

NXP Semiconductors

SDAF User Guide



1 Contents

2	Introduction and Purpose	2
3	Binaries included	2
4	volkano.dll	2
	4.1 Configuration file	11
5	volkano.exe	12
	5.1 Smartcard Authenticator usage	14
	5.1.1 General information	14
	5.1.2 Smart card init	14
6	volkano_utils.exe	14
7	Known limitations	15
8	Revision history	15

2 Introduction and Purpose

SDAF (Secure Debug Authorization Framework) is a utility software used to facilitate secure debug operations on NXP's S32 processors configured with challenge and response authentication scheme.

The purpose of this document is to describe the available functionalities for the SDAF components.

3 Binaries included

- **volkano.exe**
 - offers a command line API to volkano.dll.
 - available list of commands can be found below.
- **volkano_utils.exe**
 - utility tool that helps with key derivation and transportation.
 - available list of commands can be found below.
- **volkano.dll**
 - acts as an interface to either a host-based solution or a smart card for cryptographic operations and storage of keys.
 - can be used by explicitly linking against it, or in conjunction with volkano.exe.

*NOTE: Should you use volkano.dll by explicitly linking against it, volkano_init function should be called before calling other functions.

4 volkano.dll

Exported functions can be found in volkano_interface.h and are described below as well:

- **ULONG vlk_generate_wk();**
 - Generates an RSA-2048 key pair and stores it on a compatible smart card.
 - This pair is to be used for wrapping other keys in order to securely transport them.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_KEY_ALREADY_EXISTS** – Key already exists
- **VOLKANO_SMARTCARD_ERROR** – An error occurred during the generation process of the RSA key pair

- **ULONG vlk_export_wk(PBYTE WKPub, size_t *size);**
 - Exports the public key so that it can be used for wrapping other keys.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INVALID_PARAMETER** – size parameter is NULL.
 - **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
 - **VOLKANO_BUFFER_TOO_SMALL** – Export buffer of public key is too small
 - **VOLKANO_SMARTCARD_ERROR** – Error while read-out public wrap-key
- **ULONG vlk_init(const char *pwd);**
 - Connects to a compatible smart card if one is available.
 - This needs to be called first before trying to call any other function from this dll.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_PASSWORD_TOO_LONG** – Password exceeded 255 characters
- **VOLKANO_CONNECTION_ERROR** – Exceptions encountered during server working mode

*NOTE: pwd parameter is mandatory after the user password has been provisioned on the smart card.

- **ULONG vlk_discover(PBYTE UIDlist, size_t *ListSize, unsigned int *n);**
 - Retrieves the stored records from a compatible smart card.
 - Upon return, *UIDlist* is populated as follows:
 - UID0 (8B) || FLAG0 (1B) || UID1 (8B) || FLAG1 (1B) || etc
 - FLAG = 02 -> UID registered with ADKP
 - FLAG = 16 -> UID registered with ODAK
 - FLAG = 18 -> UID registered with ADKP + ODAK
 - *ListSize* represents the size of the *UIDlist* in bytes. Upon return, this parameter is populated with the number of relevant bytes in *UIDlist*.
 - Upon return, *n* holds the number of UIDs discovered.
 - * DEPRECATED. This function will only discover 8 byte UIDs. Please use *vlk_discover_all_uids* instead.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INVALID_PARAMETER** – *ListSize* or *n* parameter is NULL.
 - **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to *vlk_init* has not been performed prior to calling this function
 - **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
 - **VOLKANO_BUFFER_TOO_SMALL** – Input parameter UID list is NULL / Output list size is too small
- **ULONG vlk_discover_all_uids(PBYTE UIDlist, size_t *ListSize, unsigned int *n);**

- Retrieves the stored records from a compatible smart card.
- Upon return, *UIDList* is populated as follows:
 - *UID0_SIZE* (1Byte) || *UID0* (*UID0_SIZE* Bytes) || *FLAG0* (1Byte) || *UID1_SIZE* (1Byte) || *UID1* (*UID1_SIZE* Bytes) || *FLAG1* (1B) || etc
 - FLAG = 02 -> UID registered with ADKP
 - FLAG = 16 -> UID registered with ODAK
 - FLAG = 18 -> UID registered with ADKP + ODAK
- *ListSize* represents the size of the *UIDList* in bytes. Upon return, this parameter is populated with the number of relevant bytes in *UIDList*.
- Upon return, *n* holds the number of UIDs discovered.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INVALID_PARAMETER** – *ListSize* or *n* parameter is NULL.
 - **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to *vlk_init* has not been performed prior to calling this function
 - **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
 - **VOLKANO_BUFFER_TOO_SMALL** – Input parameter UID list is NULL / Output list size is too small
- **ULONG *vlk_register_key*(PBYTE UID, size_t UIDsize, PBYTE WpedKey, size_t WpedKeySize, BYTE KeyType, const char *pwd);**
 - Registers UID & ADKP
 - KeyType constants to be used:
 - KeyType = 02 -> ADKP key
 - KeyType = 16 -> ODAK key

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to *vlk_init* has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_MISSING_UID** – *UID* is NULL
- **VOLKANO_KEYSIZE_ERROR** – *WpedKey* is NULL
- **VOLKANO_INCORRECT_UID_SIZE** – *UIDsize* != 8 or *UIDsize* != 16
- **VOLKANO_KEYSIZE_ERROR** – *WpedKeySize* != 256
- **VOLKANO_UNSUPPORTED_KEY_TYPE** – *KeyType* == 16 and *UIDsize* != 16 or *KeyType* == 16 and *applet version* < 1.5 or
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the register key process

*NOTE: *pwd* parameter may be omitted in this version. Password provided when *volkano_init* was called will persist for the entire lifecycle of the process or until *volkano_init* is called again.

SDAF User Guide, Rev. G, October 2023

- **ULONG vlk_register_key_plain**(PBYTE UID, size_t UIDsize, PBYTE Key, size_t KeySize, BYTE KeyType);
 - Registers UID & ADKP
 - KeyType constants to be used:
 - KeyType = 02 -> ADKP key

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to vlk_init has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_MISSING_UID** – UID is NULL
- **VOLKANO_KEYSIZE_ERROR** – Key is NULL
- **VOLKANO_INCORRECT_UID_SIZE** – UIDsize != 8 or UIDsize != 16
- **VOLKANO_KEYSIZE_ERROR** – KeySize != 16 or KeySize != 32
- **VOLKANO_UNSUPPORTED_KEY_TYPE** – KeyType != 02
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version < 1.4
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the register key process

*NOTE: the difference from **vlk_register_key** is that here the key value is provided in plain.

*NOTE: This operation is only available on smart card starting with applet version 1.4.

- **ULONG vlk_delete_record**(PBYTE UID);
 - Deletes a record from the key storage (UID + ADKP).
 - *DEPRECATED. please use vlk_delete_uid instead

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to vlk_init has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_MISSING_UID** – UID is NULL
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version < 1.4
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the delete process

*NOTE: This operation is only available on smart card starting with applet version 1.4

- **ULONG vlk_delete_uid**(PBYTE UID, size_t UIDsize);
 - Deletes a record from the key storage (UID + ADKP).

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to `vlk_init` has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_MISSING_UID** – `UID` is NULL or provided value can't be found on smartcard
- **VOLKANO_INCORRECT_UID_SIZE** – `UIDsize` != 8 or `UIDsize` != 16
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version < 1.4
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the delete process

*NOTE: This operation is only available on smart card starting with applet version 1.4

- **ULONG vlk_update_password(const char* newPwd);**
 - Updates the password on the smart card or sets the initial password.
 - To provision the user password, the sequence would look like this:
 - Call `vlk_init(NULL);`
 - Call `vlk_update_password("userpassword");`

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_INCORRECT_USER_PASSWORD** – New password must not be NULL; also this error can occur while attempting to update password and the original password of smartcard is incorrect.
- **VOLKANO_PASSWORD_TOO_LONG** – Password exceeded 255 characters
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the update password process

*NOTE: If the user password has been provisioned, this requires authentication.

- **ULONG vlk_get_response(PBYTE UID, size_t UIDsize, PBYTE Chlg, size_t ChlgSize, PBYTE Resp, const char* pwd);**
 - Computes the response for a particular challenge
 - * DEPRECATED. Use `vlk_calculate_response` with `scheme_id = 1` instead.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to `vlk_init` has not been performed prior to calling this function

- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_BUFFER_TOO_SMALL** – *Chlg* or *Resp* buffer is NULL
- **VOLKANO_INCORRECT_UID_SIZE** – *UIDsize* != 8 or *UIDsize* != 16
- **VOLKANO_INCORRECT_CHALLENGE_SIZE** – *ChlgSize* < 32
- **VOLKANO_SMARTCARD_ERROR** – Error encountered during the calculation of response process
- **VOLKANO_MISSING_UID** – *UID* is NULL or provided value can't be found on smartcard

*NOTE: *pwd* parameter may be omitted in this version. Password provided when *volkano_init* was called will persist for the entire lifecycle of the process or until *volkano_init* is called again.

- **ULONG *vlk_calculate_response*(PBYTE *UID*, size_t *UIDsize*, PBYTE *Chlg*, size_t *ChlgSize*, BYTE *Schemeld*, BYTE *KeySelector*, PBYTE *Resp*);**
 - Computes the response for a particular challenge given the scheme id and a key selector.
 - Valid values for *Schemeld* are:
 - 1 – used for calculating response on SoCs in the S32 family based on HSE1.
 - 2 – calculate response for SoCs in the S32 family based on HSE2.
 - Valid values for *KeySelector* are:
 - 2 – ADKP;
 - Return codes (for more information regarding the error codes please see Table 1):
 - **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to *vlk_init* has not been performed prior to calling this function
 - **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
 - **VOLKANO_BUFFER_TOO_SMALL** – *Chlg* or *Resp* buffer is NULL
 - **VOLKANO_INCORRECT_UID_SIZE** – *UIDsize* != 8 or *UIDsize* != 16
 - **VOLKANO_INCORRECT_CHALLENGE_SIZE** – *ChlgSize* < 32
 - **VOLKANO_SMARTCARD_ERROR** – Error encountered during the calculation of response process
 - **VOLKANO_MISSING_UID** – *UID* is NULL or provided value can't be found on smartcard
 - **VOLKANO_INVALID_SCHEME** – invalid scheme id parameter.
 - **VOLKANO_UNSUPPORTED_KEY_TYPE** – invalid *KeySelector* parameter.
 - **VOLKANO_MISSING_KEY** – key corresponding to key selector is not registered for the provided *UID*.
 - **VOLKANO_INVALID_SCHEME** – *Schemeld* != 1 or *Schemeld* != 2
 - **VOLKANO_UNSUPPORTED_OPERATION** – *Schemeld* > 1 and applet version is < 1.5
- **ULONG *vlk_get_version*(char* *version*, size_t *allocatedSize*);**
 - Gets the current version of the dll and the version of the applet running on the smart card if one is connected.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_BUFFER_TOO_SMALL** – Export buffer used to store version is too small

*NOTE: if a compatible smart card is available, it will also output the version of the software on the smart card (the volkano applet version), the maximum number of supported UIDs and the secure status of the smart card.

- **ULONG vlk_discover_uid(PBYTE UID, size_t UIDsize, PBYTE FoundRecord);**
 - Checks for UID availability in smart card storage.
 - *FoundRecord* parameter needs to be a byte array of size equal to at least *UIDsize + 1* that will be populated as follows:
 - UID (*UIDsize* Bytes) || FLAG (1B)
 - FLAG = 02 -> UID registered with ADKP.
 - FLAG = 16 -> UID registered with ODAK.
 - FLAG = 18 -> UID registered with ADKP + ODAK.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to *vlk_init* has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_BUFFER_TOO_SMALL** – *FoundRecord* is NULL
- **VOLKANO_MISSING_UID** – *UID* is NULL or *UID* is not registered on smartcard
- **VOLKANO_INCORRECT_UID_SIZE** – *UIDsize* != 8 or *UIDsize* != 16
- **ULONG vlk_get_wk_auth_tag(PBYTE AuthTag, size_t *AuthTagSize);**
 - Computes the authentication tag over the wrap key on the smart card in order to be able to export this wrap key to another compatible smart card.
 - *AuthTag* parameter is the byte buffer where the value of the tag will be placed.
 - *AuthTagSize* is used as an input/output parameter. When calling this function, it should contain the address of a *size_t* variable that holds the size of the *AuthTag* buffer. If the call is successful, the function will update the value of the *AuthTagSize* to reflect the actual size of the calculated authentication tag.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_BUFFER_TOO_SMALL** – *AuthTag* is NULL or *AuthTagSize* < 32

- **VOLKANO_SMARTCARD_ERROR** – Failing during the process of obtaining authentication tag of public wrap key
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version is < 1.4

*NOTE: This operation is only available on smart card starting with applet version 1.4

- **ULONG vlk_set_wk2(PBYTE WK2Pub, size_t Wk2PubSize, PBYTE AuthTag, size_t AuthTagSize);**
 - Stores on the smart card a second wrap key that is corresponding to another smart card.
 - This second wrap key will be used for exporting an ADKP from the current smart card to the smart card that has this key as the main wrap key.
 - AuthTag parameter should contain the value of the auth tag calculated with **vlk_get_wk_auth_tag** on the smart card on which the ADKP will be exported to.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to **vlk_init** has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_BUFFER_TOO_SMALL** – AuthTag is NULL or AuthTagSize < 32; WK2Pub is NULL or Wk2PubSize != 256
- **VOLKANO_SMARTCARD_ERROR** – Failing while set authentication tag of public wrap key; Failing while set wrap key
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version < 1.4

*NOTE: This operation is only available on smart card starting with applet version 1.4

- **ULONG vlk_get_adkp(PBYTE UID, size_t UIDsize, PBYTE WADKP, size_t *WADKPSize);**
 - Exports an ADKP from the smart card so that it can be imported on another smart card.
 - If available, the ADKP will be encrypted with WK2.
 - WADKPSize is used as an input/output parameter. When calling this function, it should contain the address of a size_t variable that holds the actual size of the WADKP buffer. If the call is successful, the function will update the value of the WADKPSize to reflect the size of relevant data in the WADKP buffer.

If the function executes successfully, it will return **VOLKANO_OK**; otherwise, it will return one of the following error codes (for more information regarding the error codes please see Table 1):

- **VOLKANO_INCORRECT_USER_PASSWORD** – This function will require authentication. This function will return this error code if a call to **vlk_init** has not been performed prior to calling this function
- **VOLKANO_SMART_CARD_DISCONNECTED** – Smartcard is disconnected
- **VOLKANO_INCORRECT_UID_SIZE** – UIDsize != 8 or UIDsize != 16

- **VOLKANO_BUFFER_TOO_SMALL** – WADKP is NULL or WADKPSize < 256
- **VOLKANO_MISSING_UID** – UID is not registered on smartcard
- **VOLKANO_MISSING_WRAPPING_KEYS** – Missing wrap key
- **VOLKANO_ADKP_NOT_REGISTERED** – ADKP is not registered for this UID
- **VOLKANO_INVALID_PARAMETER** – WADKPSize is NULL
- **VOLKANO_SMARTCARD_ERROR** – unable to get ADKP from smart card
- **VOLKANO_UNSUPPORTED_OPERATION** – Applet version < 1.4

*NOTE: This operation is only available on smart card starting with applet version 1.4

You can also check the `volkano_interface.h` header file in the package.

The ADKP can be exported/imported from one smart card to another. A typical ADKP export/import procedure would go like this:

1. On the first smart card (the one on which ADKP will be imported) call **`vlk_export_wk`** and **`vlk_get_wk_auth_tag`** and save the returned values.
2. On the second smart card (the one from which ADKP will be exported) call **`vlk_set_wk2`** with the values from the previous step and then **`vlk_get_adkp`**.
3. On the first smart card **`vlk_register_key`** function can be used to register the ADKP exported on previous step.

*NOTE: It is assumed that both smart cards are properly initialized (have a wrap key and a user password) and the ADKP exists on the second smart card.

All functions in the API return VOLKANO_OK for success or an error code in case an error has occurred. Available error codes are listed in `error_codes.h` and in the table below:

Error	Code
VOLKANO_OK	0
VOLKANO_DB_ERROR	1
VOLKANO_KEYSIZE_ERROR	2
VOLKANO_UNSUPPORTED_KEY_TYPE	3
VOLKANO_INCORRECT_UID_SIZE	4
VOLKANO_INCORRECT_CHALLENGE_SIZE	5
VOLKANO_GENERIC_OPENSSL_ERROR	6
VOLKANO_ADKP_NOT_REGISTERED	7
VOLKANO_INCORRECT_USER_PASSWORD	9
VOLKANO_UNABLE_TO_LOAD_LIBRARY	10
VOLKANO_MISSING_ACTIVATION_CODE	11
VOLKANO_MISSING_WRAPPING_KEYS	12
VOLKANO_GENERIC_CC_ERROR	13
VOLKANO_KEY_ALREADY_EXISTS	14
VOLKANO_UNSUPPORTED_OS	15
VOLKANO_BUFFER_TOO_SMALL	16

VOLKANO_UNWRAP_ERROR	17
VOLKANO_PASSWORD_TOO_LONG	18
VOLKANO_SMARTCARD_ERROR	19
VOLKANO_MISSING_UID	20
VOLKANO_UNSUPPORTED_OPERATION	21
VOLKANO_CONFIG_FILE_MISSING	23
VOLKANO_SMART_CARD_DISCONNECTED	24
VOLKANO_INVALID_WORKING_MODE	25
VOLKANO_CONNECTION_ERROR	26
VOLKANO_MISSING_KEY	27
VOLKANO_INVALID_SCHEME	28
VOLKANO_INVALID_PARAMETER	29

Table 1. Volkano error codes

4.1 Configuration file

volkano.dll uses a configuration file named *vlk_config.ini* that contains entries in the form of key-value pairs. There are three config entries supported:

- **WORKING_MODE** – this config parameter is used for selecting the working mode regarding the smart card connection. There are three possible values for this:
 - **managed** – volkano.dll iterates through all the PC/SC readers locally connected to the PC where volkano is running and identifies if a smart card with the volkano applet installed is present. If such a smart card is identified, volkano will connect and interact with it accordingly.
 - **server** – this mode is very similar with managed mode, but instead of interacting with the smart card after connecting to it, volkano will keep the connection up and will start listening on a TCP port for incoming connections from other instances of volkano running in client working mode.
 - **client** – volkano will try to connect to another instance of volkano running in server mode. If the connection to the server is successful, the user will benefit from the same functionality offered by volkano in managed mode.
* Due to the fact that all the commands sent by volkano to the smart card will go through the network via TCP, some latency is expected.
- **SERVER_HOSTNAME** – this config parameter is relevant only when the working mode is **client**.
 - It specifies the hostname of the machine on which a volkano instance is running in server mode.
- **SERVER_PORT** – this config parameter is relevant for both **client** & **server** working modes.
 - **client** – determines the port of the remote volkano instance that the client will try to connect to.
 - **server** – determines the port on which the server instance is listening on.
- **LOGGING_ENABLED** – Used for enabling or disabling the logging system. There are two possible values for this:

- **true** – Enables logging system. This is the default value of the config parameter.
- **false** – Disables logging system.

These values need to be in lower case to work properly.

If enabled, the logging system will append logs to a local file named `volcano.log` as well as to the console if you are running in server mode.

- **LOGGING_VERBOSITY** - this config parameter controls the level of detail in the program's log output. The available verbosity levels are:
 - **ERROR** – provides messages that indicate exceptions that have occurred during the execution and data transmission errors. This is the default value of the config parameter.
 - **INFO** – provides informational messages that contains details about the normal execution flow.
 - **DEBUG** – provides detailed debugging information.

The loggers follow this format: **<Date> <Time> <Verbosity level> - <Log Message>**

These values needs to be in upper case to work properly.

5 volcano.exe

Available commands in volcano.exe:

discover	: list the UIDs and keys registered
discover_uid	: check if the provided UID is registered -uid <UID value>
generate_wrapkey	: generate the wrapping key pair [*1]
get_wrapkey_auth_tag	: get the authentication tag over wrapkey modulus [*1][*2]
set_wrapkey_2	: save the wrapkey of another card for ADKP export purposes [*2] -wk2 <second wrapkey value> -tag <authentication tag over the second wrapkey>
export_wrapkey	: export the public part of the wrapping key [*1] -binfile [path/name of the binary file to use for export] default is 'wrap_key.bin'
register_adkp	: register the key ADKP -uid <UID value> -key <ADKP plain/wrapped with the wrapping key> -keybin <input binary file>

register_odak	: register the key ODAK [*3] -uid <UID value> -key <ODAK value wrapped with the wrapping key>
delete_record	: delete a database record [*2] -uid <UID value>
update_pwd	: update the password on the smart card -new_pwd <new password> *NOTE: when using this to set the user password initially, the -pw parameter is not required. For other subsequent calls to this command targeting the same smart card, the -pw parameter is mandatory.
get_response	: get the answer to a challenge -uid <UID value> -chlg <challenge value> -key [key name] * is ignored when UID is 8 bytes long * use the key name given by discover command.
get_wrapped_adkp	: retrieve the value of an ADKP wrapped with the second wrapkey [*2] -uid <UID value>

[*1]: does not require authentication (-pw parameter will be ignored)

[*2]: only available on smart card starting with applet version 1.4

[*3]: only available on smart card starting with applet version 1.5

Parameters are provided as hex-strings except for the "-binfile" parameter of the "export_wrapkey" command.

```
E.g: volkano.exe -cmd get_response -uid 0123456789ABCDEF -chlg
1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef -pw <user_password>
```

```
E.g: volkano.exe -cmd get_response -uid 0123456789ABCDEF0123456789ABCDEF -chlg
1234567890abcdef1234567890abcdef1234567890abcdef1234567890abcdef -key ADKP -pw
<user_password>
```

*NOTE: Using the **register_adkp** command twice with the same UID and different key values will override the first value.

5.1 Smartcard Authenticator usage

5.1.1 General information

The number of unique device registrations is limited. A smart card can support maximum 32 UIDs.

Excepting the commands marked with *[*1]* in the help string, including *generate_wrapkey*, *export_wrapkey* and *get_wrapkey_auth_tag*, you will need to authenticate yourself with this password for every other command in the *volkano.exe* API using the *-pw* option like this:

```
volkano.exe -cmd discover -pw <your_password>
```

*NOTE: The above scenario does not apply if the *volkano.dll* file is used. In this case, it is necessary to provide the password only once, by calling the *vlk_init* function, and the password will be cached.

5.1.2 Smart card init

When using the S32 Debug Entry Authenticator smartcard for the first time, it is expected that before using other commands from the *volkano.exe* API or functions from *volkano.dll*, these two commands are to be used in the following order:

1. **generate_wrapkey** – Use this command first in order to create the RSA keypair used to store other security assets securely. Once the keypair is generated, it is persistent.
2. **update_pwd** – after generating the RSA keypair it is required to provision a user password. The password can be between 1 and 255 characters and is persistent on the card.

The first time you call the **update_pwd** command, the *-pw* parameter can be omitted, but for subsequent calls, it needs to be provided.

6 volkano_utils.exe

Available commands in *volkano_utils.exe*:

wrap_key	: wraps the input key with the given wrapkey (RSA-2048) -input_key <key plain value> -keybin [wrapping key binary file] -key [wrapping key value as hex string] -outputbin [output binary file] *NOTE* If -keybin is provided, -key parameter will be ignored and vice versa.
derive_adkp	: get device dependent ADKP based on the master ADKP and device UID -uid <UID value> -adkpm <master ADKP value>

NOTE Only works for 8 byte UIDs and 16 byte ADKPs

`generate_gmac` : generate GMAC authentication tag
-inputbin <input binary file>
-adkp <ADKP plain value/hash value>
-IV [Initialization Vector value]

`generate_cmac` : generate CMAC authentication tag
-inputbin <input file>
-key <key plain value>
-context <context plain value>
-label <label plain value>

Parameters are provided as hex-strings except for the "-keybin" parameter of the "wrap_adkp" command.

```
E.g: volkano_utils.exe -cmd wrap_adkp -adkp 0123456789abcdef0123456789abcdef -keybin wrap_key.bin
```

7 Known limitations

- Only Windows 10 x64 OS is currently supported.

8 Revision history

Revision	Date	Substantive changes
A	04/06/2020	Initial release
B	02/25/2021	Added support for smart card
C	06/14/2021	Added more details regarding smartcard usage
D	07/21/2021	Added notes configuration file and vlk_discover_uid
E	12/15/2022	Removed some deprecated features and marked affected commands accordingly
F	03/16/2023	Added info regarding new config file parameters
G	10/03/2023	Added new dll functions to support HSE2 based devices