Ernst Peter Tamminga is managing consultant bij XCESS expertise center b.v.

Secure in Microsoft Office System

CODE SIGNING EN .NET

In dit artikel licht de auteur de mogelijkheden toe die een developer ter beschikking heeft om toepassingen en uitbreidingen in Microsoft Office 'secure' te maken. Secure in de zin dat de code doet wat je er zelf aan hebt bijgedragen, en niet hetgeen een incidentele voorbijganger er in heeft gewijzigd. Secure betekent in dit geval: veilig in gebruik en niet mee te manipuleren. Dit alles onder het motto: de gebruiker kan erop vertrouwen dat de programmatuur die ik voor hem heb gemaakt het doet in de vorm die ik aan hem heb geleverd.

In dit artikel ga ik geen diepgaande .NET-programmeerobjecten behandelen, maar ga ik in op een manier om je eigen ontwikkelwerk in een Office-omgeving een meerwaarde te geven. Die meerwaarde zorgt dat je veilig en betrouwbaar Office-uitbreidingen kunt maken, waarvan je kunt garanderen dat ze soepel bij de omgeving van je gebruiker geladen worden en de functies uitvoeren die jij erin geprogrammeerd hebt. Kortom, nuttig, direct toepasbaar en direct een toegevoegde waarde.

Waarom Office security?

Door het gemak waarmee je in Microsoft Office System veelvoorkomende taken kunt automatiseren - en daar speelt de macro recorder een belangrijke rol in - wordt er met goede bedoelingen nogal wat afgeknutseld. Steeds meer worden developers zich ervan bewust dat het niet voldoende is code te maken die goed werkt. Nee, in aanvulling daarop moeten ze ook garanderen dat de code goed blijft werken en niet door andere partijen bedoeld of onbedoeld kan worden aangepast. Met de openheid die onze systemen hebben, de uitwisseling die we via de e-mail plegen, de vaak onervaren en wellicht argeloze Office-gebruiker staat de deur open voor personen met boze bedoelingen. En daar is in de Office-omgevingen het nodige aan en tegen te doen. Dat is wat in dit artikel aan de orde komt. Lever Office-uitbreidingen die veilig en betrouwbaar en tot in lengte van dagen een voorspelbare werking hebben, ongevoelig zijn voor (on)bedoelde wijzigingen en malversaties van buitenaf.

Beveiliging in Office System

Was de Office macro/programmeeromgeving oorspronkelijk een 'open' omgeving waarin het heel gemakkelijk was om nieuwe functionaliteit aan Office toe te voegen, sinds de eruptie van de nodige virussen is Office met elke nieuwe release en servicepack meer dicht komen te staan. Het wordt haast onmogelijk om met de standaardinstellingen, zonder bevestiging van een gebruiker extra functionaliteit in een Office-pakket te laden.

Beveiliging kun je op een aantal onderdelen regelen:
* Het gedrag voor het laden voor macro's en add-ins: macro security
* Het beveiligen van de macro's en add-ins zelf: directories/ locations, password, code signing

Het verplaatsen van VBA code naar .NET code en daarin de beveoliging strikt regelen.
De basisinstellingen van beveiliging van VBA-macro's en add-ins regel je in het menu-item "Macro Security"; zie afbeelding 1.

Bij het lezen van de toelichting wordt al duidelijk dat je alles weer kunt openzetten (Low). Uiteraard wordt dit niet aangeraden. Als je weinig zelf wilt regelen, en het aan de gebruiker over wilt laten, kies je voor een Medium Security-level. Besef wel dat daarmee eenvoudig malafide en/of niet correct werkende projecten in Office geladen kunnen worden; niet aan te raden dus. Dat betekent dat je als developer bent aangewezen op de opties High en Very High. Deze laatste optie is weer nieuw voor Office 2003. De keuze (radio buttons) op het eerste tabblad van Macro security moet gecombineerd worden met de checkboxen op het tweede tabblad; zie afbeelding 2.

Samengevat is de strategie voor de ontwikkelaar:
* Security op High of Very High zetten - lijkt me vanzelfsprekend
* Werken met trusted locations - een trusted location is een folder op de computer van de gebruiker. Voor Word 2003 zijn deze folders: User templates, Workgroup templates, en Startup folders.
* Werken met digitally signed macro's - een manier om je macro's zelf te certificeren

En in aanvulling hierop: beveilig je macroproject (VBA-macro, add-in) met een password.

Password op macroproject

Een minimumniveau om jouw zo kunstig samengestelde VBA-code te beveiligen: zet een wachtwoord op het VBA-project als geheel. Je komt hier door vanuit de VBA-editor (Alt-F11) te gaan naar de VBA-code in het project dat je hebt geopend. Kies dan Tools, Project Properties, Protection. Vanuit het tweede tabblad in dit window (zie afbeelding 3), geef je aan dat de inhoud van het VBA-project niet voor onbevoegden ter inzage beschikbaar is. Geef dan ook een wachtwoord op waarmee de inhoud wordt beveiligd.

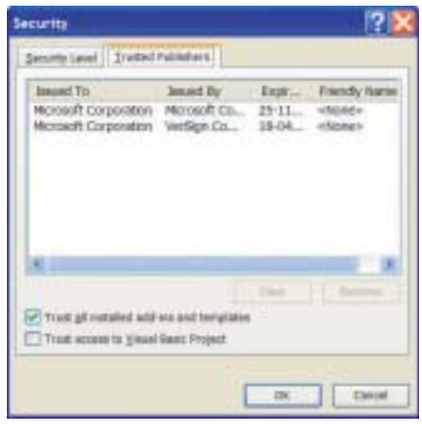
Een woord van waarschuwing: er bestaan tools om het wachtwoord te achterhalen van je aldus beveiligde VBA-project. Een wachtwoord is voor de kenner dus maar een lage drempel. Je kunt de drempel hoger maken door de standaard adviezen voor het samenstellen van wachtwoorden in ogenschouw te nemen. Minimaal 8 tekens, gebruik ook cijfers en leestekens, geen gewoon leesbare woorden, maar 'verzonnen' tekencombinaties. Vergeet niet je wachtwoord te documenteren, en beveilig het project pas in het laatste stadium van realisatie. Neem dan gelijk de gelegenheid waar om de basiseigenschappen (General, het eerste tabblad van Project Properties) in te vullen. Het kan nooit kwaad te documenteren.

Code Signing

Zoals je in de eerdere tabel hebt kunnen zien, begint het echte beveiligen en het veilig bruikbaar maken van je Office-projecten met het begrip Code Signing. Code Signing is het aanbrengen

net#5_p16-20 1.09 16 18-05-2004, 10:00:52

Afbeelding 1. Macro security



Afbeelding 2. Tweede tabblad macro security

Afbeelding 3. Inhoud VBA-project niet beschikbaar ter inzage

van een digitale signatuur aan je project, in combinatie met een certificaat waarmee achteraf de integriteit van je code kan worden vastgesteld. Om code signing toe te passen hoef je geen kennis te hebben van cryptografie of Public Key Infrastructuur; maar het mag natuurlijk wel.
De eerste keuze die je jezelf moet stellen is of het VBA-project alleen binnen jouw eigen organisatie gebruikt wordt? Of dat je de werking ook buiten de deur moet garanderen? Is het de eerste keuze, dan kun je zelf certificaten aanmaken die intern bereikbaar moeten zijn. Microsoft levert daar een programma voor: selfcert.exe. Dit hulpmiddel is ook handig om de werking van certificaten te testen.

Moet je code ook buiten de deur met code signing worden gegarandeerd, dan is het de vraag of iedereen die je project gebruikt bij de resources van jouw organisatie kan, of dat het een echt algemene, openbaar toegankelijke en verifieerbare signatuur moet zijn. Kunnen alle gebruikers een eigen Windows Server 2003 bereiken die je daarvoor beschikbaar maakt, gebruik dan de Windows Certificate Services binnen je organisatie en de kleine kring daarbuiten om de certificaten beschikbaar te maken. Moet je certificaat 'officieel' zijn, dan ben je aangewezen op een certificeringautoriteit, zoals VeriSign of Thawte. Die organisaties garanderen de gebruikers dat het door jou gebruikte certificaat ook 'echt' is. En natuurlijk zijn er aan zo'n 'echt' certificaat eenmalige kosten verbonden - enkele honderden euro's - en jaarlijkse onderhoudskosten.

Voor het vervolg behandel ik de stappen die met certificaten die je zelf kan maken met selfcert.exe:
1. Gebruik het programma selfcert.exe
2. Maak een testcertificaat
3. Signeer je VBA-project
4. Stel de Office 2003 security settings in en
5. Test de resultaten van het digitaal gesigneerd VBA-project.

Selftcert.exe is geen onderdeel van de standaard installatie van Office 2003. Je hebt dus de installatie-CD's nodig om het programma voor jezelf beschikbaar te maken. Na installatie is het bij mij beschikbaar in het Start-menu onder Office Tools. Bij het starten van het programma wordt je gevraagd een naam van het testcertificaat in te geven; zie ook afbeelding 4.

Na het ingeven van je eigen beschrijving klik je op OK. Met deze actie is onderhuids het programma makecert.exe aangeroepen (een command line utility, DOS is nooit ver weg) en is er een (test)certificaat aangemaakt. Met dit testcertificaat is je VBA-project te 'signen'. Open je VBA-project (Alt-F11 vanuit het document Windows of Tools, Macro, Visual Basic Editor). Kies vanuit de VBE-editor het project dat je wilt signen. Kies dan vanuit de

Security level	Trust all installed add-ins and templates checkbox	Digitally Signed	From a Trusted Publisher	VBA macros in a Trusted Location	Actie in Word, Excel, PowerPoint en Access
Very High	-	Ja	Ja	Ja of Nee	De add-in of macro wordt **niet geladen**
Very High	-	Ja	Nee	Ja of Nee	De add-in of macro wordt **niet geladen**
Very High	-	Nee	n.v.t.	Ja of Nee	De add-in of macro wordt **niet geladen**
Very High	Aangekruist	Ja of Nee	Ja of Nee	Nee	De add-in wordt **wel geladen**, maar de macro **niet**
Very High	Aangekruist	Ja of Nee	Ja of Nee	Ja	De add-in en de macro worden **geladen zonder bevestiging**
High	-	Ja	Ja	Ja of Nee	De add-in en macro laden **automatisch**
High	-	Ja	Nee	Ja of Nee	De **vraag wordt gesteld** of de add-in of macro beschikbaar moet
High	-	Nee	n.v.t.	Ja of Nee	De add-in of macro wordt **niet geladen**
Medium	-	Ja	Ja	Ja of Nee	De add-in en macro laden **automatisch**
Medium	-	Ja	Nee	Ja of Nee	De **vraag wordt gesteld** of de add-in of macro beschikbaar moet
Medium	-	Nee	n.v.t.	Ja of Nee	De **vraag wordt gesteld** of de add-in of macro beschikbaar moet
Low	-	Ja of Nee	Ja of Nee	Ja of Nee	De add-in en macro laden **automatisch**
High, Medium of Low	Aangekruist	Ja of Nee	Ja of Nee	Ja	De add-in en macro laden **automatisch**

Dit levert het volgende keuzepatroon op

net#5_p16-20 1.09 17
18-05-2004, 10:00:56

Afbeelding 4. Geef naam testcertificaat

Afbeelding 5. Overzicht beschikbare certificaten

VBE-omgeving voor Tools, Digital Signature. In dit window kun je een digitaal certificaat aangeven na het drukken op Choose. Je krijgt dan een overzicht van de certificaten die op jouw machine beschikbaar zijn; zie ook afbeelding 5.
Je kunt het certificaat kiezen dat je zojuist hebt aangemaakt. Klik op OK en OK, en je VBA-project heeft een digitaal certificaat gekregen! Bewaar je document (.dot-file, XLS-bestand, PPT-file). Nu moet je Office-programma nog ingesteld worden opdat deze het gesigneerde VBA-project zonder te mopperen activeert.
Start je Office-programma, bijvoorbeeld Word. Kies Tools, Macro en dan Security. Kijkend naar de eerdere tabel, zien we dat het maximum aan security dat we kunnen kiezen High is, als we besluiten dat een VBA-project vanaf iedere plek geladen moet kunnen worden en we niet kiezen om het VBA-project in een Trusted Location (User templates, Workgroup templates, en Startup folders) te plaatsen. Een strategie die een balans is tussen het continu plaatsen van VBA-project in deze bijzondere folders, dan wel een VBA-project vanaf een willekeurige plek opstarten. Heb je overigens in je opgenomen dat de instelling van deze folders dus belangrijke gevolgen heeft voor de VBA-projecten die daar geplaatst worden in combinatie met de beveiligingsinstelling? Er is sprake van een beveiligingsgat als je dit niet goed controleert.
Kies in de Macro Security voor High (eerste tabblad) en kruis in het tweede tabblad *Trust access to Visual Basic Project* aan, maar *Trust All installed add-ins and templates* uit. Door deze instelling kun je alleen VBA-projecten laden die je 'vertrouwt', oftewel die gesigned zijn. Wanneer je nu zo'n document (template) opent, krijg je de vraag of de publisher van dit VBA-project wel te vertrouwen is, zie ook afbeelding 6.
Je laat de gebruiker nu de keus of hij dit project toestaat. En aangezien ik mijzelf vertrouw, en dat altijd doe, kies ik tegelijk ervoor om *Always trust macros from this publisher* op aan te zetten. Daarmee wordt het certificaat (mijn testcertificaat) geïnstalleerd. En vanaf dat moment wordt mijn VBA-projectaranderedinggewaardeede Ccessec documenten te beveiligen (folders, encryptie, password openen/lezen, beveiligen van forms/word, als vanzelf geladen, zonder de gebruiker lastig te vallen. Op je machine wordt herkend dat het geen 'officieel' erkend certificaat is. Kijk maar eens naar de details van het certificaat (Tools, Macro, Security, Tweede tabblad (Trusted Publishers), View, Certficate derde tabblad. Zie het resultaat in afbeelding 7.
Je kan aan de Certificate Status zien dat dit certificaat geen 'Trusted Root' heeft, oftewel, de bron van dit digitale certificaat is niet te garanderen. Nu heb ik het zelf gemaakt, dus dat kan ik plaatsen. En het is voldoende voor mijn eigen organisatie. Maar als ik een certificaat wil bemachtigen dat publiekelijk beschikbaar is, dan moet ik mij tot de eerder genoemde autoriteiten wenden en aldaar een certificaat laten aanmaken. Bereid je voor op het beantwoorden van de nodige vragen, het gereed hebben van betaling, het kunnen aangeven waar jouw organisatie te autoriseren is, om aldus een officieel certificaat aan te maken.

Microsoft Outlook is speciaal. De uitvoering van security van Outlook verschilt van Word, Excel en PowerPoint. Zeker als in de loop van de tijd Outlook e-mail security-updates zijn toegepast. Dat betekent vaak dat de gebruiker toch een waarschuwing krijgt en apart toestemming moet verlenen, wanneer via VBA automation mail wordt aangemaakt, of het adresboek wordt gelezen. Dat is er natuurlijk in gekomen omdat de e-mail client een grote bron van virusverspreiding is. Er zijn geen manieren om aan deze beveiliging te ontkomen, tenzij de systeembeheerder op Exchange-niveau de beveiliging op lager dan standaard zet; hetgeen me niet aan te raden lijkt.

Microsoft Access is speciaal. Ook Access heeft de nodige uitbreiding ten opzicht van de andere Office-producten. In Access kun je met de Workgroup-administrator in combinatie met User en Group Definitions en Permissions tot in detail de toegang tot onderdelen in de Access-applicatie en database regelen. Omvangrijk met veel mogelijkheden voor een echte developer en mijns inziens een apart artikel waard.

Microsoft Word is speciaal. Omdat je sinds Word 2003 niet alleen de VBA-programmatuur kunt beveiligen, maar ook in detail de inhoud en lay-out van documenten (sjablonen) kunt beveiligen. Ook dit is een apart artikel waard.

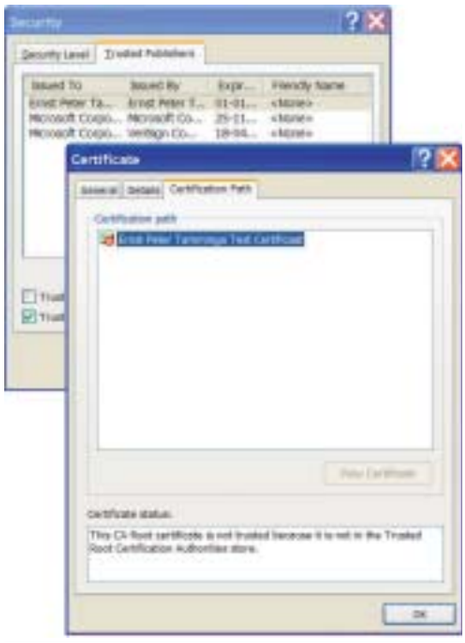
Microsoft Excel is speciaal. Omdat je binnen Excel heel selectief om kunt gaan met het delen van een workbook waarin een gebruiker mag wijzigen, dan wel gegevens invullen. Je raadt het al: ook dit is volgens mij een artikel waard.

Microsoft Office 2003 met Visual Studio Tools for Office biedt meer

In de voorgaande paragrafen zijn we nog niet aan de .NET-mogelijkheden van Office 2003 toegekomen. Visual Studio Tools for Office (VSTO), biedt de serieuze developer nu juist de mogelijk-

Afbeelding 6. Vertrouwen

net#5_p16-20 1.09 18
18-05-2004, 10:00:58



Afbeelding 7. Details certificaat

heden om .NET-projecten aan Office-applicaties (documenten) te koppelen. En als er één programmeeromgeving is waarin we security goed fijnmazig kunnen regelen, dan is dat het .NET Framework. In plaats van het gebruik van VBA in de Office-omgeving, kunnen we met VSTO doorstappen naar .NET en profiteren van al het moois dat .NET ook aan de Office-developer heeft te bieden. De VSTO-projecten voor Visual Studio .NET zorgen voor de koppeling tussen de (COM-gebaseerde) VBA/Office-omgeving en de managed code van de .NET DLL-omgeving. Het gemakkelijkste kun je een project vanuit Visual Studio .NET starten en aangeven dat je een Office-project wil aanmaken. Heb je al een bestaand Word-document of Excel-workbook waaraan je logica wilt toevoegen, dan kun je aangeven dat je .NET-project een uitbreiding is op dit document. Heb je nog geen Office-document en wil je dat je project zo'n document zelf aanmaakt, kies dan voor het aanmaken van een nieuw Office-document.
De VS.NET project-template van VSTO zorgt ervoor dat er in het document zelf een koppeling wordt gemaakt naar de .NET-assemblies die je aan het document koppelt. En wel via de custom properties: _AssemblybyName0 en _AssemblyLocation0; zie ook afbeelding 8. Open je Word- of Excel-document, kies voor File, Properties, Custom.

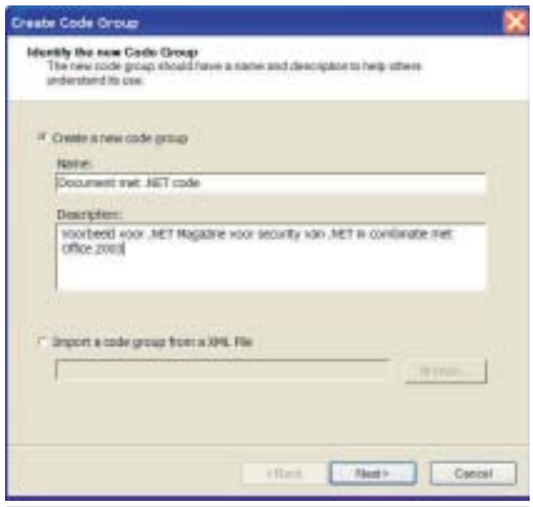
Met deze twee custom properties wordt de Assemblynaam (je .NET DLL) en de locatie van de assembly in het document (of het sjabloon) vastgelegd. Gelukkig is er de nodige vrijheid voor de _AssemblyLocation0: je kunt een instelling geven die relatief is, een volledig gekwalificeerde naam (UNC) of een http-path, waarbij het laatste natuurlijk aantrekkelijk is als je de Office-uitbreiding flexibel op een centraal punt wilt beheren. Wanneer je Visual Studio.NET gebruikt om het Office-document aan te maken, worden automatisch de benodigde .NET DLL security-settings aangemaakt. Heb je een bestaand document en wil je .NET-code er zelf aan toevoegen, of ga je de .NET-code (DLL's) verplaatsen, dan moet je de security-settings voor .NET zelf instellen. Om de .NET security-settings correct in te stellen, kun je gebruik maken van de Management Console Snap-In. Kies vanaf het startmenu voor All Programs, wijs Administrative Tools aan en click Microsoft .NET Framework 1.1 Configuration. Wat je nu opstart is een krachtig hulpmiddel om de security van .NET in te stellen. Experimenteer er op los in je ontwikkel- en testomgeving, maar wees heel nauwgezet in het instellen van rechten in een productieomgeving. Het heeft geen zin een groot hek om je gebouw te zetten, als je vergeet de poort goed op slot te doen en de sleutel weg te bergen.
Ik stel me voor dat je voor je kundig samengestelde .NET Office-extensie een nieuwe Code Group aanmaakt, die geldig is op je eigen machine. In de console-tree, klap de Runtime Security Policy node uit, en daarbinnen de Machine-node. We maken een nieuwe code group aan voor onze Office-oplossing en maken de security afhankelijk van de URL of file path van de solution die we willen laden. Kies voor Code Groups, klik deze open, klik ook All_Code open en gebruik je rechtermuis knop om een nieuwe code group aan te maken. Vul de eerste beschrijving en toelichting in, en je moet nu iets hebben dat lijkt op afbeelding 9.

Klik op Next om de locatie van de security aan te wijzen. Je krijgt een fijnmazige opgave als je kiest voor een URL, aangevuld met de filenaam-specificatie van de .NET DLL die je hebt gemaakt. Het resultaat is hetgeen je ziet in afbeelding 10.

De wildcard (*) geeft aan .NET aan dat iedere code die wordt geladen vanaf die directory of daaronder behoort tot de code groep die we aan het maken zijn.
In het volgende window moet je nog aangeven welke rechten je aan een gebruiker beschikbaar wilt stellen. In dit voorbeeld kies ik voor de 'snel thuis' manier: Full Trust. Een Next en Finish en we zijn klaar. Vanaf dit punt moet het mogelijk zijn je Word-document met de extensie die je via VSTO/.NET hebt aangemaakt betrouwbaar en veilig te kunnen gebruiken.

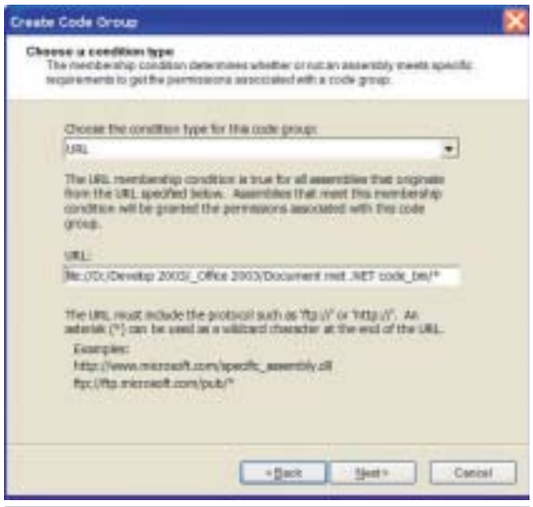
Afbeelding 8. Custom properties

Tot zover hebben we een assembly gemaakt die de URL gebruikt als 'bewijsvoering' voor de security in .NET. Om de zaak nog beter dicht te zetten, kunnen we extra stappen nemen, zoals een digitale signature, een publisher key en/of een strong name. Om een strong name voor een assembly te maken, gebruiken we een .NET-utility. Ga vanuit het startmenu, naar All Programs, Microsoft Visual Studio .NET 2003,



Afbeelding 9. Nieuwe code group

net#5_p16-20 1.09 19
18-05-2004, 10:00:59



Afbeelding 10. Locatie security

kies voor Microsoft Visual Studio .NET .NET Tools en klik op de Microsoft Visual Studio .NET 2003 command prompt. Start het programma sn om een strong name key-pair aan te maken:

```
sn -k "D:\Develop 2003\_Office 2003\Document met .NET code\Document met
.NET code.snk"
```

Start nu Visual Studio en open je Solution, kies voor AssemblyInfo.vb en configureer een versienummer en de verwijzing naar de key-pair die je zojuist hebt gemaakt:

```
<Assembly: AssemblyVersion("1.0.0.1")>
<Assembly: AssemblyKeyFile("D:\Develop 2003\_Office 2003\Document met
.NET code\Document met .NET code.snk")>
```

Bouw nu je project opnieuw met deze assembly-informatie (Build, Build Solution)
Nu moet je de URL wijzigen die je eerder hebt gebruikt om de security in .NET te regelen. Voer hierbij de volgende stappen uit:

1. Open .NET Framework Configuration Window en ga terug naar je vorige setting
2. Kies **Machine, Code Groups, All_Code**, rechter muisklik op je vorige group, **Properties**
3. Kies het tweede tabblad (**Membership Condition**), en wijzig URL in Strong Name
4. Klik op de **Import**-button, browse naar de DLL die bij je project hoort - en die je zojuist van assembly-info hebt voorzien - en klik op **Open**.
5. Selecteer **Name** en **Version** en klik op **Next**
6. Accepteer **Full Trust** door op **Next** te klikken en voltooi je actie met **Finish**
7. Start opnieuw de Visual Studio .NET command prompt en roep het volgende command line programma aan op de PublicKeyToken voor je assembly te bepalen:
   ```
   sn -Tp "D:\Develop 2003\_Office 2003\ Document met .NET code_bin\
   Document met .NET code.dll"
   ```
8. Noteer het PublicKeyToken dat wordt getoond in het command window
9. Start Word en open je document dat een verwijzing heeft naar je .NET code
10. Vanuit het **File menu**, kies voor **Properties** en het tabblad **Custom** daarbinnen
11. Wijzig de **_AssemblyName0** property, gebruikmakend van de PublicKeyToken die je eerder hebt gevonden:
    ```
    Document met .NET Code,Version=1.0.0.1,PublicKeyToken=fdg6593e1d16861
    ```
12. Sluit het properties-window, bewaar je document, sluit Word
13. Probeer of het geheel werkt door het document opnieuw te openen.

Dertien stappen die leiden tot succes. Secure werken en afleveren vergt de nodige kennis en handelingen, maar is essentieel als je langere tijd garant wilt staan voor de betrouwbaarheid van je Office-toepassingen in de 'boze buitenwereld'. En Office 2003 al dan niet in combinatie met .NET bieden je hiervoor de nodige handvatten.

Samenvatting

De onderdelen van de Office-omgeving (Word, Excel, Powerpoint, Outlook) zijn de meest gebruikte Windows-toepassingen, niet in de laatste plaats vaak in handen van minder ervaren gebruikers. Tegelijk wordt met Office een ongelofelijk goede faciliteit geboden voor aanpassing van de standaard manier van werken: de Office VBA-programmeeromgeving. Deze combinatie maakt deze toepassingen een gewild doelwit voor personen met minder goede bedoelingen, zoals virusmakers.
Gelukkig biedt de Office 2003 ontwikkelomgeving, al dan niet in combinatie met het .NET Framework, gereedschappen om Office-toepassingen te maken die aan alle toetstenen van veilige applicaties kunnen voldoen. Dit vereist nauwkeurigheid en deskundigheid van een serieus werkende developer. In dit artikel heb ik beschreven hoe je van minimale beveiliging (instellen van een macro security level in de Office-applicatie), via beveiliging van je code (met macro en digitale signatures) komt tot fijnmazige security-instellingen door het gebruik van het .NET Framework. Voor de laatste optie heb je Office 2003 en de Visual Studio Tools for Office (VSTO) nodig, maar dan beschik je als office developer over alle mogelijkheden die je als 'gewoon' .NET-developer ook hebt.

Natuurlijk is dit artikel te kort om op details van sommige Office-applicaties in te gaan. De uitgebreidste vorm van beveiliging vind je terug bij Access. Al sinds Access 2.0 kent dit product de mogelijkheid formulieren, rapporten, tabellen, queries, macro's en dergelijke te beveiligen. Niet alleen tegen het onoordeelkundig wijzigen, ook voor het in detail regelen wie welke onderdelen mag opstarten, welke tabellen mag zien, welke velden uit tabellen, welke records uit tabellen, enzovoort. Ook Outlook kent zijn eigen uitbreidingen, of beter gezegd zijn eigen beperkingen. Outlook wordt vaak gezien (en gebruikt) als motor achter het verspreiden van virussen. Dat kun je voorkomen, zowel door het bijhouden van de updates en patches die verschijnen, als door zelf bewust te zijn van de deuren die je dicht kunt zetten.
Iedere nieuwe versie van Office biedt de serieuze developer meer mogelijkheden om betrouwbare Office-uitbreidingen te maken. Office 2003 steekt alle voorgaande versies naar de kroon door de integratie met .NET-code en framework. Maak daar van gebruik. Secure je office en ga veilig op weg naar huis.

Meer informatie

Kijk voor algemene achtergrondinformatie op ontwikkelen voor integratie met de Office-omgeving op www.xcess.nl

De MSDN-site is de basis voor iedere developer in de Microsoft-omgeving. Informatie over Visual Studio Tools for Office vind je op http://msdn.microsoft.com/library/en-us/odc_vsto2003_ta/html/VSTOIntro.asp

Een stap voor stap gids voor het aanbrengen en gebruiken van een digitale signature is te vinden op http://msdn.microsoft.com/library/en-us/dnsmarttag/html/odc_dcss.asp als onderdeel van het realiseren van SmartTags (overigens ook sterk verbeterd in Office 2003).

Het Microsoft Office Development Center http://msdn.microsoft.com/office/

Ernst Peter Tamminga, is managing consultant bij XCESS expertise center b.v. (www.xcess.nl), een organisatie die zich gespecialiseerd heeft in het adviseren en realiseren van integratieoplossingen in een Microsoft-omgeving.
Hij is te bereiken via ept@xcess.nl

net#5_p16-20 1.09 20 18-05-2004, 10:01:01