ICS 35.100.70
L 79

中华人民共和国国家标准

GB/T 19713—2005

信息技术 安全技术 公钥基础设施 在线证书状态协议

Information technology—Security techniques—Public key infrastructure—Online certificate status protocol

2005-04-19 发布　　2005-10-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

本标准主要参考 IETF(互联网工程特别工作组)RFC2560 文件制定，其中对某些功能项的实施方法，结合实际经验提出了一些特别的建议。

本标准中凡涉及密码算法相关内容，按国家有关法规执行。

本标准的附录 B 为规范性附录，附录 A 为资料性附录。

本标准由中华人民共和国信息产业部提出。

本标准由全国信息安全标准化技术委员会(TC 260)归口。

本标准主要起草单位：国家信息安全基础设施研究中心、国家信息安全工程技术研究中心、中国电子技术标准化研究所、国瑞数码安全系统有限公司。

本标准主要起草人：顾青、吴志刚、邓琳、陈刚、王于、苏恒、李跃、黄峰、郭晓雷、袁文恭、李丹、上官晓丽、王利。

信息技术　安全技术　公钥基础设施　在线证书状态协议

1　范围

本标准规定了一种无需请求证书撤销列表(CRL)即可查询数字证书状态的机制(即在线证书状态协议——OCSP)。该机制可代替 CRL 或作为周期性检查 CRL 的一种补充方式,以便及时获得证书撤销状态的有关信息。本标准主要描述了以下内容:

a)　具体描述了在线证书状态协议的请求形式;

b)　具体描述了在线证书状态协议的响应形式;

c)　分析了处理在线证书状态协议响应时可能出现的各种异常情况;

d)　说明了在线证书状态协议基于超文本传输协议(HTTP)的应用方式;

e)　提供了采用抽象语法记法 1(ASN.1)描述的在线证书状态协议。

本标准适用于各类基于公开密钥基础设施的应用程序和计算环境。

2　规范性引用文件

下列文件中的条款通过本标准的应用而成为本标准的条款。凡是注日期的引用文件,其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本标准,然而,鼓励根据本标准达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件,其最新版本适用于本标准。

ISO/IEC 8824-1:2002　抽象语法记法一(ASN.1)第 1 部分:基本记忆规范

RFC 2459　因特网 X.509 公开密钥基础设施证书和证书撤销列表框架

RFC 2616　超文本传输协议(HTTP 1.1)

3　术语和定义

下列术语和定义适用于本标准。

3.1

证书扩展项　extensions

在证书的结构中,该域定义了证书的一些扩展信息。

3.2

证书序列号　certificate serial number

为每个证书分配的唯一整数值,在 CA 颁发的证书范围内,此整数值与该 CA 所颁发的证书相关联一一对应。

3.3

请求者　requester

申请在线证书状态查询服务的主体。

3.4

响应者　responder

提供在线证书状态查询服务的主体。

4　缩略语

下列缩略语适用于本标准。

CA　证书认证机构
CAs　证书认证机构群体
CRL　证书撤销列表
HTTP　超文本传输协议
OCSP　在线证书状态协议
OID　对象标识符
PKI　公钥基础设施

5　总则

5.1　概述

OCSP 作为检查定期 CRL 的替代方法或补充方法，在必须及时获得证书撤销状态的相关信息的情况下，是必不可少的。

OCSP 能够使应用程序确定某个标识证书的(撤销)状态。OCSP 可以用来满足那些需要提供比检查 CRL 更及时的撤销信息的操作要求，还可以用来获得附加的状态信息。OCSP 请求者向 OCSP 响应器发出一个状态请求，并延缓接受待查询的证书，直到响应器提供响应为止。

本标准规定了检查证书状态的应用程序和提供证书状态查询的服务器之间需要交换的数据。

5.2　请求

OCSP 请求包含以下数据：

a)　协议版本；
b)　服务请求；
c)　目标证书标识符；
d)　OCSP 响应器可处理的可选扩展，比如：OCSP 请求者的签名、随机数。

对某个请求的回应，OCSP 响应器应确定：

a)　报文格式是否正确；
b)　响应器是否配置了所要求的服务；
c)　请求是否包含响应器需要的信息。

如果上述任何一个条件未满足，则 OCSP 响应器将发出一个错误信息；否则，将返回一个明确的响应。

5.3　响应

OCSP 响应可以有不同类型，且响应由响应类型和响应实体两部分组成。本标准只规定了一种所有 OCSP 请求者和响应器都必须支持的 OCSP 基本响应类型。

对所有明确的响应报文都应进行数字签名。用于响应签名的密钥必须满足下列条件之一：

a)　签发待查询证书的 CA；
b)　可信赖的响应器，即请求者信任该响应器的公钥；
c)　CA 指定的响应器(即授权的响应器)，该响应器拥有一个 CA 直接发布的带有特殊标记扩展项的证书，该特殊标记扩展项指明该响应器可以为 CA 发布 OCSP 响应。

明确的响应消息由如下内容组成：

a)　响应语法的版本；
b)　响应器的名称；
c)　对请求中每个证书的响应；
d)　可选择的扩展；
e)　签名算法的 OID；
f)　响应的哈希签名。

对请求中每个证书的响应由如下内容组成：

a) 目标证书标识符；

b) 证书状态值；

c) 响应有效期限；

d) 可选的扩展。

本标准对证书状态值规定了如下明确的响应标识符：

a) good(好)：表示对状态查询的肯定响应。如果客户端没有使用时间戳服务器和有限期验证的安全策略，肯定的响应只能说明证书未被撤销，但并不能说明证书已被发布或产生响应的时间是在证书有效性时间范围内。响应扩展可用于传输响应器作出的关于证书状态信息的附加声明，例如发布的肯定声明、有效性等。

b) revoked(已撤销)：表示证书已被(永久地或临时地)撤销。

c) unknown(未知)：表明响应器不能鉴别待验证状态的证书(包括 OCSP 响应器证书与待验证状态的证书不是同一 CA 签发的情况)。

5.4 异常情况

当出现错误时，OCSP 响应器返回某个错误消息。这些消息不能被签名。错误可以是下列几类：

a) malformedRequest(不完整的申请)：OCSP 响应器(服务器)接收到的请求没有遵循 OCSP 语法；

b) internalError(内部错误)：OCSP 响应器处于非协调的工作状态，应当向另一个响应器再次进行询问；

c) tryLater(以后再试)：OCSP 响应器正处于运行状态，不能返回所请求证书的状态，即表明存在所需的服务，但是暂时不能响应；

d) sigRequired(需要签名)：响应器要求请求者对请求签名；

e) unauthorized(未授权)：该查询是由未授权请求者向响应器提出的。

5.5 thisUpdate、nextUpdate 和 producedAt 的语义

各响应中可包含三个时间字段，即 thisUpdate、nextUpdate 和 producedAt。这些字段的语义分别是：

a) thisUpdate：此次更新时间，所要求指明的状态是正确的时间；

b) nextUpdate：下次更新时间，表示证书在此时间之前，状态是正确的，并且在此时间可以再次获得证书状态更新的信息；

c) producedAt：签发时间，OCSP 响应器签署该响应的时间。

如果未设置 nextUpdate，响应器要指明随时可以获得更新的撤销信息。鉴于 OCSP 是实时更新有关证书的状态，本标准建议可不设置 nextUpdate 字段；thisUpdate 定义为 CA 签发证书状态的时间；producedAt 为 OCSP 签发响应的时间。

5.6 预产生响应

为说明某一特定时刻内某些证书的状态，OCSP 响应器可以预先生成某些签名响应。证书状态被认为合法的特定时刻应该就是响应的 thisUpdate 字段；关于状态再次更新的时间应反映在响应中的 nextUpdate 字段；而产生响应的时间应反映在响应中的 ProduceAt 字段。

本标准建议：鉴于 OCSP 在线服务，强调其实时性，并防重放攻击，建议可不采用预产生响应。预响应也不能对客户请求时产生的随机数作出反应。

5.7 OCSP 签名机构的委托

签署证书状态信息的密钥不必与签署证书的密钥相同。证书的发布者通过发布一个含有 extendedKeyUsage 唯一值的证书，并作为 OCSP 响应器的签名证书，来明确指派 OCSP 响应器签名机构。此证书必须由认可的 CA 直接签发给响应器。

5.8 CA 密钥泄漏

如果 OCSP 响应器知道一个特定 CA 的私钥已被泄漏，则它可以使所有由该 CA 发布的证书都返回撤销状态。

6 功能要求

6.1 证书内容

为向 OCSP 客户端请求者提供 OCSP 响应器的访问位置，CA 应该在证书扩展项中提供用于 OCSP 访问的 AuthorityInfoAccess 值(访问授权信息)(详见 RFC2459 中的 4.2.2.1)；或者，OCSP 响应器的 accessLocation(访问地址)可在 OCSP 客户端进行本地配置。

提供 OCSP 服务的 CA，不管是在本地实现还是由指定的 OCSP 响应器实现，都必须在 AccessDescription SEQUENCE 中包含 uniformResourceIndicator (URI) accessLocation 值和对象标识符 id-ad-ocsp。

主体证书中的 accessLocation(访问地址)字段的值详细说明了访问 OCSP 响应器的信息传输路径(如 HTTP)，也可以包含其他的信息(如一个 URL)。

6.2 签名响应的接收要求

a) 在把 OCSP 响应视作有效之前，OCSP 客户端应确认；

b) 收到的响应中所鉴别的证书应和请求中的证书一致；

c) 响应方的签名是有效的；

d) 响应方的签名者身份应和请求的预定接收者保持一致；

e) 签名者已被授权对响应进行签名；

f) 指明证书状态的时间(thisUpdate)应为当前最近的时间；

g) 如果设置了 nextUpdate 字段，此时间应该晚于客户端当前时间。

7 具体协议

7.1 约定

本标准采用抽象语法记法 1(ASN.1)来描述具体协议内容，ASN.1 语法引用一些 RFC 2459 定义的术语，完整的 OCSP 协议描述见附录 B。支持 HTTP 的 OCSP 请求格式和响应格式见 RFC2616 和参见附录 A。对于签名计算来说，要签名的数据是用 ASN.1 的可辨别编码规则(DER)来编码的。

如果无特殊说明，默认使用 ASN.1 显式标记。

引用的其他术语还有：Extensions，CertificateSerialNumber，SubjectPublicKeyInfo，Name，AlgorithmIdentifier，CRLReason。

7.2 请求

本条规定了确定请求的 ASN.1 规范。根据所使用的传输机制(HTTP、SMTP、LDAP 等)，实际的消息格式可能会发生相应的变化。

7.2.1 请求语法

```
OCSPRequest      ::=    SEQUENCE {
tbsRequest                  TBSRequest,
optionalSignature  [0]      EXPLICIT Signature OPTIONAL }

TBSRequest       ::=    SEQUENCE {
version            [0]      EXPLICIT Version DEFAULT v1,
requestorName      [1]      EXPLICIT GeneralName OPTIONAL,
requestList                 SEQUENCE OF Request,
```

```
requestExtensions    [2]          EXPLICIT Extensions OPTIONAL }

Signature            ::=          SEQUENCE {
signatureAlgorithm                AlgorithmIdentifier,
signature                         BIT STRING,
certs                [0]          EXPLICIT SEQUENCE OF Certificate
OPTIONAL}

Version              ::=          INTEGER { v1(0) }

Request              ::=          SEQUENCE {
reqCert                               CertID,
singleRequestExtensions               [0] EXPLICIT Extensions OPTIONAL }

CertID               ::=          SEQUENCE {
hashAlgorithm                     AlgorithmIdentifier,
issuerNameHash                    OCTET STRING, ——发布者名称的哈希
issuerKeyHash                     OCTET STRING, ——发布者公开密钥的哈希
serialNumber                      CertificateSerialNumber }
```

issuerNameHash 是发布者唯一名称的哈希值。该值对所检查证书的发布者名称字段的 DER 编码进行计算。issuerKeyHash 是发布者公钥的哈希值。该值将通过对发布者证书中的主体公钥字段(不含标记和长度)进行计算。hashAlgorithm 字段用来指明这些哈希计算所使用的哈希算法。serialNumber 是请求其状态的证书的序列号。

7.2.2 请求语法的注解

既使用 CA 公开密钥的哈希值又使用 CA 名称的哈希值来标识某个发布者的主要原因,是两个 CA 可能使用相同的名称(虽然推荐名称的唯一性,但并不强制)。但是,两个 CA 的公开密钥是不可能相同的,除非两者都决定共享私钥,或一方的密钥发生泄漏。

对任何特殊扩展域的支持是个可选项。不应将这些扩展域设置为关键性的。7.5 提出了许多有用的扩展域。在其他标准中会定义其他的附加扩展域。必须忽略那些不能识别的扩展域(除非它们有关键性的标志并且不被理解)。

请求者可以选择对 OCSP 请求进行签名。这种情况下,将针对 tbsRequest 结构来验证签名。如果请求被签名,请求者应在 requestorName 中指定其名称。同时,对于已签名的请求,请求者可在 Signature 的 certs 字段中包含有助于 OCSP 响应器验证请求者签名的那些证书。

7.3 响应

本条规定了确定响应的 ASN.1 规范。根据所使用的传输机制(HTTP、SMTP、LDAP 等),实际的消息格式可能会发生相应的变化。

7.3.1 响应语法

一个 OCSP 响应至少由一个指明先前请求的处理状态的 responseStatus 字段构成。如果 responseStatus 的值是某个错误条件,则不设置 responseBytes。

```
OCSPResponse ::= SEQUENCE {
        responseStatus          OCSPResponseStatus,
        responseBytes           [0] EXPLICIT ResponseBytes OPTIONAL }
```

```
OCSPResponseStatus ::= ENUMERATED {
        successful              (0), ——响应被有效确认
        malformedRequest        (1), ——非法确认请求
        internalError           (2), ——发布者内部错误
        tryLater                (3), ——稍候重试
        sigRequired             (4), ——必须对请求签名
        unauthorized            (5),——请求未被授权
}
```

responseBytes 的值由一个对象标识符和响应语法组成，该响应语法由按照 OCTECT STRING 编码的 OID 来标识。

```
ResponseBytes ::=       SEQUENCE {
        responseType    OBJECT IDENTIFIER,
        response        OCTET STRING }
```

对于基本的 OCSP 响应器，responseType 应为 id-pkix-ocsp-basic.

```
id-pkix-ocsp            OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic      OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
```

OCSP 响应器应能产生 id-pkix-ocsp-basic 类型的响应。相应地，OCSP 客户端应有能力接受和处理此类响应。

response 的值应为 BasicOCSPResponse 的 DER 编码。

```
    BasicOCSPResponse       ::= SEQUENCE {
    tbsResponseData         ResponseData,
    signatureAlgorithm      AlgorithmIdentifier,
    signature               BIT STRING,
    certs                   [0] EXPLICIT SEQUENCE OF Certificate OPTIONAL }
```

signature 的值应该基于 DER 编码的 ResponseData 哈希值计算得到。

```
    ResponseData ::= SEQUENCE {
    version                 [0] EXPLICIT Version DEFAULT v1,
    responderID                 ResponderID,
    producedAt                  GeneralizedTime,
    responses                   SEQUENCE OF SingleResponse,
    responseExtensions      [1] EXPLICIT Extensions OPTIONAL }

    ResponderID ::= CHOICE {
    byName                  [1] Name,
    byKey                   [2] KeyHash }
```

KeyHash ::= OCTET STRING——响应器公开密钥的 SHA-1 哈希
(不包括 tag 和 length 字段)

```
    SingleResponse ::= SEQUENCE {
    certID                      CertID,
    certStatus                  CertStatus,
```

```
        thisUpdate                      GeneralizedTime,
        nextUpdate          [0]         EXPLICIT GeneralizedTime OPTIONAL,
        singleExtensions    [1]         EXPLICIT Extensions OPTIONAL }

    CertStatus ::= CHOICE {
        good                [0]         IMPLICIT NULL,
        revoked             [1]         IMPLICIT RevokedInfo,
        unknown             [2]         IMPLICIT UnknownInfo }

    RevokedInfo ::= SEQUENCE {
        revocationTime                  GeneralizedTime,
        revocationReason    [0]         EXPLICIT CRLReason OPTIONAL }
```

UnknownInfo ::= NULL——此处可用枚举替代

7.3.2 响应语法的注解

7.3.2.1 时间

thisUpdate 和 nextUpdate 两个字段定义了有效时间间隔。这个时间间隔是和 CRLs 中的{thisUpdate,nextUpdate}间隔相对应。NextUpdate 值比本地系统时间早的响应应被认为无效。ThisUpdate 值比本地系统时间晚的响应也应被认为无效。未提供 nextUpdate 值的响应和未提供 nextUpdate 值的 CRL 意义相同。

producedAt 时间是响应被签名的时间。

7.3.2.2 授权的响应器

对证书状态信息签名的密钥和签发证书的密钥不必相同。但必须确保对该信息进行签名的实体是经过授权的。因此,证书的签发者直接对 OCSP 响应签名,或明确地指派授权给另一个实体对 OCSP 响应签名。CA 通过在 OCSP 响应器证书的 extendedKeyUsage 扩展中包含 id-kp-OCSPSigning 来指派 OCSP 响应器对响应进行签名。OCSP 响应器证书必须直接由 CA 发布,该 CA 还发布了需要验证状态的证书。

id-kp-OCSP Signing OBJECT IDENTIFIER ::= {id-kp 9}

依赖于 OCSP 响应的系统或应用必须能够检测并使用上述的 id-ad-ocspSigning 值,它们可以提供一种方法在本地配置一个或多个 OCSP 签名权威实体以及信任这些权威实体的 CA。如果用来验证响应上的签名所需的证书不能满足以下任何标准,响应必须被拒绝:

a) 本地配置的 OCSP 签名权威实体中包含了与待验证状态的证书相匹配的证书;
b) 或是签发待验证状态证书的 CA 证书;
c) 在 extendedKeyUsage 扩展中含有 id-ad-ocsp Signing 值,并且由签发待验证状态证书的 CA 发布。

附加的接受或拒绝标准可以应用于响应自身,或应用于验证响应签名的证书。

7.3.2.2.1 授权响应器的撤销检查

既然一个授权权威 OCSP 响应器可以为一个或多个 CA 提供状态信息,OCSP 客户端就需要知道如何去检查授权权威响应器的证书是否已被撤销。CA 可任选以下三种方法之一来解决这个问题:

a) CA 可指定 OCSP 客户端在响应器证书的整个生存期内信任该响应器。CA 通过在响应器证书中包含 id-pkix-ocsp-nocheck 扩展来完成该指定。这应该是一个非关键性的扩展。扩展值应为空。至少对于证书的有效期而言,发布这样一个证书的 CA 应认识到响应器密钥的泄密同用来签发 CRL 的 CA 密钥的泄密所带来的后果一样严重。CA 可以选择发布一种有效期很

短并且经常更新的证书，也就是短生命周期的证书。

id-pkix-ocsp-nocheck OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }

b) CA 可以指定如何检查响应器证书是否已被撤销。假如是使用 CRLs 或 CRL 分布点来检查的话，就能够使用 CRL 的分布点来完成，假如是用其他的方法来检查，就要用到权威实体信息访问(AuthorityInfoAccess 扩展)。在 RFC2459 中有这两种机制的详细说明。

c) CA 可选择不指定检查响应器证书是否已被撤销的方法。在此情况下，将遵循 OCSP 客户端的本地安全策略来决定是否做这项检查工作。

7.4 强制的密码算法和可选的密码算法

请求 OCSP 服务的客户端应能够处理已签名的响应，响应由 DSA sig-alg-oid(RFC2459 的 7.2.2 中指定)鉴别的 DSA 密钥签名。OCSP 响应器应支持散列算法。在国内应用时，应使用国家密码管理主管部门审核批准的相关算法。

7.5 扩展

本条定义了一些标准的扩展，这些扩展基于 X.509 的 V3 版本证书(见 RFC2459)中使用的扩展模式。对客户端和响应器而言，对所有扩展的支持都是可选的。对于每个扩展，定义指出了它被 OCSP 响应器处理时的语法，以及任何包含在相应响应中的扩展。

7.5.1 Nonce(现时)

Nonce 通过秘密地加密绑定一个请求和一个响应，以防止重放攻击。Nonce 在请求中作为请求包中的一个 requestExtensions 而包括在请求中，然而在响应中，它将作为响应包中的一个 responseExtensions 包括在响应中。在请求和响应中，Nonce 将由对象标识符 id-pkix-ocsp-nonce 标识，extnValue 中包含了 Nonce 的值。

id-pkix-ocsp-nonce OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }

7.5.2 CRL 参考

对于 OCSP 响应器来说，在 CRL 上指出一个已撤销的或在用的证书，可能更有价值。这一点在 OCSP 作为存储库和审计机制使用时非常有用。CRL 可能由一个 URL(CRL 可以从这个 URL 处获得)，一个序列号(CRL 序列号)或一个时间点(产生相应的 CRL 的时间点)指定。这些扩展被确定为 singleExtensions。该扩展的标识符为 id-pkix-ocsp-crl，值为 CrlID。

```
id-pkix-ocsp-crl      OBJECT   IDENTIFIER ::= { id-pkix-ocsp 3 }
CrlID ::= SEQUENCE {
   crlUrl             [0]      EXPLICIT IA5String OPTIONAL,
   crlNum             [1]      EXPLICIT INTEGER OPTIONAL,
   crlTime            [2]      EXPLICIT GeneralizedTime OPTIONAL }
```

选择项 crlUrl 指定适用于 CRL 的 URL，它的类型是 IA5String；选择项 crlNum 指定相关 CRL 的 CRL 序列号扩展的值，它的类型是 INTEGER；选择项 crlTime 指定发布相应 CRL 的时间点，它的类型是 GeneralizedTime。

7.5.3 可接受的响应类型

一个 OCSP 客户端可以希望指定其能理解的各种响应类型。为了达到这样的目的，它应该包含 id-pkix-ocsp-response 扩展，值为 AcceptableResponses。该扩展作为请求中的一个 requestExtensions。包含在 AcceptableResponses 中的 OIDs 是该客户端能够接受的各种响应类型(例如：id-pkix-ocsp-basic)的 OIDs。

id-pkix-ocsp-response OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }

AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER

如 7.3.1 所述，OCSP 响应器将能够产生 id-pkix-ocsp-basic 类型的响应。相应的，OCSP 的客户端也将能接收和处理 id-pkix-ocsp-basic 类型的响应。

7.5.4 存档截止

OCSP 响应器可以选择在证书过期后仍保留相应的撤销信息。从响应的 producedAt 时间减去间隔保持值而得到的时间定义为证书的"存档截止"时间。

即便是所要验证有效签名的证书很久以前就过期了，激活的 OCSP 应用也能使用 OCSP 存档截止时间，来提供数字签名是否有效的证明。

提供历史参考支持的 OCSP 服务器应该在响应包里面包括存档截止扩展。如果包括存档截止扩展，将把这个值作为 OCSP 的 singleExtensions 扩展，并由 id-pkix-ocsp-archive-cutoff 和 GeneralizedTime 语法来识别。

id-pkix-ocsp-archive-cutoff OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }

ArchiveCutoff ::= GeneralizedTime

举例，如果服务器的操作具有采用 7 年保留期的策略，且 produceAt 值为 t_1，那么响应中 ArchiveCutoff 值为(t_1－7 年)。

7.5.5 CRL 入口扩展域

所有指定为 CRL 入口扩展的扩展——见 RFC2459 的 5.3。

7.5.6 服务定位器

一台 OCSP 服务器也许以这样一种模式运作，服务器接收到一个请求，并转发该请求到能识别待验证证书的 OCSP 服务器上。为此定义了 serviceLocator 请求扩展。这个扩展作为一个 singleRequestExtensions 包括在请求中。

id-pkix-ocsp-service-locator OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }

ServiceLocator ::= SEQUENCE {

issuer Name,

locator AuthorityInfoAccessSyntax OPTIONAL }

这些字段的值可从主体证书的相应字段中获得。

8 安全考虑

为使服务有效，证书使用系统必须与证书状态服务提供者相连接。当不能获得此连接时，证书使用系统可以执行一个 CRL 处理逻辑，作为一种后备的处理手段。

一种拒绝服务攻击很明显是由于对服务器的大量查询引起的。密码签名的计算严重影响了应答产生的周期时间，因而更加加剧了这一情形。同时，未签名的错误响应可使协议遭受另外一种拒绝服务攻击，就是攻击者发送大量的虚假错误的应答。

预产生响应的使用给重放攻击提供了机会，一个以前的(好的)应答在证书已经被撤销之后但又在它过期之前被重放会导致重放攻击。部署 OCSP 响应器应当仔细地在预产生响应所带来的益处和重放攻击发生的可能性之间，以及重放攻击成功执行后所造成的损失和预防重放攻击相应的花费之间作出权衡。

请求中不包括目标应答器的任何信息，这就给攻击者将请求重放发送给任何可能的 OCSP 响应器提供了可能。

如果没有正确配置中间级服务器，或缓存管理出错，那么对 HTTP 高速缓存的信赖，可能导致一些意外的结果。在部署 OCSP OVER HTTP 时，本标准建议实施者把 HTTP 缓存机制的可靠性考虑进去。

附 录 A
（资料性附录）
HTTP 上的 OCSP

本附录描述了支持 HTTP 的 OCSP 请求格式和 OCSP 响应格式。

A.1 请求

基于 HTTP 的 OCSP 请求可以使用 GET 或 POST 方法来提交。为了使得 HTTP 缓存生效，较小的请求（经过编码后小于 255 字节）可以用 GET 方法来提交。如果 HTTP 缓存不是很重要，或者请求大于或等于 255 字节，那么请求应该用 POST 方法来提交。在保密性是一个重要需求的时候，基于 HTTP 的 OCSP 会话可以用 TLS/SSL 或其他底层的协议来保障其安全性。

一个使用 GET 方法的 OCSP 请求按如下方式进行构造：

GET {url}/{url-encoding of base-64 encoding of the DER encoding of the OCSPRequest}

其中 {url} 可以从 AuthorityInfoAccess 的值或者 OCSP 客户端的本地配置获得。

一个使用 POST 方法的 OCSP 请求按如下方式进行构造：

Content-Type 头具有值："*application/ocsp-request*"，而消息体是 OCSPRequest 的 DER 编码的二进制值。

本标准建议，实施者只使用 POST 方法，使用 POST 方法还可避免 HTTP 缓存机制带来的麻烦。客户端必须对请求进行签名。

A.2 响应

一个基于 HTTP 的 OCSP 响应由以下方式定义：

Content-Type 头具有值："application/ocsp-response"，Content-Length 头应该指定响应的长度，而消息体是 OCSPResponse 的 DER 编码的二进制值。其他的不能被客户端识别的 HTTP 头可能存在于响应中，可以被忽略。

注：因业务需求，可能有甲网用户访问乙网用户的 OCSP 服务的情况，可将有关交叉认证的协议融入其中。为了方便实现，访问者证书中应有其签发者的 KeyID，这样在响应器中的 CA 信任链中能更准确的找到验证它的上级证书。

附 录 B
（规范性附录）
采用 ASN.1 定义的 OCSP

```
OCSP DEFINITIONS EXPLICIT TAGS::=
BEGIN
IMPORTS
    --Directory Authentication Framework (X.509)
      Certificate, AlgorithmIdentifier, CRLReason
      FROM AuthenticationFramework { joint-iso-itu-t ds(5)
      module(1) authenticationFramework(7) 3 }
    --PKIX Certificate Extensions
      AuthorityInfoAccessSyntax
      FROM PKIX1Implicit88 {iso(1) identified-organization(3)
      dod(6) internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-pkix1-implicit-88(2)}
      Name, GeneralName, CertificateSerialNumber, Extensions,
      id-kp, id-ad-ocsp
      FROM PKIX1Explicit88 {iso(1) identified-organization(3)
      dod(6) internet(1) security(5) mechanisms(5) pkix(7)
      id-mod(0) id-pkix1-explicit-88(1)}
    --Cryptographic Message Syntax (CMS)
      IssuerAndSerialNumber
      FROM { iso(1) member-body(2) us(840) rsadsi(113549)
      pkcs(1) pkcs-9(9) smime(16) modules(0) cms-2001(14)}
      OCSPRequest      ::=     SEQUENCE {
      tbsRequest                   TBSRequest,
    optionalSignature   [0]   EXPLICIT Signature OPTIONAL }
     TBSRequest      ::=   SEQUENCE {
     version            [0]   EXPLICIT Version DEFAULT v1,
     requestorName      [1]   EXPLICIT GeneralName OPTIONAL,
     requestList              SEQUENCE OF Request,
     requestExtensions  [2]   EXPLICIT Extensions OPTIONAL }
     Signature       ::=      SEQUENCE {
     signatureAlgorithm  AlgorithmIdentifier,
     signature           BIT STRING,
     certs               [0]   EXPLICIT Certificates OPTIONAL }
     Version   ::=  INTEGER { v1(0), v2(1) }
     Request   ::=      SEQUENCE {
     reqCert                          ReqCert,
     singleRequestExtensions          [0] EXPLICIT Extensions OPTIONAL }
     Certificates     ::=     SEQUENCE SIZE(1..MAX) of Certificate
     ReqCert     ::=   CHOICE {
```

```
certID                          CertID,
fullCert                        [0] FullCertificate,
certIdWithSignature             [1] CertIdWithSignature }
CertID ::= SEQUENCE {
hashAlgorithm                   AlgorithmIdentifier,
issuerNameHash                  OCTET STRING,—— Hash of Issuer's DN
issuerKeyHash                   OCTET STRING,—— Hash of Issuers public key
serialNumber                    CertificateSerialNumber }
FullCertificate ::= CHOICE {
certificate                     [0]   Certificate,
attributeCert                   [1]   AttributeCertificate }
CertIdWithSignature ::= SEQUENCE {
issuerandSerialNumber           IssuerandSerialNumber,
tbsCertificateHash              BIT STRING,
certsignature                   CertSignature
}
CertSignature ::= SEQUENCE {
signatureAlgorithm              AlgorithmIdentifier,
signatureValue                  BIT STRING
}
OCSPResponse ::= SEQUENCE {
responseStatus                  OCSPResponseStatus,
responseBytes                   [0] EXPLICIT ResponseBytes OPTIONAL }
OCSPResponseStatus ::= ENUMERATED {
successful                      (0), ——Response has valid confirmations
malformedRequest                (1), ——Illegal confirmation request
internalError                   (2), ——Internal error in issuer
tryLater                        (3), ——Try again later
                                (4), —— is not used
sigRequired                     (5), ——Must sign the request
unauthorized                    (6), ——Request unauthorized
badCRL                          (8), ——Error in CRL processing
}
ResponseBytes ::=               SEQUENCE {
responseType                OBJECT IDENTIFIER,
response                    OCTET STRING }
BasicOCSPResponse                   ::= SEQUENCE {
tbsResponseData                 ResponseData,
signatureAlgorithm              AlgorithmIdentifier,
signature                       BIT STRING,
certs                       [0] EXPLICIT Certificates OPTIONAL }
ResponseData ::= SEQUENCE {
version                     [0] EXPLICIT Version DEFAULT v1,
responderID                     ResponderID,
```

```
producedAt                GeneralizedTime,
responses                 SEQUENCE OF SingleResponse,
responseExtensions    [1] EXPLICIT Extensions OPTIONAL }
ResponderID ::= CHOICE {
byName        [1] Name,
byKey         [2] KeyHash }
KeyHash ::= OCTET STRING --SHA-1 hash of responder's public key
--(excluding the tag, length and number of unused
-- bits fields)
SingleResponse ::= SEQUENCE {
reqCert                          ReqCert,
-- MUST be identical to the same field from the request
certStatus                       CertStatus,
thisUpdate                       GeneralizedTime,
nextUpdate                    [0] EXPLICIT GeneralizedTime OPTIONAL,
singleExtensions [1]             EXPLICIT Extensions OPTIONAL }
CertStatus ::= CHOICE {
good          [0]   IMPLICIT NULL,
revoked       [1]   IMPLICIT RevokedInfo,
unknown       [2]   IMPLICIT UnknownInfo }
RevokedInfo ::= SEQUENCE {
revocationTime        GeneralizedTime,
revocationReason [0]  EXPLICIT CRLReason OPTIONAL }
UnknownInfo ::= NULL -- this can be replaced with an enumeration
ArchiveCutoff ::= GeneralizedTime
AcceptableResponses ::= SEQUENCE OF OBJECT IDENTIFIER
ServiceLocator ::= SEQUENCE {
issuer     Name,
locator    AuthorityInfoAccessSyntax }
CrlLocator ::= CRLDistributionPoints
-- Object Identifiers
id-kp-OCSPSigning                OBJECT IDENTIFIER ::= { id-kp 9 }
id-pkix-ocsp                     OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic               OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
id-pkix-ocsp-nonce               OBJECT IDENTIFIER ::= { id-pkix-ocsp 2 }
id-pkix-ocsp-crl                 OBJECT IDENTIFIER ::= { id-pkix-ocsp 3 }
id-pkix-ocsp-response            OBJECT IDENTIFIER ::= { id-pkix-ocsp 4 }
id-pkix-ocsp-nocheck             OBJECT IDENTIFIER ::= { id-pkix-ocsp 5 }
id-pkix-ocsp-archive-cutoff      OBJECT IDENTIFIER ::= { id-pkix-ocsp 6 }
id-pkix-ocsp-service-locator     OBJECT IDENTIFIER ::= { id-pkix-ocsp 7 }
id-pkix-ocsp-crl-locator         OBJECT IDENTIFIER ::= { id-pkix-ocsp X }
END
```