Internet Draft
PKIX Working Group
draft-ietf-pkix-ipki2opp-00.txt

Expires in 6 months March 1997

S. Boeyen (Entrust Technologies)
R. Housley (SPYRUS)
T. Howes (Netscape)
M. Myers (Verisign)
P. Richard (Xcert)

Internet Public Key Infrastructure

Part 2: Operational Protocols

<draft-ietf-pkix-ipki2opp-00.txt>

Status of this Memo

Abstract

This is the first draft of the Internet Public Key Infrastructure X.509 Operational Protocols. This document identifies candidate protocols for retrieval of X.509 v3 certificates and v2 CRLs as well as a candidate protocol for online status checking of X.509 v3 certificates. It is proposed that this document serve as the basis for the PKIX Part 2 standardization effort. Please send comments on this document to the ietf-pkix@tandem.com mail list.

1. Introduction

This specification is Part 2 of a multi-part standard for development of a Public Key Infrastructure (PKI) for the Internet. This specification addresses the requirements to provide retrieval of certificates and CRLs from an information repository. Two protocol profiles are provided to satisfy this requirement. One is based on the Lightweight Directory Access Protocol (LDAP) and the other on the File Transfer Protocol (FTP). In addition, the requirement for a user to validate the status of a certificate online, directly from a CA is addressed and supporting protocol is specified.

1.1 Model

The PKI components, as defined in PKIX Part 1, which are involved in PKIX operational protocol interactions include:

- End Entities
- Certification Authorities (CA)
- Repository

End entities and CAs retrieve certificates and/or CRLs from the repository using either the Lightweight Directory Access Protocol (LDAP) profile defined in section 2 or the File Transfer Protocol (FTP) profile defined in section 3 of this specification.

Otherwise, entities validate the status of a certificate, by communicating directly with a CA, using the Online Certificate Status Protocol (OCSP) defined in section 4 of this specification.

2. Lightweight Directory Access Protocol (LDAP)

This section examines the retrieval of information from the certificate/CRL repository and defines a subset of the LDAPv2 protocol for providing this retrieval mechanism. Two scenarios, satisfying different sets of requirements are provided in 2.1 and 2.2 below. Section 2.1 satisfies the requirement to retrieve information (a certificate, CRL, or other information of interest) from an entry in the repository, where the retrieving entity (either an end entity or a CA) has knowledge of the name of the entry. This is termed “repository read”. Section 2.2 satisfies the same requirement as 2.1 for the situation where the name of the entry is not known, but some other related information which can be used as a filter against candidate entries in the repository, is known. This is termed “repository search”.

The subset of LDAPv2 needed to support each of these functions is described below. Note that the repository search service is a superset of the repository read service in terms of the LDAPv2 functionality needed.

Note also that all tags are implicit by default in the ASN.1 definitions that follow.

2.1 LDAP Repository Read

To retrieve information from an entry corresponding to the subject or issuer name of a certificate, requires a subset of the following three LDAP operations:

```
  BindRequest (and BindResponse)
  SearchRequest (and SearchResponse)
  UnbindRequest
```

The subset of each operation is given below.

2.1.1 Bind

2.1.1.1 Bind Request

The full LDAP v2 Bind Request is defined in RFC 1777.
An application providing a LDAP repository read service MUST implement the following subset of this operation:

```
   BindRequest ::=
       [APPLICATION 0] SEQUENCE {
            version      INTEGER (2),
            name         LDAPDN, -- MUST accept NULL LDAPDN
            simpleauth [0] OCTET STRING  -- MUST accept NULL simple
            }
```

An application providing a LDAP repository read service MAY implement other aspects of the BindRequest as well.

Different services may have different security requirements. Some services may allow anonymous search, others may require authentication. Those services allowing anonymous search may only allow search based on certain criteria and not others.

A LDAP repository read service SHOULD implement some level of anonymous search access. A Public-Key Search service MAY implement authenticated search access.

2.1.1.2 BindResponse

The full LDAPv2 BindResponse is described in RFC 1777.

An application providing a LDAP repository read service MUST implement this entire protocol element, though only the following errors codes may be returned from a Bind operation:

```
  success                     (0),
  operationsError             (1),
  protocolError               (2),
  authMethodNotSupported      (7),
  noSuchObject                (32),
  invalidDNSyntax             (34),
  inappropriateAuthentication (48),
  invalidCredentials          (49),
  busy                        (51),
  unavailable                 (52),
  unwillingToPerform          (53),
  other                       (80)
```

2.1.2 Search

2.1.2.1 SearchRequest

The full LDAPv2 SearchRequest is defined in RFC 1777. An application providing a LDAP repository read service MUST implement the following subset of the SearchRequest.

```
   SearchRequest ::=
       [APPLICATION 3] SEQUENCE {
            baseObject     LDAPDN,
            scope             ENUMERATED {
                              baseObject   (0),
                                         },
            derefAliases   ENUMERATED {
                              neverDerefAliases   (0),
                                      },
            sizeLimit      INTEGER (0),
            timeLimit      INTEGER (0),
            attrsOnly      BOOLEAN, -- FALSE only
            filter         Filter,
            attributes     SEQUENCE OF AttributeType
                                }

   Filter ::=
       CHOICE {
           present        [7] AttributeType, -- "objectclass" only
              }
```

This subset of the LDAPv2 SearchRequest allows the LDAPv2 “read” operation: a base object search with a filter testing for the existence of the objectClass attribute.

An application providing a LDAP repository read service MAY implement other aspects of the SearchRequest as well.

2.1.2.2 SearchResponse

The full LDAPv2 SearchResponse is defined in RFC 1777.

An application providing a LDAP repository read service over LDAPv2 MUST implement the full SearchResponse.

2.1.3 Unbind

The full LDAPv2 UnbindRequest is defined in RFC 1777.

An application providing a LDAP repository read service MUST implement the full UnbindResponse.

2.2 LDAP Repository Search

To search for an entry in a repository containing a user's public key using arbitrary criteria requires a subset of the following three LDAP operations:

  BindRequest (and BindResponse)
  SearchRequest (and SearchResponse)
  UnbindRequest

The subset of each operation required is given below.

2.2.1 Bind

The BindRequest and BindResponse subsets needed are the same as those described in Section 2.1.1.

The full LDAP v2 Bind Request is defined in RFC 1777.

2.2.2 Search

2.2.2.1 SearchRequest

The full LDAPv2 SearchRequest is defined in RFC 1777.

An application providing a LDAP repository search service MUST implement the following subset of the SearchRequest protocol unit.

```
   SearchRequest ::=
       [APPLICATION 3] SEQUENCE {
            baseObject     LDAPDN,
            scope          ENUMERATED {
                                baseObject     (0),
                                singleLevel    (1),
                                wholeSubtree   (2)
                                      },
            derefAliases   ENUMERATED {
                                neverDerefAliases     (0),
                                      },
            sizeLimit      INTEGER (0 .. maxInt),
            timeLimit      INTEGER (0 .. maxInt),
            attrsOnly      BOOLEAN,  -- FALSE only
            filter         Filter,
            attributes     SEQUENCE OF AttributeType
                                 }
```

All aspects of the SearchRequest MUST be supported, except for the following:

- Only the neverDerefAliases value of derefAliases needs to be supported

- Only the FALSE value for attrsOnly needs to be supported

This subset provides a more general search capability. It is a superset of the SearchRequest subset defined in Section 2.1.2.1. The elements added to this service are:

- singleLevel and wholeSubtree scope needs to be supported

- sizeLimit is included

- timeLimit is included

- Enhanced filter capability

An application providing a LDAP repository search service MAY implement other aspects of the SearchRequest as well.

2.2.2.2 SearchResponse

The full LDAPv2 SearchResponse is defined in RFC 1777.

An application providing a LDAP repository search service over LDAPv2 MUST implement the full SearchResponse.

2.2.3 Unbind

An application providing a LDAP repository search service MUST implement the full UnbindRequest.

2.3 Transport

An application providing a LDAP repository read service or a LDAP repository search service MUST support LDAPv2 transport over TCP, as defined in Section 3.1 of RFC 1777.

An application providing a LDAP repository read service or a LDAP repository search service MAY support LDAPv2 transport over other reliable transports as well.

2.4 Security Considerations

For LDAP, since the elements of information which are key to the PKI service (certificates and CRLs) are both digitally signed pieces of information, no additional integrity service is required. As neither information element need be kept secret and anonymous access to such information is generally acceptable, no privacy service is required. As CAs may have access to information elements in the repository which anonymous users will not, it is recommended that even though anonymous access can be provided to end entities, CAs should bind to the repository with a minimum of simple authentication.

3. File Transfer Protocol (FTP)

Some CAs mandate the use of on-line validation services, while others distribute CRLs to allow certificate users to perform certificate validation themselves. In general, CAs make CRLs available to certificate users by posting them in the Directory. The Directory is also the normal distribution mechanism for certificates. However, Directory Services are not available in many parts of the Internet today, and the File Transfer Protocol (FTP), defined in RFC 959, offers an alternate method for certificate and CRL distribution.

Within certificate extensions and CRL extensions, URI form of GeneralName is used to specify the location where issuer certificates and CRL may be obtained. For instance, a URI identifying the subject of a certificate can be carried in subjectAltName certificate extension. An IA5String describes the

use of anonymous, or authenticated FTP to fetch certificate or CRL.
For example:

ftp://ftp.netcom.com/sp/spyrus/housley.cer
ftp://ftp.your.org/pki/id48.cer
ftp://ftp.your.org/pki/id48.no42.crl

Internet users may publish the URI reference to a file that contains their certificate on their business card. This practice is useful when there is no Directory entry for that user. FTP is widely deployed, and anonymous FTP is accommodated by many firewalls. Thus, FTP is an attractive alternative to Directory access protocols for certificate and CRL distribution.

For convenience, the names of files that contain certificates should have a suffix of ".cer". Likewise, the names of files that contain CRLs should have a suffix of ".crl".

Note that this service satisfies the the requirement to retrieve information related to a certificate which is already identified by a URI. It is not intended to satisfy the more general problem of finding a certificate for a user about whom some other information, such as their email address or corporate affiliation, is known.

4. Online Certificate Status Protocol (OCSP)

There exists a requirement for CAs to provide an Online Certificate Status Protocol (OCSP) service characterized by a high degree of availability and a rapid response time. Instances where this service would be used include those where:

- Application vendors may not implement the syntax and semantics required for standards-compliant certificate path validation.

- Application vendors who implement compliant certificate path validation logic may not implement the logic associated with periodic Certificate Revocation Lists (CRLs).

- Application vendors may find that while CRL processing is within their reach, implementing the protocols necessary to obtain CRLs from public repositories (e.g. an X.500 Directory System) is not.

- In lieu of or as a supplement to checking against a periodic CRL, it may be necessary to obtain immediate status regarding a certificate’s revocation state (cf. PKIX Part 1, Section 3.3). Examples include high-value funds transfer or the compromise of a highly sensitive key.

Two meta-level requirements factor into the specification of OCSP. First, it should be significantly easier to implement than the corresponding local CRL processing it supplements. This will enable the rapid integration of the protocol into emerging certificate-enabled applications.

Secondly, the specification of OCSP should enable rapid assimilation and deployment of the service among CA product and service vendors. Since the task of certificate management is largely unaffected by the mode of a certificate's use, it is optimal from the CA perspective that a single OCSP implementation would meet the needs of IPSEC, S/MIME, EDI and other diverse applications. Recognizing that this goal may not be achievable, the semantics of the OCSP transaction model should remain invariant against the syntactic constraints of the transport protocol used to convey the OCSP. For example, it's easily forseeable that use of SMTP as a transport model is the path of least resistance for e-mail User Agents, while HTTP is an optimal choice for Web browsers. The OCSP syntax for each may differ according to each transport protocol's usage patterns; the semantic constructs should not.

4.1 Protocol Overview

The Online Certificate Status Protocol (OCSP) enables applications to efficiently and rapidly determine the validity and revocation state of an identified certificate. An application issues a status request to the certificate issuer (CA) and suspends further certificate acceptance processing until the CA responds with a status indication.

4.1.1 Query

An OCSP query is semantically defined by the following three elements:

1. protocol version
2. service request
3. target certificate identifier

Upon receipt of a query, the CA first determines if: 1) the message is well formed, 2) the CA provides the requested service, and 3) the CA issued the subject certificate. If any one of the prior conditions are not met, an error message is produced; otherwise, a definitive response is returned.

4.1.2 Response

All definitive response messages are authenticated with the responding CA's digital signature. A definitive response message is composed of:

1. date and time of response
2. repeat of target certificate identifier
3. certificate status value
4. signature algorithm OID
5. signature computed across hash of previous four values

This specification defines the following "definitive" response values:

1. VALID
2. INVALID
3. REVOKED
4. NOT REVOKED

5 EXPIRED

Two error response semantics are defined. The first favors service availability at the expense of security. This is a “minimal” error response. The second option provides the converse balance: enhancing the authenticity of CA error responses through the use of a signed error message, although at the risk of denial of service. (The Performance Considerations and Security Considerations sections of this document provide amplifying discussions.)

The syntactic definition of a minimal error message is expected to vary by transport protocol. For example, when using HTTP to convey OCSP, a minimal error response would be a single space character. This is viewed as sufficient to inform the requesting party that, with some degree of likelihood, the CA received the message but could not return an otherwise definitive response.
Signed error messages are semantically identical to definitive response messages, extending the set of definitive response values to include the previously identified error conditions:

1 ILLFORMED MESSAGE
2 SERVICE UNAVAILABLE
3 DID NOT ISSUE CERTIFICATE

In the case of an ill-formed message, it may not be possible for the receiving CA to parse the certificate identifier from the received message. To regularize the implementation of response generation and response processing logic, a null certificate identifier is defined.

4.2 Requirements

For the purposes of requirements specification, abstract response values are indicated by UPPER CASE. A syntactic-level interpretation of these abstract values per transport protocol is provided in Section 4.3.1 of this specification.

4.2.1 Definition of Services

Certificate status service information can be organized into three categories: 1) certificate path validation; 2) current revocation status; and 3) historical revocation status. Path validation services enable applications to defer all processing associated with determining a certificate’s validity state to a trusted third party. The current revocation status service provides the means to determine whether or not an otherwise valid certificate has been revoked within the interval of its validity period and maintains this state for some limited time thereafter. The historical revocation status service extends the current revocation status service over an extended period of time beyond a certificate’s expiration.

4.2.2 Error Responses

Upon receipt of a query which fails to parse against defined OCSP semantics, the receiving CA shall respond with an error message.

If a CA provides signed error responses, a failure to parse an incoming query shall be indicated by an ILLFORMED MESSAGE response. The value of the certificate identifier of such a response shall be NULL_CERT_ID.

For service request categories not supported by a CA, the CA shall respond with an error message. If a CA provides signed error responses, non-availability of the requested service shall be indicated by a SERVICE UNAVAILABLE response.

For service request categories supported by a CA, if the receiving CA did not issue the subject certificate, the CA shall respond with an error message. If a CA provides signed error responses, this error situation shall be indicated by a DID NOT ISSUE CERTIFICATE response.

CAs shall produce a minimal error response as described in Section 2.1.2. They may provide signed error responses as described in Section 2.1.2. They should provide the option to do both. The means by which a CA signals to a relying party which error behavior is offered should be through certificate contents.

4.2.3 Required and Optional Services

CAs which offer online certificate status services shall at a minimum provide the current revocation status service defined in Section 3.1.

Upon receipt by a CA of a current revocation status request for a certificate issued by the recipient CA, the CA shall respond with either REVOKED or NOT_REVOKED, according to the certificate’s status, throughout the duration of the certificate’s validity interval and continuing for a given number days following the date of the subject certificate’s expiration. This latter interval is identified as the certificate’s “inclusion interval”. Specification of a certificate’s inclusion interval is considered a matter of a CA’s certification practices, and should be documented in the CA’s Certification Practices Statement.

If a subject certificate was not revoked prior to the expiration of its validity period, but a current revocation status request is received by its issuer within the subject certificate’s inclusion interval, the CA shall respond with a status indicating EXPIRED.

Thereafter, CAs may respond with an error message. If a CA provides signed error responses, this error situation shall be indicated by a DID NOT ISSUE CERTIFICATE response. (That is, the CA is not required to maintain online records regarding issuance beyond some well-defined interval. The automatic mechanisms that produce OCSP responses may not therefore be able to differentiate between the expiration of a certificate previously issued and a certificate that was never issued. This requirement is not intended to establish the full extent of a CA’s record-keeping obligations. The means by which CAs enable the resolution of such queries via other mechanisms and for other purposes are beyond the scope of this specification.)

CAs may extend a certificate’s inclusion interval to some arbitrarily longer period of time, thereby providing historical revocation status service. This interval is identified as a certificate’s “online status retention interval”. Specification of a certificate’s online status retention interval is considered a matter of a CA’s certification practices, and should be documented in the CA’s Certification Practices Statement. (The same caveat applies here regarding online vs. off-line records access requirements.)

Queries on certificates beyond the online status retention interval are considered by this specification to be more properly addressed by CA Archive services. Interactions with CA Archive services are beyond the scope of this specification.

CAs may provide online certificate path validation status services; they are not required to do so. However, if a CA does provide this service, then upon receipt of a path validation request for a certificate issued by the recipient CA, the CA shall respond as follows:

If the subject certificate is:	CA responds with:
within validity interval and valid:	VALID
within validity interval and invalid:	INVALID
within validity interval and revoked:	REVOKED
within inclusion interval and not revoked:	EXPIRED
within inclusion interval and revoked:	REVOKED

4.2.4 Use of PKIX AuthorityInfoAccess Extension

In order to convey to certificate-using systems a well-known point of information access, CAs that provide online certificate status services shall provide the capability to include the AuthorityInfoAccess extension (defined in PKIX Part 1, section 4.2.2.2) in certificates intended to be applied to the service.

At a minimum this extension shall contain a value for certStatus field.

Conversely, certificate-using systems shall be capable of processing the AuthorityInfoAccess extension for the purposes of obtaining the AccessDescription value of the certStatus field.

4.2.5 Required and Optional Access Methods

CAs which provide certificate status services shall provide a value for a uniformResourceIndicator (URI) accessLocation and the OID value httpID for the accessMethod in the AccessDescription SEQUENCE of the certStatus field. (The httpID OID value is defined in PKIX Part 1, Section 8: ASN.1 Structures and OIDs.)

CAs may provide additional values of AccessDescription in the certStatus field of AuthorityInfoAccess.

Certificate-using systems are not required to implement mechanisms for all values of AccessDescription.

However, to ensure the development and deployment of a globally interoperable infrastructure with the minimum number of requirements, PKIX-compliant certificate-using systems shall be capable of recognizing the httpID accessMethod and be capable of using the corresponding URI accessLocation value in accordance with the protocol syntax and semantics defined in Section 4.3.1 of this document.

The syntax, semantics and OIDs of each additional included AccessDescription syntax of certStatus shall conform to PKIX Part 1.

For each AccessDescription included in the certStatus SEQUENCE of a given certificate, the issuing CA shall ensure that: 1) all information required to obtain a certificate’s status is included in the accessLocation value; and 2), the status response is invariant with respect to the use of any AccessDescription value included in the certStatus SEQUENCE.

4.2.6 Access Method Symmetry

For each AccessDescription for which a CA provides a certificate status service, the CA shall transmit responses using the access method used to receive the correspondingly prior query. That is, queries transmitted using HTTP will result in HTTP responses; queries transmitted using SMTP will result in SMTP responses; and so forth.

Conversely, certificate-using system which initiate a query using a given access method shall be capable of receiving the corresponding response using that same access method.

4.3 Detailed Protocol

This section specifies the details of OCSP per access method. At present, only the HTTP access method is specified. Specifications of OCSP over other access methods will follow.

4.3.1 HTTP

4.3.1.1 Query Syntax

An HTTP-based OCSP query is a text-based message composed of a URL followed by a sequence of keyword-value pairs. The following loose grammar specifies the query portion of the protocol. Quoted syntactic elements are terminal elements of the grammar.

```
OCSP_query       :      url request version cert_id
url              :      protocol “://” domain_name “/”
protocol         :      “http”
request          :      service_class “/” action
service_class    :      “status”
action           :      “check”
```

```
cert_id          :      "ID" "/" hash
hash             :      hash_of(Issuer DN | cert serial number)
```

The cert_id field could be a straightforward reiteration of the Issuer DN and certificate serial number. However, OCSP should be responsive to bandwidth issues associated with high usage frequency (i.e millions of hits per day on a responding server). Backend search efficiency should be a factor as well, for exactly the same reason.

A hash of issuer DN with certificate serial number meets these needs, both reducing the bits on the wire and also providing an unstructured index useful for high speed, random access to large data repositories.

There is no cryptographic relevance to the use of a hash in OCSP queries. The requirement is production of a compact, unique identification. MD5 meets these needs and further yields fewer bits on the wire than, for example, SHA-1. Support for other hashes will require inclusion of a hash algorithm identifier, further extending the number of bits on the wire. Consequently, the OCSP query hash value shall be the base-64 representation of a hash computed using MD5.

4.3.1.2 Response Syntax

An HTTP-based OCSP response is composed of a sequence of data fields separated by a "#" character. Response codes are returned as the ASCII encoding of a decimal number. Values with a minus sign (ASCII encoding of "-") indicate definitive error values.

```
OCSP_response      :    definitive_rsp | error_rsp
definitive_rsp     :    base status_value signature_block
error_rsp          :    minimal_error | definitive_error

minimal_error      :    0x20                            // " " //
definitive_error   :    base error_value signature_block

base               :    time "#" prior_id "#"
time               :    YYYYMMDDHHMMSSZ
prior_id           :    // cert_id value of corresponding query //

error_value        :    illformed_msg | no_service | not_my_cert
illformed_msg      :    0x2d 0x31                      // "-1" //
no_service         :    0x2d 0x32                      // "-2" //
not_my_cert        :    0x2d 0x33                      // "-3" //

status_value       :    valid|invalid|revoked|not_revoked|expired
not_revoked        :    0x31                           // "1"  //
revoked            :    0x32                           // "2"  //
invalid            :    0x33                           // "3"  //
valid              :    0x34                           // "4"  //
expired            :    0x35                           // "5"  //

signature_block    :    sig_alg_oid "#" signature
sig_alg_oid        :    // OID used to generate signature //
```

```
signature          :    // base-64 encoded value corresponding to
                           the result of using sig-alg-oid //
```

To produce a signed response, the responder first calculates a hash across the to-be-transmitted sequence

```
            { time#prior_id#response_value#sign_alg_oid# },
```

signs the hash using the algorithm indicated by sig_alg_oid, base-64 encodes the result and then concatenates it to the prior fields.

4.4 Performance Considerations

Performance considerations may motivate the use of a cache on the status server end to retain recently retrieved state information. When doing so, the effect of cache refresh rates need to be considered. It is possible when using such an approach to reduce the timeliness of the certificate status service to that approaching periodic CRL distribution.

4.5 Security Considerations

For this service to be effective, certificate using system must connect to the certificate status service provider. In the event such a connection cannot be obtained, certificate-using systems should implement CRL processing logic as a fall-back position.

A denial of service vulnerability is evident with respect to a flood of queries constructed to produce error responses. The production of a cryptographic signature significantly affects response generation cycle time, thereby exacerbating the situation. Performance studies on a preliminary implementation of OCSP capable of handling two million hits per day without degradation suggest this effect is of an orders of magnitude per response. Unsigned error responses provide a reasonable tradeoff against protection against this particular attack.

The use of unsigned error messages introduces a vulnerability to intermediation attacks. It is reasonable to ask for error messages to be signed to address this vulnerability. A request to do so however must also consider the converse risk identified above— namely that increasing the response cycle time of error messages through use of cryptographic signing increases the impact of flooding attacks. CAs that wish to offer to their relying parties the benefit of signed error responses should strongly consider the use of hardware-assisted cryptography. Do so will reduce the threat of flood attacks.

To mitigate the effects of replay attacks (by using previously signed responses), applications should match the certificate identifier and time field of the incoming response to the previous query before acting on the response.

Finally, the results delivered to the certificate acceptance decision function may be mediated by one or more software components which provide no explicit means to establish or maintain a trusted path. Ultimately, the relying party (or, in the case of automated machine processing, the owner/operator of a router, Web Server, X.400 MTA, etc.) is responsible for placing trust in the results.

Author Addresses:

Sharon Boeyen
Entrust Technologies
2 Constellation Crescent, Nepean
P.O. Box 3511,Station C
Ottawa, Ontario
Canada K1Y 4H7
boeyen@entrust.com

Russell Housley
SPYRUS
PO Box 1198
Herndon, VA 20172
USA
housley@spyrus.com

Tim Howes
Netscape Communications Corp.
501 E. Middlefield Rd.
Mountain View, CA 94043
USA
howes@netscape.com

Michael Myers
VeriSign Inc.
2593 Coast Avenue
Mountain View, CA 94043
USA
myers@psn.net

Patrick Richard
Xcert Software Inc.
Suite 1001, 701 W. Georgia Street
P.O. Box 10145
Pacific Centre
Vancouver, B.C.
Canada V7Y 1C6
patr@xcert.com