** As described above, the message integrity service defined by SSL assures that session traffic does not change as it travels over the Internet. SSL uses a combination of a shared secret (a key negotiated during the SSL handshake) and hash functions (special mathematical functions) to assure integrity. Viruses and “trojan horse” programs are another integrity concern. The Java programming language was designed from the ground up with a strong model for security in mind. Code signing allows users to verify the source of any downloaded programs. Access to system resources outside of the Java “sandbox” are granted only with explicit user permission, as described above. In addition, virus-scanning technology is included within servers such as Netscape Proxy Server. Netscape’s proxy server also helps system administrators implement their own security policies by allowing them to stop transmission of MIME types, Java and JavaScript, and ActiveX components altogether.

**Authorization (access control).** Properties contained in a CDSS-specific extension for personal certificates will determine access to particular CDSS information and services in accordance with the policies described in the protocol. Additional database access control mechanisms will assure a fine grain of access control to different portions of the knowledge base. Permission to access or change particular information (e.g., set of patient records or specific fields within those records) will be granted or denied based on individual or role identity.

1. We will track emerging W3C standards (e.g., OPS) for the publication and protection of personal information. [↑](#footnote-ref-1)