Estrutura de Chaves Públicas Brasileira

Manual de Condutas Técnicas 9 - Volume I

Requisitos, Materiais e Documentos Técnicos para Homologação de Softwares Provedores de Serviços Criptográficos (CSP) no Âmbito da ICP-Brasil

versão 1.0

São Paulo, 22 de novembro de 2007

Sumário

Controle de versão

Versão revisada	Data de emissão	Alterações realizadas

Listas de ilustrações

Lista de Figuras

Lista de Tabelas

1 Introdução

Este documento descreve os requisitos técnicos a serem observados no processo de homologação de software provedor de serviços criptográficos (CSP) [1], no âmbito da Infra-Estrutura de Chaves Públicas Brasileira, ICP-Brasil [2] [27][28][29]. Recomendações de melhores práticas também estão incluídas.

Tais requisitos e recomendações abordam tópicos relativos à interoperabilidade, segurança, funcionalidade e compatibilidade.

De maneira geral, um CSP fornece serviços como cifrar e decifrar chaves e assinatura digital além de consistir na implementação de uma interface definida por uma arquitetura criptográfica e algoritmos criptográficos. No mínimo, um CSP consiste em um conjunto de funções a serem carregadas dinamicamente (DLL, SO ou JAR) [3].

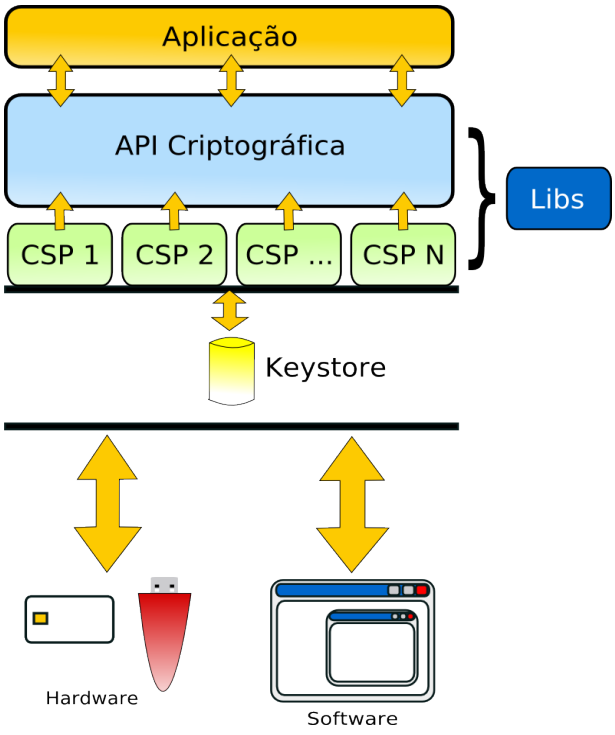

Figura 1. Arquitetura de um CSP

1.1 Objetivo da homologação

O objetivo do processo de homologação de CSP é validar a interoperabilidade e operação segura do provedor de serviços criptográficos (CSP), oferecido por meio da avaliação técnica de aderência aos requisitos técnicos definidos para este processo.

1.2 Descrição do processo de homologação

O processo de homologação é baseado em um conjunto de requisitos técnicos que devem ser atendidos por um CSP para garantia da interoperabilidade e operação segura.

Os requisitos técnicos englobam requisitos funcionais, de interoperabilidade, de documentação, de segurança e requisitos específicos como gerenciamento, exportação e importação, certificação e proteção de chaves em memória que devem ser atendidos pelo CSP.

Estes requisitos técnicos são avaliados segundo ensaios de aderência aos requisitos técnicos. Para a realização dos ensaios, a parte interessada deve submeter ao processo de homologação um conjunto de materiais requisitados, através de um procedimento denominado depósito de material.

1.3 Escopo do processo de homologação

O escopo da avaliação considera os serviços criptográficos, porém levando em consideração os possíveis riscos causados pela coexistência com outros serviços ou subsistemas.

O escopo dos requisitos técnicos e da avaliação se aplicam aos seguintes componentes:

- Documentação aderente
  - O produto corresponde ao descrito na documentação
- Avaliação dos algoritmos criptográficos
  - Implementação
  - Conformidade com as normas de especificação
  - Auto-testes

- Projeto de software
  - Documentos de requisitos (Ex. UML)
- Avaliação de interoperabilidade
  - Gerenciamento de chaves criptográficas e requisitos gerais de interoperabilidade

1.4 Verificação de integridade externa

O conjunto de arquivos especificados no CSP constitui o conjunto completo de código-fonte desse sistema. Não haverão inserções, retiradas ou alterações desse conjunto de arquivos como definido na geração do CSP.

O código compilado do CSP será verificado usando um tipo de algoritmo criptográfico como HMAC [5] utilizando função hash SHA-1 [6] e SHA-256.

Uma chave simétrica arbitrária será definida pelo ITI para ser usada na geração do valor HMAC utilizando função hash SHA-1 e SHA-256 para a checagem da integridade do sistema.

1.5 Estruturação do MCT-9

Este documento (MCT-9) está estruturado da seguinte forma:

- Parte 1: Descreve os requisitos técnicos que devem ser verificados no processo de homologação de CSP´s;
- Parte 2: Descreve os materiais que devem ser depositados para a execução do processo de homologação de CSP´s;
- Referências normativas: Descreve as referências normativas que foram utilizadas na elaboração deste documento.

Os termos e expressões usados nesse documento estão referenciados no MCT – Glossário Geral [4].

2 PARTE 1

Requisitos técnicos para homologação de CSP

2.1 Introdução

A parte 1 deste documento apresenta os requisitos técnicos que devem ser verificados no processo de homologação de CSP´s.

Os requisitos técnicos descritos nesta parte englobam:

- Requisitos de documentação;
- requisitos de segurança;
- requisitos de interoperabilidade.

2.2 Requisitos de documentação

A documentação em geral envolve os seguintes itens:

- **Manual de instalação**: Manual especificando como deve ser feita a instalação do CSP.
- **Manual do usuário**: Manual do usuário, especificando como utilizar o CSP.
- **Manual do desenvolvedor**: Manual da API para desenvolver aplicações utilizando o CSP. Especificação do próprio fornecedor.
- **Manual de integração**: Manual especificando a compatibilidade de hardwares específicos como *smart cards*, leitoras de *smart cards* ou *tokens* criptográficos utilizados para acesso das chaves criptográficas através do CSP.

A parte 2 deste documento terá um *check list* com uma lista completa de documentação requisitada para o processo de homologação de CSP.

REQUISITO II.1: A documentação deve estar escrita nos idiomas português do Brasil ou inglês.

REQUISITO II.2: A PI deve fornecer manual de instalação e configuração, especificando os processos de instalação e configuração do CSP. Além disso, o manual de instalação deve especificar os sistemas operacionais suportados pelo CSP.

REQUISITO II.3: A PI deve fornecer o manual do usuário, detalhando as ferramentas e recursos disponíveis aos operadores do CSP.

REQUISITO II.4: A PI deve fornecer o manual de desenvolvedor detalhando as APIs para desenvolvimento de aplicações utilizando o CSP.

REQUISITO II.5: A PI deve fornecer um manual especificando as compatibilidades do CSP com as APIs de mercado para dispositivos de armazenamento como *smart cards* ou *tokens*.

REQUISITO II.6: A PI deve fornecer trechos de código-fonte para utilização do CSP.

2.3 Requisitos de segurança

Esta seção descreve os requisitos mínimos de segurança que devem ser atendidos de forma comum pelos CSP´s na sua utilização.

2.3.1 Requisitos de segurança baseados no padrão FIPS

2.3.1.1 Especificação de um CSP

Um CSP pode ser um conjunto de serviços específicos tais como cifração e decifração baseado em hardware ou software.

REQUISITO III.1.1: A documentação deve especificar cada subsistema empregado pelo CSP.

REQUISITO III.1.2: Caso o CSP carregue dinamicamente subsistemas na hora de execução, deve existir um mecanismo de integridade do CSP, impedindo substituição de subsistemas por sistemas mal intencionados.

REQUISITO III.1.3: A documentação deve especificar o método para garantia de integridade do CSP.

2.3.1.2 Interfaces do CSP

REQUISITO III.1.4: A documentação técnica do CSP deve especificar claramente as seguintes interfaces:

- Entrada de dados: Parâmetros de entrada para todas as funções que aceitam entrada do invocador da API;
- saída de dados: Parâmetros de saída de funções que retorna dados como argumentos ou como valor de retorno da função;
- saída de estado: Informação retornada por meio de exceções (códigos de retorno ou *exit*).

2.3.1.3 Algoritmos criptográficos

Uma grande preocupação em um CSP são os algoritmos criptográficos implementados ou fornecidos por meio de biblioteca criptográfica. É importante que essas implementações estejam em conformidade com as respectivas especificações.

REQUISITO III.1.5: O CSP deve suportar no mínimo as seguintes funções criptográficas:

- Criptografia de dados:
    - DES (*Data Encryption Standard*) [8] nos modos de operação ECB e CBC, apenas para uso legado (conforme padrão NIST FIPS PUB 46-3);
    - *Triple*-DES (3DES ou TDES) [8] nos modos de operação ECB e CBC (conforme padrão NIST FIPS PUB 46-3);
    - AES (*Advanced Encryption Standard*) [12] com tamanho de chave 128 bits nos modos de operação ECB e CBC (conforme padrão NIST FIPS PUB 197);

    - RSA com utilização de chaves de comprimento maior do que 1024 bits, conforme padrões ANSI X9.31 [13] e PKCS#1 v. 1.5 [9].
- Autenticação e assinatura digital de dados:
    - RSA com tamanho mínimo de chaves de 1024 bits, conforme padrões ANSI X9.31 [13] e PKCS#1 v. 1.5 [9].
- Resumo criptográfico de dados (*Hash*):
    - SHA-1 (*Secure Hash Algorithm*) [6], apenas para uso legado conforme padrão NIST FIPS PUB 180-2;
    - SHA-256 (*Secure Hash Algorithm*) conforme padrão NIST FIPS PUB 180-2.

RECOMENDAÇÃO III.1.1: O CSP também pode suportar a função AES (*Advanced Encryption Standard*) [12] com utilização de chaves de comprimento de 192 e 256 bits, conforme padrão NIST FIPS PUB 197, para cifração e decifração de dados.

RECOMENDAÇÃO III.1.2: O CSP também pode suportar a função DSA (*Data Signature Algorithm*) com utilização de chaves de comprimento maior do que 512 bits, conforme padrão NIST FIPS PUB 186 [10], para autenticação e assinatura digital de dados.

RECOMENDAÇÃO III.1.3: O CSP também pode suportar as seguintes funções para a geração de resumos criptográficos de dados, conforme padrão NIST FIPS PUB 180-2 [6]:

- SHA-224;
- SHA-256;
- SHA-384;
- SHA-512.

RECOMENDAÇÃO III.1.4: O CSP também pode suportar as seguintes funções para a autenticação e integridade de dados:

- CBC-MAC baseado nos algoritmos 3DES ou AES, conforme padrão NIST PUB 800-38B [14];

- HMAC baseado nos algoritmos de resumos criptográficos implementados, conforme padrão NIST FIPS PUB 198 [5];
- CMAC baseado nos algoritmos 3DES ou AES, conforme padrão NIST PUB 800-38B [14];
- MAC-CCM baseado nos algoritmos 3DES ou AES, conforme padrão NIST PUB 800-38C [15].

RECOMENDAÇÃO III.1.5: Para a biblioteca criptográfica ICP-Brasil que suportar funções para derivação de chaves simétricas baseada em senha, é recomendável a seguinte função de derivação de chaves [16]:

- Função 2 de derivação de chaves baseada em senha, PBKDF2, como especificada em PKCS#5.

REQUISITO III.1.6: O CSP deve suportar o formato PKCS#1 para armazenamento das chaves assimétricas [9]:

- Uma chave pública RSA deve ter o tipo *ASN.1 RSAPublicKey*:

  RSAPublicKey ::= SEQUENCE {
  modulus INTEGER,
  publicExponent INTEGER
  }

  Os campos do tipo *RSAPublicKey* possuem os seguintes significados:
  - *modulus* é o módulo n.
  - *publicExponent* é o expoente público *e*.

- Uma chave privada RSA deve ter o tipo *ASN.1 RSAPrivateKey*:

  RSAPrivateKey ::= SEQUENCE {
  version INTEGER,
  modulus INTEGER,
  publicExponent INTEGER,
  privateExponent INTEGER,

```
    prime1 INTEGER,
    prime2 INTEGER,
    exponent1 INTEGER,
    exponent2 INTEGER,
    coefficient INTEGER
}
```

Os campos do tipo *RSAPrivateKey* possuem os seguintes significados:

- *version* é o número da versão. Esse valor pode ser 0 para a versão desse padrão.
- *modulus* é o módulo n.
- *publicExponent* é o expoente público *e*.
- *privateExponent* é o expoente privado *d*.
- *prime1* é o fator primo *p* de *n*.
- *prime2* é o fator primo *q* de *n*.
- *exponent1* é *d mod (p−1)*.
- *exponent2* é *d mod (q−1)*.
- *coefficient* é o coeficiente do Teorema de Resto Chinês *q−1 mod p*.

- **OBSERVAÇÃO:** Para mais detalhes sobre esses campos, ver especificação do PKCS#1 [9].
- **OBSERVAÇÃO:** Para uma chave privada RSA seria suficiente somente módulo e expoente privado, porém é desejável otimizar o processo de aplicação de expoente conforme é feito na exponenciação modular utilizando formato TCR (Teorema Chinês do Resto).

RECOMENDAÇÃO III.1.6: O CSP pode suportar o formato PKCS#12 [17] para troca de identificações pessoais. O formato de troca de informações pessoais (PFX), permite a transferência de certificados e das chaves particulares correspondentes entre computadores ou de um computador para mídia removível. Isso pode ser feito entre produtos do mesmo fornecedor ou de diferentes fornecedores.

Para tal, o CSP precisa importar como arquivo para dentro do módulo, decifrar a chave privada e depois criar objetos de PKCS#11 como chave pública e chave privada baseado nisso. O certificado também precisa ser importado e ligado ao par de chaves.

RECOMENDAÇÃO III.1.7: O CSP pode suportar o formato PKCS#8 [19] para armazenamento das chaves assimétricas:

- ASN.1 type PrivateKeyInfo:

```
PrivateKeyInfo ::= SEQUENCE {
        version INTEGER,
        privateKeyAlgorithm PrivateKeyAlgorithmIdentifier,
        privateKey PrivateKey,
        attributes [0] IMPLICIT Attributes OPTIONAL
}

PrivateKeyAlgorithmIdentifier ::= AlgorithmIdentifier
PrivateKey ::= OCTET STRING
Attributes ::= SET OF Attribute
```

Os campos do tipo *PrivateKeyInfo* possuem os seguintes significados:

- *version* é o número da versão.
- *privateKeyAlgorithm* identifica o algoritmo da chave privada.
- *privateKey* é um *"octet string*" cujos conteúdos são os valores da chave privada.
- *attributes* é um conjunto de atributos.

2.3.1.4 Auto-testes

REQUISITO III.1.7: O CSP deve executar um número de auto-testes para garantir a operação correta do mesmo.

Podemos citar as seguintes classes de auto-testes para um CSP:

- Testes de algoritmos criptográficos;
- testes da integridade de software;

- testes de carregamento de software;
- testes de funções críticas;
- testes de respostas conhecidas.

Algoritmos	Testes de resposta conhecida
AES	Cifração e decifração com chave de 128 bits
DES	Cifração e decifração
3DES (2 chaves)	Cifração e decifração
3DES	Cifração e decifração
RSA	• Teste de consistência de par de chaves de 1024 bits • Cifragem pública e decifragem privada com chave de 1024 bits • Teste de assinatura e verificação com chave de 1024 bits

REQUISITO III.1.8: Os testes de integridade devem utilizar um tipo de algoritmo criptográfico como HMAC-SHA-1, calculado em cima do código compilado de cada componente do CSP.

REQUISITO III.1.9: Os auto-testes devem ser chamados na instanciação do CSP. Adicionalmente devem ser possíveis de chamar por meio de função API como por exemplo *Executar_auto_testes()*;

REQUISITO III.1.10: Se o CSP apresentar falhas durante um auto-teste, o mesmo deve ser conduzido a um estado de erro e emitir um indicador de erro via “Interface de Saída de Estado”.

REQUISITO III.1.11: Nenhuma funcionalidade criptográfica deve estar disponível até a execução com sucesso dos auto-testes.

REQUISITO III.1.12: Quando um estado de erro ocorrer devido a falhas em um auto-teste, toda saída ou envio de dados via “Interface de Saída de Dados” deve ser impedido.

REQUISITO III.1.13: A documentação do CSP deve especificar os seguintes itens:

- Os auto-testes realizados pelo CSP;
- os estados de erro que o CSP pode entrar quando um auto-teste falha;
- as condições e ações necessárias para sair dos estados de erro e reiniciar a operação normal do CSP.

2.3.1.5 Garantia do projeto

REQUISITO III.1.14: A parte interessada deve fornecer documentação de utilização de ferramenta de controle de versão do código-fonte do CSP.

REQUISITO III.1.15: A documentação do CSP deve incluir diagramas de engenharia de software que representem a arquitetura do elemento de software.

REQUISITO III.1.16: A documentação do CSP deve incluir diagramas que ilustrem sua relação de uso por outros elementos de software ou hardware.

Nível de Segurança de Homologação 2

REQUISITO III.1.17: No caso do CSP ser *multi-threaded*, as funções que envolvem operações criptográficas devem ser *thread-safe*, ou seja, não devem colocar em risco nenhum tipo de informação protegida compartilhada contra divulgação, modificação e substituição não autorizada.

Nível de Segurança de Homologação 3

RECOMENDAÇÃO III.1.8: Os parâmetros de saída das funções das APIs do CSP podem apresentar as seguintes características:

- Nenhum destes parâmetros deve ser utilizado como variável temporária durante a sua execução, isto é, não devemos atribuir os valores dos parâmetros de uma função a uma variável temporária durante a execução desta função;

- todos os parâmetros de saída devem somente retornar os tipos que foram pré-determinados na API, isto é, não devemos modificar os parâmetros de saída de uma função na execução da mesma.

2.4 Requisitos de interoperabilidade

Os requisitos funcionais se referem à avaliação de funções relacionadas aos serviços criptográficos que podem ser invocados por aplicações de usuários por meio de uma interface de alto nível denominada API (*Application Programming Interface*).

REQUISITO IV.1: [referente ao item 2.1. do DOC-ICP-10.03] O CSP deve atender aos requisitos funcionais estabelecidos, conforme descrito nos itens a seguir. No escopo deste documento, pelo menos uma das seguintes APIs serão consideradas para análise dos requisitos funcionais:

- *Microsoft CSP (CryptoAPI) [20];*
- *PKCS#11 V.2.11 [21];*
- *JCE/JCA [22];*
- *OpenSSL Engine [23] .*

2.4.1 Gerenciamento de chaves criptográficas

REQUISITO IV.1.1: [referente ao item 2.1. do DOC-ICP-10.03] Os seguintes requisitos funcionais de gerenciamento de chaves criptográficas do CSP devem estar disponíveis por invocação via API:

- Gerar chave criptográfica assimétrica de forma randômica;
- destruir chave criptográfica assimétrica com sobrescrita de valores;
- recuperar parâmetros sobre uma determinada chave criptográfica assimétrica, tais como:
  - Algoritmo;

- expoente público (RSA);
- módulo (RSA);
- tamanho da chave;
- permissões.

2.4.2 Exportação e importação

REQUISITO IV.2.1: [referente ao item 2.1.do DOC-ICP-10.03] Os seguintes requisitos funcionais de exportação e importação devem estar disponíveis no CSP por invocação via API:

- Exportar chave criptográfica simétrica;
- importar chave criptográfica simétrica;
- exportar chave criptográfica assimétrica pública. A exportação de chave criptográfica assimétrica privada só deve ser possível para certificados dos tipos A1, A2, S1 e S2;
- importar chave criptográfica assimétrica pública ou privada;
- exportar certificado segundo padrão X.509 versão 3;
- importar certificado segundo padrão X.509 versão 3.

RECOMENDAÇÃO IV.2.1: A exportação e disponibilização de certificado digital armazenado em hardware seguro para o sistema operacional pode ser feita automaticamente por software, como uma facilidade oferecida pelo fabricante ao usuário.

2.4.3 Assinatura e certificação digital

REQUISITO IV.3.1: [referente ao item 2.1.do DOC-ICP-10.03] Os seguintes requisitos funcionais de assinatura e certificação digital do CSP devem estar disponíveis por invocação via API:

- Realizar a assinatura digital de uma mensagem conforme o padrão PKCS #1 V.1.5;
- realizar a verificação de uma assinatura digital de uma mensagem conforme o padrão PKCS #1 V.1.5.

2.4.4 Requisitos gerais de interoperabilidade

REQUISITO IV.4.1: No mínimo uma das seguintes APIs serão consideradas para análise dos requisitos de interoperabilidade:

- *Microsoft CSP (CryptoAPI);*
- *PKCS#11 v. 2.11;*
- *JCE/JCA;*
- *OpenSSL Engine.*

2.4.4.1 Requisitos gerais de um CSP

REQUISITO IV.4.2: Um CSP deve ser capaz de fazer, no mínimo, as seguintes operações:

- Gerar par de chaves especificando os componentes de chaves assimétricas em texto claro;
- cifrar e decifrar chaves especificando os componentes de chaves assimétricas em texto claro;
- importar e exportar chaves (PKCS#12) especificando os componentes de chaves assimétricas privadas criptografados. A exportação de chaves assimétricas privadas é permitida apenas no caso de certificados do tipo A1, S1, A2 e S2;
- assinar conteúdo especificando os componentes de chaves assimétricas públicas em texto claro;
- verificar assinatura especificando os componentes de chaves assimétricas públicas em texto claro.

REQUISITO IV.4.3: A implementação da interface nativa deve suportar os algoritmos criptográficos descritos na seção “Algoritmos criptográficos”.

2.4.4.2 Requisitos sobre Microsoft CSP (CryptoAPI)

REQUISITO IV.4.4: O CSP deve suportar, no mínimo, uma implementação do MS CSP (*CryptoAPI)*, versão 1.0.

REQUISITO IV.4.5: O CSP deve ser implementado na forma de DLL.

RECOMENDAÇÃO IV.4.1: É recomendável que o CSP seja implementado em apenas um módulo DLL, para reduzir problemas relativos ao contexto de execução.

REQUISITO IV.4.6: O CSP deve ser assinado pela Microsoft. Caso seja composto de mais de uma DLL, todas devem ser assinadas.

REQUISITO IV.4.7: O CSP deve possuir um software instalador, que copie os arquivos necessários a um diretório apontado no PATH e que crie as entradas no registro adequadas. Este software deve ser capaz de desinstalar os componentes do CSP, apagando as informações não compartilhadas por outros sistemas instalados.

REQUISITO IV.4.8: [referente ao artigo do MSDN: “*The Smart Card CSP Cookbook*”] O CSP deve exportar, isto é, expor sua interface, das seguintes chamadas:

- *CPAcquireContext*
- *CPCreateHash*
- *CPDecrypt*
- *CPDeriveKey*
- *CPDestroyHash*
- *CPDestroyKey*
- *CPEncrypt*

- *CPExportKey*
- *CPGenKey*
- *CPGenRandom*
- *CPGetHashParam*
- *CPGetKeyParam*
- *CPGetProvParam*
- *CPGetUserKey*
- *CPHashData*
- *CPHashSessionKey*
- *CPImportKey*
- *CPReleaseContext*
- *CPSetHashParam*
- *CPSetKeyParam*
- *CPSetProvParam*
- *CPSignHash*
- *CPVerifySignature*

Sendo obrigatória a implementação das seguintes funções:

- *CPAcquireContext* para criação e remoção de *key containers* existentes.
- *CPGenKey* tanto para chaves simétricas quanto para assimétricas;
- *CPImportKey* especificando tanto as chaves simétricas quanto as assimétricas;
- *CPGetKeyParam* para recuperação de parâmetros de permissões de acesso às chaves criadas/existentes em um *key container*;
- *CPHashData e CPSignHash* para geração de assinatura utilizando chave assimétrica;
- *CPVerifySignature* para verificação da assinatura após a importação da chave pública via *CPImportKey;*
- *CPDecrypt e CPEncrypt* para decifrar e cifrar chaves simétricas e assimétricas.

As funções não implementadas devem retornar o código de erro *E_NOTIMPL*.

- **Gerenciamento do cache do PIN:**

REQUISITO IV.4.9: Caso o CSP utilize cartão inteligente, deve ser configurável o armazenamento do PIN em cache após ter sua validade verificada pelo cartão. Mecanismo de relacionamento entre o usuário corrente e o PIN deve ser implementado.

REQUISITO IV.4.10: Caso o CSP utilize cartão inteligente, o cache do PIN deve ser apagado quando o mesmo for removido da leitora ou quando encerrar a sessão do Windows.

REQUISITO IV.4.11: Se o serviço gerenciador de cartões inteligentes do Windows (*Smart Card Service*) for desativado, todos os contextos e *handles* em uso pelo CSP devem ser invalidados.

REQUISITO IV.4.12: O CSP deve suportar hibernação. Todos os *handles*, contextos e sessões devem ser válidos após o processo de hibernação.

REQUISITO IV.4.13: A implementação de MS CSP (*CryptoAPI)* deve suportar os algoritmos criptográficos descritos na seção “Algoritmos criptográficos”.

2.4.4.3 Requisitos sobre PKCS#11

REQUISITO IV.4.14: O CSP deve suportar uma implementação PKCS#11 na versão no mínimo 2.11.

REQUISITO IV.4.15: O CSP deve suportar as seguintes chamadas de PKCS#11 (*Cryptoki*):

- *C_Initalize*
- *C_Finalize*

- *C_OpenSession*
- *C_CloseSession*
- *C_Init_Token*
- *C_Init_PIN*
- *C_Login*
- *C_Logout*
- *C_CreateObject*
- *C_DestroyObject*
- *C_GetAttributeValue*
- *C_SetAttributeValue*
- *C_EncryptInit*
- *C_Encrypt*
- *C_DecryptInit*
- *C_Decrypt*
- *C_DigestInit*
- *C_Digest*
- *C_DigestKey*
- *C_SignInit*
- *C_Sign*
- *C_VerifyInit*
- *C_Verify*
- *C_GenerateKey*
- *C_GenerateKeyPair*
- *C_DeriveKey*
- *C_GenerateRandom*

Sendo obrigatória a implementação das seguintes funções:

- *C_GenerateKey* especificando templates de chaves simétricas;
- *C_GenerateKeyPair* especificando templates de chaves assimétricas;
- *C_Sign* para realizar assinatura de um conteúdo;
- *C_Verify* para verificar a assinatura de um conteúdo;

- *C_Encrypt* para cifrar um dado com uma chave já construída;
- *C_Decrypt* para decifrar um dado com uma chave já construída;
- *C_CreateObject* especificando templates de chaves assimétricas (no mínimo chave pública);
- *C_DestroyObject* especificando o *handle* do objeto.

REQUISITO IV.4.16: A implementação PKCS#11 deve suportar os algoritmos criptográficos descritos na seção “Algoritmos criptográficos”.

2.4.4.4 Requisitos sobre Java Cryptographic Extension (JCE)

REQUISITO IV.4.17: O pacote de classes JCE deve ser suportado pela versão da máquina virtual Java (no mínimo V.1.4.2).

REQUISITO IV.4.18: O CSP deve suportar, no mínimo, as seguintes classes de JCE [Java 2 SDK]:

- *MessageDigest*
- *Signature*
- *KeyPairGenerator*
- *KeyFactory*
- *CertificateFactory*
- *KeyStore*
- *AlgorithmParameters*
- *AlgorithmParameterGenerator*
- *SecureRandom*
- *CertStore*

REQUISITO IV.4.19: A documentação deve especificar os componentes de software implementados do provedor de serviço criptográfico.

REQUISITO IV.4.20: A documentação deve especificar o processo de configuração e instalação do provedor de serviço criptográfico.

REQUISITO IV.4.21: A documentação deve especificar serviços criptográficos implementados no provedor de serviço criptográfico que não estejam na especificação JCE versão 1.4 ou superior.

REQUISITO IV.4.22: A documentação deve informar detalhes sobre o uso do provedor de serviço criptográfico como API no formato Javadoc com trechos de código-fonte.

REQUISITO IV.4.23: A implementação JCE deve suportar os algoritmos criptográficos descritos na seção “Algoritmos criptográficos”.

RECOMENDAÇÃO IV.4.2: Se aplicável, o provedor de serviço criptográfico pode ser assinado por uma chave privada ligado a um certificado digital reconhecido no âmbito ICP-Brasil.

2.4.4.5 Requisitos sobre OpenSSL

REQUISITO IV.4.24: O CSP deve ser capaz de implementar as seguintes rotinas do OpenSSL Engine:

- *ENGINE_init;*
- *ENGINE_finish;*
- *bind_fn;*
- *Engine_load;*
- *ENGINE_load_private_key;*
- *ENGINE_load_public_key;*
- *bind_helper;*
- *ENGINE_destroy;*
- Dentre as funções requeridas para operações RSA estão (RSA_METHOD):
    - *RSA_init;*

- *RSA_finish;*
- *RSA_pub_dec ou RSA_verify (1);*
- *RSA_priv_enc ou RSA_sign (1);*
- *RSA_pub_enc;*
- *RSA_priv_dec;*

OBS: (1) Por questão de compatibilidade o OpenSSL ainda mantém as duas funções, tendo um campo para setar um flag (RSA_FLAG_SIGN_VER) de que versão é suportada.

- Funções requeridas para geração de números aleatórios (RAND_METHOD):
  - *RAND_bytes;*
  - *RAND_pseudo_bytes;*
  - *RAND_status.*

REQUISITO IV.4.25: A implementação do OpenSSL Engine deve suportar os algoritmos criptográficos descritos na seção “Algoritmos criptográficos”.

3 PARTE 2

Material e documentos técnicos a serem depositados para a execução do processo de homologação de CSP

3.1 Introdução

O objetivo desta parte do documento é detalhar o material e os documentos técnicos a serem depositados pela parte interessada junto ao LSI-TEC LEA [24] para a realização dos processos de homologação de provedor de serviços criptográficos (*Cryptographic Service Provider* – CSP) no âmbito da ICP-Brasil [25][26].

O material e os documentos técnicos referidos são classificados em duas categorias:

1. Documentos técnicos: correspondem aos documentos de natureza técnica referentes aos CSPs a serem submetidos ao processo de homologação. Devem ser depositados em formato impresso ou em formato eletrônico. No caso de formato eletrônico, devem estar armazenados, preferencialmente, em mídia tipo "leitura-somente" (*read-only*). Devem estar, obrigatoriamente, escritos nas línguas portuguesa ou inglesa;
2. Componentes em softwares executáveis: correspondem aos *drivers*, bibliotecas de software, ferramentas de gerenciamento de dispositivo e/ou outros softwares executáveis, solicitados por este documento, referentes aos dispositivos a serem submetidos ao processo de homologação. Devem ser depositados, obrigatoriamente, em formato eletrônico e armazenados, preferencialmente, em mídia tipo "leitura-somente" (*read-only*).

Três Níveis de Segurança de Homologação (NSH) diferentes foram estabelecidos para CSPs:

- NSH 1: Este nível não requer depósito e análise de código-fonte em homologação;
- NSH 2: Este nível requer depósito e análise apenas de código-fonte de componentes específicos associados ao objeto em homologação.
- NSH 3: Este nível requer depósito e análise de código-fonte completo associado ao objeto em homologação.

Para os NSHs 2 e 3, a parte interessada pode depositar o código-fonte em linguagem C, C++ ou Java. Se o código-fonte estiver escrito em linguagem proprietária ou mesmo em micro código, o respectivo manual desta linguagem deve estar contido na documentação como também simuladores para compilação e execução deste código-fonte.

3.2 Material e documentos técnicos a serem depositados

Segue abaixo a relação de materiais e documentos técnicos a serem depositados junto ao LSI-TEC LEA.

3.2.1 Documentos técnicos

3.2.1.1 Nível de Segurança de Homologação 1

Os seguintes documentos técnicos devem ser depositados junto ao LSI-TEC LEA pela parte interessada:

- Projeto de software: Projeto de software do CSP;
- Política de segurança não proprietária: Se a biblioteca criptográfica já tiver sido homologada pelo padrão FIPS;
- Manual de usuário/instalação: Manual de usuário/instalação idêntico ao fornecido ao usuário;
- Manuais das interfaces de programação (API): Manuais e documentos técnicos relacionados às APIs aplicáveis, como por exemplo:
    - Microsoft CSP (*CryptoAPI)*;
    - PKCS#11 versão 2.11;
    - Interfaces associadas à plataforma JCE/JCA, versões suportadas e procedimento de configuração no JRE (*Java Runtime Environment*);
    - OpenSSL Engine.

- Projeto dos softwares de apoio: Documentos técnicos contendo a arquitetura, especificação técnica e o projeto de todo software de apoio, tais como, interfaces de programação (API), SDK (*Software Development Kits*), ferramenta de gerenciamento e bibliotecas de software suportadas;
- Relação de certificados obtidos: Relação de certificação e/ou licenças obtidas para o CSP emitidas por entidades independentes;
- Outros documentos: Projetos técnicos e suas especificações que a parte interessada julgar necessário para completar toda documentação técnica exigida.

3.2.1.2 Nível de Segurança de Homologação 2

Adicionalmente aos documentos técnicos solicitados na seção 3.2.1.1, os seguintes itens devem ser depositados junto ao LSI-TEC LEA pela parte interessada:

- Código-fonte do componente de geração de chaves;
- código-fonte do componente de armazenamento de chaves;
- código-fonte do componente de importação/exportação de chaves e sementes;

3.2.1.3 Nível de Segurança de Homologação 3

Adicionalmente aos documentos técnicos solicitados nas seções 3.2.1.1 e 3.2.1.2, os seguintes itens devem ser depositados junto ao LSI-TEC LEA pela parte interessada:

- Código-fonte do CSP: Relação de todo código-fonte de software;
- código-fonte de apoio: Relação de todo código-fonte de apoio relacionado às interfaces de programação (API), SDK (*Software Development Kits*), ferramenta de gerenciamento e bibliotecas de software suportadas pelos serviços criptográficos.

3.2.1.4 Componentes em software executável

Para os NSHs 1, 2 e 3, os seguintes componentes em softwares executáveis devem ser depositados junto ao LSI-TEC LEA pela parte interessada:

- O software CSP compilado;
- Ferramentas de gerenciamento do CSP;
- Outras bibliotecas de software e/ou programas.

3.2.1.5 Componentes físicos de apoio

Para os NSHs 1, 2 e 3, os seguintes componentes físicos devem ser depositados junto ao LSI-TEC LEA pela parte interessada:

- Material de apoio: Caso o CSP submetido precise de hardware de apoio como cartão, leitora ou *token*:
  - Cartão criptográfico ICP: Amostras nas quantidades definidas por este documento.
  - Leitora de cartão inteligente: Amostras nas quantidades definidas por este documento.
  - *Token* criptográfico: Amostras nas quantidades definidas por este documento.

3.2.2 Quantidade de material e documentos técnicos a serem depositados

A Tabela 1 apresenta os materiais e documentos técnicos a serem depositados pela parte interessada junto ao LSI-TEC LEA referente ao processo de homologação de CSPs.

Tabela 1. Quantidade de material e documentos técnicos a serem depositados pela parte interessada junto ao LSI-TEC LEA referente ao processo de homologação de CSPs.

Requisito de depósito	Material e documentos técnicos a serem depositados pela parte interessada – NSH 1	Quantidade
1	Cartão inteligente – material de apoio	2
2	Leitora de cartão inteligente – material de apoio	1
3	*Token* de acesso – material de apoio	2
4	PIN padrão	1
5	Projeto de software	2
6	Política de segurança não proprietária	2
7	Manual de usuário e manual de instalação	2
8	Manuais das interfaces de programação (APIs) e bibliotecas de desenvolvimento	2
9	Projeto de software de apoio	2
10	Relação de certificados obtidos	2
11	Outros documentos	2
Requisito de depósito	**Material e documentos técnicos a serem depositados pela parte interessada – NSH 2**	
12	Código-fonte de componentes	2
Requisito de depósito	**Material e documentos técnicos a serem depositados pela parte interessada – NSH 3**	
13	Código-fonte	2
14	Código-fonte de apoio	2
Requisito de depósito	**Componentes em software executável a serem depositados pela parte interessada – NSH 1, 2 e 3**	
15	O CSP compilado	2
16	Ferramentas de gerenciamento	
17	Outras bibliotecas de software e/ou programas	2

3.3 Plataformas para homologação

O fabricante pode escolher para qual plataforma deseja ser homologado como requisito do material a ser depositado.

Quando aplicável e possível, na arquitetura da biblioteca criptográfica, os requisitos funcionais podem estar disponíveis por invocação, via API, nas seguintes plataformas dos sistemas operacionais:

- “Linux kernel 2.4 ou versões superiores”;
- “Microsoft Windows 2000 ou versões superiores”

No caso de plataforma descontinuada (tais como Windows 98 SE) ou outra plataforma disponível específica (tais como Solaris, Mainframe, etc), o fabricante deverá fornecer o ambiente e treinamento (nos casos em que a equipe de homologação do LSI-TEC LEA não possui conhecimento da plataforma em questão) para que a homologação seja efetuada de acordo com a disponibilidade.

4 Referências normativas

[1] **Cryptographic Service Providers – MSDN (Microsoft Developer Network).** Disponível em: <http://msdn2.microsoft.com/en-us/library/aa380245.aspx>. Acesso em: 20.jul.2007.

[2] COMITÊ GESTOR DA ICP-BRASIL. **DOC ICP-01.01: Padrões e Algoritmos Criptográficos da Infra-Estrutura de Chaves Públicas Brasileira (ICP-BRASIL).** Versão 1.0. Brasília. ICP-BRASIL: 2006.

[3] MESSIER, Matt e VIEGA, John. **Secure Programming Cookbook For C And C++**. O'reilly Publisher: July 2003. ISBN 0-596-00394-3.

[4] [ITI] GLOSSÁRIO ICP-BR – INFRA-ESTRUTURA DE CHAVES PÚBLICAS BRASILEIRAS. **Glossário ICP-Brasil.** Versão 1.2. Brasília. ICP – BR: 2007.

[5] [NIST FIPS 198] **The Keyed-Hash Message Authentication Code (HMAC)**. 2002. Disponível em: <http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf>. Acesso em: 20.jul.2007.

[6] [NIST FIPS 180-2] **Secure Hash Standard (SHA).** 2001. Disponível em: <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>. Acesso em: 20.jul.2007.

[7] [FIPS / NIST] DEPARTMENT OF COMMERCE, NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY, [ITL] INFORMATION TECHNOLOGY LABORATORY. **Federal Information Processing Standards Publication.** Washington. US Government Printing Office: 2001. Disponível em: <http://www.itl.nist.gov/fipspubs/>. Acesso em: 20.jul.2007.

[8] [NIST. FIPS 46-3]. **Data Encryption Standard (DES).** 1999. Disponível em: <http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf>. Acesso em: 20.jul.2007

[9] [RSA LABORATORIES] **PKCS#1: RSA Cryptography Standard.** Version 2.1. 2002. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf>. Acesso em: 30.nov.2006.

[10] [NIST FIPS 186-2] **Digital Signatura Standard (DSS)**. 2001. Disponível em: <http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf>. Acesso em: 20.jul.2007.

[11] [NIST FIPS 196] **Entity Authentication Using Public Key Criptography**. 1997. Disponível em: <http://csrc.nist.gov/publications/fips/fips196/fips196.pdf>. Acesso em: 20.jul.2007.

[12] [NIST FIPS 197] **Advanced Encryption Standard (AES)**. 2001. Disponível em: <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>. Acesso em: 20.jul.2007.

[13] [ANSI. X9.31] AMERICAN NATIONAL STANDARDS INSTITUTE. **Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA).** 1998.

[14] [NIST Special Publication 800-38B] **Recommendation for Block Cipher Modes of Operation - The CMAC Mode for Authentication.** 2005. Disponível em: <http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf>. Acesso em: 20.jul.2007.

[15] [NIST / FIPS Special Publication 800-38C] NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. **Counter with Cipher Block Chaining-Message Authentication Code (CCM)**. 2004. Disponível em: <http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C.pdf>. Acesso em: 23.jul.2007.

[16] [RSA LABORATORIES] **PKCS#5: Password-Based Cryptography Standard.** Version 2.0. 1999. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-5v2/pkcs5v2-0.pdf>. Acesso em: 30.nov.2006.

[17] [RSA LABORATORIES] **PKCS#12: Personal Information Exchange Syntax Standard**. Version 1.0. 1999. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-12/pkcs-12v1.pdf>. Acesso em: 04.jul.2007.

[18] [RSA LABORATORIES] **CMS: Cryptographic Message Syntax Standard.** Version 1.5. 1993. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/ps/pkcs-7.ps>. Acesso em: 27.abril.2007.

[19] [RSA LABORATORIES] **PKCS#8: Private-Key Information Syntax Standard.** Version 1.2. 1993. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/ps/pkcs-8.ps>. Acesso em: 27.abril.2007.

[20] **Cryptography (Windows) – MSDN (Microsoft Developer Network).** Disponível em: <http://msdn2.microsoft.com/en-us/library/aa380255.aspx>. Acesso em: 20.jul.2007.

[21] [RSA LABORATORIES] **PKCS#11: Cryptographic Token Interface Standard**. Version 2.0. 1997. Disponível em: <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/pkcs11v2.pdf> Acesso em: 04.jul.2007.

[22] **Java Cryptography Extension (JCE) for the Java 2 SDK,** versão 1.4. Disponível em: <http://java.sun.com/products/jce/index-14.html> Acesso em: 20.jul.2007.

[23] **[OpenSSL FIPS 1402] Security Policy Object Module By the Open Source Software Institute** - Version 1.0a, 2006. Disponível em <http://csrc.nist.gov/cryptval/140-1/140sp/140sp642.pdf> Acesso em: 20.jul.2007.

[24] [LEA] LABORATÓRIO DE ENSAIOS E AUDITORIA. N**orma de Elaboração de Documentos**. versão 2.0. São Paulo. LEA: 2007.

[25] [ICP-BRASIL] COMITÊ GESTOR DA ICP-BRASIL. **Doc ICP-01.01. Padrões e Algoritmos Criptográficos da Infra-Estrutura de Chaves Públicas Brasileira (ICP-Brasil).** Versão 1.0. Brasília. ICP – Brasil: 2006

[26] [ABNT] ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. **NBR 10520: Informação e Documentação: Citações em Documentos: Apresentação**. Rio de Janeiro. ABNT: 2002.

[27] [IN 01/2007 – ITI] INSTITUTO NACIONAL DE TECNOLOGIA DA INFORMAÇÃO. **Instrução normativa 01/2007: Procedimentos administrativos a serem observados nos processos de homologação de sistemas e equipamentos de certificação digital no âmbito da ICP-Brasil.** DOC-ICP-10.01 versão 2.1. Brasília. ICP-Brasil: 2007

[28] [IN 02/2007 – ITI] INSTITUTO NACIONAL DE TECNOLOGIA DA INFORMAÇÃO. **Instrução normativa 02/2007: Estrutura normativa técnica e níveis de segurança de homologação a serem utilizados nos processos de homologação de sistemas e equipamentos de certificação digital no âmbito ICP-Brasil.** DOC-ICP-10.02 versão 2.0. Brasília. ICP-Brasil: 2007

[29] [IN 06/2007 – ITI] INSTITUTO NACIONAL DE TECNOLOGIA DA INFORMAÇÃO. **Instrução normativa 06/2007: Padrões e procedimentos técnicos a serem observados nos processos de homologação de bibliotecas criptográficas e softwares provedores de serviços criptográficos no âmbito da ICP-Brasil.** DOC-ICP-10.06 versão 1.0. Brasília. ICP-Brasil: 2007