Security

Microsoft en veiligheid lijkt een oxymoron. Toch is er het een en ander veranderd sinds Bill Gates het *trustworthy computing initiative* aankondigde. Bovendien ligt de verantwoordelijkheid voor veiligheid vooral bij de ontwikkelaars. Een interview met Richard Lamb, Security Evangelist bij Microsoft.

Van toeters en bellen naar veiligheid

Interview Steven Lamb

Dré de Man
Auteur en foto's

Steven Lamb begint aan het begin van het interview meteen al te sputteren. "Security niet sexy? Hoe kunt u dat zeggen? Ik zou sowieso voorzichtig zijn met het combineren van dat woord met iets dat niet levend is. Maar afgezien daarvan: security is heel spannend." Het is duidelijk, Lamb houdt van zijn vak. Het sputteren was het antwoord op de vraag of hij niet een moeilijke baan had. De combinatie van het begrip security met Microsoft lijkt niet zo gelukkig, en die van security met ontwikkelaars al helemaal niet. Over de suggestie dat Microsoft en security niet zo goed samengingen bleef Lamb heel rustig. Dat had hij kennelijk al vaker gehoord.
Lamb: "Ik ben vijfeneenhalf jaar geleden bij Microsoft begonnen, en het *trustworthy computing*-initiatief werd kort daarop gestart. Dat was het begin van de cultuuromslag bij de ontwikkelaars van Microsoft. Security werd interessant voor ontwikkelaars in het gehele bedrijf als gevolg van het feit dat hun omgeving veranderde. Daarvóór hielden alleen experts zich bezig met cryptografie, modules en security-functies. Mensen die algemene softwareontwikkeling deden, waren niet noodzakelijkerwijs getraind in security. De prioriteit verschoof ineens van features, functies en knipperende lichtjes naar veiligheid. Het was bovendien ook nodig om bijvoorbeeld de opleidingen te veranderen. In feite moest de hele omgeving veranderen. Een deel van de beoordeling en het salaris van mensen is nu gebaseerd op hun opstelling ten opzichte van security. Dus, hebben ze trainingen gevolgd, hebben ze hun threat-models gevolgd, zijn ze proactief geweest in het oplossen van kwetsbaarheden. Er is een hele serie van matrices rond security die in belangrijke mate bepaalt waarom mensen aangenomen of ontslagen worden, promotie maken et cetera. Dat is een fundamentele verandering. Sommige teams hebben er langer over gedaan dan andere om security een hogere prioriteit te geven dan functionaliteit of performance. Maar het kan economisch zeer interessant zijn. Ik heb informatica gestudeerd en toen geleerd dat de kosten van het oplossen van een bug in het conceptiestadium miniem zijn vergeleken met de kosten die bij het oplossen bij uitlevering ontstaan. Voor security geldt precies hetzelfde, zowel voor wat betreft het risico als voor de (herstel-)kosten.

Lamb gaat zelfs zover dat hij Microsoft ziet als een voorloper op het gebied van veiligheid.
Lamb: "We leiden de industrie als je kijkt naar de security development-levenscyclus en het gebruik van threat-models. Threat-models behandelen vragen als 'wat zijn de bedreigingen', en als ze gecombineerd worden, 'wat is dan het effect?' Als je vijf jaar geleden een kwetsbaarheid

vond in een Microsoft-product, dan wist je niet wat je moest doen. Wie zou je moeten bellen of mailen, wat was de procedure, hoe wist je dat er überhaupt iets zou gebeuren? Veiligheidsexperts en hackers neigden ertoe om op onverantwoorde wijze in de publiciteit te brengen dat Microsoft hun bevindingen niet serieus nam, en er niets aan deed, afgezien van pogingen de problemen te verbergen. Als antwoord op dat probleem is het Microsoft Security Respons Centre opgezet. Als je nu iets vindt waarvan je denkt dat het een kwetsbaarheid is in een Microsoft-product, dan kun je een e-mail sturen naar secure@microsoft.com en dan zal iemand de eigenaar worden van dat probleem. Als het een belangrijk probleem mocht blijken te zijn, dan zul jij ook de eer krijgen voor het feit dat je de eerste was die het probleem ontdekt heeft. Security maakt dus deel uit van alles wat zowel de ontwikkelaars als de rest van het bedrijf doen. Het heeft even geduurd voordat het in een stroomversnelling raakte, maar het is een fundamentele verandering. Wanneer de vent aan de top zegt dat security nummer één is - en dat is wat Bill heeft gezegd in het trustworthy computing-memo in januari 2003 - dan zet dat een grote verandering in. Stel je voor dat je een ontwikkelaar bent en al die tijd bezig bent geweest die geweldige functie te schrijven, en die wordt standaard uitgezet? Het heeft een hoop werk gekost om dat soort verwachtingen te veranderen. Als je nu een echte *coole* functie schrijft, dan is er een barrière op het punt waar hij geadopteerd moet worden. Je moet samenwerken met andere teams, marketingteams, technische teams, het is een heel andere omgeving. We hebben het effect daarvan ook in de kwetsbaarheidstatistieken gezien.

Lamb vindt het ook nodig dat managers meer belang gaan hechten aan veiligheid.
"De meeste ontwikkelaars hebben een redelijke hoeveelheid kennis van security. Ze weten misschien niet zoveel als ze zouden moeten weten, maar het valt echt mee. Ze vechten echter om hun managers zover te krijgen om tijd te besteden aan threat-models, aan dingen waarvan ze geloven dat ze juist zijn. Als security op jouw acceptatiecriterialijst staat en assessment van security deel uitmaakt van een acceptatietest, dan heb je een heel andere situatie. Als er een veiligheidstekortkoming in een Microsoft-product gevonden wordt, dan zoeken we uit wie die code geschreven heeft, onder iedereen die code ingecheckt heeft. Niet omdat we op heksenjacht willen gaan, maar omdat we willen weten waarom die kwetsbaarheid ontstaan is. Was het een probleem van het proces, een trainingprobleem, was het oude code waar een probleem mee was, hoe zat het met het reviewen van de code, zowel automatisch als handmatig? Ook bij kwetsbaarheden in concurrerende producten, open source of commercieel, kijken we gewoon naar de vraag van het ontstaan van het probleem, zodat we ervan leren en niet een vergelijkbare fout maken. We werken ook samen met universiteitsdocenten om ze te leren security te onderwijzen als deel van het leren programmeren. Het vergroten van het technische bewustzijn voor security is erg belangrijk, maar het erbij betrekken van klanten en management is minstens zo belangrijk."

Lamb heeft zelf als ontwikkelaar gewerkt én heeft ervaring opgedaan met het schrijven van onveilige code.
"Toen ik zelf nog ontwikkelaar was – ik schreef Fortran – hadden we codereviews in het team. Voordat je code kon inchecken, werd het gereviewed. Het is waarschijnlijk iets Engels, maar degene die de meeste problemen had gehad met zijn code, moest aan het einde van de week het bier betalen bij de vrijdaglunch. Dat was echt van grote invloed, want je wilde niet degene zijn bij wie de meeste fouten gevonden werden. Uiteindelijk gingen mensen te ver in het zoeken naar kwetsbaarheden in de code van de anderen. Toch zit er wat in het idee van het belonen van het zoeken naar kwetsbaarheden. Bij Microsoft hebben we het team van Secure Windows Initative. De rol van dat team is het om kwetsbaarheden

CardSpace

CardSpace maakt deel uit van het .NET 3.0 framework. Een belangrijk onderdeel van CardSpace is de Identity Selector, waarmee iemand op een InfoCard kan klikken om zich aan te melden bij een server of website. Een InfoCard is in feite een visuele weergave van een digitale identiteit, zoals een eBay-logo.
Bij het openen van een relevante website roept de browser de identiteitsselector op. De gebruiker kiest een InfoCard (een digitale identiteit) en kan dan gebruik maken van de server of website.
De gebruiker kan overigens zelf InfoCards genereren, of gebruikmaken van een digitale identiteit die hij van een externe Identity Provider krijgt.
De verzameling digitale identiteiten wordt op een beveiligde locatie in het besturingssysteem bewaard. CardSpace voldoet aan de in WSDL gedefinieerde protocollen en aan standaardprotocollen zoals SAML. Daardoor kan ook gecommuniceerd worden met niet-Microsoft producten als Firefox. CardSpace werkt ook samen met OpenID.

in onze code te vinden. Al onze testers krijgen dezelfde training. Iedereen die test, codeert, specificaties schrijft, of zich met architectuur bezig houdt, krijgt training. We hebben ook onze Blue Hat-sessies, waarbij we security-researchers, hackers en wie ook maar wil, uitnodigen om onze code te kraken voor de ogen van degenen die het geschreven hebben. Dat helpt de ontwikkelaars te begrijpen hoe security-onderzoekers en hackers code kraken, en het verhoogt hun bewustzijn over waarom ze de dingen zo moeten doen. Bovendien helpt het hen innovatief te worden in het tegengaan van bedreigingen. We doen dat nu al jaren."

Heeft het gebruik van managed code bij Microsoft de security verhoogd?
"Een soortgelijke vraag is of virtualisatie de veiligheid verhoogt. Mijn initiële antwoord is 'ja en nee'. Ja, het geeft isolatie, separatie en controle, maar nee, omdat het iets nieuws is. Voor managed code geldt hetzelfde. Er is meer controle met managed code, je kunt je grenzen aangeven voor security, aangeven wat toegestaan is et cetera, daar kun je heel specifiek in zijn. Als je om het even welke technologie niet begrijpt, of minder goed dan de vorige technologie, dan duurt het voordat je er echt handig in bent, en in die tijd zul je mogelijk fouten maken. Maar impliciet zou het veiliger moeten zijn."

Het on-line kopen van producten en doorgeven van persoonlijke informatie op websites wordt steeds meer gemeengoed. Daarnaast zien we de komst van webservices en SOA, met single sign on over verschillende domeinen. Al die ontwikkelingen maken het zorgen voor veiligheid er mijns inziens niet gemakkelijker op en de privacy zal er ook niet op vooruitgaan.
Lamb: "Daar staat tegenover dat we ook nieuwe technologieën hebben, die oplossingen voor deze problemen bieden. Wanneer ik inlog bij Microsoft, dan is dat heel eenvoudig. Ik log in op een domein, of ik nu thuis ben, of op kantoor of draadloos, en vanaf dat punt werkt het gewoon. Afhankelijk van de vraag of ik lid ben van de juiste groepen krijg ik toegang of niet. Als ik geen toegang krijg, kan ik elektronisch toegang aan de eigenaar vragen. Als ik buiten die ideale managed omgeving ga, dan kijk je naar de CardSpace-type technologieën die het voor mij als eindgebruiker mogelijk maakt om eenvoudig te authenticeren en dan de *tokens* door te geven zodat ik toegang krijg. Het idee achter CardSpace is dat je de gebruiker de controle geeft over de informatie. De card heeft een digitale identiteit en daarin zit de leeftijd, adres, creditcarddetails, et cetera. Ik als eindgebruiker bepaal wat wanneer waarom waarnaartoe gestuurd wordt. Als ik een boek koop van een webwinkel, waarom moeten ze dan weten wat mijn creditcardnummer is? Ze hoeven niet veel meer te weten dan een afleveradres, het boek dat ik wil kopen, en ze moeten erop kunnen vertrouwen dat ze een betaling zullen ontvangen. Waarom moet ik in 's hemelsnaam verschillende accounts hebben bij meerdere leveranciers, waarom moet ik persoonlijke informatie delen met hen? Card space heeft een grafische front-end, de Microsoft-implementatie ervan heet Infocard. Bij Infocard klik ik op de grafische weergave van mijn identiteit, laten we zeggen mijn eBay-identiteit, en laat ik me identificeren door hun systeem. Als gebruiker hoef ik er verder niets van te weten. Als ontwikkelaar aan de webserver-kant moet ik iets begrijpen van CardSpace. Je moet begrijpen hoe er mee te interacteren, maar dan gaat het verder ook automatisch. We hebben mensen op Sun-platforms en open source, geheel andere omgevingen, die de CardSpace-standaard ook hebben geadopteerd, ook veel on-line winkels. Het is een interessante technologie, omdat je geen accounts bij die sites meer nodig hebt, met dingen als de verplichting om mijn adres bij te houden et cetera. Belangrijk nog is dat de andere kant alleen die informatie van mij krijgt die nodig is, en niets meer.

Zorg er voor dat je input valideert, dat is zo oud als de weg naar Rome

Lamp heeft een hele reeks adviezen voor ontwikkelaars.
Lamb: "Allereerst, schrijf je code zo, dat er zo weinig mogelijk rechten nodig zijn. Het voordeel voor jou als ontwikkelaar is dat het minder waarschijnlijk is dat je de rest van het systeem zult destabiliseren, en dat het minder waarschijnlijk wordt dat je daarvoor de schuld krijgt. Je hoeft minder security-analyse te doen. Wees heel strikt in wat je doet, gebruik minder privileges, gebruik threat-models, dus die grafische middelen om *trust*-grenzen voor informatiestromen te definiëren. Dat zal je een hoop tijd besparen en het een stuk gemakkelijker maken om te zien waar je tijd moet investeren in security. Zorg er voor dat je input valideert, dat is zo oud als de weg naar Rome. Code-injectie-achtige attacks bestaan al vijftien tot twintig jaar, en toch worden

Ten immutable laws of security

Tien onveranderlijke veiligheidswetten

1: If a bad guy can persuade you to run his program on your computer, it's not your computer anymore
2: If a bad guy can alter the operating system on your computer, it's not your computer anymore
3: If a bad guy has unrestricted physical access to your computer, it's not your computer anymore
4: If you allow a bad guy to upload programs to your website, it's not your website any more
5: Weak passwords trump strong security
6: A computer is only as secure as the administrator is trustworthy
7: Encrypted data is only as secure as the decryption key
8: An out of date virus scanner is only marginally better than no virus scanner at all
9: Absolute anonymity isn't practical, in real life or on the Web
10: Technology is not a panacea

er nu nog steeds applicaties geschreven die daar kwetsbaar voor zijn. Een goede mantra is: beschouw alle input als slecht, totdat het tegendeel bewezen is. Ik zou ontwikkelaars op het hart willen drukken naar *the ten immutable laws of security* te kijken; zie kader, red.. Dat zijn tien onaanvechtbare stellingen, met zaken als: wanneer ik geen fysieke controle over mijn computer heb, dan is het mijn computer niet meer. Als ik software installeer van een derde partij van wie ik niet weet of ik die kan vertrouwen, dan is het mijn machine niet meer. Als ik niet-gevalideerde input accepteer, ook op lager niveau, dan kan ik het ergste verwachten. Je kunt dus jouw kwetsbaarheid verminderen door deze technieken te gebruiken.

Zorg er ook voor dat e managers begrijpen waarom security belangrijk is. We spraken er eerder over dat het zo vroeg mogelijk fixen van bugs goed is, dat je dan de kosten om problemen op te lossen kunt reduceren. Jouw code is beter opnieuw te gebruiken, hij is in het algemeen beter ontworpen, want beter begrepen en gedocumenteerd. Het heeft dus vele voordelen voor de ontwikkelaar. Security zou niet iets moeten zijn dat in de weg staat. Security zou datgene moeten zijn dat je competitieve voordelen geeft. Als er twee vendors zijn en de een blijkt security serieus te nemen en de ander niet, en verder zijn ze gelijk, dan zullen mensen na verloop van tijd wel eens voor die ene vendor kunnen gaan kiezen. Ik zou geen onbeheerde machine gebruiken om mijn creditcarddetails in te tikken. Sommige mensen gebruiken alles met een toetsenbord om hun password in te tikken.

Hobby

"Een collega heeft als hobby computers in hotellobby's te onderzoeken. Alle computers die hij tegengekomen is, zonder enige uitzondering, waren besmet met malware. Hetzelfde geldt voor code. Ik zou geen code installeren van een bedrijf waarvan ik niet weet of ze security serieus nemen, tenzij het op een virtual machine staat of op een andere machine die ik niet gebruik om gevoelige informatie door te geven. In Vista hebben we voor IE een protected mode. Die gebruikt verplicht integriteitcontrole. Daarmee krijgen data en executables een niveau van integrity. Iets met een hoger integriteitniveau mag niet in contact komen met iets met een lager integriteitniveau, zelfs wanneer de toegangscontroles het toestaan. Dat betekent dat er nauwelijks iets kan gebeuren wanneer je malware tegenkomt, op een malafide website of zo, zelfs als jouw antivirus en anti-al-het-andere het probleem niet oplossen. De malware kan in het ergste geval data schrijven naar de tijdelijke internetdocumentenmap, waar het alleen in contact kan komen met dingen met het allerlaagste niveau van integriteit. Protected mode gaat uit van het idee dat kwetsbaarheden bestaan. Het bepaalt hoe jouw systeem zich gedraagt op het moment dát er een kwetsbaarheid is. Dat is een heel andere mentaliteit dan daarvoor, waarbij we niet verder gingen dan pogingen om kwetsbaarheden te vermijden. Het klinkt misschien merkwaardig van iemand van Microsoft, als je denkt aan het oxymoron van het begin van ons gesprek, maar security is iets dat we heel serieus nemen. «