

A MULTI-FACTOR AUTHENTICATION  
PROVIDER FOR THE DOTNETNUKE<sup>®</sup> OPEN-  
SOURCE CONTENT MANAGEMENT WEB  
APPLICATION FRAMEWORK

*BRANDON HAYNES  
HARVARD UNIVERSITY*

*MAY 1, 2009*

## ABSTRACT

DotNetNuke<sup>®</sup> ([www.dotnetnuke.com](http://www.dotnetnuke.com)) is the leading and most widely-adopted content management system and web application framework for the Microsoft ASP.NET platform. To date, it has been downloaded over six million times and its use is approaching a half-million production installations (1).

However, DotNetNuke does not provide an out-of-the-box multi-factor authentication story. While this may be seemingly possible through existing third-party extensions (such as Active Directory or IIS plug-ins), these extensions are often not sensitive to the myriad roles – both public and internal – that may exist in a particular DotNetNuke installation (given that multi-factor authentication may be needed only for a subset of those roles). This is especially complicated by the fact that DotNetNuke may multiplex an arbitrary number live websites, each having a unique set of roles and authentication requirements. An authentication system that is not able to differentiate between a user who possesses an ordinary level of privilege on one website and elevated privilege on another – where both websites are hosted within the same DotNetNuke installation – is of little practical utility. Any satisfactory solution must be role- and website-sensitive, and since those roles are configured at the DotNetNuke framework level, must have some level of integration with DotNetNuke itself.

This authentication provider offers this level of integration, and is designed to extend core DotNetNuke functionality to allow a host to configure enhanced authentication (including, but not limited to, SMS, SMTP, YubiKey, and X.509 certificates) for any number and combination of user roles across any number of websites in a given installation. Each configured factor must be fulfilled prior to authentication being granted, serving to increase security against some forms of external threat.

This provider is open-source and available under a liberal New BSD license. The software, along with technical documentation and source, are available on the project website located at [dnnmultifactor.codeplex.com](http://dnnmultifactor.codeplex.com).

# 1. INTRODUCTION

The DotNetNuke® content management system is a mature, robust, and highly-utilized web application framework. It is the “leading and most widely-adopted content management system and web application framework for the Microsoft ASP.NET platform”, and to date is rapidly approaching a half-million production installations (1). It offers role-based authentication using the ASP.NET membership subsystem, and allows for multiplexing across any number of websites in a single installation.

Despite its popularity, DotNetNuke does not offer an out-of-the-box multi-factor authentication story. This is especially unfortunate, especially as the platform is more widely utilized as a line-of-business portal for enterprise-scale applications such as high-volume e-commerce and banking. These applications, along with the fact that the platform offers cross-website "host" accounts of maximal privilege, require a higher-than-normal level of authentication than a simple username and password combination may be able to offer.

While this level of authentication may be seemingly possible through existing third-party extensions (such as Active Directory or IIS plug-ins), these extensions are often not sensitive to the myriad roles – both public and internal – that may exist in a particular DotNetNuke installation. This is especially complicated by the fact that DotNetNuke may multiplex an arbitrary number live websites. An authentication system that is not able to differentiate between a user who might possess an ordinary level of privilege on one website and elevated privilege on another – where both websites are hosted within the same DotNetNuke installation – is of little practical utility. Any satisfactory solution must be both role- and

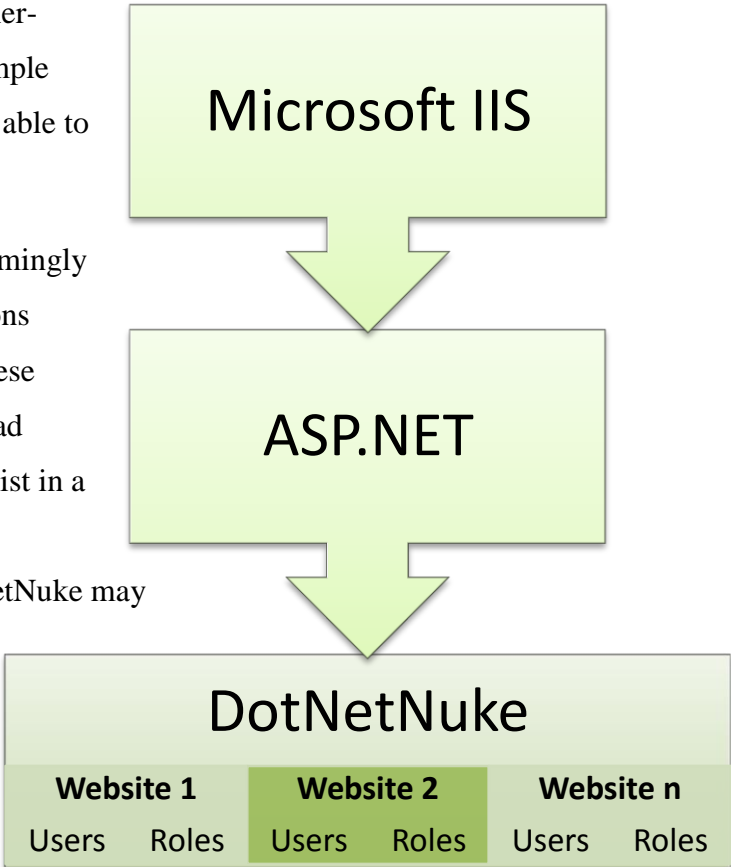


FIGURE 1: LOGICAL STRUCTURE OF DOTNETNUKE WEBSITES IN THE MICROSOFT STACK.

website-sensitive. Since those roles are configured at the framework level, any acceptable solution must have some level of integration with the DotNetNuke framework itself.

This project attempts to bridge that gap by extending existing DotNetNuke functionality to offer multi-factor authentication to those roles, users, and websites configured to require such enhanced services. Available factors include, in addition to the first-factor ASP.NET membership subsystem (which is always required): SMS, SMTP, YubiKey, and X.509 certificate-based authentication. The package is designed to be flexible and easily extended, allowing any arbitrary additional factors to be introduced by third-parties.

## 2. BACKGROUND

For those unfamiliar with one or more of the components described herein (e.g. the DotNetNuke framework, the Yubico YubiKey, et cetera) we briefly described each major component below.

### 2.1. MULTI-FACTOR AUTHENTICATION

Multi-factor authentication is the act of requiring more than one credential in the process of

binding a principal to an identifier; such credentials are typically

selected from the triad of those things that a person knows, possesses, or are inherent to that person. While variations exist, this has generally been effectuated through the use of passwords (which a person knows), tokens (which a person possesses), and biometrics (which are inherent to a person).

For some environments, multi-factor authentication is a good choice for increasing the overall security of a system (2) (but c.f. arguments to the contrary at (3)). While any security is only as

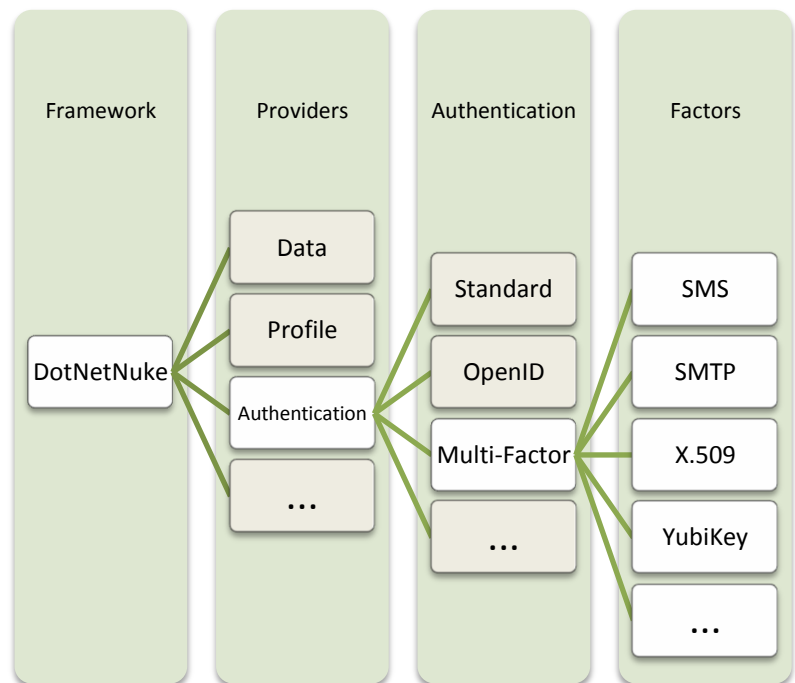


FIGURE 2: THE PROVIDER-BASED HIERARCHY ASSOCIATED WITH THE MULTI-FACTOR PROVIDER.

strong as its weakest link (which is often not authenticated-related), in many cases additional factors of authentication is a desirable goal (and, in some circumstances, is required by statute or regulation (4) (5)).

## 2.2. DOTNETNUKE®

DotNetNuke ([www.dotnetnuke.com](http://www.dotnetnuke.com)) is a popular open-source web application framework and content management system for the Microsoft ASP.NET software stack. It offers robust functionality, an active community, and a vigorous ecosystem of third-party extensions. To date, production installations are nearing a half million instances.

The DotNetNuke framework offers a wide variety of authentication options, including those developed against the ASP.NET

membership system (a provider-based system that integrates with options ranging from Microsoft Commerce Server 2009 to legacy ad-hoc systems) along with external options such as OpenID, Active Directory, and Windows LiveID.

DotNetNuke has a long history of taking the security of its product seriously, as evidenced both by its security policy, peer-reviewed source, and active response to potential vulnerabilities (6).

## 2.3. YUBIKEY®

Nominated as a “Best Innovation” in the 2009 European Identity Conference (7), the YubiKey – offered by Yubico Inc. ([www.yubico.com](http://www.yubico.com)) – is a physical authentication token that does not require client software, operates on multiple platforms, and does not require any battery. At a price point well below its competition (starting at around twenty-five USD per token), and with no other hardware or software requirements, it is an excellent low-cost solution for companies with heightened authentication needs (8).

Because of exorbitant prices, many small-to-mid-sized corporations (not to mention hobbyists, user groups, and other individuals) lack the resources to deploy robust enterprise-scale

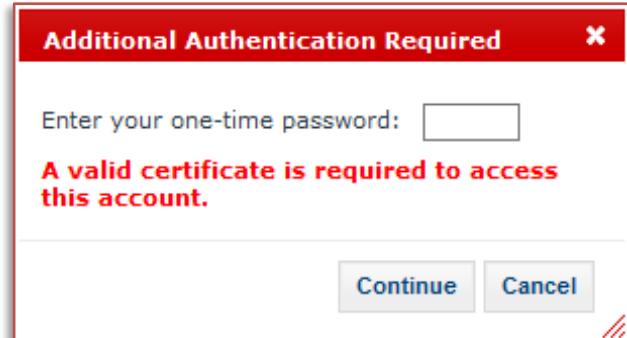


Figure 3: Authentication prompt for a DotNetNuke website configured to require one-time password and X.509 certificate to authenticate.



Figure 4: The YubiKey

authentication solutions. This includes many DotNetNuke hosts and clients. Due to this fact, the YubiKey solution was selected as the token-based factor for demonstrating the DotNetNuke multi-factor authentication system.

The factor design in this package was intended so that other token-based authentication configurations (such as the RSA SecurID system) may be easily incorporated by third-party developers, should the demand exist. The YubiKey factor implementation may be used as a model for these other options.

#### 2.4. CLICKATELL™

Clickatell ([www.clickatell.com](http://www.clickatell.com)) was selected as the secure SMS gateway for delivery of one-time passwords to an out-of-band device due to its affordability (beginning at about a \$0.05 USD per transmission), secure configuration, and its SOAP-driven webservice communication model.

The SMS authentication factor was carefully designed to be interoperable with other SMS gateways should a developer desire to utilize an alternative provider.

### 3. PROJECT LOCATION AND DOWNLOADS

This provider, along with technical documentation and source, are available on the project website located at:

<http://dnnmultifactor.codeplex.com>

Additional information, including screenshots and an in-depth technical discussion may be obtained at the website maintained by this author, located at:

<http://brandonhaynes.org>

### 4. PROJECT GOALS

We present the following goals as a satisfactory solution for the need described above, arranged in (roughly) descending order of importance:

- 4.1. ***A robust, extensible multi-factor authentication framework for the DotNetNuke content management system.*** This is the primary goal and *raison d'être* for both this document and the project itself.
- 4.2. ***Complete integration into the framework using existing extension points, with no core modifications or recompilation required.*** To maintain full compatibility with future versions of DotNetNuke, modifications to the core framework is strongly discouraged (9). This project thus will utilize only existing points of extension that are guaranteed to be stable across future releases<sup>1</sup>.
- 4.3. ***Support for host-, administrator-, and user-level configuration, with the ability to vary required factors across an arbitrary set of roles.*** A satisfactory solution must include configuration options for requiring additional factors of authentication for a wide variety of role combinations. This includes administrators, arbitrary sets and combinations of custom roles, and DotNetNuke “root user” accounts (referred to as “hosts” within DotNetNuke). Any given role may be subject to different authentication needs, and configuration should be as flexible as is feasible.
- 4.4. ***A robust set of included factor providers, including SMS, SMTP, YubiKey, and X.509 certificate.*** To increase adoption, a minimum set of core factors must be developed and maintained within the authentication provider itself. These factors also serve as examples against which other developers may create custom factors (or augment existing ones with new functionality). Based upon ease of implementation, likelihood of adoption, and cost of deployment, the following core factors are included in the initial release: SMS, SMTP, YubiKey, and X.509 client certificate.
- 4.5. ***Extension points in the authentication system allowing for development of custom factors by third parties.*** So that developers who wish to customize factors (or deploy authentication via factors not included in this distribution) are easily able to extend the provider to cover other scenarios, it is important that the authentication framework be easily extendable. This covers both new types of authentication that may be needed within an installation (e.g. biometrics), and modification to those factors that are already included (e.g. to enable an alternate SMS gateway).

---

<sup>1</sup> The last DotNetNuke version with intended breaking changes was 3.0.0, released March 2005 (11) (12)

- 4.6. ***Reliance on the existing ASP.NET membership subsystem for existing (first-factor) authentication.*** To minimize the risk of unforeseen vulnerabilities (which might allow elevation of privilege attacks within the multi-factor authentication system), the provider must rely on the existing ASP.NET membership subsystem insofar as is feasible. This includes, *inter alia*, the use of the subsystem for membership verification, along with ASP.NET membership services such as password generation (for one-time passwords<sup>2</sup>). This obviates a need to develop potentially risky ad-hoc security logic for some functionality.
- 4.7. ***Minimization of "ad-hoc security" risk by relying, insofar as is possible, upon existing security infrastructure (ASP.NET membership, DotNetNuke portal security, one-time password generation, et cetera).*** In order to minimize the need for risky and unpredictable ad-hoc security logic, the provider should rely, insofar as is possible, upon existing security infrastructure. This includes both the ASP.NET membership system and the open-source DotNetNuke portal security services.
- 4.8. ***As small an overall surface area as is possible and an absolute reliance upon the existing ASP.NET membership system as a first-factor fallback should any unforeseen vulnerability exist.*** Irrespective of configuration, authentication of existing ASP.NET membership should be enforced. This ensures that, irrespective of any potential vulnerability that might exist, a site is no weaker than it would be if it did not use multi-factor authentication.

## 5. AUTHENTICATION FACTORS

In this section we discuss the factors included in this release, cover the development steps necessary to create custom authentication factors, and describe in broad detail the extension of the SMS factor to other gateways.

---

<sup>2</sup> The utilization of well-vetted services is an important protection here; for example, we avoid the pitfalls associated with password generation (e.g. random number generation) by utilizing existing .NET cryptographic services.

## 5.1. INCLUDED FACTORS

The following factors are included in this distribution:

### 5.1.1. SMS

This factor utilizes the Clickatell SMS gateway (discussed above) for secure one-time password delivery. Factor configuration requires an account number and password; these data should be encrypted for additional protection (see implementation documentation for more details).

### 5.1.2. X.509 CLIENT CERTIFICATES

This factor utilizes the certificate presented by the client when making an authentication decision. The factor utilizes the underlying ASP.NET subsystem (which, itself, relies upon IIS) in making decisions related to the presence and validity of a client certificate. Assuming both, the certificate subject is compared to a (configurable) user property (by default the user's unique username). Alternate user profile properties may be used as a basis for comparison by including a directive in the factor definition (see the documentation for configuration details). In addition to built-in profile properties (username, e-mail, address, et al.), DotNetNuke allows for arbitrary additional profile details to be created, allowing great flexibility in binding a principal to a presented certificate.

### 5.1.3. SMTP

This factor utilizes the SMTP configuration within DotNetNuke for transmission of a one-time password. Though these settings may potentially be configured to transmit this information securely, in many circumstances this configuration should not be relied upon for true out-of-band one-time password transmission (an exception might exist for delivery to a handheld device external to the authenticating computer system via end-to-end secured transmission).

Due to these facts, this factor is primarily intended to demonstrate the abilities of the provider; significant caution should be used when relying on SMTP for secure out-of-band delivery.

### 5.1.4. YUBIKEY™

This factor requires activation of a valid YubiKey during authentication. By default, the factor authenticates against the Yubico-provided authentication service (at <https://api.yubico.com/wsapi/verify>), but this may be changed by specifying any custom verification URI. Such a configuration allows any server to authenticate a presented YubiKey, including one operated by an enterprise itself.

Note that by default if no YubiKey is associated with an authenticating account, it will be associated upon first use. Subsequent authentication requires that same YubiKey to be presented. This feature may be deactivated with a flag on the factor definition. Should this automatic association be disabled, accounts requiring YubiKey authentication must be manually associated (by a DotNetNuke host). If practical, it is a good idea to disable automatic association after all accounts have associated at least once.

Manual association may be effectuated through a properly-formatted HMAC-SHA1 on the relevant data (consult the YubiKey API for formatting specifications (10)). For convenience, this provider API exposes functionality for association as static methods defined as `YubiKeyFactor.AssociateKey` and `YubiKeyFactor.GenerateSignature` (both in the `BrandonHaynes.Membership.Factors` namespace) for signature generation. Consult the configuration documentation or source for additional details on effectuating manual association.

## 5.2. CUSTOM AUTHENTICATION FACTORS

This authentication provider is designed to be highly extensible and to allow for additional factors to be added with relative ease. Developers may deploy their own factors by implementing an interface (`BrandonHaynes.Membership.IAuthenticationFactor`) and ensuring that their class contains a required constructor accepting a string dictionary of attributes (see the configuration details for specific prototype details). Relevant factor parameters are injected through this constructor.

### 5.2.1. CREDENTIAL QUERY USER INTERFACE

Many authentication factors (such as those involving one-time passwords) must prompt a user for credential submission relevant to that factor. To accommodate this, each factor exposes a `PromptControl` property during implementation of the `IAuthenticationFactor` interface. This property is used to return a control that collects whatever additional criteria is required by the factor.

While any control may be used here, three are included herein for use in custom development:

1. `PasswordPrompt` (to input a one-time-password),
2. `MessagePrompt` (to display a simple message), and
3. `NullPrompt` (when the factor does not display or require any additional input).

Though any valid control may be used here, there exist some life-cycle implementation limitations. Consult the configuration documentation for more details.

### 5.3. CUSTOM SMS GATEWAYS

The SMS factor is designed to be extendable and allow for integration with alternate SMS gateway providers. This may be effectuated by deriving a new factor from the `BrandonHaynes.Membership.SmsFactor` class and overriding one or more of the following methods:

```
void SendSms(UserInfo user, string telephone, string onetimePassword)
string FormatE164Telephone(UserInfo user, string telephone)
```

Note that the current implementation includes only unsophisticated, United States-centric E.164 telephone number formatting. Those utilizing gateways with more complex (or non-US) formatting requirements should extend SMS functionality via the second method above.

### 5.4. MAPPING FACTORS TO ROLES

This provider supports the arbitrary mapping of defined factors to one or more roles. These roles may exist in any DotNetNuke website across an installation. When authenticating, the union of the sets associated with each of a user's roles is used to select factors. For example, consider the following configuration:

		Factors				ASP.NET (Always Required)
		SMS	SMTP	X.509	YubiKey	
Roles	Subscriber	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Premium Member	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

*FIGURE 5: SAMPLE FACTOR CONFIGURATION FOR A SINGLE DOTNETNUKE WEBSITE CONFIGURED TO REQUIRE MULTI-FACTOR AUTHENTICATION FOR TWO ROLES (SUBSCRIBERS AND PREMIUM USERS).*

A user who was *both* a subscriber and a premium member would be required to authenticate against the union of the two sets above, and would thereby be required to present a SMS one-time password, valid X.509 certificate, a YubiKey, and validate against the ASP.NET membership store.

## 6. DEPLOYMENT

This authentication extension has been designed for, tested against, and requires DotNetNuke version 5.0.1 or greater. Details about core installation and downloads are provided by DotNetNuke Corporation at (1).

## 7. KNOWN WEAKNESSES

Though there are currently no known weaknesses associated with this authentication provider, those implementing multi-factor authentication using this software are encouraged to track the open work items (and especially any security-related issues that might exist) on the project issue tracker, located at <http://dnnmultifactor.codeplex.com/WorkItem/List.aspx>. A RSS feed is available for on-demand updates.

## 8. LICENSE

The entire package along with all authentication implementation is offered under a New BSD license, which reads:

Copyright (c) 2009 Brandon Haynes  
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- \* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- \* Neither the name of Brandon Haynes nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF

MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The user-interface elements of this authentication system (located in the “Authentication Module” branch of the source code and limited to those source files containing a DotNetNuke copyright declaration) are derived from those present in the open-source DotNetNuke standard authentication system. Accordingly, those elements are covered under the following license:

Copyright (c) 2002-2008 by DotNetNuke Corporation

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Use of the Yubico-provided authentication webservices may be subject to license and use restrictions; those consuming this service are encouraged to learn more about such restrictions at (10).

Though there exist no known IPR issues associated with this provider, users are encouraged to evaluate any such issues prior to deployment. This project is not (currently) a DotNetNuke contribution, as defined under the DotNetNuke Corporation Contributor Agreement.

## REFERENCES

1. **DotNetNuke Corporation.** DotNetNuke - The Leading Open Source Web Content Management Framework for ASP.NET. [Online] [Cited: May 1, 2009.] <http://www.dotnetnuke.com/>.
2. *Comparing Passwords, Tokens, and Biometrics for User Authentication.* **O'Gorman, Lawrence.** 12, s.l. : Proceedings of the IEEE, 2003, Vol. 91.
3. **Schneier, Bruce.** Schneier on Security: The Failure of Two-Factor Authentication. *Schneier on Security.* [Online] March 2005. [Cited: May 1, 2009.] [http://www.schneier.com/blog/archives/2005/03/the\\_failure\\_of.html](http://www.schneier.com/blog/archives/2005/03/the_failure_of.html).
4. **PCI Security Standards Council, LLC.** Payment Card Industry (PCI) Data Security Standard. *PCI Security Standards Council.* [Online] October 2008. [Cited: May 1, 2009.] [https://www.pcisecuritystandards.org/security\\_standards/pci\\_dss\\_download\\_agreement.html](https://www.pcisecuritystandards.org/security_standards/pci_dss_download_agreement.html).
5. **Federal Financial Institutions Examination Council.** Authentication in an Internet Banking Environment. *Federal Financial Institutions Examination Council.* [Online] October 12, 2005. [Cited: May 1, 2009.] [http://www.ffiec.gov/pdf/authentication\\_guidance.pdf](http://www.ffiec.gov/pdf/authentication_guidance.pdf).
6. **DotNetNuke Corporation.** Security Policy. *DotNetNuke.* [Online] [Cited: May 1, 2009.] <http://www.dotnetnuke.com/News/SecurityPolicy/tabid/940/Default.aspx>.
7. **European Identity Conference.** Awards for outstanding Identity management projects. *European Identity Conference.* [Online] May 7, 2009. [Cited: May 7, 2009.] <http://www.id-conf.com/blog/2009/05/07/awards-for-outstanding-identity-management-projects/>.
8. **Datakonsult, Simon J.** Security Evaluation of Yubico. *Yubico.* [Online] 2007. [Cited: May 1, 2009.] [http://www.yubico.com/files/YubiKey\\_Security\\_Review.pdf](http://www.yubico.com/files/YubiKey_Security_Review.pdf).
9. **Walker, Shaun, et al.** *Professional DotNetNuke 5.* s.l. : Wrox, 2009.
10. **Yubico Inc.** <http://www.yubico.com/developers/api/>. *Yubico Web Service API.* [Online] 2009. [Cited: May 1, 2009.] <http://www.yubico.com/developers/api/>.

11. **DotNetNuke Corporation.** DotNetNuke 3.0. *DotNetNuke*. [Online] March 12, 2005. [Cited: 1 2009, May.] <http://www.dotnetnuke.com/TabId/792/Default.aspx>.
12. **Walker, Shaun, et al.** *Professional DotNetNuke 4*. s.l. : John Wiley and Sons, 2006.