Producten cyberveilig ontwikkelen

Houd hackers buiten – cyberveilige hard- en software

Het cyberveilig maken van een apparaat is geen actie die aan het eind van het ontwikkeltraject gedaan moet worden. ‘Nog even de cybersecurity programmeren’ als de rest klaar is, is te laat. Software en hardware gaan hand in hand in embedded apparatuur. Vanaf dag één moet aan cybersecurity gedacht worden, in alle fases van de productontwikkeling, de levering en de nazorg.

Afbeelding 1. Diverse bedrijven geven trainingen op het gebied van embedded software security, waaronder startup VDOO uit Israël.

Aandacht voor cybersecurity is tijdens de productontwikkeling vaak minimaal of zelfs geheel afwezig. De argumentatie is dan dat het functioneel niets toevoegt en de gebruiker moet de beveiliging van zijn netwerk zelf maar regelen. Dit beleid wordt echter steeds minder geaccepteerd. Overheden komen steeds meer met regelgeving en eisen, en wie het te bont maakt, vraagt om juridische problemen. Zo werd begin juli bekend dat in de Verenigde Staten de FTC (Federal Trade Commission) en het Taiwanese D-Link een schikking getroffen hebben in een rechtszaak over de falende beveiliging in D-Link-producten. Nu moet D-Link gedurende de komende 10 jaar producten cyberveilig gaan ontwikkelen, beter testen, samenwerken met beveiligingsonderzoekers en elke twee jaar een externe partij een assessment laten uitvoeren die toegang moet krijgen tot alle relevante documentatie.
Wie zich dan ook afvraagt of cybersecurity duur is, moet dan maar eens kijken naar de kosten van een rechtszaak. De vraag blijft echter, waar moet zoal allemaal rekening mee gehouden worden?

Kennis

Beveiligingslekken ontstaan vaak door gewone programmeerfouten of door een gebrek aan inzicht in gebruikte technologieën, door fouten in de architectuur of door ontwerpfouten. Fouten kunnen door iedereen gemaakt worden, bij kleine en bij grote bedrijven en bij nieuwe en ook bij gerenommeerde bedrijven.
Uiteraard is niet elke fout ook een beveiligingslek. Een beveiligingslek hoeft niet perse te leiden tot functionele problemen. De software kan 100% correct functioneren. Softwaretesters en reviewers richten zich vaak op de functionele aspecten en kijken niet met een hackersbril naar het product. Een eerste stap is dus om bewustwording op dit gebied te kweken, gevolgd door een training in veilig programmeren (en dit geldt niet alleen voor C/C++, maar ook voor andere programmeertalen). Daarnaast moet er aangeleerd worden hoe hackers naar embedded apparatuur kijken en hoe ze ‘binnen’ komen. Dit soort trainingen kan o.a. gevolgd worden in Nederland bij Riscure en SIG (Software Improvement Group - afbeelding 1).

Ontwikkelproces

Al in de eerste fase van een project moet cybersecurity meegenomen worden in het hardware- en software-deel. Mogelijk zijn speciale componenten nodig (zoals een TPM-chip, zie verderop). Veelal worden ook speciale debugging-circuits toegevoegd die alleen voor prototypes nodig zijn en op productieversies worden weggelaten (zoals RS232, JTAG). Hackers kunnen die interfaces echter later ook gebruiken (op een gekocht exemplaar). Het is daarom verstandig om ervoor te zorgen dat ze dan niet (meer) werken.

Veilig programmeren (secure programming) houdt in dat de programmeur bekend is met de zwaktes van een programmeertaal en zodanig codeert dat hackers er geen misbruik van kunnen maken. Bijvoorbeeld, in C/C++ is dit de bekende 'buffer overflow' waarbij niet genoeg geheugenruimte wordt toegewezen om bepaalde data in op te slaan, waardoor andere gegevens overschreven worden. Nog een voorbeeld is SQL Injection in databases, waarbij de database van buitenaf gemodificeerd kan worden. Op de website cwe.mitre.org en dan de sectie Top 25 Most Dangerous Software Errors is een interessant overzicht te vinden (afbeelding 2).
Reviewers en testers moeten zich ook bewust zijn van deze basisfouten. Dit kan door een handmatige review, maar er zijn genoeg software-tools die veel checks automatisch kunnen doen. Zogenaamde fuzzers kunnen automatisch random invoergegevens aan software aanbieden, puur om te zien of de invoervalidatie-algoritmes alleen zinnige invoer doorlaten (en niet crashen). Bijvoorbeeld: als een getal ingevoerd moet worden, wat is dan het geldige bereik? Zo kon ik laatst zonder problemen bij een concertorganisator een paar biljoen kaartjes voor één concert bestellen (en betalen per Ideal). Ook hiervoor is veel open-source gereedschap, zoals CPPCheck (voor C/C++), Valgrind (geheugenbeheer), Bandit (controle op Python code), AppArmor en SELinux (Linux security), iptables (firewall), maar er zijn ook commerciële tools (zoals Afl), om er maar een paar te noemen.
Software die gebruik maakt van open-source componenten (wie niet tegenwoordig) moet ook gecontroleerd worden. Wordt van elke component de laatste versie gebruikt? Worden er geen overbodige componenten gebruikt en is elke component op de juiste manier geconfigureerd? Veel software komt met standaard-instellingen die iedereen alles toestaan. Dit is immers gemakkelijk om snel aan de slag te gaan in een ontwikkelorganisatie, maar het is niet gewenst om de software ook zo uit te leveren. Deze moet zoveel mogelijk dichtgetimmerd (hardened) worden. Een voorbeeld: zorg dat verouderde encryptie-algoritmes niet meer gebruikt mogen worden zoals in SSL versie 1 en dat standaardwachtwoorden gewijzigd zijn.
Uiteraard kan alle broncode met de hand gecontroleerd worden, maar dat is op lange termijn ondoenlijk en foutgevoelig. Er bestaan geautomatiseerde omgevingen die dit automatisch kunnen, zoals van Semmle, IOTInspector en VDOO. Deze kunnen broncode analyseren en zelfs binaire code.

Broncode beveiliging

De broncode van producten is het DNA van een bedrijf. Tegenwoordig houden we dit bij in speciale archieven, die natuurlijk zelf ook beveiligd dienen te worden tegen ongewenst leegtrekken dan wel tegen ongewenste wijzigingen.
De afgelopen jaren is het al een aantal keer voorgekomen dat archieven van open-source componenten ongewenst aangepast worden. Vanuit een hacker is dit ideaal, omdat de aanpassingen dan vanzelf in heel veel andere producten terechtkomen. Ook in het eigen softwarearchief kunnen aanpassingen gebeuren die er niet zouden moeten zijn.
Men kan dus niet volstaan met het eenmalig installeren van een archiveringssysteem. Ook hier moet men bijblijven met updates. En vergeet niet met grote regelmaat een back-up te maken. U zult de eerste niet zijn die getroffen wordt door ransomware (of door een hacker die eenvoudigweg alles wist).

Wettelijke eisen

IOT apparaten moeten als elektronische apparaten voldoen aan een hele batterij wettelijke regels, bijvoorbeeld op het gebied van EMC. Voor software is er echter zoiets niet. De toenemende maatschappelijke schade door gehackte apparaten maakt dat overheden steeds meer gaan nadenken over de te stellen eisen voor IOT-apparaten. Zo wil de Nederlandse regering dat in 2020 (!) in de gehele EU een set minimale veiligheidseisen gaan gelden voor IOT-apparaten. Als eerste wordt regulering voor veilige softwareontwikkeling gemaakt, die nog dit jaar klaar moet zijn. Via het eigen inkoopbeleid wil de overheid de digitale veiligheid naar een hoger niveau brengen. Later dit jaar komt er een campagne over IoT-veiligheid.
Voor industriële toepassingen zijn er weer andere standaarden, zoals in de USA de NIST en wereldwijd de IEC-62443. Voor deze laatste standaard is het ook mogelijk om producten te certificeren via ISASecure (www.isasecure.org). Erg hard loopt het met deze certificeringen nog niet. Blijkbaar wordt het nog niet erg belangrijk gevonden door leveranciers (en/of geëist door klanten). Let er op dat een certificering ongeldig wordt nadat de software gewijzigd is of nadat een bepaalde periode verstreken is.

Licenties

Sommige producten worden geleverd met software waarvoor een licentie moet worden gekocht, bijvoorbeeld om een gebruiksduur te reguleren, bepaalde functionaliteit toe te staan of het apparaat gedurende een bepaalde tijd in een demo-mode te laten opereren. Als het licentiemechanisme gehackt raakt, bent u direct financieel benadeeld. Tegenwoordig verwachten gebruikers dat licenties via internet gekocht kunnen worden. Er moet dus ook een infrastructuur hiervoor worden opgezet (webportal, server, financiële afhandeling, etc.) die natuurlijk ook weer goed beveiligd moet worden.
Licenties zijn vaak apparaat-gekoppeld. Deze moeten dus een unieke eigenschap hebben waarmee ze herkend kunnen worden. Eenvoudige implementaties maken (bijvoorbeeld) gebruik van een Ethernet MAC-adres of een Compact-Flash ID. Die zijn op zich uniek (af-fabriek), maar kunnen met weinig moeite gewijzigd worden (afbeelding 3). Dezelfde licentie kan dus op heel veel apparaten geïnstalleerd worden. Een uniek ID is óók nodig om te voorkomen dat de hardware gekopieerd wordt. Afgezien van de financiële schade (minder omzet) is er vanwege de lagere kwaliteit van de kopieën, vaak ook nog reputatieschade als (onwetende) kopers bij u aankloppen voor verhaal.
Beter is het gebruik van een speciale TPM-chip (Trusted Platform Module). Deze worden in de fabriek voorzien van een unieke ID die niet meer te wijzigen is. Tevens kan een TPM een random-number generator hebben die échte random getallen genereert. Software kan dit vaak niet en dit maakt dat crytografische beveiligingsmaatregelen vaak gemakkelijker te hacken zijn. Een TPM kan ook nog de cryptografische sleutels intern opslaan. Die hoeven dan niet in Ram, flash of EEPROM te worden bewaard, wat maakt dat ze onbereikbaar zijn voor hackers.

Na levering

Is een product eenmaal op de markt, dan is (qua cybersecurity) het werk nog niet af. Software-componenten die 100% veilig waren, kunnen dat op een gegeven moment niet meer zijn omdat onderzoekers (of hackers) er een lek in ontdekt hebben. Met de grote hoeveelheid open-source componenten die vaak gebruikt wordt, is dit zeker geen theoretisch scenario. Controleer daarom regelmatig op de beschikbaarheid van updates.

Update mechanisme

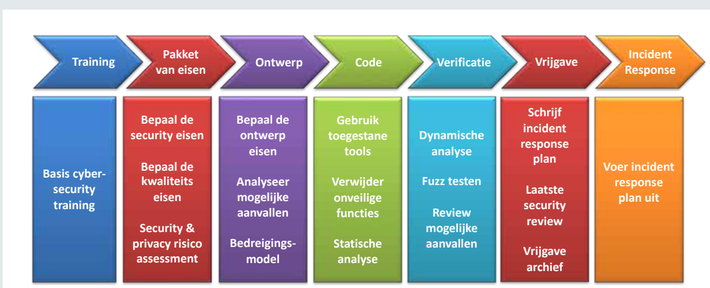
Het installeren van een update lijkt simpel, maar zonder goede beveiliging kan een hacker ook hier op inbreken en er voor zorgen dat andere (ongewenste) software wordt geïnstalleerd, of zelfs dat de bestaande software gewist wordt zodat het apparaat onbruikbaar wordt (bricked). Afhankelijk van de functie van het apparaat, dat nu dus niet meer werkt, kunnen de gevolgen ernstiger zijn, bijvoorbeeld dat het onmogelijk is om een productielijn nog op te starten.
De softwaremodule die de updates afhandelt, moet dus controleren of de nieuwe software van de juiste leverancier komt (met andere woorden de authenticiteit vaststellen). Tegenwoordig worden hiervoor digitale handtekeningen (certificaten) gebruikt en die moeten op het apparaat zelf worden opgeslagen. De nieuwe software krijgt ook een certificaat, dat getekend is met de (geheime) sleutel van de leverancier. Op de thuisbasis moet het systeem dat dit regelt dus ook goed beveiligd worden. Het lekken van de geheime sleutel is immers desastreus, omdat op elk al verkocht apparaat willekeurige software geïnstalleerd kan worden. Hackers zijn graag uit op het stelen van beveiligingssleutels, zoals in het verleden al meerdere malen voorgekomen is.

Aan de slag!

Wie zijn producten cyberveilig wil leveren, moet dus heel wat doen. Niet alleen de software van de apparatuur moet cyberveilig zijn, maar ook de hardware. Tijdens het ontwikkelproces moet er rekening gehouden worden met het veilig programmeren en met broncode-beveiliging waarna er gewerkt moet worden aan beveiligde productieprocessen. Daarna is het van belang dat er beveiligde updates geleverd worden via een veilige website voor de updates, het bijblijven met updates van gebruikte open-source componenten en niet te vergeten het informeren van klanten. Cybersecurity is op zich dan ook niet moeilijk, maar het is wel heel veel werk. •

Voor meer informatie www.etotaal.nl/achtergrond, artikel ‘Producten cyberveilig ontwikkelen’

Auteur: R.A. Hulsebos



Afbeelding 2. De SDL (Secure Development Lifecycle) zoals Microsoft die predikt is ook van toepassing voor hardware ontwikkelaars.

Afbeelding 3. Wie een Ethernet MAC-adres gebruikt als uniek ID voor het koppelen van licenties, maakt het hacken van licenties wel heel erg gemakkelijk.