INTERNET_DRAFT Printer Working Group
<draft-ietf-ipp-security-02.txt>

May 28, 1997 Expires November 28, 1997

Internet Printing Protocol/1.0: Security

Status of this memo

Abstract

This document is one of a set of documents which together describe all aspects of a new Internet Printing Protocol (IPP). IPP is an application level protocol that can be used for distributed printing on the Internet. The protocol is heavily influenced by the printing model introduced in the Document Printing Application (ISO/IEC 10175 DPA) standard, which describes a distributed printing service. The full set of IPP documents includes:

- Internet Printing Protocol/1.0: Requirements
- Internet Printing Protocol/1.0: Model and Semantics
- Internet Printing Protocol/1.0: Security
- Internet Printing Protocol/1.0: Protocol Specification
- Internet Printing Protocol/1.0: Directory Schema

This document deals with the security considerations for IPP.

Table of Contents

1.0 Introduction
2.0 Internet Printing Environments
   2.1 Client, Content and Printer in the same security domain
   2.2 Client and Printer in one security domain, Content in another
   2.3 Client and Content in one security domain, Printer in another
   2.4 Printer and Content in one security domain, Client in another
   2.5 Printer, Content, and Client all in different security domains
3.0 Security Considerations for IPP Operations
   3.1 Create Job
   3.2 Send Document
   3.3 Cancel Job
   3.4 Get Jobs
   3.5 Get Attributes

4.0 IPP Security Threats and Methods of Attack
   4.1 Threats
   4.2 Methods of Attack
   4.3 Quality of Service
5.0 Attacks vs. Security Services
6.0 Security Solutions
   6.1 Comparison of underlying technologies
   6.2 Detailed description of selected technologies
      6.2.1 S/MIME
      6.2.2 Transport Layer Security (TLS)
      6.2.3 IPSec
      6.2.4 Simple Authentication and Security Layer
      6.2.5 Digest Access Authentication
7.0 References
8.0 Author's addresses

1.0 Introduction

The purpose of this document is to describe security considerations for the Internet Printing Protocol (IPP). Internet Printing is the application of Internet technology to network printing. Using Internet technology, users want to be able to locate printers, install and configure printer software, query printers for capabilities and status, and submit and track print jobs. The Internet Printing Protocol defines the network interface for many of these functions.

It is required that the Internet Printing Protocol be able to operate within a secure environment. Wherever possible, IPP ought to make use of existing security protocols and services. IPP will not invent new security features when the requirements described in this document can be met by existing protocols and services. Examples of such services include Transport Layer Security (TLS) and Digest Access Authentication in HTTP.

It is difficult to anticipate the security risks that might exist in any given IPP environment. For example, if IPP is used within a given corporation over a private network, the risks of exposing print data may be low enough that the corporation will choose not to use encryption on that data. However, if the connection between the client and the Printer is over a public network, the client may wish to protect the content of the information during transmission through the network with encryption.

Furthermore, the value of the information being printed may vary from one use of the protocol to the next. Printing payroll checks, for example, would have a different value than printing public information from a file.

Since we cannot anticipate the security levels or the specific threats that any given IPP print administrator may be concerned with, IPP must be capable of operating with different security mechanisms and security policies as required by the individual installation. Security policies might vary from very strong, to very weak, to none at all, and corresponding security mechanisms will be required.

2.0 Security Threats and Attacks

Before discussing security concerns specifically as they relate to IPP, it will be useful to quickly discuss and categorize security threats in a general way and discuss the means by which these threats are carried out.

2.1 Threats
Security threats fall into the following broad categories:

Resource stealing: The unauthorized use of facilities, such as printers, specific printer features, media, fonts, or logos etc. resulting in some value to the perpetrator.

Vandalism: Similar to resource stealing, but usually without gain to the perpetrator. Often results in denial of service to other authorized users.

Leakage: The acquisition of information by unauthorized interceptors during transmission.

Tampering: The interception and altering of information during transmission.

2.2 Methods of Attack

The methods by which security violations can be perpetrated depend upon obtaining access to existing communication channels or establishing channels that masquerade as connections to a user with some desired authority. These methods are:

Masquerading: Submission of print jobs or performing other IPP operations using the identity and password of another user without their authority, or by using an access token or capability after the authorization to use it has expired.

Eavesdropping: Obtaining copies of documents and job instructions without authority, either directly from the network or by examining information that is inadequately protected in storage.

Document tampering: Interception documents or other print job related information and altering their contents before passing them on to the printer or print server.

Replaying: Intercepting and storing print jobs or documents, and have them submitted again later. Example: Stock Certificate Printing. Protection against replaying requires the use of a nonce and/or time stamp.

Spamming: Sending irrelevant or nonsensical print jobs or other IPP operations to a printer or print server with the objective of overloading the system and prevent legal users to get service.

Malicious Document Content Code: Sending documents that contain malicious code which will bring the printer software into a loop or even ruin hardware components in the print device. Example: Using PostScript as a programming language to run the printer into an infinite loop.

3.0 Internet Printing Environments

It is now important to understand how the threats and attacks we have discussed above apply to the various environments in which IPP will operate.

The IPP Model encapsulates the important elements required for printing into three simple objects, the Printer, the Job, and the Document. The Printer represents the functions associated with a physical output device along with the spooling, scheduling, and multiple output device management often associated with a print server. An IPP client uses the IPP protocol to invoke operations on IPP objects on other network nodes.

The initial security needs of IPP are derived from two primary considerations. First, the printing environments described in this document take into account the fact that the client, the Printer, and the document to be printed may all exist in different security domains. When objects are in different security domains the requirements for authentication and message protection are much stronger than when they are in the same domain.

Secondly, the sensitivity and value of the content being printed will vary. For example, a publicly available document does not require the same level of privacy that a payroll document requires. There are at least two parties that have an interest in the value of the information being printed, the person asking to have the information printed and the person who originated the information. This brings into the picture the need to worry about copyrights and protection of the content.

Security attacks are now described for the following IPP environments. Where examples are provided they should be considered illustrative of the environment and not an exhaustive set. Not all of these environments will necessarily be addressed in initial implementations of IPP.

3.1 Client, Document and Printer in the same security domain

This environment is typical of the traditional office where users print the output of office applications on shared work-group printers, or where batch applications print their output on large production printers. Documents may be included in a print request or printed by reference. Even though the identity of the user may be trusted in this environment, a user might want to protect the content of a document against such attacks as eavesdropping, replaying or tampering.

3.2 Client and Printer in one security domain, Document in another

In this environment, printing can only be done by reference (If the client obtains the content prior to printing then this case defaults to the previous one). Examples of this environment include printing a document from a publicly available source on the Internet, or a copy of a contract or purchase order from a business partner, on a local Printer. In this environment the most significant security requirement is protection against unauthorized access to Documents. Furthermore, since the document crosses security domains, protection against eavesdropping and document tampering are required when the document content is sensitive.

3.3 Client and Document in one security domain, Printer in another

Examples of this environment include printing a document created by the client on a publicly available printer, such as at a commercial print shop; or printing a contract on a business partner’s printer. This latter operation is functionally equivalent to sending the contract to the business partner as a facsimile. Printing sensitive information on a Printer in a different security domain requires strong security measures. In this environment authentication of the printer is required as well as protection against unauthorized use of print resources. As in the previous case, since the document crosses security domains, protection against eavesdropping and document tampering are also required. It will also be important in this environment to protect Printers against spamming and malicious document content code.

Additional security mechanisms are required for the printer to print by reference when the document is not in it’s security domain

3.4 Printer and Document in one security domain, Client in another

Printing in this environment is by reference only. Examples would include an employee at home connecting to his office through the Internet to print a document on a printer at work, or a student using the Internet to connect to the college library and asking to have the results of a literature search printed on the library’s printer. Authentication of the printer and unauthorized use of print resources are major concerns in this environment. Protection against eavesdropping, document tampering and unauthorized access to the document are also concerns if the content is sensitive. When Printers are accessible from another security domain it will be important to protect them against spamming and malicious document content code.

3.5 Printer, Document, and Client all in different security domains

Printing in this environment is by reference only. Examples include a person at home using the Internet to print a document from a remote site, at a commercial print shop. Unauthorized access to content and to print resources is a major concern in this environment. Protection against eavesdropping, document tampering and unauthorized access to the document are also concerns if the content is sensitive. When Printers are accessible from another security domain it will be important to protect them against spamming and malicious document content code.

4.0 Security Services

Now that we have decribed the security threats that exist in the various environments in which IPP may operate, we will discuss the security services that are generally available to counter these threats. Security in general encompasses the software and hardware functionality to deliver the following services:

Authorization: Only authorized users should be able to gain access to systems, applications, data or services. Authorization may be based on authenticated identity, location, time of day, role, possession of a physical device or token, or other criterion.

Authentication: Authentication is the process of proving who a user or system is, and may apply to individual identities, roles, or groups. Authentication may be done with traditional methods such as passwords or challenge-response mechansisms, or with publicly recognized methods such as certificates.

Message Protection: Access control protects data when it is within a secure system environment. However, when data must travel outside of a secure system, such as across a public network, it needs to be protected. Message protection includes the following:

Data origin authentication guarantees that the data originates from an identified source.

Privacy protection guarantees that the data cannot be observed except by authorized parties.

Integrity protection guarantees that the data cannot be undetectably modified except by authorized parties.

Non-repudiation protection guarantees that actions taken on data cannot be denied by the subjects performing those actions.

Liability: Responsibility of the user for the printed content. This holds the user accountable for making payments, usage of special resources like transparencies, color printing, etc. The printer is also responsible for the services performed and will be held responsible for it.

Provability of Service: The printer should be able to prove that it performed correctly according to the job attributes which the client/user had indeed issued. Example: The printer should be able to prove that the job request was indeed a monochrome when the user claims it issued a color copy. Provability of service requires non-repudiation.

Payment and Accounting System: It is a mistake to charge the wrong person when someone has issued a print request.

5.0 Applying Security to IPP Operations

An IPP client uses the IPP protocol to invoke operations on remote Printer and Job objects. We now need to understand which security services are required for the various IPP operations. The IPP Operations are:

CreateJob - Create an instance of a Job object
SendDocument - Append enclosed data to a Job object
PrintJob - Print the enclosed job, with attributes
Modify - Modify the state of a Printer or Job object
Validate - Validate attributes for a specific object
GetJobs - Return job queue information for a Printer object
GetAttributes - Return attribute information for a Printer of Job object

Issue: One aspect of IPP as currently defined is that different operations are directed to different URLs, even during the life of a single print job. This means that security handshaking may have to be established for each operation independently (since it has been suggested that these operations may actually be performed on different servers). Is this okay? Is this issue significant enough that we need simplify the model in this respect?

Issue: This section exposes the potential need to have different security handshaking and levels for different operations. For example, do we need the same security level for cancelling a job as we need for submitting the job in the first place? Should the initial version of IPP assume the same level of security for any operation?

5.1 CreateJob

When creating a print job, authentication of the client and the Printer are primary security considerations. Client authentication, along with authorization, protects against unauthorized use of print resources. Printer authentication guarantees the identitity of the remote Printer.

5.2 SendDocument

When sending document content to the Printer, message protection is the primary security service required.

5.3 PrintJob

PrintJob combines the functions of CreateJob and SendDocument, therefore authentication, authorization, and message protection are all required.

5.4 ModifyJob

Currently ModifyJob is only used to cancel a job. An end user may only be allowed to cancel his or her own print jobs. Therefore authentication is required to protection against unauthorized cancellation of a job.

5.5 Validate

Validate is used to validate the attributes of a remote object. Administrators may choose to restrict the ability for certain end users to see the attributes of a Printer, so authentication and authorization are required services.

5.6 GetJobs

The level of security associated with the GetJobs operation depends on the policy set by an administrator. One common policy is for the complete job queue to be returned to anyone who asks. This policy requires no security. For more secure Printers, a common policy is to list details only on the print jobs owned by the end user, while giving little or no details about other jobs. This policy requires client authentication and authorization to match the client to the print jobs.

5.7 GetAttributes

Issue: Can an administrator also determine the level of security associated with getting the attributes of a printer?

6.0 Common Security Scenarios

As discussed early in this document,we cannot anticipate the security levels or the specific threats that any given IPP print administrator may be concerned with. Security policies might vary from very strong, to very weak, to none at all, and corresponding security mechanisms will be required. In this section we will describe what we believe to be four common scenarios.

1) no security at all
2) Message protection during transmission
3) client authentication and authorization
4) mutual authentication, authorization, and message protection

Category 1

If the server requires no authorization and the client wants no message protection the client can send the print job, i.e., the job content and the job attributes without invoking any security mechanisms. The printer will print the job for the client. Note however, when documents are not publicly accessible, print by reference requires additional security requirements not available for version 1.0.

Category 2

There are two types of security that could be used to provide message protection. These are channel security and object security. In the first case, the transport medium must be made secure by mutual authentication. Then everything between the client and server is encrypted by the transport medium.

The transport medium can be either of the following: transport layer security (TLS) or network layer security (IPSec).

In the case of object security, each object is encrypted and sent over either a secure or an insecure channel. The recipient has the corresponding key to decrypt the object and get the contents. The most widely used object security mechanisms are S/MIME, S-HTTP and PGP/MIME. S/MIME and PGP/MIME are email systems.

Category 3

The third category requires client authentication which may also be used for authorization. A user ID and password may be used for authorization purposes, and may be encrypted by the lower security layer. S/MIME and TLS are good examples of this. TLS supports both one sided and mutual authentication and can also be used for this category.

Category 4

The fourth category requires mutual authentication and message protection. TLS and SSL3 are good channel level security providers in this category.

Category Selection.

A security protocol will be used by IPP depending upon the security selection made by the client. This requires that the right handshake messages be passed. These are described in more detail in following sections.

Status of Job and Event Notification.

Issue: The following paragraph needs to be worked on. I'm concerned with the possible complexity introduced here.

For knowing the status of the job, or for performing more operations on the job, the session identifier could be reused if the call needs to be made to the same server. Otherwise the whole set of selections needs to be made, the security level can be inherited from the job submission or made independently.

Issue: Does notification require any security?

7.0 Comments on existing security technologies

TLS - Transport Layer Security: Seems OK, is near completion in the IETF and existing SSL product are probably compliant, or can be made compliant without much effort.

SSL 2 and SSL 3 - Secure Socket Layer: Proprietary solution initially by Netscape, but TLS is very close.
Cannot be used as reference in an IETF RFC.

PGP/MIME - Pretty Good Privacy MIME variant: The original PGP is widely deployed (but not much liked by the US government). The PGP/MIME version is now being worked on but is still not out, not yet stable, and not yet implemented and deployed. Timing problem.

S/MIME - Secure MIME: Currently a private implementation from RSA. Although coming out as product from a number of vendors, unlikely to make it on the

IETF standards track unless RSA decides to release their proprietary products as open standards. This is unlikely to happen in the time frame that we need.

SASL - Simple Authentication and Session Layer: This seems to be winning mind share in the IETF, but is really only a security feature negotiation protocol and does not provide any security services in itself. Hence quite limited usefulness. Also it is too new to be finished in the time frame that we need, it is not yet even an Internet-Draft from a WG.

HTTP 1.1 Security Extensions, RFC 2069: This provides some limited security services, mainly only client side authentication. It transmits a cryptographic digest derived from the username, password, and a server generated challenge.

SHTTP - Secure HTTP: Although on the IETF standards track, this seems to lack some important features and does not seem to go anywhere in the market place.

PEM - Privacy Enhanced Mail. Specified in IEF RFCs 1421-1424. It was an early standard for securing email that specified a message format and a hierarchy structure for certification authorities (CAs).

MOSS - MIME Object Security Services. Offers the same functionality as PEM, but does not force a single trust model, and allows the identification of users by names that don't have any relationship to X.500, such as E-mail addresses.

IPSec is an IETF standards track protocol for security on the IP layer. It consists of two separate mechanisms. The IP Authentication Header (AH) and the IP Encapsulating Security Payload (ESP). They can be used together or separately. The IP Authentication header provides integrity and authentication of IP datagrams. The IP Encapsulating Security Payload provides integrity, authentication and privacy. IPSec allows for either host keys or user keys to be used in security. IPSec can satisfy the IPP requirements for integrity and privacy. IPP Authentication, however, would require both IPSec use user keys and that the IPP application request use their own IPSec security association. Both requirements are recommended by IPSec but are not required.

7.1 Comparison of technologies implementing object security

Technology	**Certification structure**	**Scaleability**	**Comments**
S/MIME	Hierarchies with roles of user and certifier formalized	Scaleable from small groups to large enterprises.	Interoperability with focus on email.
PGP	Key-ring or web-of-trust	Small work groups only	Specification and application.
PEM	Hierarchy	Large enterprises. Not easy to scale downward	RFC 1421-1424. Cannot handle MIME - 7bit text only.
MOSS	Hierarchy	Scaleable.	Not inter-operable between different implementations

7.2 Specific features of various technologies:

7.2.1 S/MIME: (Secure/Multipurpose Internet Mail Extensions)

Security services and features offered:

a. Sender *Authentication* is provided using digital signatures. The recipient reads the sender’s digital signature. *Non-repudiation* of origin is also achieved using digital signatures.
b. *Privacy* (using encryption).
c. *Integrity* is achieved by using hashing to detect message tampering.
d. Provides *anonymity* by using anonymous e-mailers and gateways. The digital signature and the original message are placed in an encrypted digital envelope.
e. Supports DES, Triple-DES, RC2.
f. X.509 digital certificates supported.
g. Supports PKCS #7(cryptographic message formatting, architecture for certificate-based key management) and #10(message for certification request).

Usage, implementation and interoperability:

a. Used to securely transmit e-mail messages in MIME format.
b. Public domain mailer RIPEM available.
c. RSA’s toolkit TIPEM (Toolkit for Interoperable Privacy Enhanced Messaging) can be used to build S/MIME clients. It includes C object code for digital envelopes, digital signatures and digital certificate operations.
d. Any two packages that implement S/MIME can communicate securely.
e. Compatible with IMAP (Internet Message Access Protocol - RFC 1730).

f. S/MIME works both on the Internet or any other e-mail environment.

7.2.2 Transport Layer Security 1.0 (TLS)

TLS is a two layered protocol. The lower level TLS Record Protocol that sits on top of TCP and the TLS Handshake Protocol. The TLS Handshake protocol consists of a suite of three sub protocols which are used to allow peers to agree upon security parameters for the record layer, authenticate themselves, instantiate negotiated security parameters, and report error conditions to each other. TLS is application protocol independent. It is based on SSL v3.

Security services and features offered:

a. Privacy: (optional). Uses symmetric keys. Encryption done by the TLS Record Protocol. The keys are generated for each connection by the TLS Handshake Protocol.
b. Integrity: Using keyed MAC. Hash functions (SHA, MD5) are used for MAC computations.
c. Authentication (Both one-sided and Mutual): The TLS Handshake Protocol uses public key cryptography. Encryption algorithms are negotiated.

Usage, implementation and interoperability:

a. Interoperability: Independent applications can be developed utilizing TLS and successfully exchange cryptographic parameters without knowledge of each others code. Cannot inter-operate with SSL 3.0
b. Extensibility: New encryption methods can be incorporated as necessary.
c. Efficiency: To reduce the number of sessions that need to be established from scratch, TLS provides session caching scheme.
d. Other operations: Compression, fragmentation is done by the TLS Record Protocol.

Handshake protocol steps:

1. Exchange hello messages to agree on algorithms, exchange random values, and check for session resumption.
2. Exchange the necessary cryptographic parameters to allow the client and server to agree on a premaster secret.
3. Exchange certificates and cryptographic information to allow the client and server to authenticate themselves.
4. Generate a master secret from the premaster secret and exchanged random values.
5. Provide security parameters to the record layer.
6. Allow the client and server to verify that their peer has calculated the same security parameters and that the handshake occurred without tampering by an attacker.

7.2.3 Comparison of TLS, SSL versions 2 and 3 handshake protocols

Message direction	**TLS**	**SSL 2**	**SSL 3**
C >S	ClientHello TLS clients who wish to talk to SSL 3.0 servers should send ClientHello using SSL3 format.	ClientHello TLS clients who wish to talk to SSL 2.0 servers should send ClientHello using SSL2 format.	ClientHello SSL3 Server responds with SSL3 ServerHello to TLS clients.

S > C	ServerHello Certificate* ServerKeyExchange* CertificateRequest* ServerHelloDone	ServerHello	ServerHello Certificate* CertificateRequest* ServerKeyExchange* ServerHelloDone
C >S	Certificate* ClientKeyExchange CertificateVerify* [ChangeCipherSpec] Finished	ClientMasterKey ClientFinish	Certificate* ClientKeyExchange CertificateVerify* [ChangeCipherSpec] Finished
S > C	[ChangeCipherSpec] Finished	ServerVerify ServerFinish	[ChangeCipherSpec] Finished
C > S	Application Data	Application Data	Application Data

Note: The https protocol uses port 443 regardless of which security protocol version (TLS, SSL2, SSL3) it is using.
Star (*) indicates optional messages.

7.2.4 SASL (Simple Authentication and Security Layer)

SASL provides a method for adding authentication support to connection-based protocols. A command for identifying and authenticating a user and for (optionally) negotiating a security layer for subsequent protocol interactions is included with a protocol.

Security services and features offered:
(These are layers that SASL would call. One of these could be selected.)

1. No security
2. Integrity
3. Privacy

Security mechanisms:

1. Kerberos
2. GSS-API
3. S/Key

Handshaking protocol:

1. Client sends data
2. Server returns success* with additional data (challenge).
3. Multiple authentication (s)* (Only one - the latest security layer exists during multiple authentication).
4. Registration procedures.*

Note: SASL is not relevant for HTTP based protocols, but could be relevant to IPP, if IPP decides to define an IPP specific protocol.

6.3.5 Digest Access Authentication (rfc2069)

Digest Access Authentication is a proposed standard for weak authentication in HTTP 1.1. It is intended as a replacement for Basic Access Authentication found in HTTP 1.0. While Digest authentication is on the weak end of the security spectrum, it is a considerable improvement over the completely insecure Basic authentication.

Security services and features offered:
a. Client Authentication is provided for by a client username/password pair.

A hash of the username/password (and other information) is sent from the client to the server. How the username/password is created is outside the protocol.

b. Integrity (optional) is provided for by a hash of the entity body, username/password, selected entity headers (and other information). This can be done on either messages from the client or from the server.

c. By default, the hash uses MD5. However, there are provisions for other algorithms.

d. Digest authentication is vulnerable to replay attacks, man-in-the-middle attacks, server spoofing, and attacks on the stored password on the server. Well chosen implementations can minimize, but not eliminate the vulnerability.

Usage, implementation and interoperability:

a. This is used by web servers and clients to pass authentication information.

b. This is a proposed feature addition to HTTP 1.1. As such, it is limited to HTTP 1.1 implementations (currently a small number).

c. Different implementations have proven interoperable.

Handshake protocol steps:

a. Client asks for an access-protected object and an acceptable Authorization header is not sent.

b. The Server responds with a "401 Unauthorized" status code, and a WWW-Authenticate header. The header has the fields:

   * realm - a string indicating the context for the authorization
   * domain [optional] - a list of URIs the authentication is used for
   * nonce - a data string used in authentication
   * opaque [optional] - a data string supplied by the server
   * stale [optional] - a flag indicating the previous effort used a stale nonce
   * algorithm [optional] - a token indicating the hash algorithm to use

c. The Client then asks the User for the username/password (if needed). It then calculates the needed information and retries the request with a Authorization header. The header has the fields:

   * username - the string supplied by the user
   * realm - the value supplied by the server
   * nonce - the value supplied by the server
   * uri - the URI requested
   * response - the response hash (see below)
   * digest [optional] - the digest hash (see below), used for integrity checking
   * algorithm [optional] - the algorithm used
   * opaque - the value supplied by the server

d. If authorization is granted, the Server responds with result of query, optionally including a AuthenticationInfo header. The header has the fields:

   * nextnonce [optional] - the nonce the client should use for the next request
   * digest [optional] - the digest hash (see below) used for integrity checking.

Calculation of hashes

The response hash uses the values of username, realm, password, nonce, HTTP method, and URI. It is calculated by:

```
  response = Hash(Hash(A1) ":" nonce ":" Hash(A2))
  A1 = username ":" realm ":" password
  A2 = method ":" URI
```

The digest hash uses the values of username, realm, password, nonce, HTTP method, date, URI, content-type, content-length, content-encoding, last-modified, expires, and the entity body. The values of content-type, content-length, content-encoding, last-modified and expires are all taken from the HTTP headers, and are blank if not defined. The digest hash can be sent by either the client or the server. The digest hash is calculated by:

```
   digest = Hash(Hash(A1) ":" nonce ":" method ":" date ":" entity-info ":"
Hash(entity-body))
   entity-info = Hash(URI ":" content-type ":" content-length ":"
content-encoding ":" last-modified ":" expires)
```

8.0 References:

- A. Freier, P. Karlton and P. Kocher, "The SSL Protocol Version 3.0", Internet Draft <draft-freier-ssl-version3-01.txt>, March 1996.

- K. Hickman and T. Elgamal, "The SSL Protocol", Internet Draft <draft-hickman-netscape-ssl-01.txt> (deleted), February 1995.

- X.500: The Directory -- Overview of Concepts, Models, and Service, CCITT Recommendation X.500 and , December, 1988.

- W. Yeong, T. Howes, and S. Kille, "Lightweight Directory Access Protocol", RFC 1777, 03/28/1995. (Work is also underway in the IETF to produce an extended version of LDAP.)

- R. Rivest., The MD5 Message-Digest Algorothm, RFC 1321, April 1992.

- M. Wahl, T. Howes, S. Kille, "Lightweight Directory Access Protocol (v3)", Work in Progress, Internet Draft <draft-ietf-asid-ldapv3-protocol-03.txt>, October 22, 1996.

- J. Franks, P. Hallam-Baker, J. Hostetler, P. Leach, A. Luotonen, E. Sink, L. Stewart, “An Extension to HTTP: Digest Access Authentication”, RFC-2069, Jan 1997.

- S. Dusse, “S/MIME Message Specification”, <draft-dusse-mime-msg-spec-00.txt”, Sep. 1996.

- J. Myers, “Simple Authentication and Security Layer (SASL)”, <draft-myers-auth-sasl-10.txt>, April 1997.

- T. Dierks, C. Allen, “The TLS Protocol”, <draft-ietf-tls-protocol-02.txt>, March 24, 1997.

9.0 Author's Address

Roger deBry
HUC/003G
IBM Corporation
P.O. Box 1900
Boulder, CO 80301-9191

Jerry Hadsell
1130
IBM Corporation
Rt. 100
Somers, N.Y. 10589

Daniel Manchala
Xerox Corporation
701 Aviation Blvd.
El Segundo, CA 90245

Xavier Riley
Xerox Corporation
701 Aviation Blvd.
El Segundo, CA 90245

John Wenn
Xerox Corporation
701 Aviation Blvd.
El Segundo, CA 90245

9.0 Other Contributors

Scott Isaacson
Carl-Uno Manros