`usertype` int NOT NULL, `userstatus` int NOT NULL, `usercn` varchar(100) NOT NULL, `usero` varchar(100) DEFAULT NULL, `userou` varchar(100) DEFAULT NULL, `userorgpos` varchar(100) DEFAULT NULL, `usertel` varchar(100) DEFAULT NULL, `usermp` varchar(100) DEFAULT NULL, `usere` varchar(100) DEFAULT NULL, `userfax` varchar(100) DEFAULT NULL, `userc` varchar(100) DEFAULT NULL, `userst` varchar(100) DEFAULT NULL, `userl` varchar(100) DEFAULT NULL, `userw` varchar(100) DEFAULT NULL, `usercode` varchar(100) NOT NULL, `userpin` varchar(100) NOT NULL, `macvalue` varchar(100) DEFAULT NULL, `fromtime` varchar(25) NOT NULL, `lasttime` varchar(25) NOT NULL, `lastopid` varchar(100) NOT NULL, `remark` text DEFAULT NULL, PRIMARY KEY (`userid`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `userbiz` (`bizid` varchar(100) NOT NULL, `biz2raid` varchar(100) NOT NULL, `biz2userid` varchar(100) NOT NULL, `biz2certid` varchar(100) DEFAULT NULL, `biz2certidref` varchar(100) DEFAULT NULL, `biztype` int NOT NULL, `bizstatus` int NOT NULL, `bizrefcode` varchar(100) DEFAULT NULL, `bizauthcode` varchar(100) DEFAULT NULL, `bizremark` text DEFAULT NULL, `bizinputopid` varchar(100) DEFAULT NULL, `bizinputtime` varchar(25) DEFAULT NULL, `bizcheckopid` varchar(100) DEFAULT NULL, `bizchecktime` varchar(25) DEFAULT NULL, `bizmakeopid` varchar(100) DEFAULT NULL, `bizmaketime` varchar(25) DEFAULT NULL, `macvalue` varchar(100) DEFAULT NULL, `remark` text DEFAULT NULL, PRIMARY KEY (`bizid`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `usercert` (`certid` varchar(100) NOT NULL, `cert2raid` varchar(100) NOT NULL, `cert2userid` varchar(100) NOT NULL, `certtype` int NOT NULL, `certnum` int NOT NULL, `certstatus` int NOT NULL, `certreqtype` int NOT NULL, `certcrlstatus` int NOT NULL, `certldapstatus` int NOT NULL, `certcrlcode` int DEFAULT -1, `certcrlremark` text DEFAULT NULL, `certsigntime` varchar(25) DEFAULT NULL, `certrevoketime` varchar(25) DEFAULT NULL, `certholdtime` varchar(25) DEFAULT NULL, `certc` varchar(100) NOT NULL, `certst` varchar(100) DEFAULT NULL, `certl` varchar(100) DEFAULT NULL, `certo` varchar(100) DEFAULT NULL, `certou` varchar(100) DEFAULT NULL, `certe` varchar(100) DEFAULT NULL, `certcn` varchar(100) NOT NULL, `certw` varchar(100) DEFAULT NULL, `certnotbefore` varchar(25) DEFAULT NULL, `certnotafter` varchar(25) DEFAULT NULL, `certvaluesign` text DEFAULT NULL, `certvalueenc` text DEFAULT NULL, `certtokenid` varchar(100) DEFAULT NULL, `certreqp10` text DEFAULT NULL, `certsnsign` varchar(100) DEFAULT NULL, `certsnenc` varchar(100) DEFAULT NULL, `macvalue` varchar(100) DEFAULT NULL, `fromtime` varchar(25) NOT NULL, `lasttime` varchar(25) NOT NULL, PRIMARY KEY (`certid`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;

CREATE TABLE `cacrl` (`crlid` int NOT NULL, `crltype` int NOT NULL, `crlvalue` mediumtext NOT NULL, PRIMARY KEY (`crlid`)) ENGINE=InnoDB DEFAULT CHARSET=utf8;

insert into opinfo values('20100721appca0000', '11110000', 'camgr', '', '', '', '', '', 11, 17, 1, '20100721', '', '', '','','','','camgr','camgr','','20100721','20100721','11110000','');

create user caadmin;

grant all privileges on *.* to caadmin@localhost identified by 'appca' with GRANT OPTION;

grant all privileges on *.* to caadmin@"%" identified by 'appca' with GRANT OPTION;

15.3.4　双证书技术流程设计

双证书技术流程设计如图 15-4 所示。

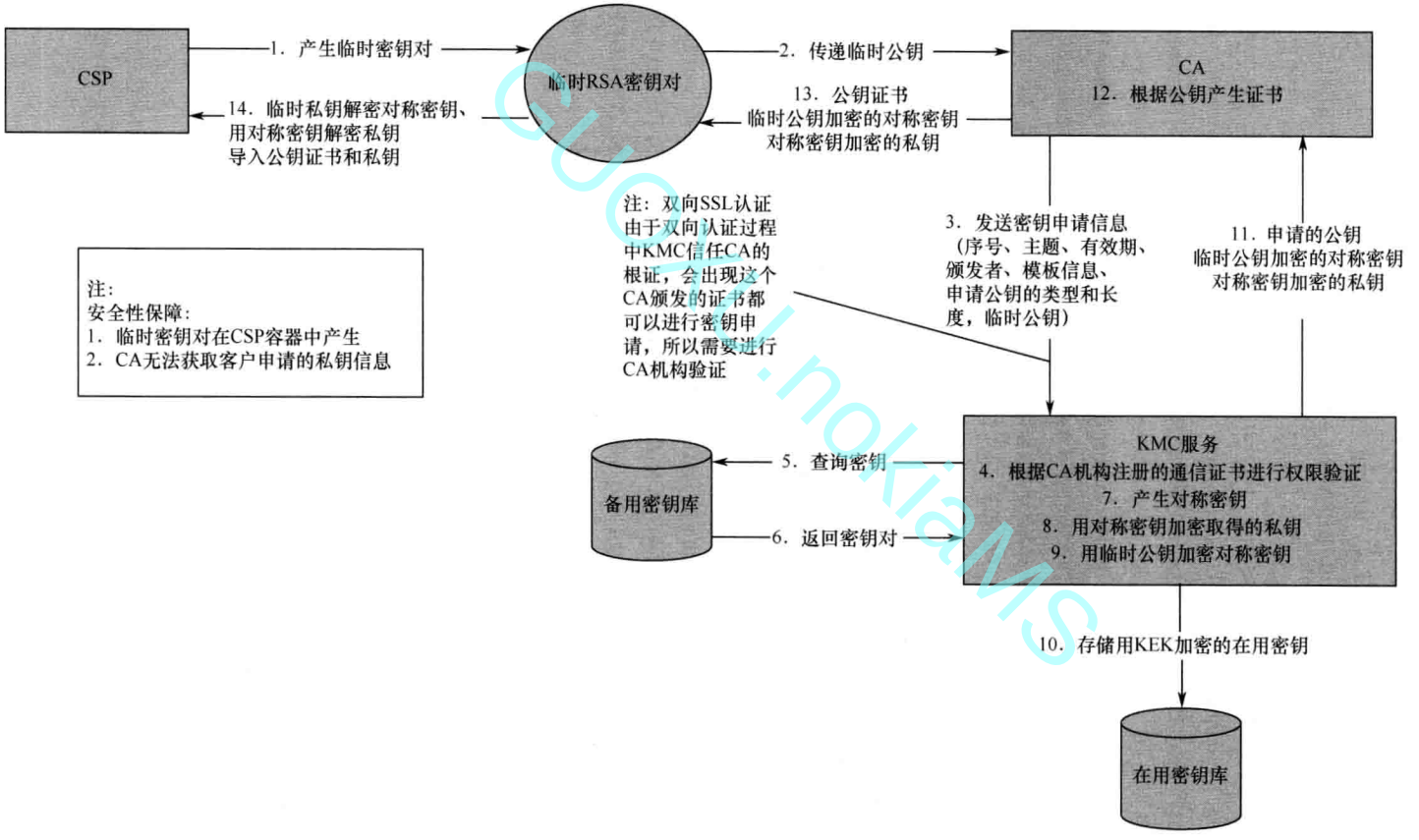

图 15-4　双证书技术流程图

双证书具体技术流程如下：

① 用户 CSP 产生临时密钥对。

② 用户 CSP 将临时公钥传递给 CA 系统。

③ CA 向 KMC 发送密钥申请信息，包括序号、主题、有效期、申请公钥类型和长度、临时公钥等。

④ KMC 根据 CA 机构注册的通信证书验证 CA 密钥申请的合法性。

⑤ KMC 从备用密钥库查询密钥对。

⑥ 备用密钥库将正式密钥对返回 KMC。

⑦ KMC 产生对称密钥。

⑧ KMC 用对称密钥加密正式私钥。

⑨ KMC 用临时公钥加密对称密钥。

⑩ KMC 将正式密钥对用 KEK 加密后存储到在用密钥库（KEK 是对密钥进行安全加密的专用密钥，需要预先设置）。

⑪ KMC 将正式公钥、对称密钥密文（临时公钥加密）、正式私钥密文（对称密钥加密）等返回给 CA。

⑫ CA 根据正式公钥产生用户正式公钥证书。

⑬ CA 将正式公钥证书、对称密钥密文（临时公钥加密）、正式私钥密文（对称密钥加密）返回给用户 CSP。

⑭ 用户 CSP 使用临时私钥解密获得对称密钥明文，用对称密钥解密获得正式私钥明文，然后将正式公钥证书和正式私钥导入密码介质。

第 16 章　对外在线服务

16.1　OCSP/SOCSP 服务

16.1.1　OCSP

OCSP 是 Online Certificate Status Protocol 的缩写，表示在线证书状态协议。CA 系统通过 OCSP 机制为用户提供在线证书状态查询服务。用户端将待查询证书序列号按照 OCSP 协议组织成 OCSP 请求包，然后将 OCSP 请求包发送给 OCSP 服务器，OCSP 服务器查询数据库获得该序列号对应证书的状态，并组织成 OCSP 响应包后返回给用户端。用户端解析 OCSP 响应包后获得该证书的当前状态。证书状态包括有效、已作废、已冻结等。

IETF RFC 2560 规定了 OCSP 请求包和响应包的具体格式。

1．OCSP 请求包

OCSP 请求包格式用 ASN.1 描述如下：

```
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, -- Hash of Issuer's DN
    issuerKeyHash       OCTET STRING, -- Hash of Issuers public key
    serialNumber        CertificateSerialNumber }
```

2．OCSP 响应包

OCSP 响应包格式用 ASN.1 描述如下：

```
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    }
ResponseBytes ::=       SEQUENCE {
    responseType    OBJECT IDENTIFIER,
    response        OCTET STRING }
id-pkix-ocsp            OBJECT IDENTIFIER ::= { id-ad-ocsp }
id-pkix-ocsp-basic      OBJECT IDENTIFIER ::= { id-pkix-ocsp 1 }
BasicOCSPResponse       ::= SEQUENCE {
    tbsResponseData     ResponseData,
    signatureAlgorithm  AlgorithmIdentifier,
    signature           BIT STRING,
    certs               [0] EXPLICIT SEQUENCE OF Certificate 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 and length
fields)
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 {
```