Minix 3

Veilig en betrouwbaar besturingssysteem

Minix 3 is een nieuw open–source besturingssysteem voor de pc. Het systeem is klein van opzet en heeft een nieuwe, modulaire opbouw waardoor het niet kwetsbaar is voor veel van de problemen waar Windows en ook Linux mee te kampen hebben. De auteurs beschrijven het ontwerp van Minix 3 en laten zien hoe het bijdraagt aan de veiligheid en betrouwbaarheid van besturingssystemen.

Jorrit Herder, Herbert Bos, Ben Gras, Philip Homburg en Andrew Tanenbaum

Het besturingssysteem van een computer vormt het fundament onder alle activiteiten van de gebruiker. Het schermt de gebruiker af van de hardware en biedt een vriendelijke omgeving waarin applicaties kunnen draaien. Hierbij zijn betrouwbaarheid en veiligheid van cruciaal belang, maar helaas schieten huidige besturingssystemen veelal tekort. Zo worden beveiligingsproblemen in Windows bijvoorbeeld misbruikt door virussen en wormen en zijn gebruikers inmiddels gewend aan fatale fouten waardoor de computer opnieuw moet worden opgestart. Veel van deze problemen zijn terug te leiden tot het ontwerp van het besturingssysteem.
De huidige generatie besturingssystemen wordt namelijk gekenmerkt door een *monolithisch ontwerp*. Daarbij draait het hele besturingssysteem als één groot programma, ook wel de *kernel* genoemd, zonder bescherming tussen de verschillende modules en met alle mogelijke toegangsrechten tot het systeem. Door de benodigde interactie met de hardware zitten *device drivers* (stuurprogramma's) ook in de kernel van deze besturingssystemen. Dit ontwerp leidt ertoe dat een enkele fout in een willekeurige device driver, bijvoorbeeld de printerdriver, het hele systeem kan laten crashen. Hoewel veel mensen Linux en Mac OS zien als concurrenten van het alom bekende Windows, zijn deze besturingssystemen op dezelfde leest geschoeid en hebben ze daardoor veelal dezelfde tekortkomingen als Windows.
Minix 3 daarentegen heeft een nieuwe, modulaire opbouw, waardoor dergelijke problemen kunnen worden voorkomen. Daarnaast kunnen veel fouten zelfs automatisch worden verholpen zonder dat de gebruiker hier iets voor hoeft te doen. Dit draagt bij aan de veiligheid en betrouwbaarheid van het besturingssysteem.

Historisch perspectief

Het besturingssysteem Minix kent een rijke historie, die teruggaat tot de jaren tachtig van de vorige eeuw. De eerste versie van Minix is in 1987 geschreven door prof. Andrew S. Tanenbaum van de Vrije Universiteit Amsterdam en was bedoeld als onderwijsinstrument. Minix had toen al een revolutionair *multiserverontwerp*, waarin belangrijke onderdelen van het besturingssysteem, waaronder het bestandssysteem en het geheugenbeheer, als onafhankelijke programma's boven op een kleine microkernel draaien. Deze modulaire structuur biedt veel voordelen ten opzichte van monolithische systemen. Door de kleine omvang en modulaire opbouw van Minix is het besturingssysteem eenvoudig te doorgronden. De broncode van Minix is gepubliceerd als appendix in het bijbehorende boek *Operating*

Samenvatting

Het open-source besturingssysteem Minix 3 is modulair opgebouwd. De kernel biedt slechts de minimaal benodigde functionaliteit, andere taken zijn als aparte programma's boven op de kernel gerealiseerd. Alle programma's zijn strikt gescheiden en draaien met beperkte rechten, zodat fouten zich niet door het systeem kunnen verspreiden. Foutieve modules worden automatisch gedetecteerd en vervangen.

Systems Design and Implementation. Wereldwijd gebruiken vele universiteiten Minix nu voor hun onderwijs over de basisprincipes van besturingssystemen.
Eerdere versies van Minix inspireerden Linus Torvalds tot het schrijven van het bekende open–source besturingssysteem Linux. Hoewel Torvalds zowel het Minix-boek als het besturingssysteem tot zijn beschikking had, heeft hij niet voor een multiserverontwerp gekozen. In plaats daarvan heeft hij Linux een traditioneel, monolithisch ontwerp gegeven, waarbij alle functies van het besturingssysteem weer in de kernel zitten. Deze stap terug in de tijd resulteerde in 1992 in een wereldberoemde discussie tussen Tanenbaum en Torvalds op de Usenet-nieuwsgroep comp.os.minix: 'Linux is obsolete' (Linux is achterhaald).
Door de tekortkomingen van standaardbesturingssystemen zoals Windows en Linux is het onderzoek naar en de ontwikkeling van Minix in 2003 weer opgepakt door de afdeling Computer Systemen van de Vrije Universiteit Amsterdam. Tijdens zijn afstuderen en daarna als promovendus heeft Jorrit Herder het ontwerp van Minix geanalyseerd en het systeem omgebouwd tot een geheel modulair systeem waarbij nu ook alle device drivers als aparte programma's buiten de kernel draaien. Verdere ontwikkeling met het oog op veiligheid en betrouwbaarheid heeft geleid tot de eerste publieke release van Minix 3 eind 2005, die in de eerste maanden na de bekendmaking al meer dan 50.000 maal is gedownload.

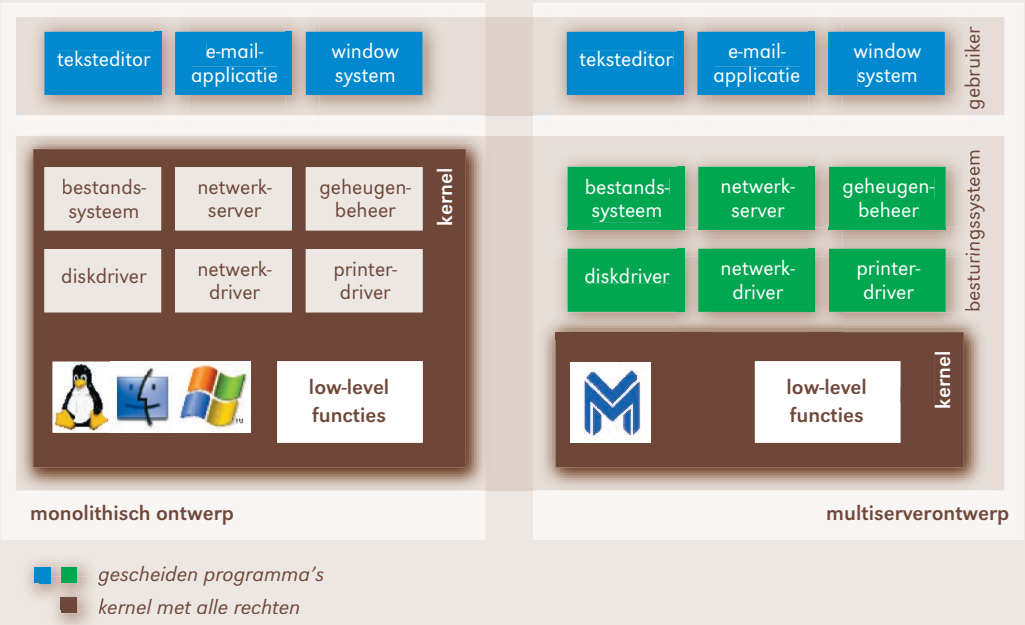
Hoe werkt Minix 3?

Voor de gebruiker werkt Minix 3 net als andere besturingssystemen. Het besturingssysteem stuurt de hardware aan en biedt de gebruiker een vriendelijke interface om de computer te bedienen. Minix 3 ondersteunt meerdere gebruikers tegelijkertijd en heeft dezelfde mogelijkheden als een normaal Unix-systeem. De standaard command-line-interface ondersteunt meerdere *virtual terminals*. Er zijn al meer dan vierhonderd Unix-programma's geschreven voor of overgezet naar Minix 3, waaronder verschillende shells (ash, bash, rsh, zsh), teksteditors (ed, emacs, vi, elle), compilers en interpreters (awk, cc, gcc, perl, python), netwerkprogramma's (telnet, ssh, ftp, lynx, mail, talk, pine, wget), bestandshulpmiddelen (cat, cp, mv, dd, tar, zip) en systeembeheerhulpprogramma's (adduser, backup, fsck, mkfs, mount, cron). En recent is ook een grafische interface, het *X Window System*, naar Minix 3 overgezet; dit wordt nu standaard meegeleverd in de 3.1.2-distributie.
Tot zo ver de overkomsten met Unix. Technisch gezien zit Minix 3 heel anders in elkaar. Traditionele Unix-systemen hebben een monolithisch ontwerp, waarbij het hele besturingssysteem in de kernel zit. Dit maakt het systeem kwetsbaar, omdat de kernel alle mogelijke rechten heeft en zelfs een kleine fout de werking van het hele systeem kan verstoren. Minix 3 heeft daarentegen een multiserverontwerp, waarbij de kernel sterk is gereduceerd en het besturingssysteem is gesplitst in onafhankelijke modules met beperkte rechten. De microkernel bevat slechts de minimale functionaliteit die nodig is om de rest van het besturingssysteem te ondersteunen. Alle functies die niet per se in de kernel hoeven te zitten, zoals het bestandssysteem, geheugenbeheer en alle device drivers die de hardware aansturen, draaien als aparte, strikt van elkaar gescheiden programma's. Deze programma's werken samen om de functionaliteit van een Unix-besturingssysteem te bieden. Het verschil in structuur tussen Minix 3 en het traditionele monolithische ontwerp dat onder andere door Windows, Mac OS en Linux wordt gebruikt, is geïllustreerd in figuur 1.

De kernel van Minix 3 is zo klein mogelijk gemaakt en bevat alleen de meest essentiële functionaliteit die niet als los programma kan worden gerealiseerd. Op het laagste niveau instrueert de kernel bijvoorbeeld welk programma hij nu moet draaien, wordt geheugen toegekend aan programma's en worden aanvragen van randappa-

ratuur opgevangen. De kernel beheert tevens de administratie van de cruciale gegevens van actieve programma's en bepaalt welk programma wanneer en hoe lang mag draaien. Beslissingen omtrent het starten en stoppen van programma's worden echter genomen door een speciale server, de *process manager*, die boven op de kernel draait. Andere servers en drivers kunnen ook gebruikmaken van de diensten van de kernel.
Een belangrijke verantwoordelijkheid van de Minix 3-kernel zijn de functies waarmee de programma's van het besturingssysteem met elkaar kunnen communiceren. Dit wordt ook wel *interprocess communication* (ipc) genoemd. Doordat alle programma's strikt gescheiden zijn, kunnen ze elkaar niet direct aanroepen. De kernel biedt echter de mogelijkheid kleine berichten uit te wisselen door ze van het ene programma naar het andere te kopiëren. Het standaardtype bericht bevat de afzender, het type verzoek en alle benodigde informatie om het verzoek af te handelen. Zo kan het bestandssysteem bijvoorbeeld een bericht naar de device driver van de harde schijf sturen met het verzoek gegevens te lezen of te schrijven.

Wanneer de driver het verzoek heeft ontvangen en afgehandeld, stuurt deze een bericht met het antwoord terug. Doordat alle berichten via de kernel worden verstuurd, kunnen allerlei beperkingen worden opgelegd aan de communicatiemogelijkheden van programma's. Voor elk programma wordt bijvoorbeeld bijgehouden welke ipc-functies mogen worden gebruikt en met welke andere programma's communicatie is toegestaan.
Een andere belangrijke taak is het bieden van zogenaamde *kernel calls* om de programma's boven op de kernel te ondersteunen. Kernel calls voeren beschermde operaties uit die programma's niet zelf kunnen of mogen verrichten. Ze kunnen worden aangeroepen door een bericht, met daarin het verzoek een kernel call uit te voeren, naar de kernel te sturen. Voorbeelden van kernel calls zijn starten en stoppen van programma's, kopiëren van geheugen tussen programma's, aansturen van de randapparatuur en instellen van privileges en beperkingen van programma's. Door dergelijke gevoelige operaties alleen in de kernel toe te staan kunnen ook hier restricties worden opgelegd aan programma's. Zo mogen device drivers in Minix 3 niet direct de computerhardware aansturen, maar moeten ze een verzoek om te lezen van of te schrijven naar de harde schijf naar de kernel sturen. De kernel kan een dergelijk verzoek vervolgens controleren en, afhankelijk van de privileges van de driver, uitvoeren of weigeren. De precieze instellingen kunnen in Minix 3 per programma



Figuur 1. **Monolithisch versus modulair ontwerp**

Links de structuur van monolithische systemen zoals Windows, Mac OS en Linux zonder bescherming tussen de verschillende functies; rechts het multiserverontwerp van Minix 3, waarbij alle modules van het besturingssysteem als gescheiden programma's draaien

verschillen. Daarentegen kunnen zulke restricties niet worden afgedwongen in een monolithisch besturingssysteem omdat alle modules in de kernel draaien en dezelfde rechten hebben.

Veiligheid en betrouwbaarheid

De modulaire structuur van Minix 3 is zeer belangrijk voor de veiligheid en betrouwbaarheid van het besturingssysteem. Omdat het besturingssysteem niet langer één geheel vormt, is het mogelijk bescherming aan te brengen tussen de verschillende modules, zodat fouten in één module zich niet kunnen verspreiden naar andere modules. Ook is het mogelijk de rechten van elke module nauwkeurig af te bakenen, waardoor de schade die een fout zou kunnen aanrichten tot een minimum beperkt blijft. Enkele voorbeelden van mogelijke restricties zijn al aan de orde gekomen. Deze bescherming van Minix 3 is vergelijkbaar met het gebruik van compartimenten bij schepen: als een lek optreedt, stroomt alleen dat bepaalde compartiment vol en niet het hele ruim, zodat het schip blijft varen en het lek naderhand kan worden gedicht.

»Fouten in één module kunnen zich niet verspreiden naar andere modules«

Uit onderzoek naar de betrouwbaarheid van besturingssystemen blijkt dat de device drivers in Linux drie- tot zevenmaal meer fouten bevatten dan de rest van de kernel. Dergelijke fouten kunnen in traditionele besturingssystemen tot grote problemen leiden omdat een fout in de driver van de printer het hele systeem kan aantasten. Onder Windows wordt bijvoorbeeld 85 procent van alle crashes door drivers veroorzaakt. Daarom zijn in Minix 3 alle device drivers afgeschermd van de rest van het systeem. Zo draait de printerdriver als een apart programma en heeft alleen toegang tot de printer. Een eventuele fout kan ertoe leiden dat de printer niet meer kan worden aangestuurd, maar deze kan niet langer het hele systeem ontregelen.

Omdat de kernel alle rechten heeft en dus de bescherming tussen de programma's kan omzeilen, is het zeer belangrijk dat de kernel correct functioneert. Er zijn verschillende studies gedaan naar de dichtheid van fouten in software met uitkomsten van tussen de 2 en 75 fouten per 1000 regels code. Om het aantal fatale fouten te beperken is het dus van groot belang de kernel zo klein mogelijk te maken. De Linux-kernel omvat zo'n 2,5 miljoen regels code, wat met een voorzichtige schatting van 10 fouten per 1000 regels code betekent dat deze 25.000 fouten bevat. Bij Windows is het probleem met zo'n 5 miljoen regels code in de kernel zelfs nog groter. De kernel van Minix 3 omvat daarentegen slechts 4000 regels code, waardoor het aantal fouten in het meest kwetsbare deel van het besturingssysteem veel kleiner zal zijn. De beperkte omvang van de kernel maakt het systeem ook eenvoudig te begrijpen en te bouwen, wat de kans op fouten nog kleiner maakt.

Een ander aspect van Minix 3 dat tot meer betrouwbaarheid leidt, is de mogelijkheid om automatisch fouten te herstellen. De modules van het besturingssysteem worden gecontroleerd door een speciale module, de *reïncarnatieserver* geheten, die verantwoordelijk is voor de juiste werking van het systeem. Wanneer een netwerkdriver onverhoopt vastloopt, wordt deze automatisch verwijderd uit het systeem en vervangen door een nieuw exemplaar. Ook kan de systeembeheerder automatisch met een e–mail op de hoogte worden gebracht van het probleem. Dit alles gebeurt zonder interventie van de gebruiker en leidt slechts tot een kleine vertraging van het systeem. Eventuele openstaande netwerkverbindingen worden niet beïnvloed, zodat bijvoorbeeld de internetbrowser blijft werken. Experimenten laten zien dat het ophalen van een groot bestand van internet slechts 8 procent langer duurt wanneer er elke 4 seconden een fout optreedt in de netwerkdriver en deze herhaaldelijk opnieuw moet worden gestart. Deze eigenschappen maken Minix 3 niet alleen geschikt voor de pc, maar ook voor systemen die zeer betrouwbaar moeten zijn, zoals de ingebedde computers van auto's, vliegtuigen of ruimtesondes.

Minix 3 is niet alleen een betrouwbaar maar ook een veilig besturingssysteem. De bescherming van Minix 3 voorkomt namelijk ook veel beveiligingsproblemen die worden veroorzaakt door fouten in de software. Veel virussen en wormen maken bijvoorbeeld misbruik van zogeheten *buffer-overrunfouten* om hun eigen code in het systeem te injecteren en uit te voeren. Omdat de code van alle programma's door Minix 3 beschermd wordt, is het uitvoeren van geïnjecteerde code niet langer mogelijk en zijn zulke fouten veel moeilijker te

misbruiken. Wanneer een aanval leidt tot een foutieve handeling of afwijkend gedrag in een deel van het besturingssysteem, merkt de reïncarnatie-server dit direct en zal de slecht functionerende module vervangen door een nieuw exemplaar. Een interessant voorbeeld is de zogenaamde *ping of death*, waardoor een kwaadwillende persoon een Windows-server kan laten crashen met een enkel fout verzoek vanaf internet. Moderne varianten van de ping of death zijn recentelijk aangetroffen in de interactie met de remote desktopserver van Windows. Een dergelijke aanval zal in Minix 3 slechts een lokale fout veroorzaken, waarna het besturingssysteem zich in veel gevallen automatisch kan herstellen.

Tot slot

Minix 3 is een nieuw besturingssysteem dat ontworpen is met het oog op de veiligheid en betrouwbaarheid voor de eindgebruiker. In tegenstelling tot traditionele systemen zoals Linux en Windows is het besturingssysteem opgesplitst in aparte modules. De kernel van Minix 3 biedt slechts de minimale functionaliteit die nodig is om de rest van het besturingssysteem te ondersteunen. Alle taken die niet per se in de kernel moeten zitten, zoals het bestandssysteem en device drivers, zijn als aparte programma's boven op de kernel gerealiseerd. De Minix 3-kernel biedt functies waarmee de programma's kleine berichten kunnen uitwisselen, alsmede kernel calls om beschermde operaties uit te voeren. Zodoende kunnen de kernel en alle programma's samenwerken om de functionaliteit van een normaal Unix-systeem te bieden.

Het ontwerp van Minix 3 biedt vele voordelen voor de betrouwbaarheid en veiligheid van het besturingssysteem. Alle programma's zijn strikt gescheiden en draaien met beperkte rechten, zodat programmeerfouten en andere problemen zich niet door het systeem kunnen verspreiden. Hierdoor wordt de schade die een fout kan aanrichten beperkt. Omdat de modules van Minix 3 klein en eenvoudig zijn, zijn ze gemakkelijk te begrijpen en beheren, waardoor de kans op fouten vermindert. Omdat de werking van het besturingssysteem continu wordt gecontroleerd en foutieve modules automatisch gedetecteerd en vaak zelfs vervangen kunnen worden, is Minix 3 uiterst geschikt voor systemen die zeer betrouwbaar moeten zijn. Momenteel wordt aan de Vrije Universiteit Amsterdam actief onderzoek gedaan om de veiligheid en betrouwbaarheid van Minix 3 nog beter te maken en het systeem verder te ontwikkelen. Daarnaast is er een groeiende gemeenschap van geïnteresseerden en vrijwilligers die aan het systeem bijdragen. Het besturingssysteem is al meer dan 50.000 maal gedownload en het aantal gebruikers groeit gestaag.

Meer informatie

• Bezoek de officiële Minix 3-website (www.minix3.org) voor een download en meer informatie of ga naar de nieuwsgroep (comp.os.minix) voor discussies over Minix 3.

• Neem contact op met Jorrit Herder (jnherder @cs.vu.nl) van de Vrije Universiteit Amsterdam voor meer informatie of vragen over Minix 3.

Literatuur

Tanenbaum, A.S. & A.S. Woodhull (2006). *Operating Systems Design and Implementation*, 3e editie. Englewood Cliffs: Prentice–Hall.

Link

www.minix3.org

Jorrit Herder
heeft een M.Sc.-graad in Computer Science van de Vrije Universiteit Amsterdam en is daar momenteel promovendus. Hij is nauw betrokken bij de ontwikkeling van Minix 3. E-mail: jnherder@cs.vu.nl.

Herbert Bos
is universitair docent Informatica aan de Vrije Universiteit Amsterdam. E-mail: herbertb@cs.vu.nl.

Ben Gras
heeft een M.Sc.-graad in Computer Science van de Vrije Universiteit Amsterdam en is daar werkzaam bij de afdeling Computer Systemen Amsterdam. Hij werkt als programmeur mee aan het Minix 3-project. E-mail: beng@few.vu.nl.

Philip Homburg
is gepromoveerd aan de Vrije Universiteit Amsterdam op het gebied van grootschalige gespreide systemen en werkt mee aan het Minix 3-project. E-mail: philip@cs.vu.nl.

Andrew S. Tanenbaum
is professor op het gebied van Computer Science aan de Vrije Universiteit Amsterdam. E-mail: ast@cs.vu.nl.