

# Middleware Universale

## Indice

|  |    |
|--|----|
| DEFINIZIONI, ACRONIMI, ABBREVIAZIONI .....                                     | 3  |
| MIDDLEWARE UNIVERSALE.....   | 4  |
| CARATTERISTICHE DEL MODULO CSP .....   | 5  |
| CARATTERISTICHE DEL MODULO PKCS#11 .....                                       | 5  |
| SUPPORTO ALLE CURVE ELLITTICHE .....   | 7  |
| ESEMPIO DI TEMPLATE PKCS#11 PER L'IMPORTAZIONE DI UNA CHIAVE PRIVATA ECC:..... | 8  |
| ESEMPIO DI TEMPLATE PKCS#11 PER GENERAZIONE COPPIA DI CHIAVI ECC: .....        | 8  |
| ESEMPIO DI GENERAZIONE COPPIA DI CHIAVI ECC CON MICROSOFT CNG: .....           | 9  |
| CARATTERISTICHE DEL MODULO DI GESTIONE DEL PIN.....                            | 10 |
| INSTALLAZIONE .....  | 12 |
| MICROSOFT WINDOWS .....  | 12 |
| APPLE MACOS .....  | 12 |
| DISTRIBUZIONI LINUX .....  | 12 |
| FILE DI CONFIGURAZIONE DEL MU.....   | 13 |

## Definizioni, Acronimi, abbreviazioni

|                                  |  |
|----------------------------------|--|
| <b>PKCS#11</b>                   | interfaccia di programmazione (API) standard, multi piattaforma, per l'accesso a generici token crittografici, quali le smart card, sviluppata da RSA  |
| <b>Libreria o modulo PKCS#11</b> | Modulo software che implementa della API PKCS#11 specifica per uno o più token crittografici di un determinato produttore.   |
| <b>CSP</b>                       | Interfaccia di programmazione (API) proprietaria Microsoft che permette di aggiungere funzioni crittografiche, anche fornite da hardware come le smart card, nei sistemi operativi Windows; un CSP è un modulo software che può essere utilizzato esclusivamente tramite API crittografiche del sistema operativo (CryptoAPI)              |
| <b>CryptoSPI</b>                 | Acronimo che sta per Crypto Service Provider Interface, indica in modo specifico la API che un modulo CSP deve implementare.   |
| <b>API</b>                       | Acronimo che sta per Application Programming Interface, indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per un determinato compito  |
| <b>CryptoAPI</b>                 | Acronimo che sta per Cryptographic Application Programming Interface; rappresenta l'interfaccia di programmazione che i sistemi operativi Windows mettono a disposizione delle applicazioni per l'uso della crittografia.  |
| <b>CryptoAPI NG (CNG)</b>        | Acronimo che sta per Cryptographic Application Programming Interface Next Generation; rappresenta l'interfaccia di programmazione che i sistemi operativi Windows 8/10 mettono a disposizione delle applicazioni per l'uso della crittografia avanzata, tra cui AES256, SHA256, RSA PSS, Curve Ellittiche.                                 |
| <b>Tray-bar</b>                  | Nei sistemi operativi Microsoft Windows rappresenta l'area localizzata tra la barra delle applicazioni e l'orologio, in cui le applicazioni possono installare un'icona che le rappresenti quando non sono in primo piano.   |
| <b>PIN</b>                       | Acronimo di Personal Identification Number; nell'ambito delle smart card rappresenta un codice che permette di accedere alle funzioni il cui uso è riservato esclusivamente al possessore della carta; generalmente il PIN si blocca dopo un numero predefinito di tentativi con valori errati, bloccando dunque l'accesso alla smart card |
| <b>PUK</b>                       | Acronimo che sta per Pin Unblocking Key; nell'ambito delle smart card è uno codice del tutto simile al PIN, il cui scopo generalmente è esclusivamente quello di sbloccare un PIN bloccato dai troppi tentativi con valori non corretti.   |
| <b>CNS</b>                       | Carta Nazionale dei Servizi; in questo documento può indicare la specifica CNS rilasciata dal CNIPA oppure le sole specifiche del file system.   |
| <b>MU</b>                        | Middleware Universale, il software in oggetto  |
| <b>file system</b>               | Nell'ambito delle smart card indica la struttura ed il formato dei file e dei dati presenti un una smart card e che servono a implementare una determinata funzionalità/applicazione.  |
| <b>ATR</b>                       | Acronimo che sta per <b>Answer To Reset</b> ; è un codice restituito da una smart card quando viene inserita nel lettore o resettata che viene spesso utilizzato per identificare il tipo di smart card in maniera univoca.  |
| <b>Store di certificati</b>      | Rappresenta il punto in cui il sistema operativo Windows memorizza i certificati di sicurezza, in modo che possano essere utilizzati dalle applicazioni che fanno uso delle CryptoAPI  |
| <b>SSL</b>                       | Acronimo che sta per <b>Secure Socket Layer</b> : protocollo standard di comunicazione cifrata che permette anche la mutua autenticazione tra le parti comunicanti (SSL Server authentication e Client Authentication)   |
| <b>TLS</b>                       | Acronimo che sta per <b>Transport Layer Security</b> : è il successore del protocollo SSL.   |
| <b>FS</b>                        | Acronimo che sta per File System   |
| <b>DS</b>                        | Acronimo che sta per Digital Signature: indica il file system di firma digitale (in questo documento può indicare anche un file system non CNS)  |

## Middleware Universale

Il "Middleware Universale" (in seguito MU) consiste in:

- Modulo di libreria che espone una API compatibile con specifica di standard PKCS#11 v2.11.
- Modulo di sistema che espone una API compatibile con specifica di standard Microsoft Cryptographic Service Provider (CryptoSPI/2006).
- Modulo di sistema "certificate store" che implementa un meccanismo di importazione automatica dei certificati nello store utente windows.
- Modulo utente per la gestione PIN/PUK (cambio PIN, sblocco PIN).

Tali moduli offrono ai software che ne utilizzano le relative interfacce di programmazione la possibilità di utilizzare le smart card supportate come token crittografici.

Le funzionalità garantite dall'interfaccia PKCS#11 sono le seguenti.

- Firma digitale e decifra tramite la chiave RSA di autenticazione CNS,
- Lettura e scrittura del certificato di autenticazione CNS,
- Lettura e scrittura del file dei dati personali CNS (PDATA)
- Generazione di una coppia di chiavi RSA di autenticazione CNS
- Cancellazione degli oggetti

Le funzionalità garantite dall'interfaccia CSP sono le seguenti:

- Firma digitale e decifra tramite la chiave RSA di autenticazione CNS,
- Lettura e del certificato di autenticazione CNS,

Il modulo utente di gestione PIN-PUK è una applicazione attivabile mediante una icona presente nella tray-bar. Consente il cambio del PIN, lo sblocco del PIN mediante PUK.

Il MU comunica con le smart card attraverso un lettore di smart card controllato dallo strato PC/SC implementato nei sistemi operativi Microsoft.

I moduli che espongono le due interfacce PKCS#11 e CSP si appoggiano su di un unico "motore" che gestisce gli oggetti di sicurezza presenti sulle smart card CNS. Tale motore ha la possibilità di essere esteso attraverso un meccanismo di "plug-in".

Il modulo che espone l'interfaccia PKCS#11 viene interfacciato direttamente dalle applicazioni che utilizzano tale API per utilizzare i servizi offerti dalle smart card. Attraverso l'interfaccia PKCS#11 è possibile leggere il certificato di autenticazione e utilizzare la chiave privata RSA presente sulla smart card.

Il modulo che espone l'interfaccia CryptoSPI viene interfacciato dal sistema operativo che integra le funzioni esposte con quelle di più alto livello del CSP che poi saranno messe a disposizione delle applicazioni.

Il modulo "certificate store" si occupa di aggiungere e rimuovere i certificati nello store "My", quando viene inserita una smart card riconosciuta. Tale modulo consente l'uso dei certificati presenti sulla smart card da parte delle applicazioni che fanno uso delle CryptoAPI in maniera del tutto automatica e trasparente. Alla rimozione della smart card i certificati verranno automaticamente rimossi. Il modulo "certificate store" può essere disattivato ed in tal caso sarà il sistema operativo, all'inserimento della carta, ad importare automaticamente i certificati nello store dei certificati personali; in tal caso i certificati saranno visibili nel sistema anche dopo la rimozione della smart card.

Il modulo "gestione PIN/PUK" non espone una interfaccia di programmazione (API) ma ha una interfaccia utente (GUI) che consente all'utente di svolgere le minimali attività per il cambio e lo sblocco del PIN. L'interfaccia utente può essere attivata intervenendo su di una icona presente nella tray-bar che fornisce inoltre anche informazioni sull'attuale stato di attività della smart card.

## Sistemi operativi supportati

- Windows 8, intel 64bit
- Windows 8.1, intel 64bit
- Windows 10, intel 64bit
- Distribuzioni Linux con Kernel versione 4 o successivi, intel 64bit
- OSX da 10.14 a 11.0, intel 64bit

## Caratteristiche del modulo CSP

1. Libreria compatibile con la specifica CSP di Microsoft

1. Funzionamento del CSP limitato all'utilizzo nei seguenti contesti applicativi:
  - a. autenticazione HTTPS/TLS1.2 con Google Chrome
  - b. funzione di "Firma leggera per Attestazione" per applicazioni web
  - c. funzione di "Firma Forte" per applicazioni di firma digitale
2. Sistemi operativi sui quali Bit4id certifica il pieno funzionamento ed il superamento dei propri test prima del rilascio:
  - a. Windows 10
3. Rilascio libreria in formato binario sotto forma di libreria a "link dinamico".
4. Applicazione con la quale Bit4id certifica il pieno funzionamento con il superamento dei propri test prima del rilascio: Google Chrome
5. Caricamento dei certificati presenti sulla smart card in maniera trasparente per l'utente all'inserimento nel lettore.

L'interfaccia CSP implementa le seguenti funzioni:

- CryptGetProvParam (PP\_NAME, PP\_CONTAINER, PP\_UNIQUE\_CONTAINER)
- CryptSetProvParam (PP\_SIGNATURE\_PIN)
- CryptAcquireContext
- CryptReleaseContext
- CryptCreateHash
- CryptSetHashParam
- CryptGetHashParam (HP\_ALGID, HP\_HASHSIZE, HP\_HASHVAL)
- CryptHashData,
- CryptDestroyHash
- CryptSignHash
- CryptGetUserKey
- CryptDestroyKey
- CryptGetKeyParam (KP\_CERTIFICATE)
- CryptExportKey (PUBLICKEYBLOB)

## Caratteristiche del modulo PKCS#11

1. Libreria compatibile con la specifica PKCS#11 di RSA (v. 2.11)
2. Funzionamento del PKCS#11 limitato all'utilizzo nei seguenti contesti applicativi:
  - a. autenticazione HTTPS/TLS1.2 con Google Chrome
  - b. funzione di "Firma leggera per Attestazione" per applicazioni web
  - c. funzione di "Firma Forte" per applicazioni di firma digitale
  - d. funzione di enrollment di credenziali CNS, di firma forte, di autenticazione tramite Software CA di Infocamere
  - e. Lettura della smart card tramite Token Manager Alladin eToken View ver. 3.50.138
3. Sistemi operativi sui quali Bit4id certifica il pieno funzionamento ed il superamento dei propri test prima del rilascio:
  - a. Windows 10
4. Rilascio del modulo PKCS#11 in formato binario come libreria a "link dinamico".
5. Applicazione con la quale Bit4id certifica il pieno funzionamento con il superamento dei propri test prima del rilascio: Browser Mozilla Firefox

L'interfaccia PKCS#11 implementa le seguenti funzioni:

- C\_Initialize
- C\_Finalize
- C\_GetInfo
- C\_GetSlotList
- C\_GetSlotInfo
- C\_GetTokenInfo
- C\_GetMechanismList
- C\_GetMechanismInfo
- C\_OpenSession
- C\_CloseSession
- C\_CloseAllSessions
- C\_GetSessionInfo
- C\_Login
- C\_Logout
- C\_SetPIN
- C\_FindObjectsInit
- C\_FindObjects
- C\_FindObjectsFinal
- C\_GetAttributeValue

- C\_GetObjectSize
- C\_SignInit (meccanismi RSA\_PKCS, RSA\_SHA1\_PKCS, RSA\_SHA\_256\_PKCS, CKM\_ECDSA)
- C\_Sign
- C\_DecryptInit (meccanismo RSA\_PKCS)
- C\_Decrypt
- C\_DigestInit (meccanismo SHA\_1, SHA\_256),
- C\_Digest
- C\_DigestUpdate
- C\_DigestFinal
- C\_CreateObject
- C\_GenerateKeyPair (CKM\_RSA\_PKCS\_KEY\_PAIR\_GEN, CKM\_EC\_KEY\_PAIR\_GEN)
- C\_SetAttributeValue
- C\_DestroyObject

**Mechanisms supportati:**

CKM\_RSA\_PKCS\_KEY\_PAIR\_GEN

CKM\_EC\_KEY\_PAIR\_GEN (opzionale)

CKM\_RSA\_PKCS (firma, decifra)

CKM\_ECDSA (firma)

RSA\_SHA1\_PKCS (firma)

RSA\_SHA\_256\_PKCS (firma)

CKM\_SHA\_1 (digest)

CKM\_SHA\_256 (digest)

**Creazione dei diversi tipi di oggetti: CNS, Firma Forte, Autenticazione**

Per stabilire la tipologia di oggetto da creare vengono utilizzati alcuni attributi specificati alla creazione o generazione degli oggetti:

- Quando il CKA\_ID di un oggetto ha come valore '**CNS0**' viene creato un oggetto nel filesystem CNS; se l'oggetto che si vuole creare esiste già viene generato un errore.
- Quando CKA\_LABEL di un data object (un oggetto per cui CKA\_CLASS=CKO\_DATA) è impostato a 'PDATA' viene creato il file dati personali del filesystem CNS. Il valore dei dati personali va specificato come attributo CKA\_VALUE.
- Quando il CKA\_ID è 'DS', oppure 'DS0', 'DS1', 'DS2'; ovvero quando CKA\_LABEL è 'Firma\_CNS' oppure 'Firma\_CNS0'..'Firma\_CNS2': viene creato un oggetto di firma forte
- In tutti gli altri casi viene creato un oggetto di autenticazione non CNS.

---

## Supporto alle Curve Ellittiche

Se la Smart card in uso è abilitata all'uso delle curve ellittiche sarà possibile generare, importare ed usare chiavi appartenenti alle seguenti curve:

Curve supportate da interfacce Microsoft CNG e PKCS#11

NIST P256 (secp256r1, OID "1.2.840.10045.3.1.7")

NIST P384 (secp384r1, OID "1.3.132.0.34")

NIST P521 (secp521r1, OID "1.3.132.0.35")

Curve supportate solo da interfaccia PKCS#11:

Secp256k1 (curva dei "BITCOIN", OID "1.3.132.0.10" )

L'interfaccia PKCS#11 supporta il campo CKA\_EC\_PARAMS nel suo formato DER NAMED CURVE ed il tipo di chiave (attributo CKA\_KEY\_TYPE) CKK\_EC.

Per l'uso e la creazione di questo tipo di oggetti fare riferimento alla documentazione Microsoft CNG oppure alla specifica PKCS#11 v3.0

## ESEMPIO di template PKCS#11 per l'importazione di una chiave privata ECC

```
CK_KEY_TYPE ckkType = CKK_ECDSA;
CK_ULONG ckoClassPrivateKey = CKO_PRIVATE_KEY;
CK_BYTE pbPrivValue = {}; // private key value
CK_BYTE pEcParams = {}; // DER Representation of OID
CK_ATTRIBUTE pPrivateKeyTemplate[] = {
    {CKA_ID, pbCkald, cbCkald},
    {CKA_LABEL, (void*)sz_ID, (CK_ULONG)strlen(sz_ID)},
    {CKA_CLASS, &ckoClassPrivateKey, sizeof(ckoClassPrivateKey)},
    {CKA_KEY_TYPE, &ckkType, sizeof(ckkType)},
    {CKA_TOKEN, &cbbTrue, sizeof(cbbTrue)},
    {CKA_PRIVATE, &cbbTrue, sizeof(cbbTrue)},
    {CKA_SENSITIVE, &cbbTrue, sizeof(cbbTrue)},
    {CKA_DECRYPT, &cbbFalse, sizeof(CK_BBOOL)},
    {CKA_SIGN, &cbbTrue, sizeof(CK_BBOOL)},
    {CKA_VALUE, (void*)pbPrivValue, (CK_ULONG)dwKeyBytes},
    {CKA_EC_PARAMS, (void*)pEcParams, (CK_ULONG)sizeof(pEcParams)},
};
```

## ESEMPIO di template PKCS#11 per Generazione coppia di chiavi ECC

```
CK_BYTE * pEcParams = {}; // DER Representation of OID
CK_ULONG ulEcParamsLen = sizeof(pEcParams);
CK_ULONG ulKeyBits = 256; // replace with actual key bits

CK_ATTRIBUTE PublicKeyTemplateECC[] = {
    { CKA_MODULUS_BITS, &ulKeyBits , sizeof(ulKeyBits) },
    { CKA_TOKEN, &CK_True , 1 },
    { CKA_DERIVE, &CK_False, 1 },
    { CKA_WRAP, &CK_False, 1 },
    { CKA_VERIFY, &CK_True, 1 },
    { CKA_ENCRYPT, &CK_False, 1 },
    { CKA_EC_PARAMS, (void*)pEcParams, (CK_ULONG)ulEcParamsLen },
    { CKA_ID, (void*)pCKA_ID, uCKA_ID },
};

CK_ATTRIBUTE PrivateKeyTemplateECC[] = {
    { CKA_SENSITIVE, &CK_True, 1 },
    { CKA_TOKEN, &CK_True, 1 },
    { CKA_PRIVATE, &CK_True, 1 },
    { CKA_DERIVE, &CK_False, 1 },
    { CKA_UNWRAP, &CK_False, 1 },
    { CKA_SIGN, &CK_True, 1 },
    { CKA_DECRYPT, &CK_False, 1 },
    { CKA_EC_PARAMS, (void*)pEcParams, (CK_ULONG)ulEcParamsLen },
    { CKA_ID, (void*)pCKA_ID, uCKA_ID },
    { CKA_LABEL, (void*)pCKA_LABEL_PRIVKEY, uCKA_LABEL_PRIVKEY },
};
```

---

## ESEMPIO di Generazione coppia di chiavi ECC con Microsoft CNG

```
WCHAR*pAlgoName=BCRYPT_ECDSA_P256_ALGORITHM;
WCHAR* wszKeyName = pAlgoName;
NCRYPT_PROV_HANDLE hNHandle = 0;
LPCWSTR szProviderName = L"Bit4id Universal Middleware Provider";
SECURITY_STATUS status;

status = NCryptOpenStorageProvider(&hNHandle, szProviderName, 0);
if (status != ERROR_SUCCESS)
{
    exit(1);
}
NCRYPT_KEY_HANDLE hKey = NULL;
status=NCryptCreatePersistedKey(hNHandle,&hKey,pAlgoName,wszKeyName,0, 0);
```



## Caratteristiche del modulo di gestione del PIN

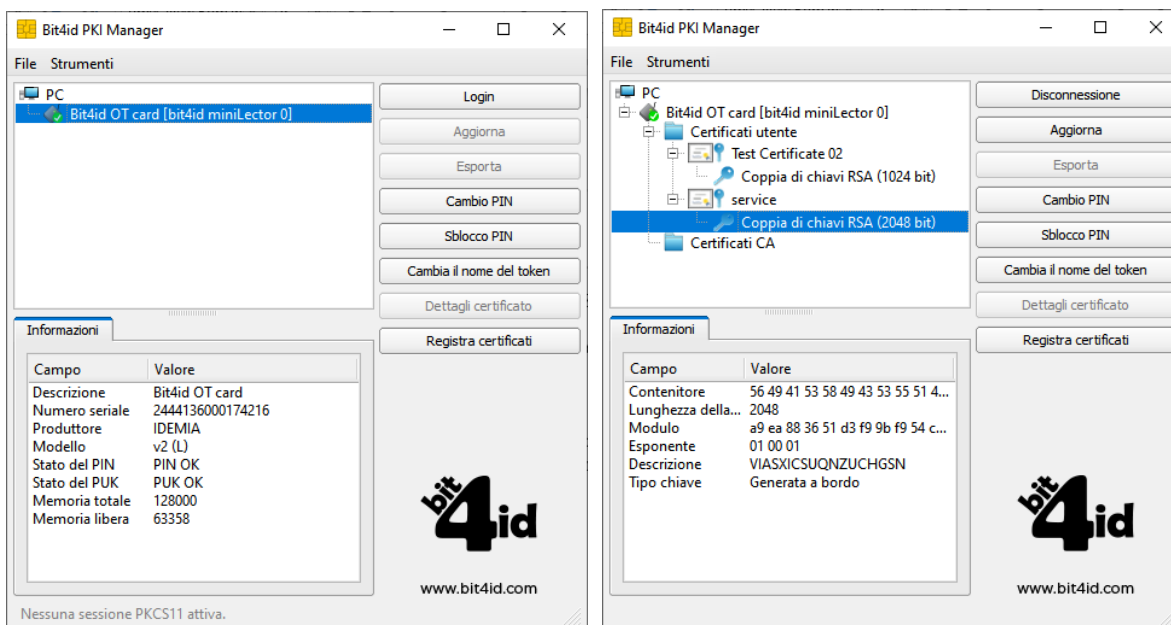
Il modulo di gestione PIN è un'applicazione dotata di interfaccia utente (GUI) che permette di gestire il PIN delle smart card supportate dal MU.

L'applicazione permette di eseguire le seguenti operazioni:

- Ottenere informazioni sulla smart card inserita, quali il numero di serie, il modello e il lettore in uso.
- Cambiare il valore del PIN
- Sbloccare il PIN bloccato usando il codice PUK

Tutte le operazioni sono eseguite utilizzando l'interfaccia PKCS#11 messa a disposizione dal MU.

L'applicazione è caratterizzata da un'interfaccia che mostra in alto a sinistra un albero che contiene tutti i lettori di smartcard collegati, mentre sulla destra sono presenti i comandi che è possibile eseguire.



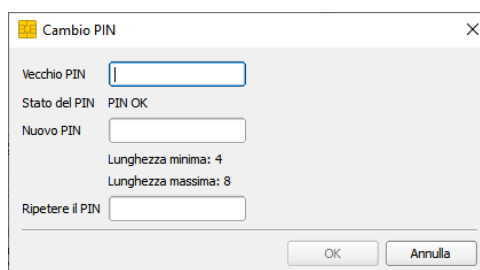
Per mostrare il contenuto di una carta, dopo aver selezionato il lettore, è necessario cliccare sul tasto “login” ed inserire il PIN. In seguito, verrà mostrato il contenuto della carta suddiviso in certificati utente e certificati di CA.

Cliccando su un oggetto in basso in verranno mostrati i dettagli, come il nome, il valore della chiave o il nome associato al certificato (Common Name).

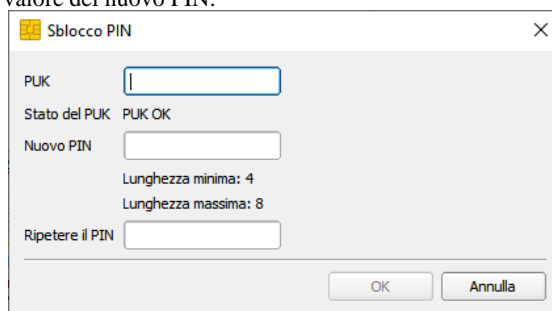
Il tasto aggiorna permette di rileggere la carta inserita, mentre il tasto “Registra certificati” esegue manualmente l'operazione che normalmente è eseguita in automatico all'inserimento di una carta, cioè la propagazione dei certificati utente presenti sulla smart card all'interno degli store di Microsoft Windows, permettendone l'uso con applicazioni compatibili come Google Chrome o Adobe Reader.

Cambia Nome token permette di cambiare l'etichetta della smart card (operazione non supportata da tutti i modelli).

Il tasto “Cambio PIN” permette di modificare il valore del PIN utente. Verrà richiesto il PIN corrente e di inserire e confermare il nuovo PIN.



In caso di PIN bloccato oppure se non si ricorda più il valore del PIN, conoscendo il valore del codice di sblocco (PUK) sarà possibile reimpostare il PIN usando il pulsante “Sblocco PIN”. In questo caso verrà richiesto il valore del codice di sblocco, o PUK, e di inserire e confermare il valore del nuovo PIN:



A screenshot of a software dialog box titled "Sblocco PIN" with a close button (X) in the top right corner. The dialog contains the following fields and labels:

- PUK**: A text input field with a cursor.
- Stato del PUK**: Labeled "PUK OK".
- Nuovo PIN**: A text input field.
- Lunghezza minima**: 4
- Lunghezza massima**: 8
- Ripetere il PIN**: A text input field.

At the bottom right, there are two buttons: "OK" and "Annulla".

## Installazione

### Microsoft Windows

Il MU è distribuito sotto forma di installer per Windows.  
È necessario eseguire l'installer con i diritti amministrativi.

Su Windows il MU è composto dai seguenti moduli:

| Nome   | Destinazione      |
|--|-------------------|
| bit4xpki.dll   | %WINDIR%\System32 |
| Descrizione  |                   |
| Il modulo principale del MU va installato nella cartella <b>System32</b> .<br>Tale modulo contiene l'interfaccia PKCS#11 |                   |

### Apple MacOS

La versione OSX del MU è distribuita sotto forma di pacchetto di installazione Apple (**PKG**), contenuto in un file immagine disco (**DMG**).

Su MacOS il MU è composto dai seguenti moduli:

| Nome   | Destinazione                             |
|--|--|
| libbit4xpki.dylib  | /Library/bit4id/pkcs11<br>/usr/local/lib |
| Descrizione  |  |
| Il modulo principale del MU. Tale modulo contiene l'interfaccia PKCS#11. |  |

### Distribuzioni Linux

La versione Linux del MU è distribuita o sotto forma di librerie sciolte oppure sotto forma di pacchetti specifici per la distribuzione.

In particolare sono distribuiti i pacchetti in formato DEB, per distribuzioni Debian/Ubuntu e compatibili. Ed in formato RPM, per distribuzioni Fedora/RedHat e compatibili.

Poiché non tutte le distribuzioni sono pensate per installare direttamente dei pacchetti DEB, potrebbe essere necessario installare le dipendenze in maniera manuale. Fare riferimento alla documentazione della distribuzione per le istruzioni precise.

Qui di seguito elenchiamo i passi necessari sui sistemi Debian ed Ubuntu. Da terminale, come **root** eseguire:

```
#> apt-get install libccid
```

```
#> apt-get install libglade2-0
```

Su Linux il MU è composto dai seguenti moduli:

| Nome   | Destinazione                  |
|--|-------------------------------|
| libbit4xpki.so   | /usr/lib/bit4id/<br>/usr/lib/ |
| Descrizione  |                               |
| Il modulo principale del MU. Tale modulo contiene l'interfaccia PKCS#11. |                               |

## File di configurazione del MU

Il MU ha un file di configurazione che permette di variarne il comportamento.

Il file di configurazione ha lo stesso nome della libreria con aggiunta dell'estensione **".conf"**. Il file DEVE sempre trovarsi nella stessa cartella in cui si trova il modulo principale.

### Formato del file di configurazione

Il file di configurazione è composto da una serie di righe ed ogni riga contiene una stringa del tipo NomeValore=Valore. Sono ammesse righe vuote.

### Contenuto del file di configurazione

Le impostazioni utilizzabili nel file di configurazione sono le seguenti:

| Nome                 | Possibili Valori | Descrizione   |
|----------------------|------------------|---|
| oDSPinIsCnsPin       | true o false     | Se impostato a "true" viene indicato forzata l'uguaglianza del pin primario con il pin di firma.<br>Valore di default: false  |
| oDSPinUseGui         | true o false     | Se impostato a "false" la libreria PKCS#11 gestisce un eventuale PIN secondario che protegge gli oggetti di firma in maniera compliant col la specifica PKCS#11, restituendo dunque l'errore CKR_USER_NOT_LOGGED_IN in seguito ad una chiamata alla funzione C_Sign su una chiave protetta da quel PIN, aspettandosi immediatamente dopo una chiamata a C_Login(CKU_CONTEXT_SPECIFIC) col PIN di firma. Se impostato a "true" verrà utilizzato il plugin per la richiesta del PIN secondario tramite interfaccia grafica (GUI). Il parametro è ignorato quando DSPinIsCnsPin è impostato a "true".<br>Valore di default: true |
| oHideCacheDsPinCheck | true o false     | Se impostato a true viene nascosto dall'interfaccia utente che richiede il PIN di firma forte il checkbox che permette di utilizzare lo stesso PIN per più operazioni di firma in sequenza, fino al logout dalla carta.<br>Valore di default: false   |