

YubiKey Configuration Utility for Windows

User Guide

Version: 2.2

May 24, 2012

Introduction

Yubico is the leading provider of simple, open online identity protection. The company's flagship product, the YubiKey®, uniquely combines driverless USB hardware with open source software. More than a million users in 100 countries rely on YubiKey strong two-factor authentication for securing access to computers, mobile devices, networks and online services. Customers range from individual Internet users to e-governments and Fortune 500 companies. Founded in 2007, Yubico is privately held with offices in California, Sweden and UK.

Disclaimer

The contents of this document are subject to revision without notice due to continued progress in methodology, design, and manufacturing. Yubico shall have no liability for any error or damages of any kind resulting from the use of this document.

The Yubico Software referenced in this document is licensed to you under the terms and conditions accompanying the software or as otherwise agreed between you or the company that you are representing.

Trademarks

Yubico and YubiKey are trademarks of Yubico Inc.

Contact Information

Yubico Inc
228 Hamilton Avenue, 3rd Floor
Palo Alto, CA 94301
USA
info@yubico.com

Contents

Introduction.....	2
Disclaimer.....	2
Trademarks	2
Contact Information.....	2
1 Document Information.....	5
1.1 Purpose	5
1.2 Audience	5
1.3 Related documentation	5
1.4 Document History.....	5
1.5 Definitions.....	5
2 Introduction.....	6
2.1 System requirements	6
2.2 Random numbers.....	6
2.3 Security and cryptographic practices	6
3 Installing the application.....	7
4 Using the application.....	8
4.1 The Start page.....	8
4.2 The Task page	9
4.3 Common configuration items	10
4.3.1 Binary strings.....	10
4.3.2 Remember settings	10
5 Create a dynamic configuration (OTP)	11
5.1 Specify public identity.....	11
5.2 Specify private identity	12
5.3 Specify cryptographic key	13
5.4 Specify output parameters	13
5.4.1 Output format flags.....	14
5.4.2 Output speed throttling.....	14
5.4.3 Strong password policy	14
5.4.4 Serial number visibility	15
5.5 Specify configuration protection	15
5.6 Writing the configuration to the YubiKey.....	17
5.6.1 Lock / protect configuration 2	17
5.6.2 The programming sequence	18
5.6.3 Access code protection confirmation	19

6	Create a YubiKey for the Yubico server	21
6.1	Passing parameters	22
6.2	Completing the Web form	23
7	Create a dynamic configuration (OATH-HOTP).....	24
7.1	Specify OATH-HOTP specific data	24
7.2	Specify cryptographic key (shared secret)	25
7.3	Specifying output parameters	25
7.4	Specifying configuration protection	25
7.5	Writing the configuration to the YubiKey	25
8	Create a static configuration (Password)	26
8.1	Simplified mode.....	26
8.2	Standard (advanced) mode.....	26
8.3	Specifying output parameters	28
8.4	Specifying configuration protection	28
8.5	Writing the configuration to the YubiKey	28
9	Create a challenge-response configuration	29
9.1	Specify challenge-response mode of operation.....	29
9.2	Specifying private identity (Yubico OTP mode only)	30
9.3	Specifying key or secret	30
9.4	Specifying output parameters	30
9.5	Specifying configuration protection	30
9.6	Writing the configuration to the YubiKey	30
10	Remove an existing configuration	31
10.1	Specifying configuration protection	31
10.2	Writing the configuration to the YubiKey	31
11	Check the YubiKey version	32
12	Test the OTP output of a YubiKey	33
13	Convert between different formats	34
14	Program settings	35
14.1	General settings	35
14.2	Programming output.....	35
14.3	Sounds at programming.....	36
15	Specify a text file for configuration input	37

1 Document Information

1.1 Purpose

The purpose of this documentation is to provide an in-detail understanding of the YubiKey configuration process using the YubiKey Configuration Utility.

The document does not cover a “systems perspective”, but rather focuses on the process of configuring the YubiKey itself.

1.2 Audience

This document is intended primarily for readers with a technical/IT background. The document assumes knowledge of basic security concepts and terminology.

Furthermore, basic knowledge of YubiKey concepts is assumed. More information about this topic can be found in the “Related documentation section”

1.3 Related documentation

- The YubiKey Manual – Usage, configuration and introduction of basic concepts
- YubiKey configuration COM API – Describes the configuration component
- YubiKey Client COM API – Describes the client-side API
- YubiKey Server COM API – Describes the server-side API
- Yubico online forum – <http://forum.yubico.com>
- RFC 2104 - HMAC: Keyed-Hashing for Message Authentication
- RFC 4226 – HOTP: An HMAC-Based One-Time Password Algorithm

1.4 Document History

Date	Version	Author	Activity
2009-09-09	2.0	JE	New release
2009-12-03	2.1	JE	Added OATH-HOTP
2010-07-08	2.2	JE	Added 2.2 features

1.5 Definitions

Term	Definition
YubiKey device	Yubico’s authentication device for connection to the USB port
USB	Universal Serial Bus
HID	Human Interface Device. A specification of typical USB devices used for human interaction, such as keyboards, mice, joysticks etc.
AES	Advanced Encryption Standard, FIPS-197
UID	Unit Identity, a.k.a. Private Id or Secret Id
Ticket	A general term for an access code generated by the YubiKey, a.k.a. OTP.
OTP	One Time Password
Modhex	Modified Hexadecimal coding

2 Introduction

The YubiKey Configuration Utility, YubiKeyConfig.exe, is a Microsoft Windows application designed to configure and verify a YubiKey authentication device. The application follows a step-by-step approach to make configuration easy to follow and understand, while still being powerful enough to exploit all functionality both of the YubiKey 1 and YubiKey 2.

The YubiKey Configuration Utility provides the following main functions:

- Programming a YubiKey in dynamic “OTP” mode
- Programming a YubiKey in static “password” mode
- Removing (invalidating) an existing configuration from a YubiKey
- Checking the type and firmware version of a YubiKey
- Test/verify the OTP output of a programmed YubiKey

The YubiKey Configuration Utility provides basic means of batch processing where a larger number of keys can be configured sequentially. Configuration input can be automatic and/or read from a text file. Output of the configuration process can be written to a text file which later can be imported into a third-party verification tool.

The YubiKey Configuration Utility is intended to be a stand-alone application. For integration of configuration functionality into a custom application in the Microsoft Windows environment, a configuration component is provided. Please refer to appropriate documentation in section 1.3.

Basic YubiKey concepts and terms are not covered by this document. For related information, please refer to section 1.3.

2.1 System requirements

The YubiKey Configuration Utility is designed to run on all Microsoft Windows Win32 environments from Windows 2000 and onwards. The configuration utility is a single-file executable that runs without any other dependencies.

One free USB port is required.

2.2 Random numbers

Where random number generation is used in the application, the random values are generated using the Win32 Crypto API function `CryptGenRandom`, which should satisfy most needs. There is no special seeding or additional obfuscation added.

2.3 Security and cryptographic practices

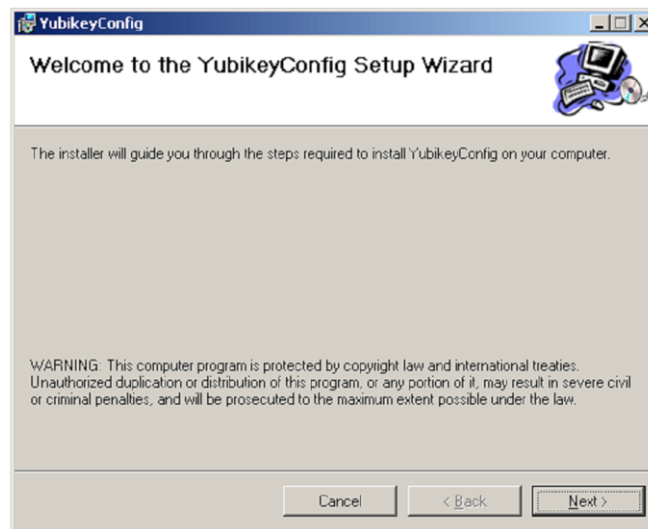
The user must be aware of the appropriate security- and cryptographic practices needed to maintain the integrity of the generated configurations.

There is absolutely “no black magic” with the application in this respect, but given that cryptographically sensitive information is handled and

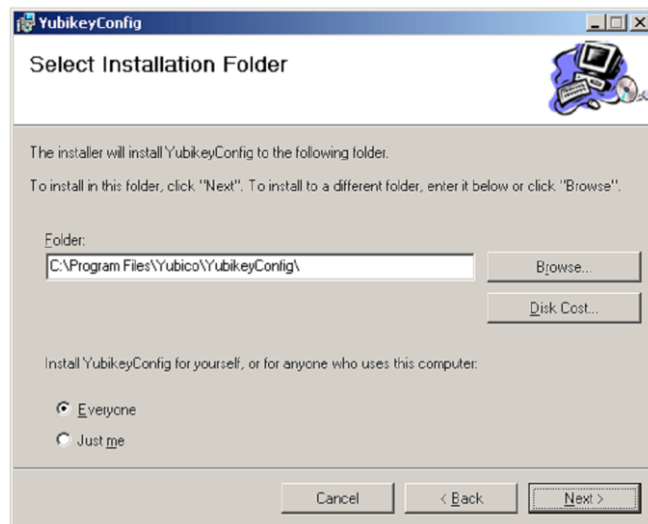
3 Installing the application

The YubiKey Configuration utility is a stand-alone application that runs without any other dependencies. This means that the application file **YubiKeyConfig.exe** alone can simply be distributed to a second computer without an installer being run.

However, a basic Windows installer is provided for convenience of the first-time user:



Press Next to select installation location



Press Next > two times and then Close and the application and documentation (this document) should show up under the Start menu / All programs like:

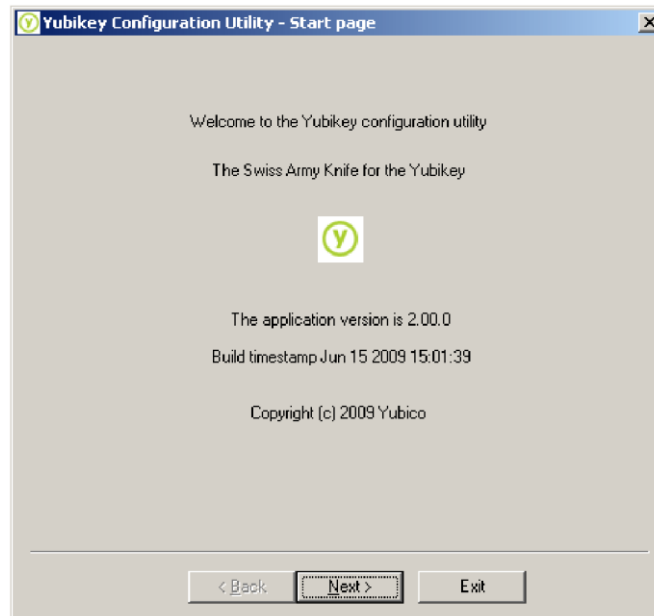


4 Using the application

Start the YubiKey Configuration Utility by launching the **YubiKeyConfig.exe** application.

The YubiKey Configuration Utility is designed as a series of “pages”, following a task-oriented, step-by step process. Pages have “Next” and “Previous” buttons and when the final step in a task has been reached, the “Next” button is replaced with a “Finish” button.

4.1 The Start page

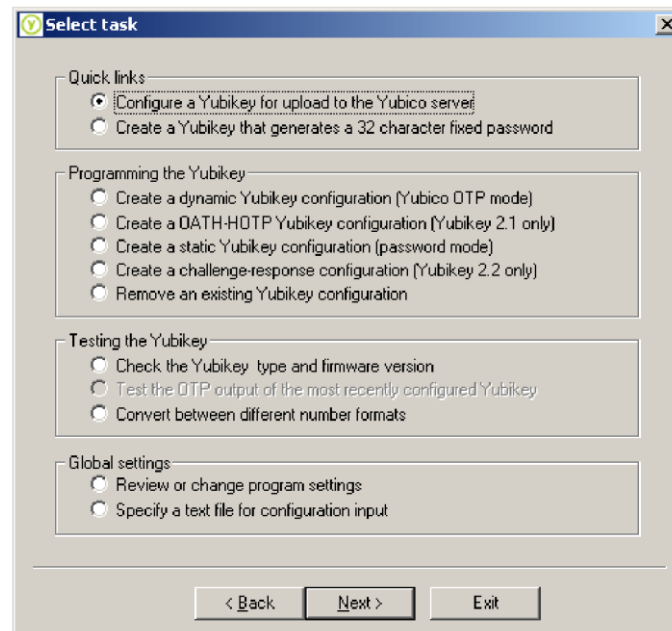


The start page displays the typical “About” information, including the application version number and build timestamp.

Press “Next >” to proceed to the task page

4.2 The Task page

The “main menu” of the YubiKey Configuration Utility is the Task page where the desired task is selected:



Select one of the following tasks and press Next >

- **Configure a YubiKey for upload to the Yubico server**
A simplified process to create a key for usage with the Yubico validation service
- **Create a dynamic YubiKey configuration (YubiKey OTPmode)**
A series of pages follows where input data is collected to create a configuration for a dynamic YubiKey configuration.
- **Create a OATH-HOTP YubiKey configuration**
This creates a dynamic configuration for the OATH-HOTP ecosystem
- **Create a static YubiKey configuration (password mode)**
A few steps are needed to create a static YubiKey, typically used to generate a “hard to remember, impossible to guess” password. The “Quick link” option provides a simplified setup.
- **Create a challenge-response configuration**
This creates a configuration that enables challenge-response operation, facilitated with a client-side interface program
- **Remove an existing YubiKey configuration**
This invalidates (removes) an existing configuration from a YubiKey
- **Check the YubiKey type and firmware version**
This selection provides a quick way to check the firmware version.
- **Test the OTP output of a configured YubiKey**
This feature is primarily intended for “lab”- and tutorial purposes, where the OTP output of a newly generated YubiKey can be tested.
- **Convert between different number formats**
This utility allows quick conversion between decimal-, hex- and Modhex formats
- **Review or change program settings**
Here, a few persistent settings for the application can be set or reviewed.

- **Specify a text file for configuration input**

Here, a text file holding pre-generated configuration input data can be selected.

4.3 Common configuration items

There are a few common items found on more than one page.

4.3.1 Binary strings

In the case where binary strings are to be supplied, there is a common entry is similar at all instances

One of four options can be selected:

- **Fixed value:** A single fixed value is used for all devices being configured
- **Increment by one:** After each configuration has been written successfully, the string is incremented by one. The increment is done from right-to-left.
- **Randomize:** Prior to each program operation, a new randomized value is generated.
- **Read from file:** The input for this field is taken from the text file specified from the Task page. If no input file is specified (see section 15), this option is grayed out. Items will be taken from columns in the order they are referenced. First item is taken from column 1, second item from column 2 etc.

Single rand generates a single random number and sets this as the current fixed value.

4.3.2 Remember settings

For most users, some settings are seldom changed or the default settings are sufficient. In this case, the settings can be remembered and the page will be skipped next time.

The selection is cleared each time the application starts and if the "< Back" button is used.

5 Create a dynamic configuration (OTP)

Creating a dynamic YubiKey configuration is the task that requires the most input although most fields have appropriate default values.

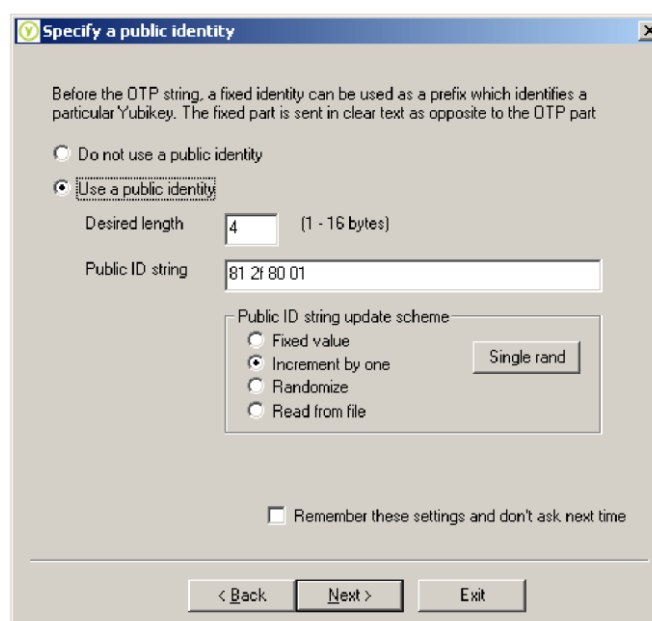
The process requires the following sequence of steps:

1. Specify public identity
2. Specify private identity
3. Specify cryptographic key
4. Specify output parameters
5. Specify configuration protection
6. Write the configuration to the YubiKey

If the OTP YubiKey is intended to be used with the Yubico validation server, there is a simplified way to configure a compatible key. See section 6 for further reference.

5.1 Specify public identity

The public identity is the first optional fixed part of the OTP string, used to identify a YubiKey. This field is sent in clear text.



The public identity is optional. If there is no requirement for it, check the “Do not use a public identity”.

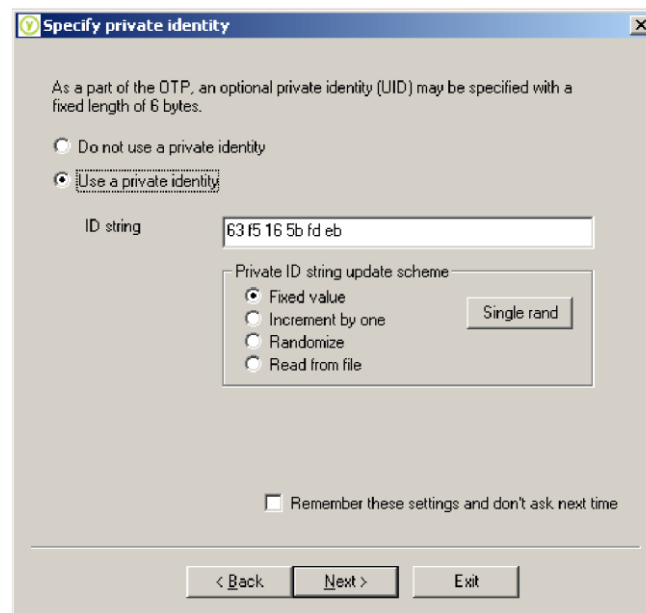
If used, a length between 1 and 16 bytes has to be specified. Any length between 1 and 5 bytes is considered a “private scope” and won’t create any interoperability issues. A public ID length of 6 bytes or more is for use with the Yubico validation server architecture or for future extensions. A unique customer prefix can be acquired from Yubico. The customer prefix is set in the configuration, see section 14.1. If a customer prefix is set in the configuration, a public ID length of 6 bytes is enforced, where the first three bytes contain the unique customer prefix.

The example above shows a public id string of 4 bytes. Converted to Modhex coding, this will be sent as **jbdvjccb**.

The public identity string generation is performed as described in section 4.3.1.

5.2 Specify private identity

The private identity is a secret field, included as an input parameter in the OTP generation algorithm.



Utilizing the private identity field is optional. If there is no requirement for it, check the “Do not use a private identity” and the field will be forced to all zeroes.

If used, the private identity string generation is performed as described in section 4.3.1. The example shows a fixed random value that was generated by pressing the **Single rand** button.

5.3 Specify cryptographic key

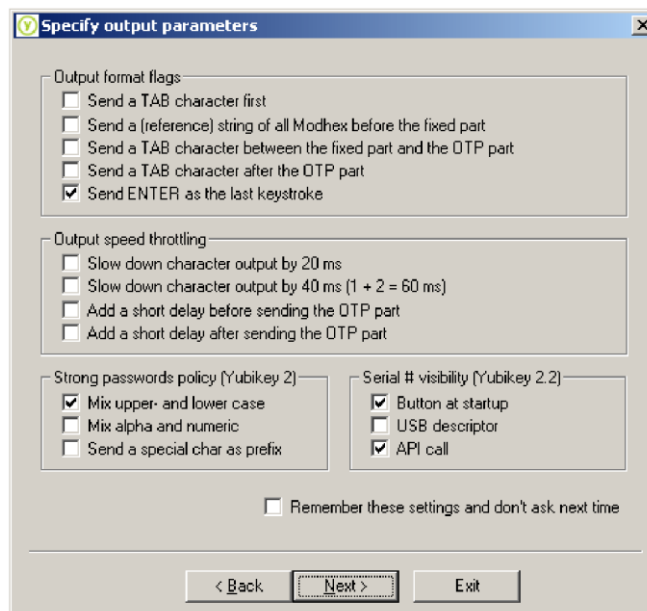
The cryptographic key is used to encrypt the OTP.



The key generation is performed as described in section 4.3.1. The example shows a key value that that will be read from the input file at the time of programming.

5.4 Specify output parameters

A collection of binary flags are provided to set various format- and behavioural parameters to either on or off.



The flags are organized in four groups:

5.4.1 Output format flags

Send a TAB character first

The first character will be a TAB, typically used to move the cursor to the next input field

Send a (reference) string of all Modhex characters before the fixed part

If there is an issue where keyboard mappings cause ambiguities, a reference string of all 16 Modhex characters can be sent first. This function cannot be used if “Mix characters and numeric digits” is set.

Send a TAB character between the fixed part and the OTP part

In order to separate the fixed (public identity) part and the OTP part, a TAB can be inserted to move the cursor to the next input field

Send a TAB character after the OTP part

A final TAB can be sent after the OTP part, typically used to move the cursor to the next input field after the password.

Send ENTER as the last keystroke

This selection sends an ENTER as the final keystroke, typically used to trigger a default OK button or to complete input from a command prompt.

5.4.2 Output speed throttling

Normally, the USB host polls the IN interrupt endpoint at the rate it can receive characters. The default poll rate is 10 ms which means that at full speed, a key entry is sent every 10ms. Each complete keystroke comprises a key-down and a key-up entry, which means that about 50 characters per second can be output.

If there are issues with lost characters due to a too high character output rate, the output can be slowed down.

Slow down character output by 20 ms

This selection adds a 20 ms additional delay for each keystroke (10 at down and 10 at up). Given a default endpoint poll rate of 10 ms, this throttles the rate to about 25 characters per second.

Slow down character output by 40 ms

To further slow down the output, an additional 40 ms delay can be added. Combining the 20 ms and 40 ms flags give an overall additional delay of 60 ms.

Add a short delay before sending the OTP part

If there is some parsing - or GUI rendering delay for a particular application, an additional 500 ms delay can be inserted before sending the OTP.

Add a short delay after sending the OTP part

If there is some parsing - or GUI rendering delay for a particular application, an additional 500 ms delay can be inserted after the OTP has been sent.

5.4.3 Strong password policy

Although passwords with a 32 character length can be seen as strong enough, certain legacy systems enforce a stricter policy where a combination of uppercase and lowercase characters or a combination of numeric and alpha characters are required. The goal with this function is not to add additional bits of entropy but rather to meet legacy standards.

Mix upper- and lower case

This flag enforces the two first Modhex characters to become shifted.

Consider the OTP string:

```
jfdkibjkegielgbgjkbejktuhurirnjt
```

Setting this bit would create the following output

```
JFdkibjkegielgbgjkbejktuhurirnjt
```

Mix characters and numeric digits

This flag enforces the two first Modhex characters in the range 0..7 to be replaced with digits 1...8. As this is not a part of the Modhex alphabet, it may cause incompatibilities with existing validation servers.

Setting this bit would create the following output:

```
j53kibjkegielgbgjkbejktuhurirnjt
```

Setting both bits would create the following output:

```
J53Kibjkegielgbgjkbejktuhurirnjt
```

Send a special character as prefix

Setting this flag will add a SHIFT-1 prefix, typically resulting in a '!' character. However, using this bit should be done with caution as SHIFT-1 has different mappings for different keyboard layouts. Furthermore, as this is not a part of the Modhex alphabet, it may cause incompatibilities with existing validation servers.

This function requires the "Mix upper- and lower case" flag to be set.

5.4.4 Serial number visibility

Introduced with YubiKey 2.2, a non-alterable, factory programmed unique serial number is included. The serial number has no direct link to the public identities configured. The configuration bits are logically ORed between the two configurations so if a bit is set in at least one of the configurations 1 or 2, that specific serial number feature is enabled.

Button at startup

Checking this option allows the serial number to be read at device powerup. Simply hold the YubiKey button while inserting the YubiKey in the USB port. Then release the button after one second and within 5 seconds. The serial number is outputted as keystrokes so keep a text editor or something open while performing this action.

USB descriptor

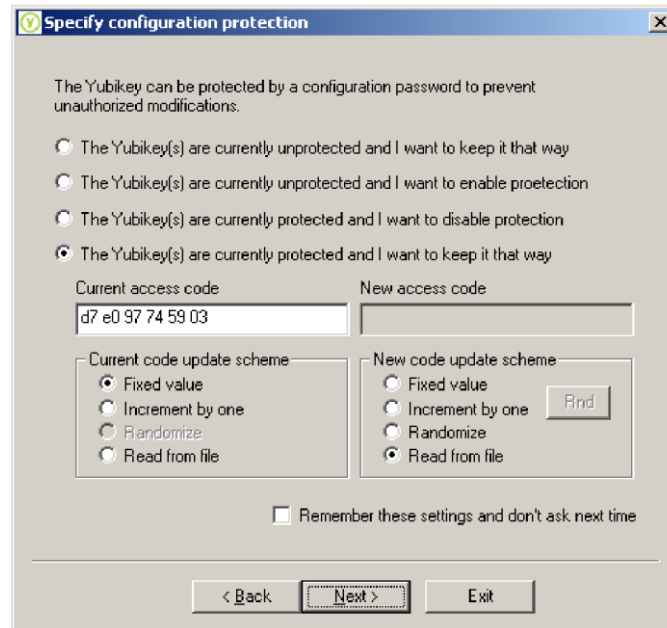
Checking this option makes the device serial number visible in the serial number field in the USB device descriptor. Please note that the device must be removed and reinserted after this bit is set in order for the operating system to recognize the updated device descriptor.

API call

Checking this option allows the device serial number to be read via a client-side application via a software interface, such as the YubiKey Client API.

5.5 Specify configuration protection

To protect against unauthorized update of a specific configuration, a protection access code can be added. Then, in order to update or remove this configuration, the corresponding access code must be used, otherwise the request is rejected.

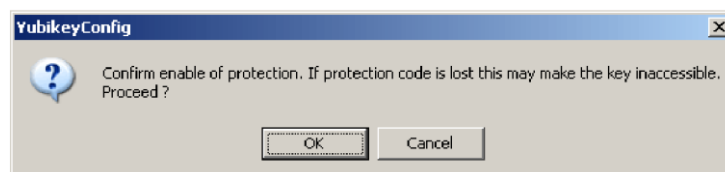


Four combinations are available:

- Unprotected -> Unprotected
- Unprotected -> Protected
- Protected -> Unprotected
- Protected -> Protected

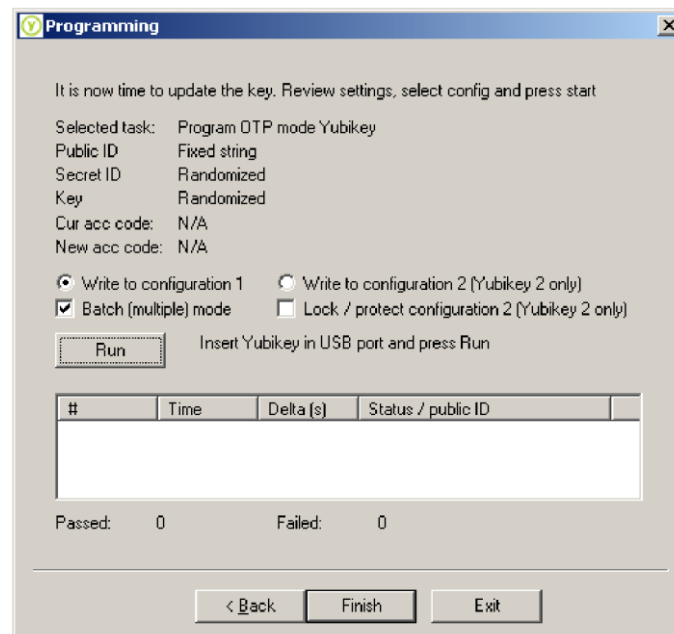
Generation of access codes is performed as described in section 4.3.1. In the example above, a key with a fixed access code **d7e097745903** will be changed to values read from the input file.

Caution: When protection is enabled the key can practically become “read-only” if the access code is lost. In order to enable protection, a confirmation dialog has to be confirmed first.



5.6 Writing the configuration to the YubiKey

The final step is to write the collected data to the configuration memory of the YubiKey.



Depending on the configuration settings (section 14.2) data about the programming is shown brief or verbose.

Before starting the programming, configuration 1 or 2 must be selected. Configuration 2 is valid for YubiKey 2 only.

The “Batch” option allows automatic programming when a new key is inserted, see section 5.6.2 for further details.

5.6.1 Lock / protect configuration 2

This function allows the second configuration to be locked or prevented from being locked in a scenario where two different “owners” or “holders” control the first and second configuration respectively.

- When set, writing to configuration 1 prevents configuration 2 from being written, i.e. the “holder” of configuration 1 can block configuration 2.
- Writing to configuration 2 with this bit set will prevent the “holder” of configuration 1 from setting its lock bit.

Together with the configuration protection (see section 5), the YubiKey can be protected from unauthorized updates.

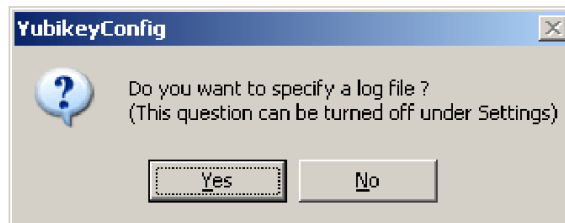
Example 1: A YubiKey is issued by a bank that uses configuration #1. The configuration is protected by an access password. The bank does not want the holder of the YubiKey to change the behaviour of the YubiKey and therefore sets the lock bit for protection #2. A casual user can then not write to configuration #2. Configuration #1 cannot be changed as there is an access password for it.

Example 2: Assume a user writing to configuration #2. This user does not know about the “owner” of configuration #1 and wants to protect itself from that “owner” from setting its lock

bit. If the user set the protect bit when writing to configuration 2, the owner of configuration #1 cannot lock update of configuration #2.

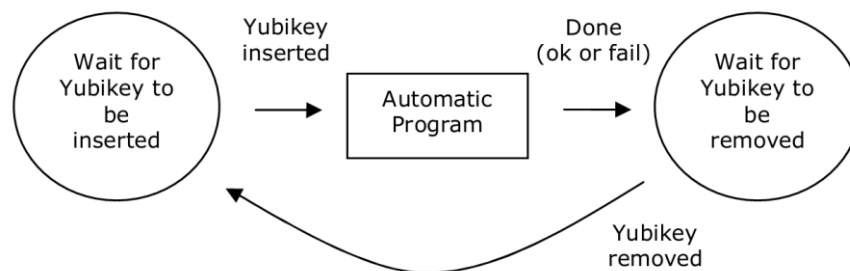
5.6.2 The programming sequence

When ready to start the programming, press Run. If the configuration is set to prompt for a log file (see section 14.2), the following question will appear:

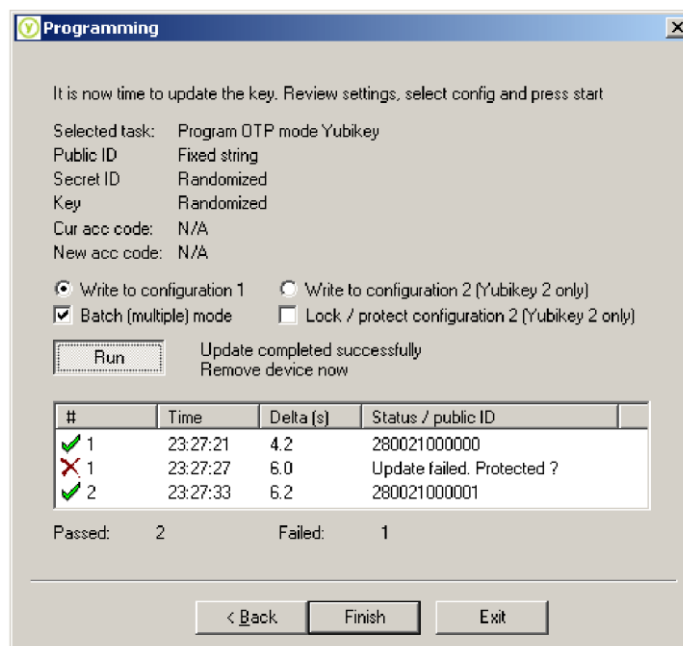


If Yes is pressed, a common “Save as...” dialog appears where the name and location of the log file is specified. Three different output formats are available – comma delimited, semicolon delimited or tab delimited.

Now, when the Run button is pressed, YubiKeys can be inserted, programmed and removed sequentially in a batch-like fashion if the Batch mode option is selected. If not selected, the Run button needs to be pressed for each programming operation.



The status output gets updated as programming progresses



Basic statistics show the number of passed and failed keys. In the example above, the second update failed where a YubiKey with “access protection code” set didn’t pass. Just remove the failing YubiKey and insert the next one and the programming resumes.

The column **Delta (s)** shows the number of seconds elapsed since the previous programming operation. This gives a hint of the programming throughput, i.e. number of keys that can be programmed per time unit.

For example, an average delta of 15 seconds gives a theoretical throughput of $3600 / 15 = 240$ programmed keys per hour.

If set, the log file is opened for each successfully updated YubiKey and a new entry is added. Depending on the output settings (see section 14.2 either the public id is written or both (public+ secret) fields are written.

In order to support efficient batch programming, sounds can be enabled under sound settings (see section 14.3). If set, two different tunes can be played depending if the programming operation passed or failed.

5.6.3 Access code protection confirmation

In order to prevent unintentional setting of the configuration password, a final confirmation dialog has been added which appears if no output log file has been selected and a new access code has been selected.



The user may here copy the access password to the clipboard and then paste it somewhere else as a reference if the particular YubiKey is to be reconfigured in the future.

As there is a configuration option to copy the most recent public id to the clipboard after the configuration has been written to the YubiKey (section 14.2), there is a risk that the clipboard data may be overwritten. Therefore, always paste the clipboard data before pressing OK here.

Checking the “Don’t show this message again” will not bring up the confirmation dialog at the next key to be programmed. The dialog is by default always brought up at the first key programmed after the program has been launched.

If randomize mode is selected for the configuration password, the confirmation dialog cannot be disabled and the “Don’t show this message again” box is therefore disabled.

The dialog is not shown if a log file is specified.

6 Create a YubiKey for the Yubico server

There is a simplified method of creating a dynamic (OTP mode) key for usage with the Yubico validation server. This process further comprises a simplified way to upload the generated key information to the Yubico validation server key storage.



Automatically send ENTER after the OTP

When checked, an ENTER keystroke is automatically sent after the OTP. This is useful when there is a default OK option or in command line operation

Slow down the character output rate

Some applications cannot receive keystrokes at full speed. Checking this option slows down the output rate.

Store the in the second configuration

When checked, the data is written to the second configuration on a YubiKey 2.

Automatically launch browser to go to upload site

When checked, the currently registered web browser is automatically launched to navigate to the YubiKey key upload site

Automatically pass identity (prefix)

When checked, the static part (public identity) is automatically passed as a parameter in the upload URL

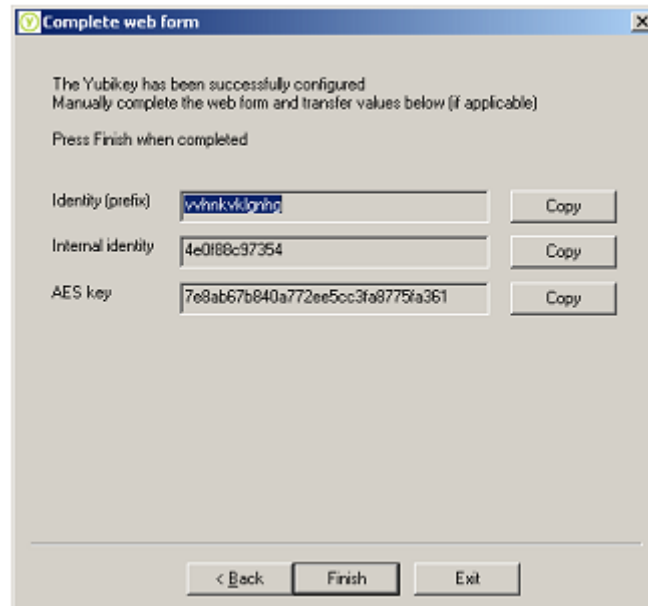
Pass secret parameters as well

When checked, the secret parameters (private id and key) are also passed as parameters in the upload URL. It is important to understand that this has some security implications. Furthermore, some web browsers store all URL information, including the parameters in the recently visited web site list.

To write the configuration to the YubiKey, press Next

6.1 Passing parameters

When the configuration has been successfully written to the YubiKey, a parameter passing panel is displayed.



Complete web form

The Yubikey has been successfully configured
Manually complete the web form and transfer values below (if applicable)
Press Finish when completed

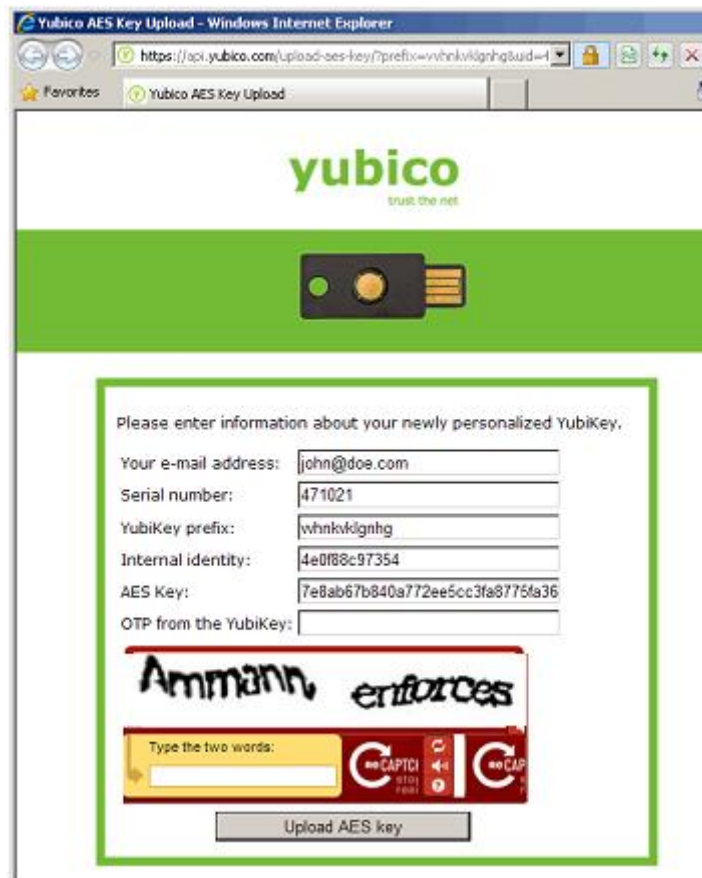
Identity (prefix)	vvhmkvldgnhp	Copy
Internal identity	4e0f88c97354	Copy
AES key	7e8ab67b840a772ee5cc3fa9775fa361	Copy

< Back Finish Exit

The public id, internal (private) identity and AES key can be manually cut and pasted via the clipboard by pressing the corresponding Copy button.

6.2 Completing the Web form

If the “Automatically launch browser”, the currently registered web browser is automatically launched.



The screenshot shows a web browser window titled "Yubico AES Key Upload - Windows Internet Explorer". The address bar displays the URL: <https://api.yubico.com/upload-aes-key/?prefix=whnkvlgnhg&uid=1>. The page features the Yubico logo and a green banner with a YubiKey icon. Below the banner is a form titled "Please enter information about your newly personalized YubiKey." with the following fields:

Your e-mail address:	<input type="text" value="john@doe.com"/>
Serial number:	<input type="text" value="471021"/>
YubiKey prefix:	<input type="text" value="whnkvlgnhg"/>
Internal identity:	<input type="text" value="4e0f88c97354"/>
AES Key:	<input type="text" value="7e8ab67b840a772ee5cc3fa8775fa36"/>
OTP from the YubiKey:	<input type="text"/>

Below the form is a CAPTCHA image showing the words "Ammann" and "enforces" in a stylized font. Underneath the CAPTCHA is a text input field labeled "Type the two words:" and a button labeled "Upload AES key".

Complete the screen by entering the requested Captcha and then output an OTP in the OTP field.

7 Create a dynamic configuration (OATH-HOTP)

The OATH-HOTP configuration allows the YubiKey to be used in an OATH-HOTP ecosystem as outlined by the RFC 4226 specification.

The process requires the following sequence of steps:

1. Specify OATH-HOTP specific data (Token identifier, HOTP format and optional seed)
2. Specify secret (key)
3. Specify output parameters
4. Specify configuration protection
5. Write the configuration to the YubiKey

The OATH-HOTP functionality is available from firmware version 2.1.

7.1 Specify OATH-HOTP specific data

The YubiKey supports the Token Identification Specification as outlined by openauthentication.org. If enabled, the YubiKey can automatically output a unique identification string preceding the HOTP.

The OMP is the OATH Manufacturer Prefix as assigned by openauthentication.org. Yubico owns manufacturer prefix **ub** to allow the OMP to be outputted as Modhex.

The TT is the Token Type, which can be any alphanumeric value.

The MUI is the Manufacturer Unique Identifier string, generated as described in section 4.3.1.

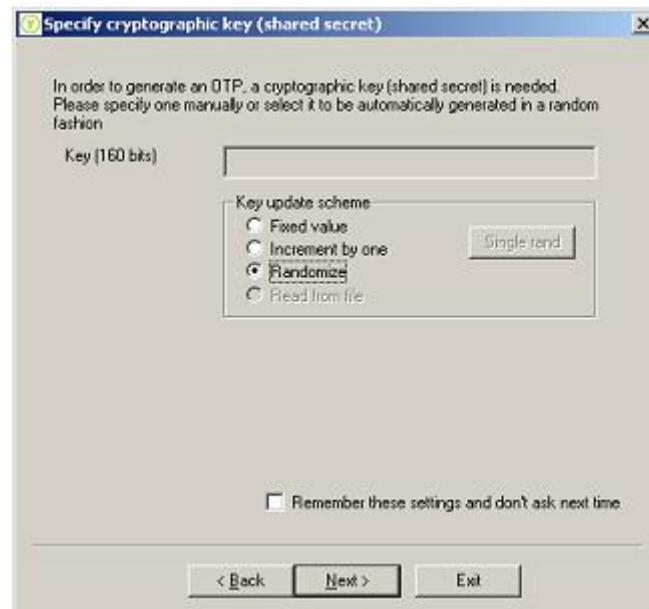
If a customer prefix is set under settings (section 14.1), OMP 'ub' is enforced and the TT and upper three digits of the MUI reflect the unique prefix. The lower 5 digits are then open to be used for a client specific range.

The HOTP truncated length can be set to either 6 or 8 digits.

Introduced in YubiKey 2.2, a seed can be assigned to the HOTP Moving Factor. The seed can either be fixed at zero, at a fixed integer value or a randomized number generated for each key in a sequence. The seeding number is 20 bits long with its four lower bits fixed at zero. This means that the seed is an even multiple of 16.

7.2 Specify cryptographic key (shared secret)

The shared secret is a 160-bits (20 bytes) binary string.



The key generation is performed as described in section 4.3.1. The example shows a key value that that will be read from the input file at the time of programming.

7.3 Specifying output parameters

This is the same options as for dynamic configuration. Follow the instructions in 5.4.

7.4 Specifying configuration protection

This is the same options as for dynamic configuration. Follow the instructions in 5.5.

7.5 Writing the configuration to the YubiKey

This is the same options as for dynamic configuration. Follow the instructions in 5.6.

8 Create a static configuration (Password)

The static mode is provided to create “hard to guess and remember” password. There are two options for password configuration – simplified and advanced.

8.1 Simplified mode

The simplified mode gives a quick path to configure a YubiKey in static mode with 32 randomized Modhex characters.



Automatically send ENTER after the password

When checked, an ENTER keystroke is automatically sent after the password. This is useful when there is a default OK option or in command line operation

Slow down the character output rate

Some applications cannot receive keystrokes at fullspeed. Checking this option slows down the output rate.

Store the password in the second configuration

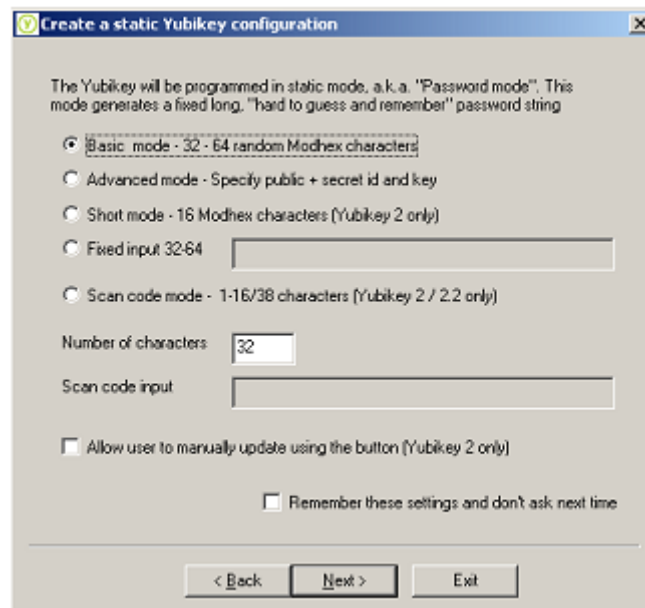
When checked, the password is written to the second configuration on a YubiKey 2.

Pressing Finish executes and completes the programming operation.

8.2 Standard (advanced) mode

The advanced mode provides more options to select the exact mode of operation. The advanced process requires the following sequence of steps:

1. Select mode
2. Specify output parameters
3. Specify configuration protection
4. Write the configuration to the YubiKey



Depending on the requirements, four different modes are available.

Basic mode

This is the simplest mode which applies default values to some underlying fields

Advanced mode

When there is a desire to have partial compatibility with the dynamic / OTP mode, all underlying parameters can be specified. This adds additional steps, like programming the key for dynamic mode. See sections 5.1 to 5.3 for a description of the dynamic mode parameters.

Short mode

This is like the basic mode but with the difference that the string is truncated to 16 digits. This feature is available for YubiKey 2 only.

Fixed input

In settings where input has been created by an external tool, a complete hexadecimal string can be pasted in here. The input will be automatically adjusted into the fixed and key fields.

Scan code mode

The scan code mode provides a mechanism to generate a string based on any arbitrary keyboard scan code. This mode may create incompatibilities if different national keyboard layouts are used as the mapping varies between countries. This function available for YubiKey 2 only. YubiKey 2.2 adds support for up to 38 characters

Simply place the cursor in the Scan code input field and type the desired string. The keystrokes are converted to scan codes.

Allow the user to manually update using button

This function can be used with legacy password systems where the user can update the device secret ID part by holding the YubiKey button for 8-15 seconds and then release it. The YubiKey LED then flashes and a single press confirms the update. When pressed, the secretID part is updated with a new random number and the new static password is outputted automatically.

A typical password update in a legacy system can therefore be performed as:

1. Select a “update password function”.
2. In the “Enter previous password” field, press the YubiKey button short to enter the previous password
3. In the “New password field”, press and hold the YubiKey button for 10 seconds and release. When the LED flashes slowly, press again shortly and the new password is outputted.
4. In the “Confirm password” field, press the YubiKey button again and the update is committed.

This function does not work in scan code mode.

8.3 Specifying output parameters

This is the same options as for dynamic configuration. Follow the instructions in 5.4.

8.4 Specifying configuration protection

This is the same options as for dynamic configuration. Follow the instructions in 5.5.

8.5 Writing the configuration to the YubiKey

This is the same options as for dynamic configuration. Follow the instructions in 5.6.

9 Create a challenge-response configuration

Challenge-response operation allows interaction between a client-side application and the YubiKey by support of a client-side application and interface software, such as the YubiKey Client API.

The challenge-response scheme can either be Yubico OTP compatible mode or HMAC-SHA1.

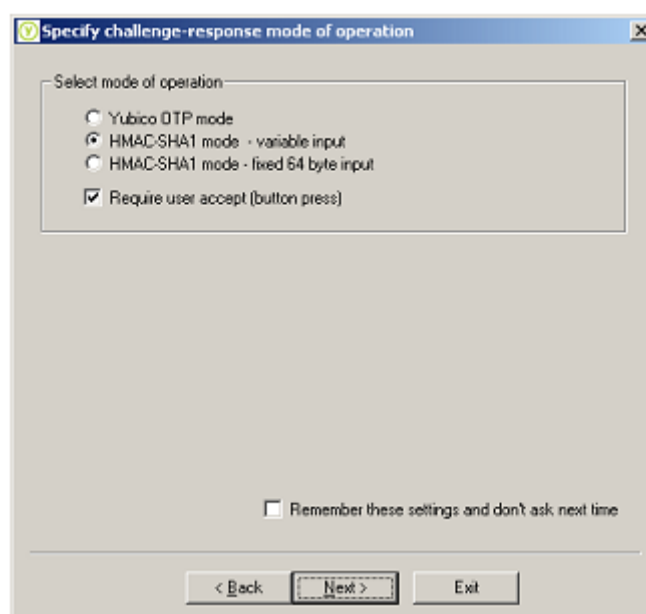
The process requires the following sequence of steps:

1. Specify challenge-response mode of operation
2. Specify private id (Yubico OTP mode only)
3. Specify secret (key)
4. Specify output parameters
5. Specify configuration protection
6. Write the configuration to the YubiKey

The OATH-HOTP functionality is available from firmware version 2.2.

9.1 Specify challenge-response mode of operation

First select the mode of operation:



Yubico OTP mode

The response is formed as a Yubico OTP where the challenge is first exclusive-ored into the private ID field prior to the encryption. Yubico OTP mode inserts timer- and counter fields and therefore creates a different response even if the challenge is identical.

HMAC-SHA1 mode

The response is formed as a HMAC-SHA1 operation of the challenge. The secret is fixed 20 bytes (160 bits). The challenge (data) can either be variable 0-63 bytes input or a fixed 64-byte string. In order to be compatible with the low-level USB interface, these modes have to be explicitly selected at the time of configuration.

Require user accept (button pres)

If this option is checked, the user has to manually acknowledge the challenge-response operation before the response is sent back.

9.2 Specifying private identity (Yubico OTP mode only)

Here, the private identity is specified that is used in the exclusive-or operation with the challenge. Follow the instructions in section 2.

9.3 Specifying key or secret

Here the Yubico OTP key (128 bits) or the HMAC-SHA1secret (160 bits) is set. Follow the instruction in section 5.2.

9.4 Specifying output parameters

Here, only the serial number visibility parameters are set. Follow the instructions in 5.4.

9.5 Specifying configuration protection

This is the same options as for dynamic configuration. Follow the instructions in 5.5.

9.6 Writing the configuration to the YubiKey

This is the same options as for dynamic configuration. Follow the instructions in 5.6.

10 Remove an existing configuration

This task removes (invalidates) an existing configuration. The process requires the following sequence of steps:

1. Specify configuration protection
2. Write the configuration to the YubiKey

10.1 Specifying configuration protection

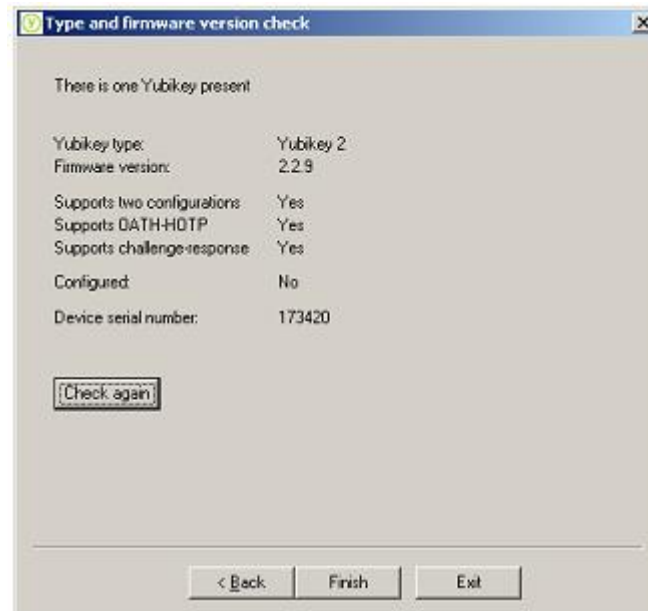
This is the same options as for dynamic configuration. Follow the instructions in 5.5.

10.2 Writing the configuration to the YubiKey

This is the same options as for dynamic configuration. Follow the instructions in 5.6.

11 Check the YubiKey version

This task provides a quick way to read out the YubiKey firmware revision, type and capabilities.



The "Configured" status shows status on a per-configuration basis from YubiKey firmware 2.1.

The "Device serial number" shows the factory programmed serial number from YubiKey firmware 2.2.

12 Test the OTP output of a YubiKey

This task is for advanced users to provide a “lab” tool, primary for testing and tutorial purposes. Here, the OTP output is decrypted based on the most recently written configuration and the individual output fields are displayed.

The test feature currently only Yubico OTP mode. Additional features include all modes can be tested with the Yubico Server API and associated test applications.

The cursor defaults to the first of the three input fields. Press the YubiKey trigger button and the output is captured and then decoded.

The application automatically restores focus to the first input field but moving around the cursor and then pressing the YubiKey trigger button may lead to invalid results as the output may not appear where intended.

How to interpret the meaning of the decoded output is out of the scope of this document please refer to related documentation is listed in section 1.3.

Field 1, 2 and 3 show the captured data entered into the input fields. The **Trigger by ENTER** shows if the output was finished with an ENTER keystroke or not.

Private id shows the decoded private id if successful. Otherwise the text “Invalid” is shown here. The distinction between a valid and an invalid OTP is determined only by verifying the decoded OTP timestamp.

Timestamp shows the current YubiKey timestamp output and the difference from the last received. The later field shall be related to the field **PC tstp**, which shows the delta in milliseconds taken from the PC clock.

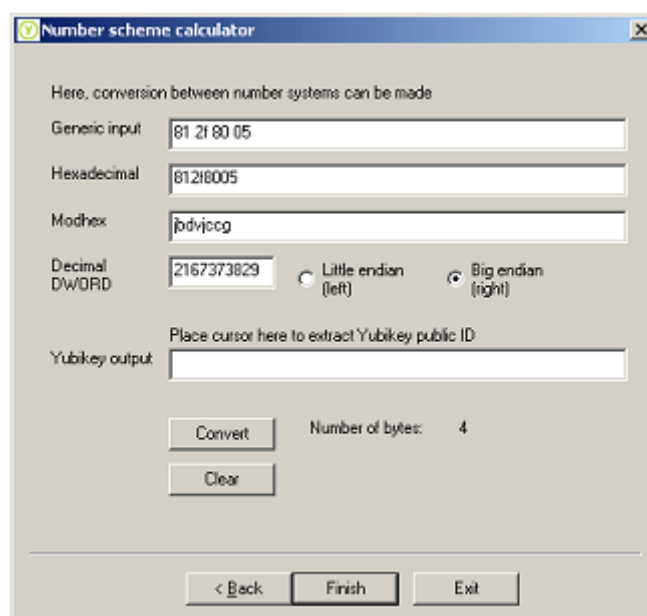
Ratio shows the ratio between the YubiKey timestamp and the PC timestamp. This value should be around 8 as this is the YubiKey timestamp frequency (in Hertz).

Counters shows the use counter and the session counter values.

Rnd shows the random number value.

13 Convert between different formats

A simple “calculator” is provided to allow quick conversion between different numeric representation formats.



Generic input

This field represents the common input field used throughout the application. This field type will parse the input string to determine if it is a delimited string, a packed hexadecimal or a Modhex one.

Hexadecimal

This field shows and accepts only packed (non-delimited) hexadecimal strings.

Modhex

This field shows and accepts only packed (non-delimited) Modhex strings.

Decimal (DWORD)

This field uses the first four bytes to represent a 32-bit unsigned long integer (DWORD). Byte ordering can either be Little Endian (LSB leftmost) or Big Endian (MSB leftmost).

YubiKey output

This field allows automatic extraction and calculation of the public ID from a YubiKey output string. The general format is to extract the first 32 – string-length characters.

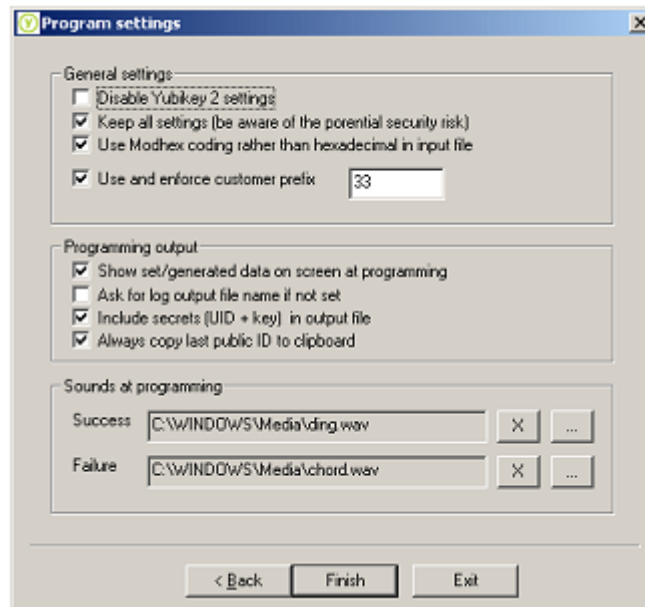
Consider the YubiKey output string

```
ccccccccftkbufkhlebivdrntelblbgkriigbjkugevb
```

Here, the first 12 bytes are the public id. Placing the cursor in this field and pressing the YubiKey button causes the first 12 bytes (**ccccccccftkb**) to be extracted and converted. The numeric ID 19857 is then displayed.

14 Program settings

Global persistent settings for the program are collected on a single page



14.1 General settings

Disable YubiKey 2 settings

When set, all YubiKey 2 specific options are disabled

Keep all settings

Normally, all selections for binary strings and schemes are reset when the program is exited so no security related information is stored persistently. Setting this flag causes all settings to be kept.

Use Modhex coding rather than hexadecimal in input files

The default format in input files is hexadecimal. Setting this flag uses Modhex coding instead.

Use and enforce customer prefix

If a customer prefix has been acquired, fill it in here and all public identities will have this customer prefix enforced

14.2 Programming output

Show/set generated data on screen at programming

For debugging and tutorial purposes, the generated fields (public id, secret id, key and access codes) can be displayed as they are to be written to the YubiKey.

Ask for log output file name if not set

Data written to the YubiKey(s) can be written to a sequential log text file. Setting this flag causes the application to prompt for a valid text file the first time a programming operation is initiated.

Include secrets (UID + key) in output file

Setting this flag included the secrets (private id and cryptographic key) to be included in the output log file.

Always copy last public ID to clipboard

When set, the last generated public ID is copied to the clipboard after a programming operation has been completed

14.3 Sounds at programming

To (potentially) make batch programming of keys more efficient, sounds can be added to programming operations. X clears the sound, ... browses for a file.

Success sound

When a programming operation is completed successfully, the sound specified here is played.

Failure sound

When a programming operation ends in a failure, the sound specified here is played.

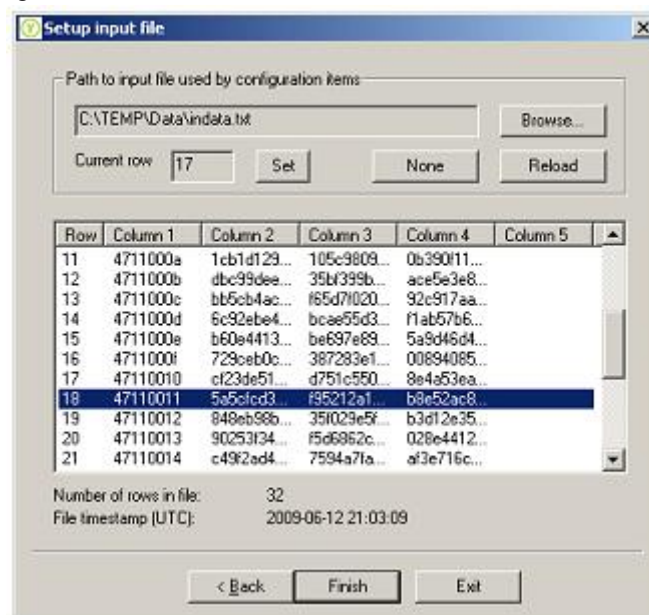
15 Specify a text file for configuration input

Input to data fields can be read from a file rather than being generated by the application. Any text file will do where hexadecimal- or Modhex items are separated by any non-valid character as delimiter

A sample input text file `indata.txt` comprising the data...

```
47110000,d92c3cb03cf7,b6e18a9ef958ba331958177d586a1cd0,2e69054a195a
47110001,bdffb816243a,0ce571abb39e87df473fb7165e343854,3f007f4efc96
47110002,a70bafb07afc,a5a94934d0beed9feb1d0cb1682ec750,fa945da85285
... following lines cut
```

... gives the following result:



The application then uses the columns in the order they are referenced. In the example above, the text file contains four columns targeted for public id, private id, key and access code.

Path to input file used by configuration items

Browse...

Browses for a file. Any delimited text file will do.

Reload

If the file has been modified while the application is running, pressing Reload will read in the newly updated data.

None

Clears the current file and flushes any buffered data.

Current row

This item shows the current row offset in the file. Each time a programming operation is been completed successfully, the row offset is incremented by 1 and the next row is read from the file.

Set

The file offset can be changed at any time by marking the appropriate row and then click “Set”. Double-clicking a row yields the same result.

Number of rows in file

This field represents the number of rows read from the file.

File timestamp (UTC)

This field represents the file’s OS timestamp, represented in Universal Time Coordinates (UTC) format.