

# DigiD-authenticatie in SharePoint

Matthijs Hoekstra

Omdat SharePoint is gebaseerd op ASP.Net is het mogelijk de authenticatie uit te breiden via membership en role providers. In dit artikel beschrijf ik hoe SharePoint gebruik kan maken van DigiD-authenticatie.


**De overheid maakt** in toenemende mate gebruik van DigiD-authenticatie. DigiD is net als Microsoft's LiveID en OpenID een identity provider. Identity providers zijn handig omdat je het account- en passwordbeheer kunt uitbesteden. In het geval van DigiD verzorgen zij voor de overheid het beheer rondom loginnaam en password. Ben je je password vergeten? Zij zorgen via een brief dat je de juiste logingegevens weer ontvangt. DigiD verzorgt ook alle acties rondom het activeren van een nieuw account en het resetten van passwords.

DigiD is ooit gestart door een aantal grote uitvoeringsorganisaties zoals Belastingdienst, CWI, UWV, IB-Groep en SVB en later overgedragen aan de gemeenschappelijke beheerorganisatie GBO. Overheid.

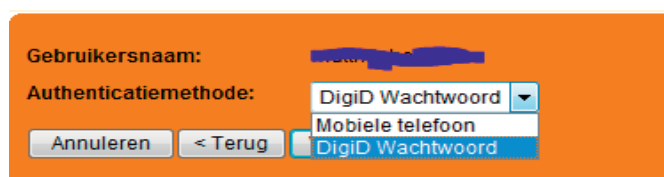
Ik heb tijdens een project voor een overheidsklant aan een portal mogen werken waar elke Nederlandse burger met een DigiD-account op moet kunnen inloggen. Na inloggen verschijnt er persoonlijke informatie, dus de noodzaak voor een goed security-systeem is aanwezig.

## Zelf kiezen voor niveau

DigiD kent drie niveaus van zekerheden: basis, midden en hoog. Op het basisoniveau maakt DigiD gebruik van een gebruikersnaam en wachtwoord. Op het middenniveau gebruikt DigiD een gebruikersnaam met wachtwoord en een transactiecode via sms. Op het moment van schrijven wordt het hoge niveau nog niet gebruikt, maar het is de bedoeling dat in de toekomst in te vullen met elektronische Nederlandse Identiteitskaart (eNIK). Als de gebruiker inlogt via DigiD kan hij zelf kiezen welk niveau van authenticatie hij wil gebruiken.



AFBEELDING 1. INLOGGEN DIGID



AFBEELDING 2. AUTHENTICEREN MET PASSWORD OF SMS-CODE

Na het invoeren van een loginnaam wordt de gebruiker gevraagd met het DigiD-wachtwoord of met de mobiele telefoon te authenticeren.

## Hoe werkt DigiD?

De volgende informatie komt uit de documentatie van DigiD. In afbeelding 3 zie je een diagram met de flow tussen onze applicatie (Webdienst), de gebruiker (Burger) en het DigiD systeem (DigiD). Het is handig te weten hoe de authenticatie technisch verloopt om dit artikel te begrijpen.

### \* Fase 0

De burger komt op onze website maar is nog niet geauthenticeerd.

### \* Fase 1

De gebruiker vraagt toegang tot onze website. Onze website neemt via een API contact op met het systeem van DigiD en verzoekt DigiD een authenticatie af te handelen. DigiD handelt authenticatieverzoeken alleen af als de webdienst zich authenticceert. We hebben hiervoor een certificaat en een 'secret' nodig. Deze worden aangeleverd door DigiD tijdens de aansluitprocedure van de webdienst. DigiD zal antwoorden met een sessieID en een url die we gebruiken om de gebruiker naar door te sturen.

### \* Fase 2

De gebruiker redirecten we naar de url van DigiD, zodat authenticatie door DigiD kan plaatsvinden. De SessieID slaan we tijdelijk op. DigiD kan nu de identiteit van de gebruiker bepalen met een bepaalde betrouwbaarheid.

### \* Fase 3

DigiD stuurt de gebruiker terug naar onze website. In de url worden daarbij gegevens over de afgehandelde authenticatiesessie meegestuurd.

### \* Fase 4

Wij controleren de meegestuurde gegevens via de API van DigiD. Als de gegevens kloppen, krijgen we op dat moment de identiteit (BSN) en het betrouwbaarheidsniveau terug.

### \* Fase 5

We weten nu wie de gebruiker is en kunnen een sessie starten met het BSN als identiteit.

We moeten dus als webapplicatie een aantal keren communiceren met de systemen van DigiD om gegevens op te halen en te controleren. DigiD biedt hier een aantal mogelijkheden voor: een SOAP-interface en een CGI-interface. Nu had ik het geluk dat een collega

al eerder met DigiD heeft gewerkt en er toen achter kwam dat de SOAP-implementatie te wensen overlaat en dus koos voor de CGI-interface. Deze interface hebben wij ook toegepast.

## Membership provider

In eerste instantie was ik begonnen met het maken van een membership provider. We kwamen er al snel achter dat deze voor een koppeling met DigiD niet geschikt is. Ik heb geen loginscherm met loginnaam en password. Ik hoef het password van de gebruiker niet eens te weten want daar maakt DigiD zich druk om. De meeste methodes voor het implementeren zouden voor een membership provider niet eens mogelijk zijn, zoals GetPassword, GetUserNameByEmail, createUser, et cetera.

We hebben dus een andere oplossing nodig en wel afhandelen van het login-proces via één aspx-pagina.

Op het moment dat SharePoint via de central admin forms authenticatie aanzet, wordt er een aanpassing in de web.config gemaakt om Forms-authenticatie mogelijk te maken. Via de central admin-pagina van SharePoint stellen we default-zone in op Forms Authenticatie.

De waarde voor de Membership Provider-naam is willekeurig. We gebruiken deze niet maar zijn wel verplicht hem in te voeren. De formslogin-configuratie passen we vervolgens aan door middel van het editen van de web.config zodat we onze eigen login-pagina voor DigiD kunnen maken en de bestaande login.aspx-pagina niet te hoeven overschrijven.

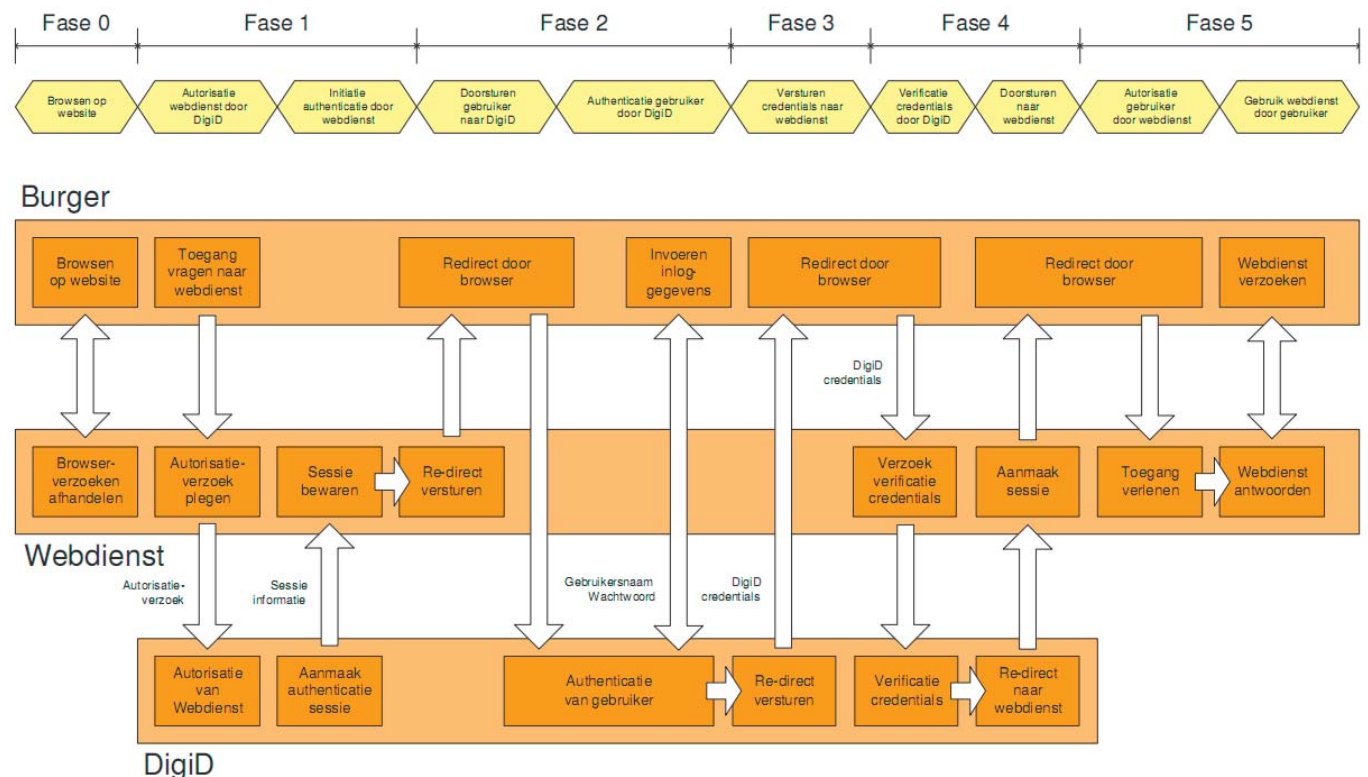
```
<authentication mode="Forms" >
  <forms loginUrl="/_layouts/loginDigiD.aspx" />
</authentication>
```

Normaal staat hier dus loginUrl="/\_layouts/login.aspx". Op het moment dat we SharePoint configureren dat iemand alleen geauthenticeerd op de site mag komen, is de loginDigiD.aspx pagina de enige anoniem toegankelijke pagina. Bij het openen van een andere url redirect SharePoint de gebruiker automatisch naar deze pagina.

Zoals het processchema voor DigiD-authenticatie laat zien maken we vanuit onze applicatie twee keer contact met de systemen van DigiD. De eerste keer om een sessie te starten bij DigiD en de authenticatie-url op te halen. De tweede keer om de gegevens die we via de url van de gebruiker krijgen te controleren bij DigiD en het BSN op te halen.

Zoals beschreven maken we gebruik van de CGI-interface van DigiD. Het gaat om een simpele HTTPS/GET-interface waarbij key value pairs worden meegegeven. Om een authenticatie te starten bij DigiD moeten we een "request-authenticate"-verzoek versturen. De volgende parameters zijn nodig om dit succesvol te doen: request, app\_url, app\_id, shared\_secret, a-select-server. De app\_url is de volledige url van onze website. DigiD zal na succesvol authenticeren de gebruiker naar die URL doorsturen. De voor onze applicatie unieke app\_id wordt uitgegeven door DigiD.

De shared\_secret is een authenticatiecode die we ook van DigiD krijgen. De a-select-server is de Server-id van DigiD.  
 Onze website is: <https://www.voorbeeld.nl/secure>  
 Het applicatie-id: dotnetmagazine\_digid\_portal  
 De authenticatiecode: 123456-kd2e-s3kg-72kf-k2f3-mk2e-aoe3  
 DigiD heeft als URL: <https://server.digid.nl/was/server>  
 DigiD heeft als ID: DigiD1



AFBEELDING 3. AUTHENTICATIEPROCESS-SCHEMA DIGID

## Hoe ga je iemand met een DigiD-account rechten geven op een onderdeel van je SharePoint-omgeving?

Als we alle parameters achter elkaar plakken, krijgen we het volgende voorbeeld:

[https://server.digid.nl/was/server?request=authenticate&app\\_url=https://%3A%2F%2Fwww%2Evoorbeeld%2Enl%2Fsecure&app\\_id=dotnetmagazine\\_digid\\_portal&shared\\_secret=123456-kd2e-s3kg-72kf-k2f3-mk2e-aoe3&a-select-server=DigiD1](https://server.digid.nl/was/server?request=authenticate&app_url=https://%3A%2F%2Fwww%2Evoorbeeld%2Enl%2Fsecure&app_id=dotnetmagazine_digid_portal&shared_secret=123456-kd2e-s3kg-72kf-k2f3-mk2e-aoe3&a-select-server=DigiD1)

DigiD zal vervolgens in één regel het antwoord terugsturen. Bijvoorbeeld:

[Rid=A77C582B33C03912&as\\_url=https://server.digid.nl/aselectserver/server?request=login1&a-select-server=DigiD1&result\\_code=0000](http://rid=A77C582B33C03912&as_url=https://server.digid.nl/aselectserver/server?request=login1&a-select-server=DigiD1&result_code=0000)

Anonieme gebruikers hebben alleen rechten op de loginDigiD.aspx-pagina. Daarom zorgen we ervoor dat na de start van het authenticatieverzoek en inloggen bij DigiD de gebruiker weer terugkeert op onze loginDigiD.aspx-pagina. De gebruiker is op dat moment wel geauthenticeerd bij DigiD maar nog niet op onze website. Dus blijft de loginDigiD.aspx-pagina de enige pagina waar de gebruiker op mag komen.

Zone  
Default

Authentication Type

Windows

Forms

Web single sign on

Enable anonymous access

Membership provider name:

Role manager name:

Enable Client Integration?

Yes  No

Save Cancel

AFBEELDING 4. FORMS AUTHENTICATIE VOOR ONZE SHAREPOINT SITE

Laten we eens naar wat code kijken. In de Page\_Load van onze loginDigiD.aspx-pagina controleren we of we de eerste keer op de loginpagina komen of dat we al bij DigiD zijn geweest als gebruiker. Door te controleren of er al een sessie bestaat en of we parameters zoals rid, aselect\_credentials en de a-select-server kunnen uitlezen, bepalen we of de gebruiker voor de eerste of tweede keer de loginpagina bezoekt. Als het de eerste keer is, moeten we een authenticatiesessie starten en de gebruiker naar DigiD sturen.

```
protected void Page_Load(object sender, EventArgs e)
{
    if (IsRedirectedFromDigid() || HttpContext.Current.Session.IsNewSession)
    {
        if (!IsRedirectedFromDigid())
        {
            StartAuthentication();
        }
        else
        {
            FinishAuthentication();
        }
    }
}
```

```
public void StartAuthentication()
{
    String requestUrl = String.Format(
        "{0}?request=authenticate&app_url={1}&app_id={2}&shared_secret={3}&a-select-server={4}",
        "https://server.digid.nl/was/server",
        "https://www.voorbeeld.nl/loginDigiD.aspx",
        "dotnetmagazine_digid_portal",
        "123456-kd2e-s3kg-72kf-k2f3-mk2e-aoe3",
        "DigiD1");

    // Call the DigiD API.
    StringDictionary returnedValues = CallCgiApi(requestUrl);

    // Put together the redirect url.
    string redirectUrl = String.Format(
        "{0}&rid={1}&a-select-server={2}",
        returnedValues["as_url"],
        returnedValues["rid"],
        returnedValues["a-select-server"]);

    //Store RID in Session to compare when user returns
    HttpContext.Current.Session["DigidRid"] = returnedValues["rid"];

    // Redirect the user to the DigiD website for filling in his/her DigiD credentials.
    HttpContext.Response.Redirect(redirectUrl);
}

private static StringDictionary CallCgiApi(string requestUrl)
{
    HttpWebRequest request = (HttpWebRequest)WebRequest.Create(requestUrl);

    string certificateFile = Path.Combine(
        HttpContext.Current.Request.MapPath("~/App_Data"),
        config.ClientCertificate);
```

```

X509Certificate2 certificate = new X509Certificate2(certifi
cateFile);
request.ClientCertificates.Add(certificate);

HttpWebResponse response = (HttpWebResponse)request.
GetResponse();

StreamReader reader = new StreamReader(response.GetRespon
seStream());
string responseAsString = reader.ReadToEnd();

response.Close();

// Return the data from the response body as name-value
pairs.
StringDictionary returnedValues = ParseResponseBody
(responseAsString);

return returnedValues;
}

```

#### CODEVOORBEELD 1

In de `IsRedirectedFromDigiD()`-methode checken we dus de drie parameters zoals hiervoor beschreven. Bij de eerste stap zijn we nog niet bij DigiD geweest en zal de functie `StartAuthentication()` worden aangeroepen. Deze ziet er uit als in codevoorbeeld 1.

Voor de communicatie met DigiD is een certificaat van PKIOverheid nodig. Voor de testomgeving voldoet een selfsigned certificaat. In dit geval laden we deze vanaf disk maar in de praktijk is het slimmer om hem in de certificate store te plaatsen en het SharePoint-account rechten te geven het certificaat uit te lezen. Als reactie van DigiD krijgen we, als het goed is, een resultcode van 0000. In ons voorbeeld hebben we die controle overgeslagen. We redirecten de gebruiker naar de url van DigiD en wachten tot hij weer terugkeert op onze login-pagina.

Als een gebruiker na succesvolle authenticatie terug is, kunnen we de tweede fase van ons authenticatie proces uitvoeren (codevoorbeeld 2). We verifiëren bij DigiD of de gegevens kloppen, die we hebben binnengekregen via de Querystring. Bij succes krijgen we via de "uid" het BSN van de gebruiker terug. Daarnaast krijgen we een melding over het betrouwbaarheidsniveau. We moeten zelf controleren of het niveau voldoende is.

Als alle controles zijn uitgevoerd kunnen we de gebruiker authenticeren. Dat gebeurt door de regel:

```
FormsAuthentication.RedirectFromLoginPage(returnedValues["uid"],
false);
```

We loggen de gebruiker in met het BSN als identiteit en het formscookie is niet persistent (false).

## Roleprovider

Nu we in staat zijn met DigiD te authenticeren lijkt het alsof we klaar zijn. We lopen echter tegen een probleem aan. Hoe ga je iemand met een DigiD-account rechten geven op een onderdeel van je SharePoint-omgeving? Normaal zou je via de site actions' permissies uitdelen aan een gebruiker. We hebben echter in dit geval geen database met accounts. Deze worden namelijk beheerd door DigiD en we krijgen alleen de mededeling of iemand wel of niet succesvol is aangemeld. Dus hoe ga je een gebruiker aan

SharePoint toevoegen waar je zelf geen account voor hebt? De truc vond ik in het voorbeeld van de LiveID provider in de Community Kit voor SharePoint (CKS), namelijk het schrijven van een roleprovider.

```

public void FinishAuthentication()
{
    String requestUrl = String.Format(
        "{0}?request=verify_credentials&rid={1}&aselect_
credentials={2}&a-select-server={3}&shared_secret={4}",
        "https://server.digid.nl/was/server",
        HttpContext.Request.QueryString["rid"],
        HttpContext.Request.QueryString["shared_secret"],
        HttpContext.Request.QueryString["a-select-server"],
        "123456-kd2e-s3kg-72kf-k2f3-mk2e-aoe3");

    // Call the DigiD API.
    StringDictionary returnedValues = CallCgiApi(requestUrl);

    string rid = (string)HttpContext.Current.
Session["DigidRid"];
    if (rid != returnedValues["rid"])
    {
        //Throw security exception,
    }
    else
    {
        FormsAuthentication.RedirectFromLoginPage(returnedVal
ues["uid"], false);
    }
}

```

#### CODEVOORBEELD 2

We maken een roleprovider en zorgen ervoor dat alle ingelogde DigiD-gebruikers de rol 'Authenticated DigiD user' krijgen. Deze rol kunnen we in SharePoint bijvoorbeeld leesrechten geven. Een roleprovider maken we door de interface `System.Web.Security.RoleProvider` te implementeren. Codevoorbeeld 3 laat een klein stukje uit de implementatie zien.

In onze situatie zitten alle gebruikers in de rol 'Authenticated DigiD Users'. Maar je kunt je voorstellen dat je in andere situaties het BSN van de gebruiker tegen een eigen database aanhoudt. Om zo te bepalen wat voor rol deze gebruiker moet hebben en vervolgens deze rol terug te geven aan het systeem. Je past dan de code aan voor de `IsUserInRole`-methode. Nu geven we altijd de "DigiD Authenticated Users"-rol terug. De overige methods van de class hebben we niet geïmplementeerd. Het aanmaken en verwijderen van rollen ondersteunen we niet in onze implementatie.

Om deze roleprovider te laten werken moet deze in SharePoint geconfigureerd worden. Dat gaat als volgt in de `web.config`:

```

<roleManager enabled="true" defaultProvider="DigiDRoles">
  <providers>
    <add name="DigiDRoles" type="DigiD.DigiDRoleProvider, DigiD"
/>
  </providers>
</roleManager>

```

Om personen rechten te geven, moet je als beheerder op de site kunnen inloggen. Als we alleen de DigiD roleprovider configureren kun je uitsluitend als DigiD user inloggen. Dit hebben we opgelost door de roleprovider toe te voegen aan de central admin site. Vanaf die site waren we in staat de rechten op onze SharePoint site te laten aanpassen door een beheerder. Dit hebben we

In dit artikel is expres gebruik gemaakt van ‘eenvoudige’ code. In de praktijk zul je hier nog grondig doorheen moeten lopen om security checks in te bouwen en exception handling toe te voegen.

gedaan door de volgende web.config-aanpassing te maken in de config van de central admin:

```
<roleManager enabled="true" defaultProvider="AspNetWindowsTokenRoleProvider">
  <providers>
    <add name="DigiDRoles" type="DigiD.DigiDRoleProvider, DigiD" />
  </providers>
</roleManager>
```

De truc is dus de defaultProvider `AspNetWindowsTokenRoleProvider`. Deze stelt je in staat ook nog steeds als Windows-gebruiker in te loggen. In ons geval is de website alleen toegankelijk voor DigiD-gebruikers en niet voor gebruikers met een Windows-account. Op de central admin site mogen beheerders met een Windows-account uiteraard wel inloggen. Nog een laatste tip. Bij het toevoegen van de rol in SharePoint kun je een speciale notatie gebruiken, namelijk de naam van de provider die we in SharePoint hebben geconfigureerd, gevolgd door een dubbele punt en de rolnaam die je wilt configureren. Dus in ons voorbeeld: "DigiDRoles:Authenticated Digid user". Ik heb gemerkt dat zonder de naam van de provider SharePoint de naam niet wil resolvable.

## Conclusie

Met dit artikel heb ik laten zien hoe je DigiD-authenticatie toevoegt aan je eigen SharePoint- of ASP.Net-applicatie. Ik probeer zo snel mogelijk een stub van de DigiD-service online te zetten, zodat je zelf alvast kan beginnen met ontwikkelen. Het duurt soms even voordat je bent aangesloten op de testomgeving van DigiD. Daarnaast is de stub een ideale manier om zelf al stress testen uit te voeren zonder dat de systemen van DigiD daar overlast van ondervinden.

Check mijn blog voor updates over deze stub (zie onder kopje 'Links').

Wanneer je zelf aan de slag gaat met DigiD kun je een starterspakket aanvragen. Dit pakket bevat uitgebreide documentatie op technisch vlak en daarnaast een aantal handreikingen hoe je DigiD kan implementeren in het eigen systeem.

Let op. In dit artikel is expres gebruik gemaakt van 'eenvoudige' code. In de praktijk zul je hier nog grondig doorheen moeten lopen om security checks in te bouwen en exception handling toe te voegen. Bovendien is het altijd verstandig het security-gedeelte te laten reviewen door een expert.



```
public class DigiDRoleProvider : RoleProvider
{
    protected string appName;
    private const string DIGIDROLENAME = "Authenticated DigiD Users";

    public override void Initialize(string name, System.Collections.Specialized.NameValueCollection config)
    {
        if (string.IsNullOrEmpty(name))
        {
            name = "DigidRoleProvider";
        }
        appName = name;
        base.Initialize(name, config);
    }

    public override string ApplicationName
    {
        get
        {
            return appName;
        }
        set
        {
            appName = value;
        }
    }

    public override string[] GetAllRoles()
    {
        return new string[] { DIGIDROLENAME };
    }

    public override string[] GetRolesForUser(string username)
    {
        return new string[] { DIGIDROLENAME };
    }

    public override bool IsUserInRole(string username, string roleName)
    {
        return (string.Compare(roleName, DIGIDROLENAME, StringComparison.OrdinalIgnoreCase) == 0);
    }

    public override bool RoleExists(string roleName)
    {
        return (string.Compare(roleName, DIGIDROLENAME, StringComparison.OrdinalIgnoreCase) == 0);
    }
}
```

### CODEVOORBEELD 3

#### Links

DigiD: [www.digid.nl](http://www.digid.nl)

CKS LiveID provider: [www.codeplex.com/CKS](http://www.codeplex.com/CKS)

Blog: [blogs.microsoft.nl/mhoekstra/](http://blogs.microsoft.nl/mhoekstra/)

Matthijs Hoekstra is Developer Evangelist bij Microsoft Nederland.