Version 1.00 – 2/1/03

SECURE AUDITING FOR LINUX –

NETWORK TRAFFIC ANALYSIS

February 2003

SecureAudit uses SSLv3 to transfer the system call packages to a remote server for analysis and storage. http://wp.netscape.com/eng/ssl3/3-SPEC.HTM#7-5 (see below) describes the handshaking protocol, which happens at the beginning of the SSL session (when the client first connects to the server). Our client and server agree on a cipher suite, exchange certificates and RSA keys, and verify each other. This completes the handshaking and the client begins transmitting its system call packages.

In the attached Excel document are excerpts from the output of tcpdump and ssldump during an execution of our SecureAudit client and server. The first two lines show the TCP connection being established. The next block shows the client hello, including a list of possible cipher suites. The third shows the server's response, including the cipher suite decision, the certificate verification, and a request for the client's certificate. The fourth block shows the client's reply, including its certificate, encryption key, certificate verification, and a confirmation that it is switching to the chosen cipher suite. Finally, the server confirms the switch to the new cipher suite. After this are many lines of "application_data". These are the system call packages. The last two lines show the client and server closing their respective TCP connections.

------ excerpt from http://wp.netscape.com/eng/ssl3/3-SPEC.HTM#7-5 ------

Handshake protocol overview

The cryptographic parameters of the session state are produced by the SSL Handshake Protocol, which operates on top of the SSL Record Layer. When a SSL client and server first start communicating, they agree on a protocol version, select cryptographic algorithms, optionally authenticate each other, and use public-key encryption techniques to generate shared secrets. These processes are performed in the handshake protocol, which can be summarized as follows:

The client sends a client hello message to which the server must respond with a server hello message, or else a fatal error will occur and the connection will fail. The client hello and server hello are used to establish security enhancement capabilities between client and server. The client hello and server hello establish the following attributes: protocol version, session ID, cipher suite, and compression method. Additionally, two random values are generated and exchanged: ClientHello.random and ServerHello.random.

Following the hello messages, the server will send its certificate, if it is to be authenticated. Additionally, a server key exchange message may be sent, if it is required (e.g. if their server has no certificate, or if its certificate is for signing only). If the server is authenticated, it may request a certificate from the client, if that is appropriate to the cipher suite selected.

Now the server will send the server hello done message, indicating that the hello-message phase of the handshake is complete. The server will then wait for a client response.

If the server has sent a certificate request message, the client must send either the certificate message or a no certificate alert. The client key exchange message is now sent, and the content of that message will depend on the public key algorithm selected between the client hello and the server hello. If the client has sent a certificate with signing ability, a digitally-signed certificate verify message is sent to explicitly verify the certificate.

At this point, a change cipher spec message is sent by the client, and the client copies the pending Cipher Spec into the current Cipher Spec. The client then immediately sends the finished message under the new algorithms, keys, and secrets. In response, the server will send its own change cipher spec message, transfer the pending to the current Cipher Spec, and send its Finished message under the new Cipher Spec. At this point, the handshake is complete and the client and server may begin to exchange application layer data. (See flow chart below.)