Bio Enrollment

authenticatorUserVerification

This command is used by the platform to provision/enumerate/delete bio enrollments in the authenticator.

It takes the following input parameters:

Parameter name	Data type	Required?	Definition
modality (0x01)	Unsigned Integer	Optional	The user verification modality being requested
subCommand (0x02)	Unsigned Integer	Optional	The authenticator user verification sub command currently being requested
subCommandParams (0x03)	CBOR Map	Optional	Map of subCommands parameters.
pinProtocol (0x04)	Unsigned Integer	Optional	PIN protocol version chosen by the client. For this version of the spec, this SHALL be the number 1.
pinAuth (0x05)	Byte Array	Optional	First 16 bytes of HMAC-SHA-256 of contents using pinToken.
getModality (0x06)	Boolean	Optional	Get the user verification type modality. This MUST be set to true.

The type of modalities supported are as under:

modality Name	modality Number
fingerprint	0x01

The list of sub commands for fingerprint(0x01) modality is:

subCommand Name	subCommand Number
enrollBegin	0x01
enrollCaptureNextSample	0x02
cancelCurrentEnrollment	0x03
enumerateEnrollments	0x04
setFriendlyName	0x05

removeEnrollment	0x06
getFingerprintSensorInfo	0x07

subCommandParams Fields:

Field name	Data type	Required?	Definition
templateId (0x01)	Byte Array	Optional	Template Identifier.
templateFriendlyName (0x02)	String	Optional	Template Friendly Name.
timeoutMilliseconds (0x03)	Unsigned Integer	Optional	Timeout in milliSeconds.

On success, authenticator returns the following structure in its response:

Parameter name	Data type	Required?	Definition
modality (0x01)	Unsigned Integer	Optional	The user verification modality.
fingerprintKind (0x02)	Unsigned Integer	Optional	Indicates the type of fingerprint sensor. For touch type sensor, its value is 1. For swipe type sensor its value is 2.
maxCaptureSamplesRequiredForEnroll (0x03)	Unsigned Integer	Optional	Indicates the maximum good samples required for enrollment.
templateId (0x04)	Byte Array	Optional	Template Identifier.
lastEnrollSampleStatus (0x05)	Unsigned Integer	Optional	Last enrollment sample status.
remainingSamples (0x06)	Unsigned Integer	Optional	Number of more sample required for enrollment to complete
templateInfos (0x07)	CBOR ARRAY	Optional	Sequence of templateInfo’s

TemplateInfo definition:

Field name	Data type	Required?	Definition
templateId (0x01)	Byte Array	Required	Template Identifier.
templateFriendlyName (0x02)	String	Optional	Template Friendly Name.

lastEnrollSampleStatus types:

lastEnrollSampleStatus Name	lastEnrollSampleStatus Value	Definition
CTAP2_ENROLL_FEEDBACK_FP_GOOD	0x00	Good fingerprint capture.
CTAP2_ENROLL_FEEDBACK_FP_TOO_HIGH	0x01	Fingerprint was too high.
CTAP2_ENROLL_FEEDBACK_FP_TOO_LOW	0x02	Fingerprint was too low.
CTAP2_ENROLL_FEEDBACK_FP_TOO_LEFT	0x03	Fingerprint was too left.
CTAP2_ENROLL_FEEDBACK_FP_TOO_RIGHT	0x04	Fingerprint was too right.
CTAP2_ENROLL_FEEDBACK_FP_TOO_FAST	0x05	Fingerprint was too fast.
CTAP2_ENROLL_FEEDBACK_FP_TOO_SLOW	0x06	Fingerprint was too slow.
CTAP2_ENROLL_FEEDBACK_FP_POOR_QUALITY	0x07	Fingerprint was of poor quality.
CTAP2_ENROLL_FEEDBACK_FP_TOO_SKEWED	0x08	Fingerprint was too skewed.
CTAP2_ENROLL_FEEDBACK_FP_TOO_SHORT	0x09	Fingerprint was too short.
CTAP2_ENROLL_FEEDBACK_FP_MERGE_FAILURE	0x0A	Merge failure of the capture.
CTAP2_ENROLL_FEEDBACK_FP_EXISTS	0x0B	Fingerprint already exists.
CTAP2_ENROLL_FEEDBACK_FP_DATABASE_FULL	0x0C	Fingerprint database storage is full.

CTAP2_ENROLL_FEEDBACK_NO_USER_ACTIVITY	0x0D	User did not touch/swipe the authenticator.
CTAP2_ENROLL_FEEDBACK_NO_USER_PRESENCE_TRANSITION	0x0E	User did not lift the finger off the sensor.

Feature detection

To detect whether authenticator supports this preview feature, following conditions MUST be met:

- Authenticator MUST return "FIDO_2_1_PRE" in authenticatorGetInfo as one of version it supports in addition to "FIDO_2_0".
- Authenticator MUST return "userVerificationMgmtPreview" in options fields of authenticatorGetInfo.
    - Presence of this key indicates that the authenticator supports auehenticatorUserVerification commands.
    - True value indicates that authenticator has atleast one bio enrollment already provisioned.
    - False value indicates that authenticator has not been provisioned with any bio enrollment yet.
- For this preview feature, authenticatorUserVerification command is choosen from vendor command space and its value is MUST be 0x40.

Get user verification modality

Following operations are performed to get user verification modality supported by the authenticator:

- Platform sends authenticatorUserVerification command with following parameters:
    - getModality (0x06): true.
- Authenticator returns authenticatorUserVerification response with following parameters:
    - modality (0x01): It represents the type of modality authenticator supports. For fingerprint, its value is 1.

Get fingerprint sensor info

Following operations are performed to get fingerprint sensor information:

- Platform sends authenticatorUserVerification command with following parameters:
    - modality (0x01): fingerprint (0x01).
    - subCommand (0x02): getFingerprintSensorInfo (0x07)
- Authenticator returns authenticatorUserVerification response with following parameters:

- fingerprintKind (0x02):
  - For touch type fingerprints, its value is 1.
  - For swipe type fingerprints, its value is 2.
- maxCaptureSamplesRequiredForEnroll (0x03): Indicates the maximum good samples required for enrollment.

Enrolling fingerprint

Following operations are performed to enroll a fingerprint:

- Platform gets pinToken from the authenticator.
- Platform sends authenticatorUserVerification command with following parameters to begin the enrollment:
  - modality (0x01): fingerprint (0x01).
  - subCommand (0x02): enrollBegin (0x01).
  - subCommandParams (0x03): Map containing following parameters
    - timeoutMilliseconds (0x03) (optional): timeout in milliseconds
  - pinProtocol (0x04): Pin Protocol used. Currently this is 0x01.
  - pinAuth (0x05): `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || enrollBegin (0x01) || subCommandParams), 16)`.
- Authenticator on receiving such request performs following procedures.
  - Authenticator verifies pinAuth by generating `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || enrollBegin (0x01) || subCommandParams), 16)` and matching against input pinAuth parameter.
    - Authenticator does same semantics checks for pinAuth protection as done in authenticatorClientPin command.
    - If there is no space available, authenticator returns CTAP2_ERR_KEY_STORE_FULL.
  - Authenticator cancels any unfinished ongoing enrollment if any.
  - Authenticator generates templateId for new enrollment.
  - Authenticator sends the command to the sensor to capture the sample.
  - Authenticator returns authenticatorUserVerification response with following parameters:
    - templateId (0x04): template identifier of the new template being enrolled.
    - lastEnrollSampleStatus (0x05) : Status of enrollment of last sample.
    - remainingSamples (0x06) : Number of sample remaining to complete the enrollment.
- Platform sends authenticatorUserVerification command with following parameters to continue enrollment in a loop till remainingSamples is zero or authenticator errors out with unrecoverable error or platform wants to cancel current enrollment:
  - Platform sends authenticatorUserVerification command with following parameters
    - modality (0x01): fingerprint (0x01).

        - subCommand (0x02): enrollCaptureNextSample (0x02).
        - subCommandParams (0x03): Map containing following parameters
            - templateId (0x01) : template identifer platform received from enrollBegin subCommand.
            - timeoutMilliseconds (0x03) (optional): timeout in milliseconds
        - pinProtocol (0x04): Pin Protocol used. Currently this is 0x01.
        - pinAuth (0x05): `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || enrollCaptureNextSample (0x02) || subCommandParams), 16)`.
    - Authenticator on receiving such request performs following procedures.
        - Authenticator verifies pinAuth by generating `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || enrollBegin (0x01) || subCommandParams), 16)` and matching against input pinAuth parameter.
            - Authenticator does same semantics checks for pinAuth protection as done in authenticatorClientPin command.
            - If there is no space available, authenticator returns CTAP2_ERR_KEY_STORE_FULL.
        - If fingerprint is already present on the sensor, authenticator waits for user to lift finger from the sensor.
        - Authenticator sends the command to the sensor to capture the sample.
        - Authenticator returns authenticatorUserVerification response with following parameters:
            - lastEnrollSampleStatus (0x05) : Status of enrollment of last sample.
            - remainingSamples (0x06) : Number of sample remaining to complete the enrollment.

Cancel current enrollment

Following operations are performed to cancel current enrollment:

- Platform sends authenticatorUserVerification command with following parameters:
    - modality (0x01): fingerprint (0x01).
    - subCommand (0x02): cancelCurrentEnrollment (0x03).
- Authenticator on receiving such command, cancels currnet ongoing enrollment, if any, and returns CTAP2_OK.

Enumerate enrollments

Following operations are performed to enumerate enrollments:

- Platform gets pinToken from the authenticator.
- Platform sends authenticatorUserVerification command with following parameters:
    - modality (0x01): fingerprint (0x01).
    - subCommand (0x02): enumerateEnrollments (0x04).
    - pinProtocol (0x04): Pin Protocol used. Currently this is 0x01.

    - pinAuth (0x05): `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || enumerateEnrollments (0x04)), 16)`.
- Authenticator verifies pinAuth by generating `LEFT(HMAC-SHA-256(pinToken, enumerateEnrollments (0x04)), 16)` and matching against input pinAuth parameter.
    - Authenticator does same semantics checks for pinAuth protection as done in authenticatorClientPin command.
    - If there are no enrollments existing on authenticator, it returns CTAP2_ERR_INVALID_OPTION.
- Authenticator returns authenticatorUserVerification response following parameters:
    - templateInfos (0x07) : Sequence of templateInfo's for all the enrollments available on the authenticator.

Rename/Set FriendlyName

Following operations are performed to remove a fingerprint:

- Platform gets pinToken from the authenticator.
- Platform sends authenticatorUserVerification command with following parameters:
    - modality (0x01): fingerprint (0x01).
    - subCommand (0x02): setFriendlyName (0x05).
    - subCommandParams (0x03): Map containing following parameters
        - templateId (0x01) : template identifer.
        - templateFriendlyName (0x02): Friendly name of the template
    - pinProtocol (0x04): Pin Protocol used. Currently this is 0x01.
    - pinAuth (0x05): `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || setFriendlyName (0x05) || subCommandParams), 16)`.
- Authenticator verifies pinAuth by generating `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || setFriendlyName (0x05) || subCommandParams), 16)` and matching against input pinAuth parameter.
    - Authenticator does same semantics checks for pinAuth protection as done in authenticatorClientPin command.
    - If there are no enrollments existing on authenticator for the passed templateId, it returns CTAP2_ERR_INVALID_OPTION.
    - If there is an existing enrollment with that identifier, rename its friendly name and return CTAP2_OK.

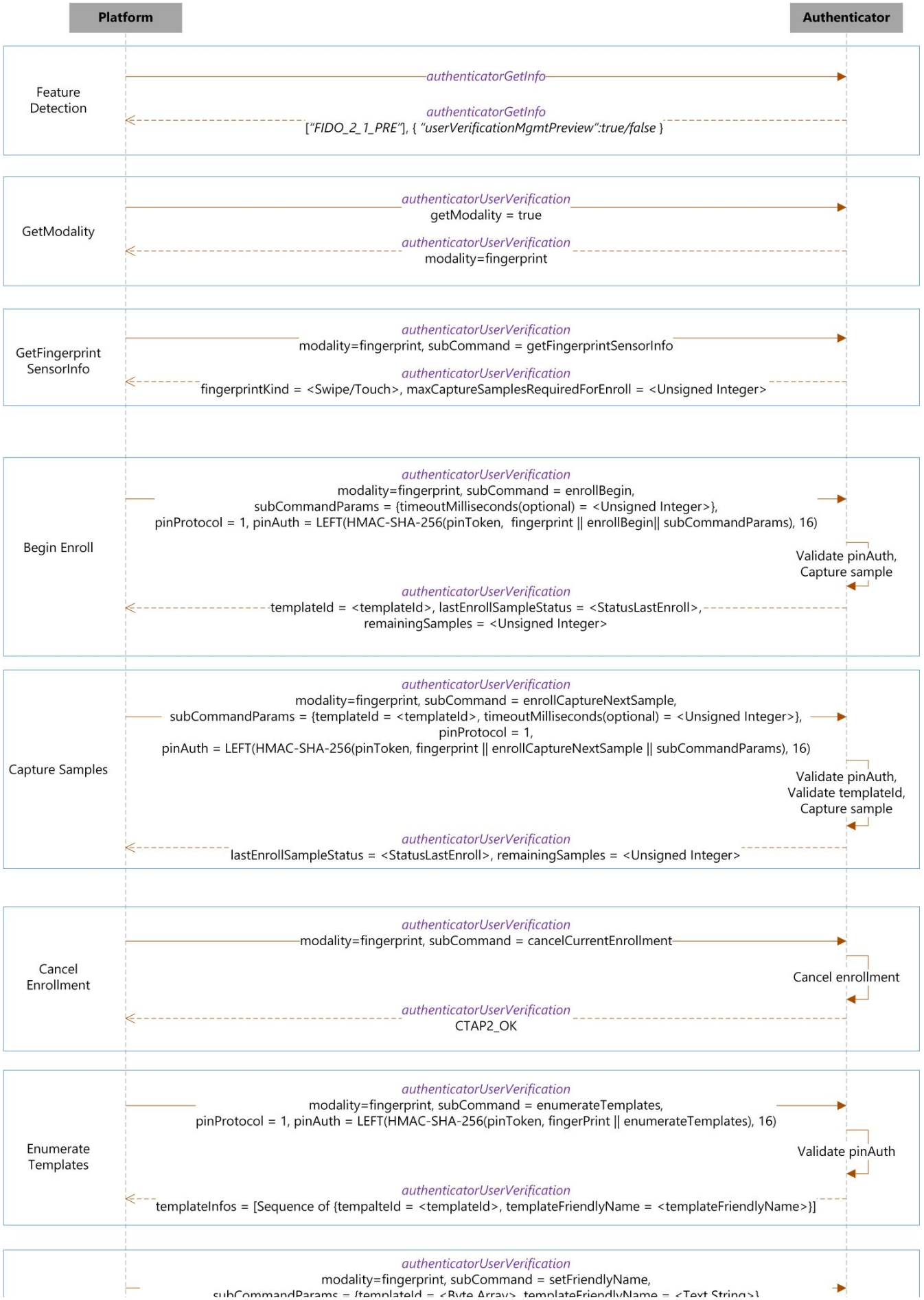
Remove enrollment

Following operations are performed to remove a fingerprint:

- Platform gets pinToken from the authenticator.
- Platform sends authenticatorUserVerification command with following parameters:
    - modality (0x01): fingerprint (0x01).

- subCommand (0x02): removeEnrollment (0x06).
- subCommandParams (0x03): Map containing following parameters
  - templateId (0x01) : template identifer.
- pinProtocol (0x04): Pin Protocol used. Currently this is 0x01.
- pinAuth (0x05): `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || removeEnrollment (0x05) || subCommandParams), 16)`.

- Authenticator verifies pinAuth by generating `LEFT(HMAC-SHA-256(pinToken, fingerprint (0x01) || removeEnrollment (0x05) || subCommandParams), 16)` and matching against input pinAuth parameter.
  - Authenticator does same semantics checks for pinAuth protection as done in authenticatorClientPin command.
  - If there are no enrollments existing on authenticator for passed templateId, it returns CTAP2_ERR_INVALID_OPTION.
  - If there is an exiting enrollment with passed in templateInfo, delete that enrollment and return CTAP2_OK.


Platform
Authenticator
Feature Detection
authenticatorGetInfo
authenticatorGetInfo
["FIDO_2_1_PRE"], { "userVerificationMgmtPreview":true/false }
GetModality
authenticatorUserVerification
getModality = true
authenticatorUserVerification
modality=fingerprint
GetFingerprint SensorInfo
authenticatorUserVerification
modality=fingerprint, subCommand = getFingerprintSensorInfo
authenticatorUserVerification
fingerprintKind = <Swipe/Touch>, maxCaptureSamplesRequiredForEnroll = <Unsigned Integer>
Begin Enroll
authenticatorUserVerification
modality=fingerprint, subCommand = enrollBegin,
subCommandParams = {timeoutMilliseconds(optional) = <Unsigned Integer>},
pinProtocol = 1, pinAuth = LEFT(HMAC-SHA-256(pinToken, fingerprint || enrollBegin|| subCommandParams), 16)
Validate pinAuth, Capture sample
authenticatorUserVerification
templateId = <templateId>, lastEnrollSampleStatus = <StatusLastEnroll>,
remainingSamples = <Unsigned Integer>
Capture Samples
authenticatorUserVerification
modality=fingerprint, subCommand = enrollCaptureNextSample,
subCommandParams = {templateId = <templateId>, timeoutMilliseconds(optional) = <Unsigned Integer>},
pinProtocol = 1,
pinAuth = LEFT(HMAC-SHA-256(pinToken, fingerprint || enrollCaptureNextSample || subCommandParams), 16)
Validate pinAuth, Validate templateId, Capture sample
authenticatorUserVerification
lastEnrollSampleStatus = <StatusLastEnroll>, remainingSamples = <Unsigned Integer>
Cancel Enrollment
authenticatorUserVerification
modality=fingerprint, subCommand = cancelCurrentEnrollment
Cancel enrollment
authenticatorUserVerification
CTAP2_OK
Enumerate Templates
authenticatorUserVerification
modality=fingerprint, subCommand = enumerateTemplates,
pinProtocol = 1, pinAuth = LEFT(HMAC-SHA-256(pinToken, fingerPrint || enumerateTemplates), 16)
Validate pinAuth
authenticatorUserVerification
templateInfos = [Sequence of {tempalteId = <templateId>, templateFriendlyName = <templateFriendlyName>}]
authenticatorUserVerification
modality=fingerprint, subCommand = setFriendlyName,

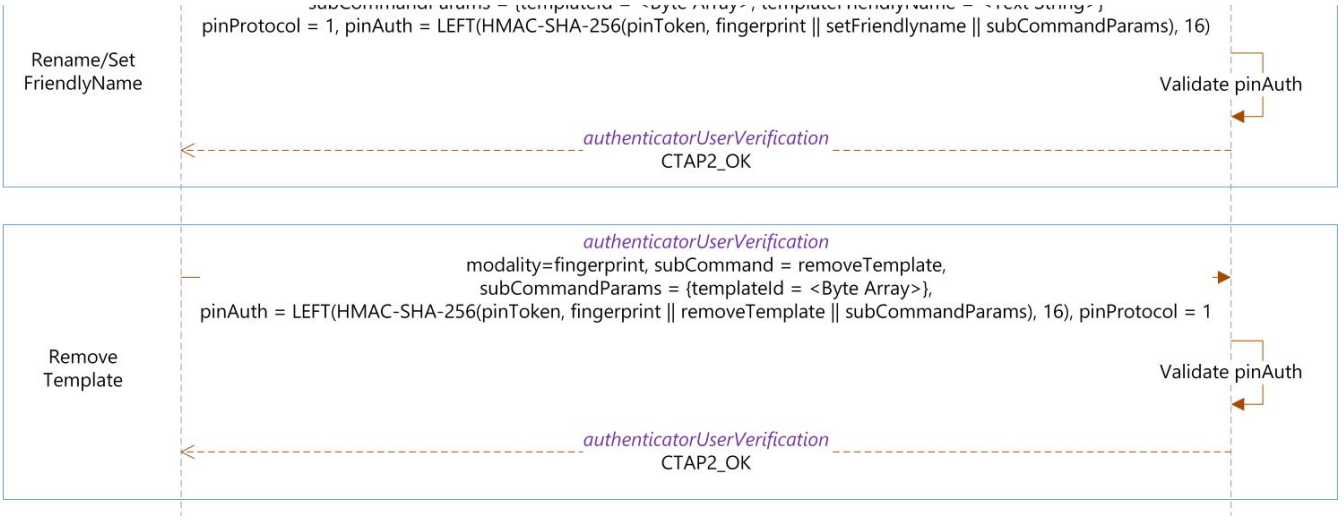


Figure 1 *User Verification Modality - Fingerprint*

Commands

For each command that contains parameters, the parameter map keys and value types are specified below:

Command	Parameter Name	Key	Value type
authenticatorUserVerification	modality	0x01	Unsigned Integer. (CBOR major type 0)
	subCommand	0x02	Unsigned Integer. (CBOR major type 0)
	subCommandParams	0x03	CBOR definite length map (CBOR major type 5).
	pinProtocol	0x04	Unsigned Integer. (CBOR major type 0).
	pinAuth	0x05	byte string (CBOR major type 2).
	getModality	0x06	Boolean

Responses

For each response message, the map keys and value types are specified below:

Response Message	Member Name	Key	Value type
authenticatorUserVerification_Response	modality	0x01	Unsigned integer (CBOR major type 0).
	fingerprintKind	0x02	Unsigned integer

			(CBOR major type 0).
	maxCaptureSamplesRequiredForEnroll	0x03	Unsigned integer (CBOR major type 0).
	templateId	0x04	byte string (CBOR major type 2).
	lastEnrollSampleStatus	0x05	Unsigned integer (CBOR major type 0).
	remainingSamples	0x06	Unsigned integer (CBOR major type 0).
	templateInfos	0x07	CBOR definite length map (CBOR major type 5).