uFR digital signing and verification tools

DL Signer cards provides digital signing of data and documents in the cards themselves using RSA or ECDSA asymmetric cryptographic algorithms. PKI infrastructure is supported and in the DL Signer cards it is possible to store X.509 certificates that are related to pairs of cryptographic keys generated in the cards itself. It is supported to store all X.509 certificates that make up the chain of trust from the root certificate to the end-entity certificate.

The public key which is generated in the DL Signer cards is placed in the body of the request while creating the certificate signing requirement (hereinafter CSR). The request is signed in the card itself with an appropriate private key that never leaves the card itself and in no way it can be read after generating key pairs. Further, the CSR is sent to the certification body in order to create and sign the X.509 certificate based on it. This end-entity certificate is placed in the DL Signer card with other certificates from the chain of trust and is ready to digitally sign data and documents. The user can send CSR to any certification body whose services he wishes to use. Digital Logic has provided a mechanism for issuing end-entity certificates for the purpose of testing the system. One of the basic characteristics of the end-entity certificate is that the private key, which is paired with the public key that such certificates contain, must not be used to sign other certificates.

The Windows software tools that initiate the generation of cryptographic keys pairs, generates CSRs, manages the PIN and PUK codes of the DL Signer cards, manipulates the contents of the X.509 certificates and signs the data and files, is distributed as "ufr-signer".

"Signature-verifier" is a Windows application validating RSA and ECDSA digital signatures.

Digital signing and validation of signatures can also be done from Adobe Acrobat Reader DC application using ufr-pkcs11 module that we developed for this purpose. Our PKCS#11 module can also be used with popular Mozilla's e-mail client and web browser, as well as with other software tools that are compatible with the PKCS#11 specification.

We also provided web services for online checking of X.509 certificates and signed pdf files.

uFR Signer

"uFR Signer" is a software tool that initiates the generation of cryptographic key pairs, generates CSR requests, serves to manage the PIN and PUK codes of the DL Signer cards, manipulates the contents of the X.509 certificates and signs the data and files.

The application is divided into several logical units using tab control visual component. The tabs are labeled by the names of these units:

"RSA Keys" and "EC Keys" are used to create and manipulate RSA or ECC key pairs.

RSA (Rivest, Shamir, & Adleman) and ECC (Elliptic Curve Cryptography) represent contemporary asymmetric cryptographic algorithms. DL Signer cards supports storing of 3 RSA and 3 ECC keys separately. Each of the cryptographic keys can be of different lengths and characteristics and it is indicated by a cryptographic algorithm and a key index.

The "PIN Codes" tab refers to managing and logging the user PIN code to the DL Signer card located in the uFR reader field. The PIN is an abbreviation of "Personal Identification Number". In addition to PIN codes, on this tab you can also unlocks eventually blocked cards using PUK codes. PUK stands for "PIN Unlock Key".

The "Card Objects" tab is used to manage CA certificates, and end-entity certificates that are associated with their respective cryptographic keys through their indexes. Certificates must be in

accordance with the X.509 version 3. The end-entity credentials must contain a public key originally generated in the DL Signer card in pairs with the associated private key. The primary purpose of the certificate is for use with preparation for signing through the PKCS#11 module. Presence of the X.509 certificates is not mandatory for use DL Signer Card with the proprietary applications.

On the "Signature" tab there are options for creating digital signatures. It is possible to sign an array of bytes, a text input, or a file. These signatures can be verified using the "signature-verifier" application.

To address the performance issues efficiently, DL Signer Cards are designed to sign blocks of data as concisely as possible. Therefore, it is a practice to digitally sign the data with an RSA algorithm as defined by the PKCS#1 v1.5 scheme. For ECDSA algorithm, digital signature generation procedure, padding and alignment mechanisms are defined in RFC 6979.

The latest version of the uFRSigner application is 1.5.3.0 and it is necessary to use the uFCoder library version 5.0.1 or higher and the uFR firmware version 5.0.7 or higher.

PIN Codes

PIN code is an abbreviation of "Personal Identification Number". The DL Signer Card contains 2 different PIN codes. These are SO (Security Officer) PIN and user PIN code. The so-called "Security Officer" is actually a user who have administrative privileges for accessing security objects on the DL Signer Card. SO PIN code should be different from the user PIN code.

"Security Officer" is required to be logged in to access the card in cases when it is necessary to change the PIN and PUK codes and to change the contents of the storage for the keys and certificates. Loging in with a user PIN code is necessary to get digital signature of a hashed data string.

PIN codes on the DL Signer Cards can have a minimum of 4 characters and a maximum of 8 characters. Here, under the character there is any alphanumerical (case sensitive) or any printable character. Printable characters mainly refer to punctuation marks on the standard keyboards. When changing PIN codes, it is not recommended the use of specific characters that can be found only on individual localized keypads, but only characters that are in ASCII standard and that exist on standard US English keyboards.

In all DL Signer Cards, the default PIN and user PIN codes are set initially, consisting of eight consecutive numerical characters '0' (zero) or "00000000".

The maximum number of incorrect consecutive PIN code entered is 5. If the number of incorrect successive attempts to enter the PIN code is exceeded, that PIN code is blocked. While the PIN code is not blocked, entering the correct PIN code resets the incorrectly entered PIN codes counter.

The only way to unblock your PIN is to enter the correct PUK code. PUK is the abbreviation of "PIN Unlock Key". SO PUK code serves exclusively to unblock SO PIN code and user PUK to unblock user PIN code. In the case of 10 consecutive incorrectly entered PUK codes, the PUK code becomes unusable, and the functionality of the card on which the blocked PIN code relates remains blocked forever.

RSA Keys

On the "RSA Keys" tab there are options for creating and managing RSA keys. Before working with RSA keys it is necessary that the DL Signer Card be in the uFR reader field that is connected to the computer running the ufr-signer application. In addition, SO (Security Officer) is required to be logged in.

EC Keys

On the "EC Keys" tab there are options for creating and managing EC key. Before working with the EC keys, it is necessary that the DL Signer Card be in the uFR reader field that is connected to the computer running the ufr-signer application. In addition, SO (Security Officer) is required to be logged in.

The DL Signer Cards supports the following standard ECC curves:

DL Signer 22:

secp112r1, secp112r2, secp128r1, secp128r2, secp160k1, secp160r1, secp160r2, secp192k1, secp192r1, secp224k1, secp224r1, secp256k1, secp256r1, secp384r1, secp521r1, sect113r1, sect113r2, sect131r1, sect131r2, sect163k1, sect163r1, sect163r2, sect193r1, sect193r2, sect233k1, sect233r1, sect239k1, sect283k1, sect283r1, sect409k1, sect409r1.

DL Signer 30:

secp192k1, secp192r1, secp256k1, secp256r1, secp384r1.

DL Signer 145:

secp160k1, secp160r1, secp160r2, secp192k1, secp192r1, secp224k1, secp224r1, secp256k1, secp256r1, secp384r1, secp521r1.

Generating Certificate Signing Request (CSR)

It is defined by the PKCS#10 standard and represents a standardized record that, among other things, contains basic information about the user of the certificate in the distinguished name. On the basis of the characteristic name, the so-called Subject (subject field) is formed in the final X.509 certificate. Furthermore, CSR records may also contain extensions specified by the X.509

standard that the Certificate Authority (CA) may consider or discard, depending on its own certification policy.The basic part of the CSR is certainly a public key and its parameters. All these data, packaged in the form defined by the PKCS#10 standard, are finally passed through the appropriate cryptographic digest algorithm and the result is signed in the card with the appropriate private key. The digital signature thus acquired becomes an integral part of CSR.

The CSR is sent to the desired issuer of the certificate, which represents the so-called Certificate Authority (CA). The certificate issuer will generate your X.509 end-entity certificate, which is signed by their respective private key, now associated with the public key contained in the appropriate intermediate or root CA certificate. In this way, your end-entity certificate becomes part of the chain of trust guaranteed by the Certification Authority (CA).

The "Certificate Signing Request (CSR)" dialog is a separate group for the input of "Distinguished Name (DN)", "Extensions" and on the upper right side of the groups of combo boxes, for controlling the hash algorithm and definition of the cryptographic key. Below right is a group of buttons for choosing activities related to generating CSR.

A distinguished name, referred to as DN, consists of a Relative Distinguished Name (RDN) group representing the attribute group. When defining a DN, the order of the RDN field is very important. It is important to note that it is possible to repeat a single RDN field several times within the DN. It must be remembered that the Certificate Authority (CA) is not obliged to issue a certificate with exactly the same DN as defined in the CSR, but the DN can form on the basis of its own rules and data obtained during the previously implemented verification of the user's identity .

The formation of the DN is done by selecting the appropriate RDN from the combo box and typing the desired value into the text box. By pressing the "Put" button, the declared RDN will be placed on the list (List Box) that defines DN. If one of the RDN fields has been selected on the list, the new RDN will be inserted between the selected and the previous field. In the event that nothing is selected on the DN list, a new RDN field is inserted at the end of the list. To cancel the selection on the list, press the "Deselect" button. Incorrect RDN can be removed from the list by pressing "Remove". The order of the RDN field can be changed using the "Move Up" and "Move Down" buttons that affect the redundancy of the selected RDN in the list.

Extensions are an optional part of the CSR and the Certification Authority (CA) can consider or reject them, depending on their own certification policy. It is possible to assign more attributes, and it is desirable to define an e-mail address within the subject's alternate subject (subject alternative name, abbreviated subjectAltName), because this is the most common certificate issuer in the usual place for this purpose. For extensions, the order of individual attributes is not important. It is envisaged that the desired purpose of keys, the expanded purpose of keys and statements related to qualified certificates can still be defined. Once again, it should be emphasized that issuers of the certificate can ignore extensions, and only some of them can issue so-called qualified electronic certificates. In any case, within the extensions, the user can indicate the desired elements of the future certificate, but, once again, the final elimination of the X.509 certificate depends exclusively on their issuer and that all details need to be fully familiarized with their policies before the conclusion of the issuing contract.

The choice of the hash algorithm is done from the combo box marked with the "signer digest algorithm". The default choice is SHA-256, which relates to the SHA2 algorithm that has 256 bits at the output or 32 bytes and this is recommended to use for CSR. SHA1 is definitely not recommended anymore, and SHA2 with a higher number of bytes at the output (384 and 512) and the SHA3 algorithm are planned for more frequent use in the future.

The keys and their parameters ("signer cipher algorithm" and "signer key index (card)") can not be changed here and there are already defined on the tab from which you have opened this dialog ("RSA Keys" or "EC Keys") and their purpoise here is only informative. If the "RSA Keys" or "EC Keys" from which you opened the CSR dialog, there was no corresponding public key, previously read from the card, on the label indicating the size of the RSA key, The ECDSA curve will be indicated as "Public key not set". When a public key is not set, it is not possible to execute "Sign and Save CSR", but it is possible to load DN and extensions from an existing CSR or so-called TBS CSR.

TBS CSR is our internal record format so-called "To Be Signed" request that can serve as a template for creating CSR requests with multiple common features. TBS CSR does not contain cryptographic keys but only stores DN and extensions.

Pressing the "Clear Entries" button removes all items in the DN and the extension so that the dialog is prepared for the new entry.

By pressing the "Sign and Save CSR" key, signing the CSR in the card and storing it in the selected file is done. If the card is not in the field of the connected uFR reader or entered the wrong user PIN code, you will receive an error message with the appropriate description.

The last thing to do after generating the CSR is sending it to the certificate issuer in order to to receive the X.509 certificate. You can choose any of the commercial or non-commercial certificate issuers or by pressing the "Get Certificate Online" button you can send the CSR to our web

service in order to obtain a demonstration, test certificate issued by "Digital Logic Ltd." Test Certificates. Demonstration, test certificate is intended for test usage only and its validity period is 3 month. Accompanying root and intermediate CA certificates you can download from http://ca.d-logic.com.

If you click on "Get Certificate Online" button you will be prompted for a previously saved CSR file with a ".pem" extension that will be sent to the HTTP server. If connection with a server on the https://certificates.d-logic.com is successfull and the X.509 certificate is issued, you will be prompted for the file name to save the certificate. Otherwise you will receive an appropriate error message.

X.509 Objects

This tab is intended to manage the X.509 objects, related to the certificates on DL Signer Cards. To read X.509 objects from a card, it is not necessary to be loged in with any of the PIN codes. To change the contents of the storage for the X.509 objects on the card, you need to be loged in with the SO PIN code on the "PIN Codes" page.

On the "X.509 Objects" tab, application immediately tries to read the card which is present in the field of the uFR reader. If there is no DL Signer Card is in the field, an error message will appear, as shown on the picture below:

Error — X

❌ Card error code: 0x00000008

OK
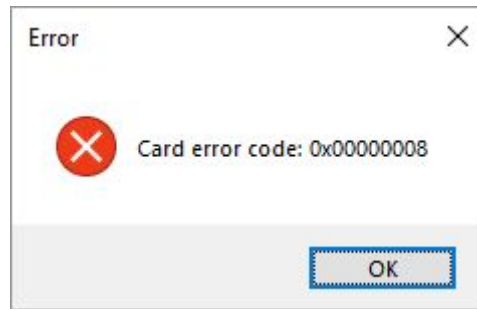
If the X.509 objects storage is successfully read from the card, the certificate display lists will be populated with that content. You can refresh these lists at any time by placing the desired card in the reader field and pressing the "Refresh objects from the card" button.

The X.509 certificate file selection is done by pressing the "open certificate file" key. It is also possible to read certificates from files in PEM format (Base64 encoded), which usually have the extension ".pem" or from the binary files written in ASN.1 standard (DER encoded), which can have extensions ".crt", ".cer ", or ".der ". If a valid file is selected, a system dialog showing the content of the selected X.509 certificate will be displayed. After checking the certificate items, it is sufficient to confirm this by pressing the "OK" button.

To store the selected certificate in a card, you need to enter the desired object ID (arbitrary alphanumeric character string) that must be unique in relation to other certificates that are stored on the card. The object ID is entered in the text box labeled "Objects ID:". The proposal is that certificates containing RSA public keys should be marked with an ID of eg. "0001" to "0003" and those containing ECDSA public keys will be marked with an ID of eg. "1001" to "1003". CA certificates must also have a unique ID tag on the card, so it is a suggestion to tag them with e.g. "5001" to "5012". It is still necessary to select a key type. For RSA and ECDSA type, the private key index in the card is bound to that certificate, and these indices must be consistent. For the CA Certificate Authority, order of the index is not relevant, but due to the transparency by recommendation, they should be entered in order, one after the other in pairs, for example, from root to intermediate. Applications that support the PKCS#11 module and use X.509 certificates, work by reading all public objects from the card and then checking the chain of trust based on the contents of the certificates themselves. In the end, you need to press the button with a descriptive name "Put certificate from a file in the card with a designated ID, object type and index".

We mentioned root and intermediate pairs of CA certificates and it may be necessary to further clarify this. Here we have assumed that the end-entity certificate in the chain of trust is established through the intermediate to the root CA certificate. This is the usual way of forming a chain of trust by the official issuers of the certificate. However, this is not a strict rule and other configurations are possible to alter CA certificates that form a chain of trust. It is important to emphasize that there are always two final certificates, at the beginning of the chain, the so-called root (root) CA certificate and at the end of the chain end-entity certificate (leaf certificate)

Signature

On the "Signature" tab there are commands for obtaining digital signatures from the card. A set of data from the input line labeled "M:" (message) or a file whose path can be set by clicking on "Set file to sign" radio button can be signed. The data can be entered in hexadecimal (HEX) format,

Base64 encoded or ASCII code layout.

The hexadecimal (HEX) format includes pairs of hexadecimal digits that can be separated by spaces. Base64 format is often used in cryptography and PEM records of the X.509 certificates. Here we will not deal with Base64 format in details. The ASCII code layout is a commonly used standard for recording textual data sets that includes all alphanumeric characters as well as all standard punctuation. In principle, everything that can be entered through the standard US English keyboard is covered by the ASCII code layout. If, by any chance, characters that are not part of the ASCII code layout are entered, and this option is selected, these characters will be replaced internally by the character '?'. Characters that are not part of the ASCII code layout can be entered via some localized keyboards or by selecting the "paste" option from the context menu of the "M:" text box.

When some data is entered in the input line, conversion to another type of record is done by simply selecting the desired record format (HEX / Base64 / ASCII options) except when there is an input error.

Clicking on the "Set file to sign" radio button opens a file selection dialog that is standard for Windows operating system. When the file is selected, its path is set to the "M:" text box.

By pressing the "Save message to binary file" button, it is enabled to store an array of bytes entered in the "M:" text box to the binary file.

By selecting the "Get signature", to be signed data passes through the selected hash algorithm (Message digest algorithm) whose result is sent to the card. The card then generates a signature based on the selected cryptographic algorithm from the "Cipher algorithm" combo box (RSA or ECDSA) and the key index in the card ("Key index in card" combo box). In order to sign the card, it is necessary to be loged in with the user PIN code beforehand.

After successful signing, the value of the digital signature is displayed in the "Signature" text box in HEX or Base64 format, depending on the selected option. By changing the HEX / Base64 selection, it is possible to convert the signature display. The signature can be saved in the binary

file by pressing the "Save signature to binary file" button.

Optional calculating the hash value of the digital signature has also been implemented. The choice of hash algorithms for this purpose also includes the obsolete MD5 algorithm for historical reasons, as some old cryptographic systems still depend on this mechanism. The hash value of a digital signature can be stored in a binary file by pressing the "Save hash to file" button.

1.0

Appendix: Change log

Date	Description	revision	refers to the uFCoder lib version / uFR firmware ver.
2019-01-28	First edition	1.0	5.0.1 / 5.0.12