yubico
the key to the cloud

YubiKey Authentication Module Design Guideline

Yubico Application Note

Version 1.0

May 7, 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

1 Document Information

1.1 Purpose

YubiKey Authentication Modules are developed to add YubiKey based strong two-factor authentication capabilities to server-side applications.

The growing popularity and adoption of YubiKey has resulted in a number of partner enterprise solutions and open source projects offering server-side YubiKey modules to fulfill the need for strong authentication.

However, Yubico and its customers have noticed a large variance in the deployment, management, authentication and recovery capabilities offered by these modules. So based on customer feedback and Yubico's experience, we propose module designers to follow a set of general guidelines for their module design that will help to improve on customer experience.

This document provides the general guidelines for the development of a YubiKey authentication module so that it will work smoothly for the majority of use cases we have been exposed to. It covers the capabilities we recommend to be supported and the considerations to be made when designing and developing a comprehensive and configurable YubiKey authentication module for server-side applications. The document also provides recommendations for how the module should be implemented, administered and maintained.

The document does not cover any specific platform or programming language details.

1.2 Audience

This document is intended for application and module designers interested in developing YubiKey Authentication Module for providing strong two-factor authentication using YubiKey.

1.3 Related Documentations

- The YubiKey Manual – Usage, configuration and introduction of basic concepts
- Getting Started Writing Clients
- YubiKey OTP Validation Protocol Version 2.0

1.4 Definitions

Term	Definition
YubiKey device or YubiKey	Yubico's authentication device for connection to a USB port
USB	Universal Serial Bus
OTP	One Time Password
YubiCloud	Yubico's online OTP validation service

2 Introduction and Prerequisites

YubiKey is a unique USB-key for secure, easy and affordable login to networks and services. Because of its ability to emulate a USB keyboard, it works from any computer for any number of applications with only user level rights and no client software needed.

Designed for any application where username/password is no longer secure enough, it is overall far easier to use and deploy/integrate than traditional display based tokens or smartcards.

The YubiKey is offered with free open source validation server and client components, a free authentication/validation service to meet the requirements of today's fast moving developers, enterprises, e-services and consumers. Yubico also supports open identity standards (SAML, OpenID, etc.). The free cloud based validation service enables one YubiKey to be used for multiple applications. This has resulted in a growing range of partner enterprise solutions and open source projects offering server-side YubiKey based strong authentication capabilities for many standard application frameworks and services.

A number of open-source YubiKey authentication modules have been developed for various open-source applications and frameworks by both Yubico and by the open-source community.

Some examples of applications and frameworks for which YubiKey Authentication Modules have been developed include Crasman, Drupal, Joomla, MediaWiki, phpBB, osCommerce, WordPress and so on. For a detailed list, please visit http://wiki.yubico.com

However, Yubico customers have noticed a large variance in how the modules are implementing support for deployment, management, authentication and recovery capabilities. So in order to guide module designers to more easily build comprehensive functionality and flexible yet powerful management control in the Authentication module, we put together this guideline and best practices which will also enhance customer experience.

Based on Yubico's experience and feedback from our customers, this document discusses various considerations to be made and provides best practices guidelines for application and module designers in order to provide comprehensive functionality and powerful yet easy to use deployment + management control that is flexible enough to meet a variety of real-life deployment needs for a strong two-factor authentication solution using YubiKey.

The document assumes that the reader is familiar with the following:

- Yubico OTP technology (OTP generation and validation)
- YubiKey OTP Validation Protocol Version 2.0

3 Overview

A two-factor authentication system authenticates the user based on two factors, something the user knows which normally is a combination of their user name and/or password and something the user has in their physical possession which in our case is the YubiKey. Two-factor authentication provides high security and is many times required for compliance with regulations. However, there are use cases where one-factor authentication is desired for simplified access or convenience with moderate level of security. We will cover development of both two-factor and one-factor authentication modes.

3.1 YubiKey One-Factor and Two-Factor Authentication

The YubiKey emits a 44 character One Time Password (OTP) of which the first 12 characters is a unique public identifier of the YubiKey itself and the characters that follows is the dynamic part of the OTP. A sample output from a YubiKey where the button has been pressed three times may look like this:

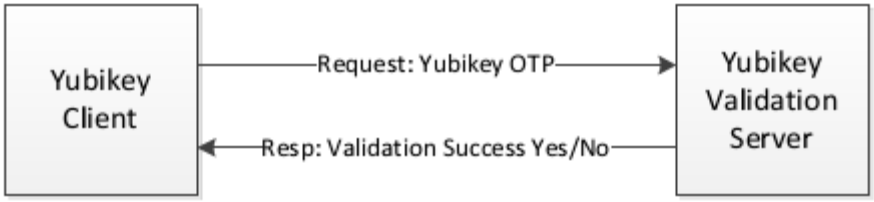
Public ID	**OTP**
`fifjgjgkhchb`	**`irdrfdnlnghhfgrtnnlgedjlftrbdeut`**
`fifjgjgkhchb`	**`gefdkbbditfjrlniggevfhenublfnrev`**
`fifjgjgkhchb`	**`lechfkfhiiuunbtnvgihdfiktncvlhck`**

This fact that the YubiKey emits both a static identifier part and a dynamic ever changing part of the OTP makes it possible to develop combinations of two-factor authentication and one-factor authentication system.

3.2 Base Components

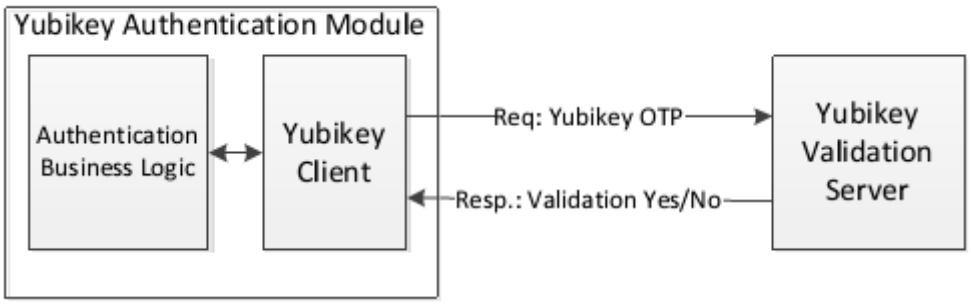
Yubico authentication framework server side components are available as open source. The base components provide validation of the YubiKey OTP linking it back to which YubiKey that was used and whether or not the OTP was emitted was valid.

Base components in a typical YubiKey authentication system are a so called “client” module responsible for implementing the transport protocol of the OTP and parse the response back from the second component, the Validation Server which is doing the actual validation of OTP.



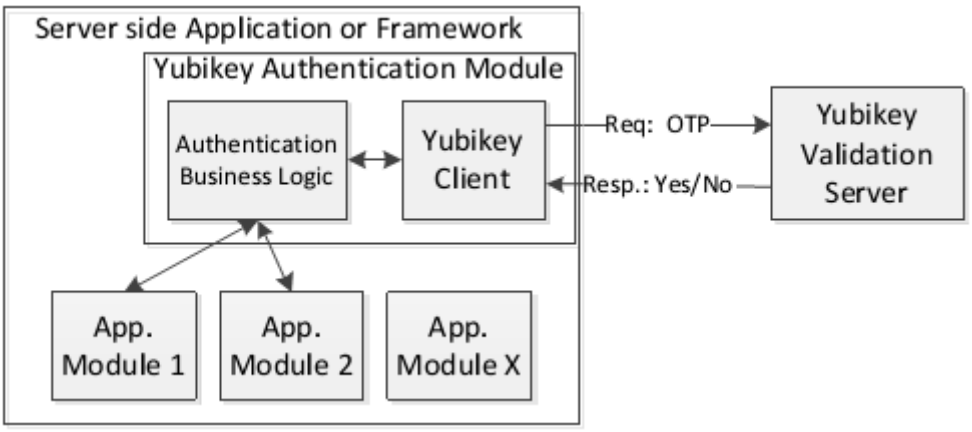
The YubiKey Client module is what you will integrate into your server side YubiKey Authentication Module. Clients are available in many programming languages http://www.yubico.com/web-api-clients and you can also write your own client (see references above in section 1.3).

A YubiKey Client implemented in a YubiKey Authentication Module would, at a high level, typically look something like the following:



The Authentication Business Logic implements YubiKey and user bindings and management of the YubiKeys.

The Validation module is then made part of a server side framework or application where the authentication module interacts with other application modules and components such as a database and user management and login modules.



Depending on the nature of the application or framework the designer has to take scaling, YubiKey provisioning technique, module installation, uninstallation and ongoing management into consideration.

3.3 Validation Server vs. Online Validation Service

The YubiKey Authentication Module can validate the OTP against either own Validation Server or against the Yubico Online Validation Service. You will have to decide which model works best for your application.

The main benefit with your own server is that you are in full control over all AES keys programmed into the YubiKeys.

The main benefit of using the Yubico online Validation Service is that the YubiKeys are already ready to use with the Online Validation Service out of the box (no additional handling needed).

3.3.1 Using onsite Validation Server

When validating against your own server, you are in control over the YubiKeys internal keys. The AES keys can be programmed using a personalization tool (available for free on the Yubico site) and each YubiKey's AES key will be recorded into a file that then in turn can be imported into the validation server database. The benefit is that you are in full control over your own keys.

3.3.2 Using Yubico Online Validation Service

The main benefit using the Validation Service is that the YubiKeys are already ready to use with the Online Validation Service out of the box. This also means that a user can use it with both your service as well with other 3rd party applications and services such as Google Apps for business users and LastPass for consumer password management. It also enables users that already have a YubiKey to sign up for your service.

3.4 Authentication Modes, Scaling, Provisioning and Administration

You must take authentication modes, scaling, provisioning of the YubiKeys and Administration into consideration when designing the module.

3.4.1 Authentication Modes

In order to be compliant with many industry legislations, two-factor authentication is required. You can choose to only support modes for two- factor authentication. However, if you design a YubiKey authentication module that is planned to be part of a framework then we strongly recommend you support the four basic authentication modes described below and provide an option for administrator to enforce a desired mode for each user group based on the security needs and policy requirements:

a. User Name + Password + YubiKey OTP
b. User Name or YubiKey OTP + Password
c. YubiKey OTP only
d. User Name + Password

3.4.2 Scaling

Depending on if you are creating an application or a module to work within a lager framework you may or may not know how many users you will have to plan for. For a framework such as Drupal, there can be sites with only a few users but also sites with millions of uses. Even if you may not design for the extreme, please consider modules working within frameworks to support 10,000 users as a minimum.

Also for scaling (if applicable) consider a hierarchical structure for administration and helpdesk functions.

As a best practice, Yubico recommends you to design the YubiKey Authentication module in such a way that it does not limit or adversely affect the scalability, administration and helpdesk function (if present) of the original framework/application in any way.

3.4.3 Provisioning

Provisioning of a YubiKey should be possible to perform by an administrator as a minimum.

The administrator shall be able to assign YubiKey(s) to a user. The administrator should be able to administer all other aspects i.e. mapping one or more YubiKeys to a user.

If Self-provisioning is used (optional requirement; but highly recommended in case of a large user base), then design the module so that it can support large consumer sites as well as sites with a more limited user base.

See more information in the details section below.

3.4.4 Administration

Administration of the module should be kept to a minimum. Self-provisioning, self-management by users should be considered to help minimize the administration burden especially for large sites.

See more information in the details section below.

4 Capabilities and Functionality Considerations

This section describes the various capabilities to be supported and the considerations to be made when designing the YubiKey authentication module for server-side applications.

4.1 YubiKey Provisioning

Users can normally have one or more YubiKeys associated with their accounts. However, as a sound security practice, it is highly recommended that the module should not allow assigning the same YubiKey to more than one user.

The first part (fixed 12 characters) of the YubiKey OTP is a unique identifier for the YubiKey and it is known as YubiKey ID. It should be used to maintain the User – YubiKey association in the module.

To associate the YubiKeys with user accounts, the module should implement support for one or both of the following two provisioning models depending on the needs and size of the application/service:

1. Provisioning by Administrator (this should always be supported)
2. Self-Provisioning by User (Optional but great for scaling a service)

The following sub-sections describe these provisioning models in more detail.

4.1.1 Provisioning by Administrator (Required)

Under this provisioning model, the administrator should have the ability to associate the YubiKeys with the user accounts.

This provisioning model provides tighter control over the users accessing the application and is recommended for applications with higher security needs and small user base. However, if the application user base is large, the YubiKey provisioning by the administrator can become quite cumbersome.

4.1.2 Self-Provisioning by User (Optional)

Under this provisioning model, the individual user has the ability to associate one or many YubiKeys with his/her account.

In self-provisioning model, the module should enable the user to register a YubiKey as part of the registration process or at first use of the application/service after installing and enabling the module. Until the point the first YubiKey is successfully assigned by the user, the module should enforce the original method (e.g. typically User Name + Password) for user authentication.

This provisioning model reduces the administrative burden and is recommended for applications with large user base.

4.2 Authentication Modes

A YubiKey authentication module should be designed to be very flexible. Based on customer feedback and Yubico's experience, we recommend that the module should normally support the following four authentication modes that cover the majority of use cases for usage of YubiKeys.

Authentication modes to be supported:

1. Username + Password + YubiKey OTP (Most secure)
2. Password + YubiKey OTP
3. (Username or YubiKey OTP) + Password (Convenience mode)
4. YubiKey OTP only

The administrator should be able to choose one of the above YubiKey authentication modes as the global selected user authentication mode for the application. The chosen authentication mode should be made binding for all users of all types.

The following sub-sections describe the possible authentication modes (as defined above) in more detail.

4.2.1 Username + Password + YubiKey OTP

This authentication mode requires the user to provide Username, Password and YubiKey OTP in order to login to the application.

It is the most secure mode of the four and should normally be set as the default for the module (after install).

We highly recommend you to consider an administrative configuration option to “Make OTP optional until a YubiKey is assigned to a user account” in a case there is a possibility of rolling out the YubiKey Authentication Module for applications already in production and having a moderate to large user base since distributing and assigning YubiKeys to users may take time. This option will allow administrators to create a window of time to allow uninterrupted service for users while they receive and assign their YubiKeys and after a reasonable period of transition move to stricter security enforcement.

4.2.2 Password + YubiKey OTP

With this authentication mode, the users can login to the application using their Password and YubiKey OTP.

YubiKey ID can be used to identify/map to a username for simplifying the logon process and offer better user experience.

4.2.3 (Username or YubiKey OTP) + Password

With this authentication mode, the users can login to the application using either their (Username and Password) or (YubiKey OTP and Password).

This mode does not provide any extra security. Instead it is pure convenience since the user can choose Username and Password without having to use a YubiKey.

It can however be useful in a transition phase while deploying YubiKeys to all users. In such a scenario the administrator can later on switch to the first mode where all three factors are required to login.

4.2.4 YubiKey OTP only

Using this authentication mode, the users can login to the application by simply providing their YubiKey OTP.

YubiKey ID can be used to identify/map to a username for simplifying the logon process and offer better user experience. This mode offers convenient one-factor authentication using the YubiKey. But it is not as secure as two factor authentication since if the YubiKey is lost or stolen, then (until disabled) the YubiKey can be used to login to a service by anyone by just pressing the button (provided the unauthorized user is aware of the service for which the YubiKey is enabled).

4.3 YubiKey OTP Validation

Yubico provides an online OTP validation service (called YubiCloud) to verify the OTPs generated by the YubiKey. Alternatively, organizations can choose to host their own YubiKey OTP validation servers.

The YubiKey authentication module should be designed to support both validating YubiKey OTPs against the Yubico's Online (OTP) Validation Service (YubiCloud) AND against an internal YubiKey OTP validation service.

4.3.1 YubiCloud - Yubico Online Validation Service

YubiCloud is Yubico's online OTP validation service that can be used to verify the OTPs generated by the YubiKey. YubiKeys by default come programmed for use with YubiCloud (out of the box).

YubiCloud uses the normal Yubico two tier approach (Validation server frontend and a KSM backend). It is based on a replicated architecture and operates with multiple validation servers in different geographical locations to avoid a single point of failure, reduce network delays and improve availability.

4.3.2 Internal YubiKey OTP Validation Server

Organizations and application/service providers can choose to host their YubiKey OTP validation servers. Hence, YubiKey authentication module should incorporate support for validating YubiKey OTPs with internally hosted YubiKey OTP validation server(s).

We recommend selecting between internal and external validation would be a configuration option under global configuration.

4.4 YubiKey Management

Managing the YubiKeys is an important part of supporting YubiKeys. In the simplest case the administrator would perform all management when needed. For larger sites, this would not scale so you may consider having the user to perform some tasks or having a hierarchy of administrators and helpdesk personnel.

4.4.1 YubiKey Management by Administrator

The module should enable administrators to manage the YubiKeys associated with user accounts. The administrator should be able to activate/deactivate and delete the YubiKeys associated with users. Only YubiKeys with "active" status can log into the application.

The administrator should always be able to assign the YubiKeys to the user accounts. This is particularly important – and mandatory - if self-provisioning of YubiKeys is not supported.

4.4.2 YubiKey Management by User

If support for self-provisioning of YubiKeys is implemented, then the module should normally allow the users to manage the YubiKeys associated with their accounts. This can be implemented differently depending on the type of service you provide.

For business, management of YubiKeys by users can be limited to reporting lost or stolen key (automatically suspending the associated YubiKey) and implementing a way for users to temporarily gain access to the service or some parts of it using some temporary means until a new YubiKey is received by the user.

For consumer based services (and if your security policy allows), users should be able to add new YubiKeys to their accounts. The user should also be able to activate/deactivate and delete the YubiKeys associated with the account as well as report lost or stolen YubiKeys. Depending on your service it may or may not be necessary to support users gaining temporary access to the services if a key as reported as lost or stolen (see section below). In some cases having multiple keys associated with the same account solves this problem especially for financial linked services.

4.5 Handling of lost, stolen or damaged YubiKeys

The module should incorporate a mechanism to gracefully handle the cases where the YubiKey is either lost or damaged. The module should facilitate the user to report and confirm the loss/damage of YubiKey so as to block any access to the server-side application by an unauthorized person finding the lost YubiKey. And if the self-provisioning by user is enabled, then the module should enable the user, whose YubiKey is lost/damaged, to add a new YubiKey to his/her account so that user can login to the application.

5 Implementation Recommendations

The YubiKey authentication module should be designed and implemented as a plug-in (whenever possible). The module should not be implemented by changing the core files of the underlying platform. This helps to ensure that the module is not broken when the underlying application platform is upgraded.

This section provides the recommendations for how the YubiKey authentication module should be implemented and maintained.

5.1 Installation

The administrator should be able to install, configure and enable the YubiKey authentication module from the admin console.

Installing the module should be easy and made as part of the module installation process; the following activities should be carried out:

1. Default configuration settings for the module should be defined
2. All database modification like creating new tables, defining module configurations etc. should be performed without affecting the other modules
3. Based on the underlying application platform requirement, the module should be registered with the application

It is recommended that the installation script provide support for all database systems supported by the underlying application platform.

5.2 Uninstallation

The module should in most cases support a way to gracefully uninstall the module if needed. In such a case the module should before uninstalling offer the administrator to back up the data used by the module in a file that should be possible to restore later if needed.

The backup does not have to be implemented as part of the uninstallation process and can instead be an option the administrator can select using the normal administration screen of the module. A uninstall of the YubiKey authentication module should not leave any other modules in a state where they will not function as normal.

5.3 Configuration

Once the YubiKey authentication module is installed and enabled, the administrator should be able to configure the global settings for the module.

The module should provide a module configuration interface where the administration can configure the following global settings for the module:

- YubiKey Authentication Mode
- Validation Service to use
- Enable/Disable module

The following sub-sections describe in detail various settings that should be configurable.

5.3.1 Choosing Authentication Mode

All suggested authentication modes may not have to be supported for all applications since the requirements may dictate that only some of the modes are compliant and therefore only those modes can be implemented. However for frameworks (e.g. Drupal) that will be used for multiple purposes we recommended that all modes are implemented and that you leave it to the site administrator to decide which modes to use.

From the module configuration interface, the administrator should be able to choose one of the following four authentication modes as the selected user authentication mode:

1. Password + YubiKey OTP
2. (Username or YubiKey OTP) + Password
3. YubiKey OTP only
4. Username + Password + YubiKey

For “Username + Password + YubiKey OTP” authentication mode, the interface should also provide an option to select “Make OTP optional until assigned” option. When this option is selected, users should be able to log into the application using Username and Password for as long as no YubiKey is assigned to their account. Once a YubiKey is assigned to a particular user account, then that user should be forced to provide an OTP from the assigned YubiKey in addition to the Username and Password in order to log in.

The option facilitates smooth roll out of YubiKey strong authentication solution for existing instances of an application having existing active users. For a large user base administrators can potentially use this option to provide uninterrupted access to users for certain limited time period (e.g. 15 days) until the YubiKeys are assigned and distributed to all the users and then the administrator can select to switch off this option or select other authentication mode.

Based on the selected user authentication mode, the login interface for the application should also be changed to avoid any user confusion. For example, if the selected authentication mode is “Username + Password + YubiKey OTP” with “Make OTP optional until assigned” option as selected, then the login interface should have three input fields namely Username, Password and YubiKey OTP in the sequence and it should be clearly mentioned that YubiKey OTP field is optional until assigned. Similarly, if the selected authentication mode is “Password + YubiKey OTP”, then the login interface should have two input fields namely Password and YubiKey OTP.

5.3.2 Configuring Validation Service

On the module configuration interface, the administrator should be able to choose between Yubico’s online OTP validation service and internal YubiKey OTP validation service. By default, the module should use the Yubico’s online validation service.

To use the YubiKey OTP validation service, the module needs to get its own Yubico API key and API ID. The module developer(s) can generate the API key using Yubico online API key generator at the following link:

https://upgrade.yubico.com/getapikey/

On the interface, the administrator should be able to configure the following validation service parameters:

- API Key – It is a shared symmetric key that is used to sign the OTP validation request and to verify the OTP validation response
- API ID - It is unique API ID that is used by the validation server to retrieve correct shared secret for signing the response
- HTTPs – It is a flag indicating that the module wants to use HTTPs to access the validation service
- Timeout – It is the number of seconds to wait for authentication responses before the request times out

If the administrator chooses to use the internal YubiKey OTP validation service, then the interface should enable the administrator to configure URLs to the internal validation server instances.

5.4 Integrating Yubico package

Yubico's online OTP validation service is available through web service APIs. To simplify the integration with the YubiCloud, Yubico offers a number of ready to use implementations of the Web Service API (called Yubico Clients) in various programming languages. The client libraries are available in C, PHP, Java, .Net, Ruby, PERL, Python and so on. For more information on these client libraries, please visit http://www.yubico.com/web-api-clients

It is highly recommended that the module developer use the latest version of one of these client libraries for integrating YubiKey OTP validation into the YubiKey Authentication Module.

5.5 YubiKey Management

5.5.1 YubiKey Management by Administrator

The module should provide an easy to use interface for YubiKey Management by Administrator on the administrator console.

The interface should be designed and implemented keeping in mind that there can be a large number of users with associated YubiKeys within an application or in a framework using the module.

We recommend that the interface should list the user YubiKey details in the tabular format with the following columns:

1. Username
2. YubiKey ID
3. Status – Active/Deactivated
4. Actions – Activate/Deactivate, Delete

The module could also list additional details like "Last Used" date time information and additional fields for helpdesk use etc. The interface should display a fixed number of records at a time and should implement appropriate paging functionality for easy navigation.

The administrator should also be able to search a particular user/YubiKey using username/YubiKey ID or portions thereof in a search function.

5.5.2 YubiKey Management by User

When requirements allow we recommend that the module should provide an (easy to use) interface for YubiKey Management by the user on the user console.

We recommend that the interface should list the user's YubiKey details in the tabular format with the following columns:

1. YubiKey ID
2. Status – Active/Deactivated
3. Actions – Activate/Deactivate, Delete

The module should also provide a form for adding a new YubiKey to a user's account. The user should be asked to provide a YubiKey OTP. The module should perform the following checks before adding the YubiKey to the user account:

1. OTP is a valid YubiKey OTP
2. YubiKey OTP is successfully validated with the validation service
3. YubiKey ID doesn't belong to any other existing user (if the requirements are same YubiKey cannot be used for multiple accounts)

Note: The module should make sure that every time it gets a YubiKey OTP from an end-user; it should consume that OTP (by sending a validation request to the server) thereby ensuring that the same OTP cannot be reused again.

5.6 Reporting of lost/damaged YubiKey

In case a user loses their YubiKey, the module should provide a mechanism to easily block the YubiKey to prevent any access to the site by an unauthorized person finding the lost YubiKey. On the login interface, the module should provide a link that users can use to report loss of their YubiKey.

It is recommended that the mechanism is designed as three step process:

1. Reporting
2. Confirmation (to avoid malicious/accidental locking out of other user accounts)
3. Resetting

The following sub-sections describe the above mentioned steps in more detail.

5.6.1 Reporting of lost Yubikey

In order to report the loss of YubiKey, the user should be asked to provide the Username/e-mail Address and Password (optional if user has forgotten his/her password) associated with his/her account.

On successful validation of user's username/e-mail address and password (if provided), a one-time confirmation link should be mailed to the user's registered e-mail address.

5.6.2 Confirmation

In order to confirm the loss of a YubiKey, the user needs to visit the one-time confirmation link sent in the confirmation mail.

When a user confirms the loss of his/her YubiKey, the following activities should be performed:

- All the YubiKeys associated with the user account should be temporarily blocked

- If self-provisioning of YubiKey is supported, then the user should be redirected to the "Reset YubiKey" interface
- If self-provisioning of YubiKey is not supported, then a notification email should be sent to all the concerned people (normally administrators) and the user should be notified whether he/she can log in or not before being assigned a new YubiKey
  a. And if allowed to login how i.e. what temporary method can be used until a new YubiKey has been received.

5.6.3 Resetting

Resetting should enable the user to either add a new YubiKey to the account or activate one of the existing YubiKeys.

The user should be asked to provide a YubiKey OTP from the YubiKey. The following activities should be carried out while resetting a YubiKey:

1. YubiKey OTP should be validated with the validation service
2. YubiKey ID should be checked to ensure that the YubiKey doesn't belong to any other existing user
3. If YubiKey already belongs to the user, activate this YubiKey
4. If YubiKey doesn't belong the user, associate the YubiKey with the user account and enable it

5.7 Logs and Reports

The module should log all the YubiKey based events with appropriate severity for regular monitoring by the administrators.

The module should provide various reports for the administrator and as a minimum the following reports:

- User-YubiKey mapping
- Users with no YubiKeys
- Deactivated YubiKeys
- Activity log – Record date/time, Public ID and activity (success/fail) for each authentication attempt (successful or not)

5.8 Documentation

The module should provide a detailed step by step configuration guide for the administrators and application developer interested in providing strong two- factor user authentication using YubiKey. The document should guide the readers through the steps required for installation and configuration of YubiKey authentication module for the application to enable YubiKey based strong two-factor user authentication capabilities. It should also describe a high level administration of the module.

The configuration guide should describe the following items in detail:

Prerequisites for the module

- Module capabilities
- Installation instructions
- Configuration instructions
- Un-installation instructions
- Module administration (Users and YubiKeys)

6 Checklist

This section provides a checklist of capabilities to be supported and the considerations to be made when designing the YubiKey authentication module for server-side applications.

No.	Design Consideration	Your Selection (check all that apply)
1	Installation	□ Yes □ No
2	Uninstallation	□ Yes □ No
3	Configuration Options • YubiKey Authentication Modes	□ User Name + PWD + OTP □ PWD + OTP □ (User Name or OTP) + PWD □ OTP only □ Make OTP optional until a YubiKey is assigned to a user account
	• Validation Service to use	□ YubiCloud Validation Service □ Your local validation server
	• Module enable/disable support	□ Yes □ No
4	Use of Yubico open source validation client	□ Yes □ No
5	YubiKey Management	
	• By site administrator	□ Yes □ No
	• By site users	□ Yes □ No
	• Self-provisioning support	□ Yes □ No
6	Assign multiple YubiKeys to users	□ Yes □ No
7	Report loss of YubiKeys by users	□ Yes □ No
8	Logs and Reports	□ Yes □ No
9	Module documentation	□ Yes □ No

Copy this checklist to the Yubico Wiki page where you describe your module. This way interested users will get a quick overview of supported functionality even before installing the module.

