Platform Informatiebeveiliging

z/OS Security Server (RACF) versie 1.5

Document Versie 3

2 december 2004

Deze PI-standaard bevat 249 pagina's

Inhoudsopgave

1 Voorwoord

Het Platform Informatiebeveiliging (PI) heeft als doel ‘het bevorderen van de beveiliging van alle belangen betreffende gegevensverwerking, -opslag en -transport, alles in de ruimste zin van het woord’. Binnen deze doelstelling wordt het ontwikkelen van aanvaardbare richtlijnen voor de praktische inrichting van informatiebeveiliging als essentieel onderwerp gezien. Door het gezamenlijk opstellen van dergelijke richtlijnen kan worden gebruikgemaakt van praktijkervaringen, zodat een doeltreffende richtlijn ontstaat die ook uitvoerbaar is.

De PI-richtlijnen worden in werkgroepverband ontwikkeld onder auspiciën van een bestuurslid in de rol van projectleider. Deze ziet er onder meer op toe dat de PI-kwaliteitsrichtlijnen door de werkgroep worden gehandhaafd. De deelnemers van de werkgroepen zijn primair afkomstig uit de organisaties, die aangesloten zijn bij PI, maar niet uitsluitend. Het betreft beveiligingsfunctionarissen en ICT-auditors van uiteenlopende bedrijven en instellingen, die zich kenmerken door de hoge eisen die zij in hun advies- en controlewerkzaamheden aan organisaties moeten stellen in verband met de sterke automatiseringsgraad en de belangen die met de geautomatiseerde informatievoorziening zijn gemoeid. Door deze achtergrond vormen de deelnemers een representatieve afspiegeling van de aanwezige ICT-beveiligingsexpertise in Nederland en bieden zij een draagvlak om gezag te verlenen aan de ontwikkelde richtlijnen, hetgeen bevorderlijk is voor de acceptatie door het algemene en het ICT-management.

De PI-beveiligingsrichtlijnen zijn primair bedoeld voor functionarissen die zijn belast met het implementeren van IT-systemen, zoals systeembeheerders en technische ontwerpers en bouwers.

Daarnaast zijn de richtlijnen van betekenis voor de volgende doelgroepen:

- ICT-beveiligingsfunctionarissen (security officers en administrators). De ICT-beveiligings-functionaris binnen een organisatie is verantwoordelijk voor het (doen) treffen van beveili-gingsmaatregelen. De richtlijnen bieden hierbij ondersteuning.
- ICT- en algemeen management. Het management is (eind)verantwoordelijk voor de infor-matiebeveiliging en geeft hieraan invulling door het (doen) analyseren van risico's en het bepalen van (globale) beveiligingsdoelstellingen. De keuzen in de richtlijnen zijn hierbij een handvat.
- ICT-auditors. De richtlijnen geven de vereiste beveiligingsmaatregelen aan. Hierdoor kunnen de richtlijnen ook worden gehanteerd als toetsingsnorm bij ICT-audits.

Aldus bieden de richtlijnen enerzijds een handreiking aan beveiligingsfunctionarissen en het algemene en ICT-management om een toereikende en evenwichtige beveiliging van de infor-matievoorziening te implementeren en bieden zij anderzijds een basis aan ICT-auditors voor de normstelling bij de beoordeling van de beveiliging van een ICT-systeem.

Voor eenvoudige herkenning van tekstwijzigingen ten opzichte van versie 2 zijn verticale lijnen (|) naast de toegevoegde tekst aangebracht. De lezers die versie 2 van deze PI RACF-standaard reeds hebben geïmplementeerd zullen aangepaste normen opnieuw moeten evalueren en indien van toepassing implementeren.

De werkzaamheden van KPMG IRM voor versie 1 zijn begeleid en getoetst door een werkgroep bestaande uit de volgende leden:

▪ Henk van Ballegooy	Rabobank
▪ Bart Bokhorst	Belastingdienst Centrum voor ICT, B/CICT
▪ Laure Closset	Elceda
▪ Frank Engel	KPMG Information Risk Management, IRM
▪ Mike Greene	Achmea Security Management
▪ Bart van den Heuvel	Atos Origin
▪ Henk Keller	ING, ITC/ISS/CSA

- Theo Krens KPMG Information Risk Management, IRM
- Leen van Rij KPMG Information Risk Management, IRM

Ondersteunende adviezen werden gegeven door:
- Peter Mienes KPMG Information Risk Management, IRM

Revisoren van de exposure draft versie 0.6 waren:
- Kees Jongejans Rabobank
- André Premereur Fortis België
- Rob Velthuis Belastingdienst Centrum voor ICT, B/CICT

Deelnemers PI RACF-discussiegroep versie 1:
- Jos Haring IN IT
- Henk Hebing Atos Origin
- Ary van der Most Civility
- Coen Robeers Rabobank
- John Rudolph CTG
- Hans Schoone Consul Risk Management
- Thom Snels ABN Amro
- Tom van der Toorn ABN Amro

Deelnemers PI RACF-werkgroep versie 2 waren:
- Lorenz Kenter Kenter AutoVision
- John Rudolph LogicaCMG
- Frank Engel KPMG Information Risk Management
- Theo Krens KPMG Information Risk Management
- Peter Mienes KPMG Information Risk Management
- Leen van Rij KPMG Information Risk Management

Revisoren van de exposure draft versie 2 waren:
- Laure Closset Elceda
- Lorenz Kenter Kenter AutoVision
- John Rudolph LogicaCMG

Deelnemers PI RACF-werkgroep versie 3 waren:
- Theo Krens KPMG Information Risk Management
- Leen van Rij KPMG Information Risk Management

De auteur,

Leen van Rij KPMG Information Risk Management, IRM

mailto:VanRij.Leen@kpmg.nl

2 Inleiding

De toenemende integratie van automatisering met de bedrijfsprocessen en de eveneens toenemende complexiteit van automatiseringsoplossingen noodzaken tot voortdurende aandacht voor zowel het vereiste niveau van informatiebeveiliging als de technische realisering hiervan. Ook gezien de ontwikkelingen op het gebied van wet- en regelgeving met betrekking tot informatiebeveiliging is deze aandacht noodzakelijk.

Objectivering van het vereiste niveau van informatiebeveiliging en van de effectiviteit van gekozen technische oplossingen is voor veel organisaties een probleem, doordat slechts in beperkte mate standaarden voorhanden zijn. Beschikbare standaarden kennen ofwel een te beperkt werkingsgebied, of richten zich te veel op de organisatorische kant van de informatiebeveiliging. Door het gebrek aan deugdelijke standaarden zijn organisaties gedwongen zelf oplossingen te ontwikkelen en hieraan veel energie te besteden. De gevolgen zijn suboptimale oplossingen, verspilling doordat vele malen opnieuw het wiel moet worden uitgevonden en moeizame acceptatie door de afwezigheid van geobjectiveerde criteria.

Tegen deze achtergrond is het initiatief ontstaan om in werkgroepverband concrete, geobjectiveerde richtlijnen te ontwikkelen voor de inrichting respectievelijk de beoordeling van technische beveiligingsmaatregelen. Deze aanpak heeft de volgende voordelen:

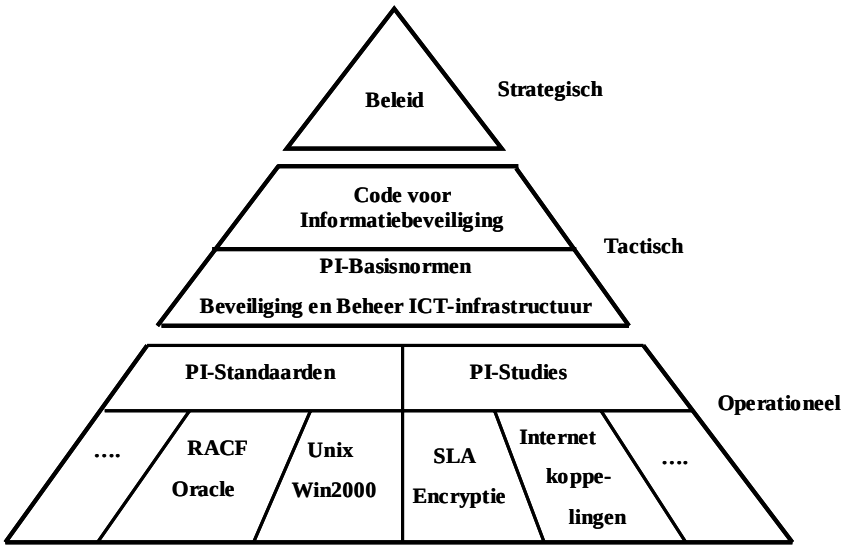
- door uitwisseling van kennis, ervaring en inzicht ontstaat een belangrijk synergie-effect tussen de deelnemers; de deelnemers kunnen elkaar ondersteunen bij de keuze en implementatie van beveiligingsmaatregelen;
- met behulp van de ingebrachte kennis en inzichten kan worden gekomen tot de vaststelling van technische beveiligingsrichtlijnen die op een breed draagvlak kunnen rekenen;
- toepassing van de opgestelde beveiligingsrichtlijnen leidt bij de betrokken organisaties tot een verhoging van de effectiviteit van de beveiliging.

De technische beveiligingsmaatregelen die in de richtlijnen worden beschreven vormen een onderdeel van de bredere context van het gehele samenstel van beveiligingsmaatregelen om de kwaliteit van de geautomatiseerde informatievoorziening te waarborgen. Beveiligingsmaatregelen binnen deze bredere context zijn bijvoorbeeld beschreven in de publicatie 'Code voor Informatiebeveiliging. Standaard voor Beleid en Implementatie[1]' . Deze Code, die vooral in het bedrijfsleven wordt gebruikt, richt zich in het bijzonder op het tactische niveau binnen organisaties en bestrijkt het gehele terrein van informatiebeveiliging. Door het abstractieniveau geeft de Code echter weinig concrete handvatten voor het implementeren van beveiligingsmaatregelen bij ICT-systemen. Hetzelfde geldt voor het besluit Voorschrift Informatiebeveiliging Rijksdienst (VIR)[2], dat voor de rijksoverheid van toepassing is.

De PI-richtlijnen kunnen dan ook worden beschouwd als een verdere uitwerking van de Code en de baselinebeveiliging van het besluit VIR en zijn vooral gericht op het operationele niveau binnen organisaties. Daarnaast kan nog een strategisch niveau worden onderkend, dat betrekking heeft op de eindverantwoordelijkheid voor informatiebeveiliging van het topmanagement. De samenhang tussen deze drie niveaus is schematisch weergegeven in figuur 1.

[1] De Code voor Informatiebeveiliging is een uitgave van het Nederlands Normalisatie-instituut, mogelijk gemaakt door het Ministerie van Economische Zaken in samenwerking met een groep toonaangevende bedrijven en organisaties in Nederland.

[2] Het besluit Voorschrift Informatiebeveiliging Rijksdienst 1994 is uitgegeven door het Ministerie van Binnenlandse Zaken in samenwerking met een begeleidingsgroep waarin alle ministeries waren vertegenwoordigd en enkele toonaangevende organisaties.



Figuur 1. Informatiebeveiligingspiramide

Aangezien voor het tactische niveau van informatiebeveiliging en voor de beleidsmatige en organisatorische maatregelen die op het strategische en tactische niveau moeten worden getroffen, al veel literatuur voorhanden is, wordt hier in de PI-richtlijnen niet nader op ingegaan. Het uitgangspunt van de PI-richtlijnen is dat op dit gebied voldaan is aan de Code voor Informatiebeveiliging en vergelijkbare standaarden, hetgeen inhoudt dat er een beveiligingsbeleid is, dat er functiescheiding is tussen ontwikkeling en productie, etc., etc.

Bij de implementatie van een product of architectuur moet een evenwicht worden gevonden tussen risico's en daarmee samenhangend beveiligingsniveau, gebruikersgemak, invoerings- en beheerkosten en gevolgen voor de prestaties van het systeem. De richtlijnen bieden hierbij een praktische leidraad, doordat beargumenteerd wordt aangegeven waarom bepaalde keuzen zijn gemaakt. Door deze aanpak kunnen organisaties de vertaalslag maken naar hun eigen specifieke omstandigheden.

2.1 PI-Beveiligingsrichtlijnen

De beveiligingsrichtlijnen die in PI-verband zijn en worden ontwikkeld, betreffen de technische maatregelen en voorzieningen die moeten worden getroffen ter waarborging van de integriteit, vertrouwelijkheid, beschikbaarheid en controleerbaarheid van de gegevens die met een ICT-systeem worden opgeslagen, verwerkt en/of getransporteerd.

2.1.1 Kwaliteitsaspecten

De richtlijnen geven dus aan hoe de (beveiligings)functies van een ICT-systeem die relevant zijn voor de genoemde kwaliteitsaspecten, moeten worden ingesteld. Zij bevatten tevens aanwijzingen voor de organisatorische inbedding hiervan, maar primair gaat het om de techniek.

Met de richtlijnen wordt vooral beoogd te bevorderen dat de integriteit, vertrouwelijkheid, beschikbaarheid en controleerbaarheid van de gegevens in voldoende mate is gewaarborgd. Dat wil niet zeggen dat andere kwaliteitsaspecten, zoals efficiëntie (bedieningsgemak, performance, kostenbeheersing), uit het oog worden verloren. Juist door de inbreng vanuit de praktijk wordt gestreefd naar een optimaal evenwicht tussen beveiligingsniveau en praktische realiseerbaarheid. De richtlijnen vormen de neerslag van de gezamenlijke kennis, inzichten en praktische ervaringen van de werkgroep- en PI-leden.

2.1.2 Scope

De richtlijnen kunnen betrekking hebben op elk component van de ICT-infrastructuur in de breedste zin van het woord (aangeduid als ICT-systeem). De ICT-infrastructuur betreft het geheel van apparatuur, besturings- en hulpprogrammatuur, faciliteiten voor data-, spraak- en videocommunicatie, alsmede de fysieke beveiligingsfaciliteiten van de geautomatiseerde informatievoorziening. Ook generieke toepassingen en diensten (zoals e-mail en file transfer) vallen onder het begrip ICT-systeem.

De beveiligingsrichtlijnen van PI vallen uiteen in twee categorieën:
1. PI-standaarden. PI-standaarden zijn technische implementatiehandleidingen voor concrete objecten (ICT-producten), bijvoorbeeld een besturingssysteem van een bepaalde leverancier en een bepaalde versie;
2. PI-studies. PI-studies zijn technische beveiligingshandleidingen voor objecttypen, bijvoorbeeld een bepaalde categorie besturingssystemen, generieke ICT-architecturen, bijvoorbeeld een firewall, Internet-verbinding of inbelfaciliteit, generieke diensten, bijvoorbeeld directory services en dergelijke.

Bij nieuwe versies van producten en bij nieuwe technologische of maatschappelijke ontwikkelingen bestaat er behoefte aan vroegtijdige risico-inschatting en standpuntbepaling met betrekking tot de invulling van beheer- en beveiligingsaspecten. In die gevallen zijn de richtlijnen meer het resultaat van een researchinspanning dan dat zij - zoals bij bestaande producten en architecturen - zijn gebaseerd op eigen praktische ervaring ('best practice').

Het te bereiken beveiligingsniveau dient te zijn afgestemd op de waarde van het te beveiligen belang. Aangezien dit voor elke organisatie verschillend zal zijn, zijn de PI-richtlijnen primair gebaseerd op de beveiligingsmogelijkheden van het ICT-systeem, dat wil zeggen op het optimaal benutten van de beveiligingsfaciliteiten die het biedt. Dit wordt het beginsel van goed huisvaderschap genoemd. Bij het opstellen van de richtlijnen wordt echter tevens nagegaan aan welk niveau het systeem redelijkerwijs zou moeten voldoen, gegeven de Code voor Informatiebeveiliging en andere gezaghebbende literatuur, de ‘state of the art’ van de beveiligingstechniek, de gemeenschappelijke ‘common sense’ van de werkgroepleden, enzovoort.

2.2 Status document

Deze versie van de RACF-standaard is nog aan verandering onderhevig. Het is een **werkdocument** dat nog vele toevoegingen zal krijgen, zoals de inhoudsopgave reeds laat zien. De reeds beschreven onderwerpen zullen nog worden aangepast met informatie en opmerkingen die vanuit de praktijk worden aangedragen. Een gedeelte van dit document is geschreven voor een decentrale beheerorganisatie. Dit is vooral terug te vinden in de hoofdstukken over groepen, gebruikers en dataset protectie. Andere hoofdstukken zullen op later tijdstip worden herzien voor toepassing voor decentraal beheer.

De lezer wordt gevraagd om informatie aan te leveren voor het verhogen van de kwaliteit en de bruikbaarheid van het gehele document. Lezers worden dan ook van harte uitgenodigd hun opmerkingen te zenden aan:

KPMG Information Risk Management
Platform Informatiebeveiliging
Leen van Rij
Burgemeester Rijnderslaan 20
1185 MC Amstelveen
e-mail: VanRij.Leen@kpmg.nl

2.3 Uitgangspunten

Bij het lezen/toepassen van deze standaard dienen de volgende uitgangspunten in acht te worden genomen:

Uitgangspunt 1: implementatie keuze

RACF geeft de mogelijkheid, om voor eenzelfde doelstelling, verschillende implementatievormen te gebruiken. Door deze flexibiliteit kunnen, indien niet vroegtijdig onderkend, verkeerde keuzes ontstaan die slechts met veel inspanning te wijzigen zijn. Een voorbeeld hiervan is de gekozen groepenstructuur; als deze niet in overeenstemming is met de criteria, die wij daarvoor hebben ontwikkeld, kan het dagelijkse beveiligingsbeheer veel inspanning vergen. Indien mogelijk geeft dit document een keuze tussen twee of meer te volgen implementatiepaden voor de betreffende RACF-functionaliteiten. Voor het maken van de betreffende keuze is een argumentatie opgenomen die kan worden gebruikt om de keuze te bepalen.

Uitgangspunt 2: koppelingen

Deze standaard gaat er vanuit dat bij gebruik van bijvoorbeeld: Parallel Sysplex, Multi Access Spool (MAS), shared DASD, etc. alle gekoppelde systemen hetzelfde beveiligingsniveau hanteren. Indien één van de systemen een lager beveiligingsniveau heeft zal dit het uiteindelijke niveau van het gehele complex bepalen.

Uitgangspunt 3: aanvullende beveiligingsmaatregelen

Als normen vooraf worden gegaan met de tekst ***AANVULLENDE MAATREGEL*** geeft dit aan dat de implementatie van deze norm een verhoging van het basisniveau implementeert. Het hoofdstuk ‘gebruikers- en data-categorisering’ beschrijft de implementatie van beveiligingslevels, beveiligingscategorieën en beveiligingslabels volgens het B1 beveiligingsniveau. Het B1 beveiligingsniveau is gedefinieerd door de Department of Defense (DoD) van USA. Het gehele B1 beveiligingsniveau wordt in deze standaard, alhoewel hierin meerdere implementatievormen kunnen worden onderkend, in zijn geheel als aanvullende maatregel gezien.

Uitgangspunt 4: soorten beveiligingsfunctionarissen

In deze PI-standaard wordt onderscheid gemaakt tussen een viertal beveiligingsfunctionarissen. De eerste drie gebruikte functienamen zijn conform de RACF documentatie, en zijn:

- Security officer, verantwoordelijk voor de aanpak en keuzes van de beveiliging en beveiligingsimplementatie conform het voor de organisatie geldende ICT-beveiligingsbeleid;
- Security administrator, zorgt voor de dagelijkse verwerking van autorisatieaanvragen en voor de definities in de RACF-database conform de normen en richtlijnen opgesteld door de Security officer. Deeltaken kunnen worden gedelegeerd naar decentrale Security administrators;
- Security auditor, zorgt voor de dagelijkse controle op de beveiligingsimplementatie en het -beheer door monitoring en rapportages. Deze functie kan deels worden gedelegeerd naar decentrale Security auditors.

Naast deze Security functionarissen kunnen er ook ICT-auditors worden onderkend, die aan het hoger management zoals een directie of een raad van bestuur rapporteren over de inrichting en werking van het beveiligings- en eventueel het RACF-beheer.

Uitgangspunt 5: beheerfunctionarissen

In deze PI-standaard is ervan uitgegaan dat de Security administrators alle groeps- en resource-profielen beheren, terwijl de HRM-afdeling de gebruikers- en de connect-profielen beheert. Hierbij gaan we er vanuit dat de HRM-afdeling als ondersteunende afdeling t.b.v. het lijnmanagement volledig op de hoogte is van de wijzigingen in het medewerkersbestand. Het gaat hierbij zowel om de permanente als de tijdelijke medewerkers (contracters). Voorwaarde hierbij is dat de HRM-afdeling, of andere hieraan gelijk te stellen afdelingen, middelen voor het definiëren van gebruikers moeten hebben.

Uitgangspunt 6: gebruik van deze PI-standaard

Voor het implementeren van een basisbeveiligingsniveau is het noodzakelijk alle hoofdstukken door te nemen ongeacht of het betreffende product toepast wordt. In elk hoofdstuk wordt namelijk aangegeven wat moet worden gedaan als het product wel of juist niet wordt gebruikt.

Uitgangspunt 7: specifieke uitgangspunten

Indien er voor een component specifieke uitgangspunten worden gehanteerd zijn deze in het betreffende hoofdstuk opgenomen. Een voorbeeld hiervan is DB2 waarbij afwijkende uitgangspunten moesten worden toegepast.

Uitgangspunt 8: kennis van RACF

Om op een verantwoorde wijze gebruik te kunnen maken van deze PI-standaard is voldoende inhoudelijke kennis en inzicht in de werking van RACF een vereiste. Het toepassen van deze PI-standaard zonder voldoende achtergrondkennis van RACF kan ertoe leiden dat de beveiligingsmaatregelen verkeerd worden geïnterpreteerd, waardoor er een zwakkere beveiligingsstructuur kan ontstaan dan wellicht de bedoeling is.

Uitgangspunt 9: beperkingen bij de uitwerking van deze PI-standaard

In deze PI-standaard is afgezien van het geven van enige vorm van productuitleg, omdat dat voor de deskundigen niet nodig is en voor de minder deskundige lezer al gauw als ontoereikend zal worden ervaren. Wij verwijzen hiervoor naar de RACF-manuals en de beveiligingsparagrafen uit de verschillende product-manuals van de leverancier, die zoveel als mogelijk zijn gevolgd voor de opbouw van dit document. Afwijkingen van de indeling van de manuals komen alleen voor om het betreffende onderwerp extra te kunnen belichten. Verder is in deze PI-standaard geen algemene opsomming van risico’s bij de maatregelen gegeven omdat dit onvoldoende informatie zou toevoegen.

2.4 Leeswijzer

Voor deze PI-standaard is een zodanige notatie gekozen dat snel de belangrijke parameters kunnen worden herkend. Hieronder is een RACF commando aangegeven waarbij de nadruk ligt (bold aangegeven) op IBMUSER omdat dit het belangrijkste gegeven is in onderstaand voorbeeld:

ALTUSER **IBMUSER** NAME('IBM Installatie Userid')

In de tekst zijn RACF-commando's opgenomen die de beveiligingsprofielen in de RACF-database definiëren om zo aan de gestelde norm te voldoen. Alhoewel is aangegeven dat het RACF-commando de implementatie is, moet de Security administrator ermee rekening houden dat bepaalde weergegeven commando's, zoals definities van gebruikers, groepen, datasets, etc., meerdere malen moet worden gegeven afhankelijk van de specifieke wensen en opbouw van de installatie.

Indien is aangegeven dat het RACF-commando een voorbeeld is dan zal in de praktijk de installatiespecifieke wensen en eisen moeten worden gevolgd. Het RACF-commando zoals beschreven, geeft dan een indicatie voor de Security administrator om de definitie te kunnen maken.

Het RACF-commando moet in de praktijk worden aangepast naar de wensen en naamgevingconventie zoals door de Security officer is aangegeven. Als in het RACF-commando een installatiespecifieke waarde moet worden gebruikt dan worden hiervoor *italic* karakters gebruikt. Een voorbeeld:

SETROPTS **RVARYPW(SWITCH(***HFUKSNCE***) STATUS(***KDUEFTJD***))**

In de RACF-commando's zijn uitsluitend die parameters opgenomen die noodzakelijk zijn voor de implementatie van de genoemde norm. Voor het correct uitvoeren van de RACF-commando's zal het over het algemeen noodzakelijk zijn om nog meer parameters te definiëren (zie hiervoor de betreffende RACF documentatie).

2.5 Disclaimer

Hoewel bij deze uitgave de uiterste zorgvuldigheid is nagestreefd, kunnen fouten en onvolledigheden niet geheel worden uitgesloten. De leden van PI en/of leden van de werkgroepen en/of het secretariaat aanvaarden derhalve geen enkele aansprakelijkheid, hoe dan ook genaamd, uit welken hoofde dan ook voor enig gevolg rechtstreeks of indirect voortvloeiend uit het gebruik van deze uitgave.

3 Installatie

3.1 Introductie

De hieronder beschreven RACF-installatienormen geven informatie over die aspecten waaraan moet worden voldaan om het basisbeveiligingsniveau te waarborgen.

3.2 Status

Dit hoofdstuk is gereed voor commentaar door een ieder die toevoegingen of verbeteringen ziet.

3.3 RACF-databases

RACF dient naast de primairy database een actieve backup database geïnstalleerd te hebben. Deze RACF databases dienen via verschillende I/O-paden bereikbaar te zijn zodat voorkomen wordt dat in de hardwareconfiguratie een ‘single point of failure’ voorkomt. Verder dienen de RACF-databases op verschillende volumes te worden geplaatst. De RACF-databases dienen op DASD device te worden gedefinieerd waarbij de beschikbaarheid (bijvoorbeeld RAID5 devices) en performance is gewaarborgd.

3.4 RACF-database sharing

Het sharen van RACF-databases met een ander systeem is niet toegestaan als deze systemen een ander beveiligingsniveau hebben. Als voorbeeld: sharing test- en productiesysteem is niet toegestaan, maar wel tussen productie-A en productie-B. In één Sysplex, productie, acceptatie of test Sysplex, dienen alle systemen in dat Sysplex dezelfde RACF databases te sharen. Opmerking: een testsysteem verschilt van een productiesysteem door veelal meer en andere bevoegdheden voor de gebruikers. De gebruikers zijn vaak geen eindgebruikers maar applicatie/systeemontwikkelaars. Bevoegdheden die veelal op een testsysteem worden uitgegeven zijn doorgaans niet op een productiesysteem toegestaan en die op het productiesysteem zijn niet op een testsysteem toegestaan.

3.5 IBMUSER

Het installatie userid ‘IBMUSER’ dient direct na de eerste IPL van het systeem te worden gebruikt om een nieuw TSO userid met de SPECIAL attribuut aan te maken. Nadat geconstateerd is dat het nieuwe userid correct werkt en gebruikt kan worden voor het uitvoeren van RACF commando’s en het definiëren van RACF-profielen moet het ‘IBMUSER’ userid daarna direct worden gedeactiveerd. Een voorbeeld:

```
ADDUSER  SECADM01  NAME('O.N. Veilig')  SPECIAL

ALTUSER  IBMUSER  NAME('IBM Installatie userid')
NOPASSWORD  REVOKE
NOSPECIAL  NOOPERATIONS  NOAUDITOR
```

3.6 RACF en SAF exits

Gebruik van RACF- en SAF-exits, behalve de noodzakelijke standaard exits, is niet toegestaan tenzij aangetoond kan worden dat dit het beveiligingsniveau verhoogt. Het gebruik van de standaard ICHDEX01 en ICHDEX11 exit (beide met fixed return code 8) dient echter wel te worden geïmplementeerd om wachtwoord verificatie uitsluitend via het DES-algoritme te laten verlopen. Als exits worden toegepast dienen deze als USERMOD door middel van SMP/E te worden geïnstalleerd.

3.7 SETROPTS definities

De definities die de algehele werking van RACF beïnvloeden worden met het SETROPTS (Set RACF options) commando uitgevoerd. De SETROPTS-parameters worden gebruikt om specifieke functionaliteiten te activeren of de werking aan te passen. De SETROPTS-parameters zijn daarom in de specifieke hoofdstukken bij de betreffende functionaliteiten op genomen. De waarden voor de SETROPTS-parameters dienen aan het organisatie specifieke informatiebeveiligingsbeleid te worden ontleend. Wij raden aan om de voorgestelde en vereiste SETROPTS parameters eerst te implementeren/activeren voordat een installatie voor eindgebruikers toegankelijk wordt gemaakt.

3.8 Database backup

De RACF-databases dienen vanaf dat deze zijn geïnitialiseerd periodiek te worden veiliggesteld. De RACF-database backups dienen veilig te worden bewaard zodat ongeautoriseerde personen geen toegang hiertoe kunnen krijgen. Men dient rekening te houden dat de backup van een RACF database nooit de actuele situatie kan herstellen omdat de database restore van RACF geen roll-foreward mechanisme heeft. Zoals beschreven in paragraaf RACF-databases is het noodzakelijk om naast de primaire een actieve backup (secondaire) database toe te passen. Met de actieve backup database wordt een ‘single point of failure’ voorkomen en wordt het belang van een statische backup gereduceerd. Voor de statische backup wordt aangeraden deze minstens dagelijks veilig te stellen op tape.

3.9 Recovery procedure

Voordat RACF in productie wordt genomen dienen alle recovery procedures aanwezig, geverifieerd en getest te zijn.

3.10 Performance

Doordat RACF door vele componenten veelvuldig wordt geraadpleegd dienen alle functies die de performance positief kunnen beïnvloeden te worden gebruikt. Zo kan gebruik worden gemaakt van de Virtual Lookaside Facility (VLF) en het in-storage plaatsen, bijvoorbeeld het RACLISTen, van RACF-profielen te worden geïmplementeerd. De performancefactor is bij het veelvuldig gebruik van beveiligingsfuncties hierdoor geen belemmering meer. In het RACF-manual “Security Administrators Guide” zijn de classes opgenomen die in aanmerking komen voor RACLIST.

Indien de RACF-database door utilities of door meerdere systemen wordt gebruikt moet worden voorkomen dat de RACF-database wordt gelocked aangezien dit een performance vermindering

geeft. Eventuele langdurige updates van de RACF-database moeten worden uitgevoerd in een relatief rustige periode.

3.11 Console

Alle door RACF gegenereerde messages dienen naar een console te worden gestuurd, en mogen niet door de Message Processing Facility (MPF) worden onderdrukt.

AANVULLENDE MAATREGEL. Voor het snel onderkennen van beveiligingsincidenten dienen minimaal een primaire en een secondaire (backup) console voor uitsluitend RACF gerelateerde boodschappen te worden gedefinieerd en aanwezig te zijn.

3.12 Database Switch en Status commando

Voor het beheren van de RACF-databases dient het RACF console commando RVARY te worden afgeschermd voor ongeautoriseerd gebruik. Beide commando's kunnen worden uitgevoerd via het z/OS operator, hardware, TSO console of via een batch job. Hiervoor dienen de standaard wachtwoorden van deze commando's binnen RACF te worden gewijzigd. De wachtwoorden dienen onder beheer van de Security administrators te vallen en moeten worden veiliggesteld met behulp van bijvoorbeeld een enveloppe procedure. Uitsluitend geautoriseerde personen mogen toegang krijgen tot de enveloppe met de wachtwoorden. De implementatie:

SETROPTS RVARYPW(**SWITCH**(*HFUKSNCE*) **STATUS**(*KDUEFTJD*))

3.13 RACF commando's

RACF commando's dienen met de RACF OPERCMDS class te worden gecontroleerd en uitsluitend toegankelijk te zijn voor geautoriseerde personen. De implementatie voor het afschermen van RACF commando's is beschreven het hoofdstuk 'z/OS faciliteiten'.

3.14 RACF Classes

RACF kan voor veel subsystemen en faciliteiten de beveiligingsprofielen beheren. Hiervoor zijn verschillende zogenaamde resource classes aanwezig. Standaard resource classes zijn de USER, GROUP en DATASET class. Andere classes worden General Resource classes genoemd, voor algemene zogenaamde resource managers als database managers, transactie managers, networking managers, etc. Indien classes niet worden gebruikt dienen deze altijd te worden gedeactiveerd. Een voorbeeld:

SETROPTS **NOCLASSACT**(*ALCSAUTH*)

Opmerking: indien een general resource class wordt gedeactiveerd met NOCLASSACT worden alle general resource classes met hetzelfde POSIT ID gedeactiveerd. Voor informatie over welk POSIT ID elke afzonderlijke general resource class heeft is te vinden in de actieve Class Descriptor Table (CDT). Een voorbeeld van een gelijk POSIT ID is de class TCICSTRN, voor het definiëren van individuele CICS transacties, en de class GCICSTRN, voor het definiëren van groepen CICS transacties. Bij deactivering van GCICSTRN zal ook TCICSTRN worden gedeactiveerd wat waarschijnlijk niet de bedoeling is aangezien de CICS transacties dan niet meer door RACF worden beschermd.

3.15 Class Descriptor Table (CDT)

De Class Descriptor Table wordt gebruik om aan te geven welke classes moeten worden beschermd. Voor IBM producten zijn deze classes reeds in de CDT gedefinieerd. Indien een niet IBM-product met behulp van externe security (RACF) beveiligd moet worden, moet de door de leverancier aangegeven class in de Class Descriptor Table worden opgenomen. Het gebruik van deze niet-standaard classes dient tot het minimum te worden beperkt en alleen te worden toegepast indien geen gebruik gemaakt kan worden van één van de standaard IBM classes.

Bij het gebruik van niet standaard classes dient men namen te gebruiken die niet door IBM zullen worden gebruikt. Aan te raden is de te definiëren class naam met een speciaal karakter te laten beginnen, in de praktijk meestal een $ (dollar) teken. Een voorbeeld: $MYCLAS. Zo gebruiken leveranciers een speciaal teken zoals het @ sign. Een voorbeeld: PR@PANEL.

3.16 RACF parameters

RACF heeft een library waarmee parameters voor RACF-faciliteiten kunnen worden ingesteld.

Indien de RACF parameter library wordt gebruikt dient deze uitsluitend door de Security auditor te kunnen worden benaderd. De implementatie:

```
ADDSD    'SYS1.RACF.PARM' GENERIC OWNER(SECADM) UACC(NONE)
PERMIT   'SYS1.RACF.PARM' ID(SECADM) ACCESS(READ)
```

Opmerking: de dataset naam 'SYS1.RACF.PARM' in dit voorbeeld is een keuze en wordt bepaald in de JCL van de RACF-subsysteem started task procedure. De individuele parametermembers hebben de standaard prefix *IRROPTxx*.

3.17 RACF Subsystem

Met het RACF-subsysteem kunnen de meeste RACF-commando's door middel van operator-commando's worden uitgevoerd. Verder is het RACF-subsysteem de basis voor de RACF Remote Sharing Facility (RRSF) waarmee wachtwoorden, gebruikers- en beveiligingsprofielen met remote RACF-systemen kunnen worden synchroniseerd.

Goede redenen om het RACF-subsysteem te activeren alhoewel het voor de toegangsbeveiliging niet direct noodzakelijk is aangezien RACF blijft functioneren als het subsysteem niet actief is. Indien het RACF-subsysteem wordt gebruikt moet deze voor RRSF de RACF-parameter library en APPC-profielen kunnen benaderen. Dit kan door het privilege of trusted attribuut voor de RACF-subsysteem started task te definiëren. De implementatie:

```
ADDUSER  RACF  DFLTGRP(STCGROUP)  OWNER(SECADM)
         NOPASSWORD

SETROPTS GENERIC(STARTED)
SETROPTS CLASSACT(STARTED)

RDEFINE  STARTED  RACF.*  STDATA(USER(RACF)  GROUP(STCGROUP)
         TRUSTED(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

Opmerking: zie voor verdere informatie over started tasks het hoofdstuk 'Started tasks en started jobs'.

3.18 Dynamic Parser table

De Dynamic Parser table, noodzakelijk voor de interpretatie van segment gerelateerde definities zoals voor het TSO segment, CICS segment, enz. moet na elke IPL van het systeem worden geactiveerd. Dit moet worden uitgevoerd met behulp van parameters in het IRROPTxx member uit de RACF parameter library. Een voorbeeld:

```
ALLOCATE FILE(SYSUT1) DATASET('SYS1.RACF.PARM(IRRDPSDS)') SHR
IRRDPI00 UPDATE
FREE     FILE(SYSUT1)
```

4 Groepen en Gebruikers

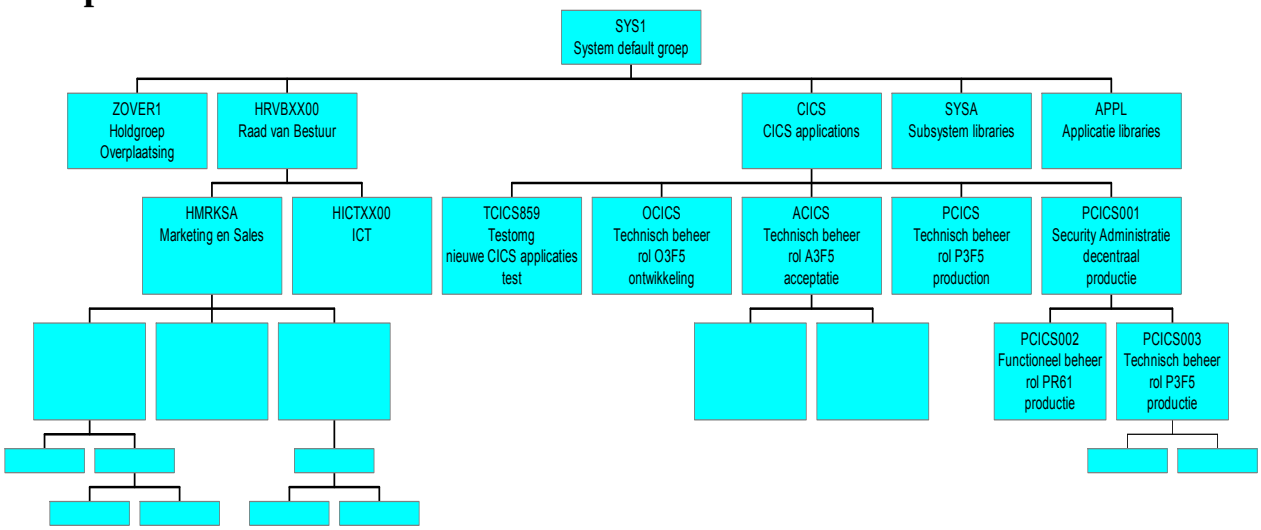
4.1 Introductie

Voor een beheersbare RACF-omgeving is het van groot belang de opzet van de groepenstructuur, waarmee gebruikers kunnen worden gegroepeerd en daarmee toegangsrechten toegekend, helder en vooraf te beschrijven. Dit hoofdstuk beschrijft een aantal mogelijkheden voor de inrichting van de groepenstructuur. Verder geeft het een hulpmiddel door middel van autorisatiematrices om de activiteiten die binnen een organisatie moeten worden uitgevoerd te beschrijven en te vertalen naar RACF-definities noodzakelijk om de gewenste autorisaties te implementeren.

4.2 Status

Dit hoofdstuk is gereed voor commentaar.

4.3 Groepen



Figuur 2. Voorbeeld RACF-groepenstructuur

4.3.1 Introductie groepenstructuur

RACF dwingt af dat gebruikers minimaal in één groep worden ondergebracht. Voor de groepenstructuur geeft RACF verder geen voorkeur. Binnen het raamwerk van deze standaard zijn een 6-tal verschillende groepstructuren onderscheiden die elk een eigen visie hebben op het onderbrengen van gebruikers in groepen en het verlenen van toegang tot de noodzakelijke systeemmiddelen. De standaard gaat er vanuit dat toegang uitsluitend via een groepsautorisatie wordt verleend en gebruikers daarom in principe niet op een toegangslijst voorkomen.

In de volgende paragraaf wordt een beschrijving gegeven van de verschillende soorten RACF-groepen. Hierbij moet worden vermeld dat de soort RACF-groep een aanduiding is waarvoor deze wordt toegepast en niet voor de technische implementatie binnen RACF. Technisch gezien zijn alle soorten groepen gelijk, het gebruik is echter anders gedefinieerd. Na de beschrijving van

de afzonderlijke soorten groepen volgt een mogelijke combinatie van groepen met een argumentatie, die is opgesteld naar 'best practice' binnen verschillende grote ICT-organisaties.

4.3.2 Groepsdefinitie

De omschrijving van groepsprofielen dient helder en eenduidig met het gebruik gerelateerd te zijn. Een voorbeeld:

```
ADDGROUP  PCICS003   DATA('Technisch beheer, rol P3F5')
```

De groepsprofielen dienen een (gedelegeerd) eigenaar (OWNER) te hebben die voor het beveiligingsbeheer van de groeps-resources verantwoordelijk is. In de praktijk zou de (gedelegeerd) eigenaar een willekeurige groepsnaam, zoals bijvoorbeeld SECADM, moeten kunnen zijn. Echter bij een groepsdefinitie dwingt RACF af dat de eigenaar (OWNER) van een groepsprofiel gelijk moet zijn aan de bovenliggende groep (SUPGROUP). Een voorbeeld:

```
ADDGROUP  PCICS003  SUPGROUP(PCICS001)
                    OWNER(PCICS001)  DATA('Technisch beheer, rol A3F5')
```

Bij de definitie van groepsnamen dient er op te worden gelet dat deze niet overeenkomen met gebruikersnamen. Een aanpak kan zijn dat groepsnamen 8 karakters lang worden gemaakt en gebruikersnamen 7 karakters lang. In dit document is rekeninggehouden met de naamgevingconventie van zowel groepsnamen als gebruikersnamen zodat hiermee geen probleem ontstaat. In de volgende paragraaf is een mogelijke groepsnaamgevingconventie beschreven.

4.3.3 Soorten groepen

Voordat groepen worden gedefinieerd dient een naamgevingconventie vastgesteld en gedocumenteerd te worden. Alle groepsnamen dienen aan een vooraf vastgestelde naamgevingconventie te voldoen waarbij tot uiting komt waarvoor een groep (groupid) wordt gebruikt. Zo kunnen de volgende soorten groepen worden onderscheiden:

- Resource-groepen, gelijk aan de naamgevingconventie van de high-level qualifier voor datasets zoals b.v. SYS1, de standaard naam voor z/OS systeem-libraries. De SYS1-groep is reeds standaard binnen RACF aanwezig en hoeft niet te worden gedefinieerd. Een voorbeeld:

  ```
  ALTGROUP   SYS1   DATA('System en subsystem libraries')
  ```

  De resource-groepen dienen direct onder de groep SYS1 te worden gedefinieerd waarbij het aantal niveaus beperkt dient te blijven tot dit niveau. Een voorbeeld:

  ```
  ADDGROUP  SYSA   SUPGROUP(SYS1)   OWNER(SYS1)
            DATA('Subsystem libraries')
  ADDGROUP  APPL   SUPGROUP(SYS1)   OWNER(SYS1)
            DATA('Applicatie libraries')
  ADDGROUP  DB2    SUPGROUP(SYS1)   OWNER(SYS1)
            DATA('DB2 libraries')
  ```

- Hold-groepen, zijn voor het vastleggen van gebruikers die tijdelijk niet tot een andere groep kunnen worden gerekend. Dit kan zijn bij b.v. bij overplaatsing, ontslag, (tijdelijk) arbeidsongeschiktheid, etc. Als er meerdere hold-groepen worden gebruikt hoeven deze geen bepaalde hiërarchie te hebben maar wel een duidelijke plaats in de groepenstructuur. Een voorbeeld voor de naamgevingconventie:
  - één letter (b.v. Z als laatste letter van het alfabet)
  - daarna volgen 4 letters die het type hold aangeven

- afsluitend met een 3-cijferig volgnummer

Een voorbeeld:

ADDGROUP ***Z**OVER001* SUPGROUP(*SYS1*) OWNER(*SECADM*)
DATA(*'Holdgroep voor overplaatsingen'*)

De hold-groepen dienen direct onder de groep SYS1 te worden gedefinieerd waarbij het aantal niveaus beperkt dient te blijven tot dit niveau. Een voorbeeld:

ADDGROUP *ZOVER002* **SUPGROUP(*SYS1*)** OWNER(*SECADM*)
DATA(*'Holdgroep voor overplaatsingen'*)

- Organisatorische groepen, zijn voor het vastleggen van de afdeling waarin de gebruikers participeren. Deze groepen, die uitsluitend toegangsrechten geven tot afdelings-resources zoals b.v. voor interne communicatie, zijn opgebouwd conform de organisatiestructuur en kunnen daardoor een groot aantal niveaus bereiken. Een voorbeeld voor de naamgevingconventie:
  - één letter (b.v. H van Hiërarchie)
  - daarna volgen 5 letters die de afdeling aangeven
  - afsluitend met een 2-cijferig volgnummer

Een voorbeeld:

ADDGROUP ***H**RVBXX00* DATA(*'Raad van Bestuur'*)

De Organisatorische groep van de hoogst geplaatste afdeling in de organisatiehiërarchie dient direct onder de groep SYS1 te worden gedefinieerd. De organisatiehiërarchie bepaalt het aantal niveaus. Een voorbeeld:

ADDGROUP *HRVBXX00* **SUPGROUP(*SYS1*)** DATA(*'Raad van Bestuur'*)
ADDGROUP *HMRKSA00* **SUPGROUP(*HRVBXX00*)** DATA(*'Marketing Sales'*)
ADDGROUP *HMRKPR01* **SUPGROUP(*HMRKSA00*)** DATA(*'Marketing Grp 1'*)
ADDGROUP *HMRKPR02* **SUPGROUP(*HMRKSA00*)** DATA(*'Marketing Grp 2'*)
ADDGROUP *HICTXX00* **SUPGROUP(*HRVBXX00*)** DATA(*'ICT'*)

- Functionele groepen, zijn voor het verstrekken van toegangsrechten conform de functie die een medewerker in de organisatie moet kunnen uitvoeren. Gebruikers worden bij een functionele groepenstructuur in principe aan één groep geconnect. De functionele groepen moeten om de gewenste functie te kunnen uitvoeren toegangsrechten hebben op, voor de functie noodzakelijke, verschillende soorten resources. De toegangsrechten van de functionele groepen moeten zo zijn gedefinieerd dat de gewenste functiescheiding gewaarborgd is tussen o.a.:
  - Functioneel beheer
  - Technisch beheer
  - Operationeel beheer
  - Test, Ontwikkeling, Acceptatie en Productie voor applicatiebeheer/ontwikkeling
  - Test, Acceptatie en Productie voor z/OS-, subsysteem- en tool-beheer
  - Bedrijfsfuncties

Een voorbeeld:

- één letter voor tijdelijk, test, ontwikkeling, acceptatie of productie (respectievelijk b.v. X, T, O, A of P)
- daarna volgen 4 letters die de toepassing aangeven
- daarna 1 letter voor de functietype
- afsluitend met een 2-cijferig volgnummer

Een voorbeeld:

ADDGROUP ***XCICSX86*** DATA(*'Tester, Tijdelijk, nieuwe CICS versie'*)
ADDGROUP ***TCICSX79*** DATA(*'Tester, Test, nieuwe beheer tools'*)
ADDGROUP ***OCICST34*** DATA(*'Technisch beheerder, Ontwikkeling, CICS34'*)
ADDGROUP ***ACICST34*** DATA(*'Technisch beheerder, Acceptatie, CICS34'*)
ADDGROUP ***PCICST45*** DATA(*'Technisch beheerder, Productie, CICS45'*)

De functionele groepen dienen, voor eenduidig beheer, in een hiërarchie te worden ondergebracht waarbij geadviseerd wordt het aantal niveaus te beperken tot maximaal 2 niveaus. De primaire functionele (hoofd)groep dient direct onder de groep SYS1 te worden gedefinieerd. Een voorbeeld:

ADDGROUP *PCICS001* **SUPGROUP(*SYS1*)**
DATA(*'Security Administratie CICS001'*)

ADDGROUP *PCICS002* **SUPGROUP(*PCICS001*)**
DATA(*'Functioneel beheer, rol PR61'*)

ADDGROUP *PCICS003* **SUPGROUP(*PCICS001*)**
DATA(*'Technisch beheer, rol P3F5'*)

Opmerking: Als een gebruiker geconnect is aan hoofdgroep *PCICS001* krijgt hij/zij niet de toegangsrechten van de andere groepen *PCICS002* en *PCICS003*. RACF geeft de gebruiker uitsluitend die toegangsrechten die de groepen hebben waaraan de gebruiker is geconnect. Als de gebruiker in dit voorbeeld de functies *PCICS002* en *PCICS003* moet kunnen uitvoeren zal hij/zij aan beide groepen moeten worden geconnect. De hoofdgroep *PCICS001* wordt in dit voorbeeld als administratieve groep gebruikt om de samenhang tussen de onderliggende groepen vast te leggen en geeft verder geen toegangsrechten.

- Activiteiten- of Rolgroepen, zijn voor het verstrekken van toegangsrechten gerelateerd aan een activiteit of rol, één RACF-groep per onderkende activiteit of rol. Een medewerker voert in het algemeen binnen zijn/haar functie meerdere activiteiten of rollen uit waardoor bij implementatie van activiteiten- of rolgroepen gebruikers aan meerdere groepen moeten worden geconnect om hun functie uit te kunnen voeren. De toegangsrechten die een gebruiker krijgt door connectie aan meerdere activiteiten- of rolgroepen moeten zo worden gedefinieerd dat de gewenste functiescheiding gewaarborgd is.

  Een voorbeeld voor de naamgevingconventie:

  - één letter voor tijdelijk, test, ontwikkeling, acceptatie of productie (respectievelijk b.v. X, T, O, A of P)
  - daarna volgen 4 letters die de toepassing aangeven
  - daarna 1 letter voor het type activiteit
  - afsluitend met een 2-cijferig volgnummer

Een voorbeeld:

ADDGROUP ***XCICSC37*** DATA(*'Alle commandos Tijdelijke CICS37'*)
ADDGROUP ***TCICSC34*** DATA(*'Alle commandos Test CICS34'*)
ADDGROUP ***OCICSD34*** DATA(*'Display commandos Ontwikkel CICS34'*)
ADDGROUP ***ACICSS03*** DATA(*'Starten en Stoppen Acceptatie CICS03'*)
ADDGROUP ***PCICSC45*** DATA(*'Alle commandos Productie CICS45'*)

De activiteitengroepen dienen in een groepshiërarchie te worden ondergebracht waarbij geadviseerd wordt het aantal niveaus te beperken tot maximaal 2 niveaus. De primaire activiteitengroepen dienen direct onder de groep SYS1 te worden gedefinieerd. Een voorbeeld:

ADDGROUP *PCICS**C**34* **SUPGROUP(*SYS1*)**
DATA(*'Alle commandos Productie CICS34'*)

ADDGROUP *PCICS**D**34* **SUPGROUP(*PCICSC34*)**
DATA(*'Display commandos CICSP34'*)

ADDGROUP *PCICS**S**34* **SUPGROUP(*PCICSC34*)**
DATA(*'Starten en Stoppen CICSP34'*)

Opmerking: RACF geeft de gebruiker uitsluitend die toegangsrechten die de groepen hebben waaraan de gebruiker is geconnect. Om de juiste toegangsrechten te krijgen die bij zijn/haar functie behoren moet de gebruiker daarom veelal aan meerdere activiteiten- of rolgroepen worden geconnect. In het voorgaande voorbeeld zijn dit bijvoorbeeld de groepen *PCICSD34* en *PCICSS34*. De hoofdgroep *PCICSC34* wordt in dit voorbeeld als administratieve groep gebruikt om de samenhang tussen de onderliggende groepen vast te leggen en heeft verder geen toegangsrechten.

- Combinatiegroepen (hoofdgroepen), worden gebruikt om verschillende activiteiten- of rolgroepen te bundelen tot functionele groepen, hierbij rekeninghoudend met de gewenste functiescheiding. Toegangsrechten verlenen via combinatiegroepen is binnen RACF niet mogelijk aangezien een gebruiker geen toegangsrechten krijgt van de ene naar de andere groep (geen overerving). Een connect aan een combinatiegroep geeft geen toegangsrechten tot de hiërarchisch gekoppelde activiteiten- of rolgroepen. Om rechten te krijgen die een activiteiten- of rolgroep geeft zal de gebruiker aan deze groepen moeten worden geconnect. Combinatiegroepen kunnen om administratieve redenen worden gedefinieerd om zo de samenhang tussen de verschillende groepen vast te leggen en het beheer inzichtelijker te maken.

  Bij de opbouw van combinatiegroepen krijgt men een hiërarchie van drie niveaus. De SYS1-groep, de combinatiegroepen en de activiteiten- en/of rol groepen. Deze hiërarchie is binnen RACF toe te passen, echter als de organisatie deze opzet ook binnen andere besturingssystemen wil toepassen, waar geen RACF aanwezig is, zal over het algemeen niet van een groepenstructuur van 3 niveaus gebruik kunnen worden gemaakt. Bijvoorbeeld bij UNIX- of Linux-systemen inclusief z/OS Unix System Services (USS) is deze 3 niveau structuur niet mogelijk.

  Verder is het gebruik van combinatiegroepen binnen RACF niet aan te raden omdat in een groepsprofiel, dus ook bij een activiteiten- of rolgroepprofiel, uitsluitend één Superior group te definiëren is. In de praktijk behoort één activiteit of rol vaak bij meerdere functies dus bij meerdere combinatiegroepen. Om de activiteiten- of rolgroep met meerdere combinatiegroepen te kunnen verbinden moeten bij de activiteitengroep meerdere Superior groups kunnen worden gedefinieerd, hetgeen door RACF technisch niet wordt ondersteund.

Producten in de markt die Role Based Access Control ondersteunen geven veelal de mogelijkheid om combinatiegroepen te definiëren en connecten de gebruiker conform de definities van de combinatiegroep aan de verschillende activiteiten- of rolgroepen.

Een voorbeeld voor de naamgevingconventie:

- één letter voor tijdelijk, test, ontwikkeling, acceptatie of productie (respectievelijk b.v. X, T, O, A of P)
- daarna één letter om aan te geven dat het om een combinatiegroep gaat (bijvoorbeeld 'X')
- daarna volgen 3 letters die de toepassing aangeven
- daarna 1 letter voor de functietype
- afsluitend met een 2-cijferig volgnummer

Een voorbeeld:

```
ADDGROUP  XXHYPV45    DATA('View alle Hypotheken Tijdelijke omgeving')
ADDGROUP  TXHYPU44    DATA('Update alle Hypotheken Testomgeving')
ADDGROUP  OXHYPA34    DATA('Accepteer alle Hypotheken Ontwikkel')
ADDGROUP  AXHYPI23    DATA('Inbrengen Hypotheken Acceptatie')
ADDGROUP  PXHYPO54    DATA('Opheffen Hypotheken Productie')
```

- Een started task groep, die voor één of meerdere started tasks worden gebruikt dienen zoveel mogelijk herkenbaar te zijn. Een voorbeeld voor de naamgevingconventie waarbij het groepid begint met een 'S' van started en eindigt op een 'GRP' van groep:

```
ADDGROUP  SFTPGRP1  DATA('Started Task voor FTP')
```

4.4 Selectie groepenstructuur

De hiervoor beschreven groepen kunnen in combinatie worden toegepast. Verder kan een selectie worden gemaakt van de verschillende soorten groepen. Hieronder is een mogelijke invulling gegeven met daarbij de argumentatie voor de keuze.

Organisatiestructuur	**Statische**		**Dynamische**	
Activiteiten / **soort groep**	**Veranderende**	**Statische**	**Veranderende**	**Statische**
Resource	Altijd	Altijd	Altijd	Altijd
Hold	Optioneel	Optioneel	Optioneel	Optioneel
Organisatorische	Optioneel	Optioneel	Optioneel	Optioneel
Functionele	Optioneel	Aanbevolen	Optioneel	Optioneel
Activiteiten	Aanbevolen	Optioneel	Aanbevolen	Aanbevolen
Combinatie	Niet aanbevolen*	Niet aanbevolen*	Niet aanbevolen*	Niet aanbevolen*

Opmerking: ***** *= tenzij een Role Based Access Control product wordt toegepast dat dit ondersteunt.*

Figuur 3. Keuzematrix groepenstructuur

De selectiecriteria voor de te implementeren soorten groepen zijn:

- Organisatorisch:
  - Als verwacht mag worden dat de organisatiestructuur regelmatig zal veranderen, spreekt men in deze context van een dynamische organisatiestructuur.
  - Als verwacht mag worden dat de organisatie nagenoeg niet zal veranderen, spreekt men in deze context van een statische organisatiestructuur. Gedacht kan worden aan een toekomst vaste organisatiestructuur, waarbij het te verwachten is dat uitsluitend marginale aanpassingen zullen plaatsvinden.
- Activiteiten:
  - Als verwacht wordt dat de huidige activiteiten sterk zullen veranderen of anders in de organisatie zullen worden ondergebracht, spreekt men in deze context van veranderende activiteiten.
  - Als verwacht mag worden dat de huidige activiteiten nagenoeg niet zullen veranderen of niet anders in de organisatie zullen worden ondergebracht, spreekt men in deze context van statische activiteiten.

De hierboven beschreven tabel geeft de aanbevolen combinatie aan van de soorten groepen en soorten organisaties. Overwogen kan worden om functionele groepen en activiteitengroepen binnen één organisatie te combineren afhankelijk van het organisatieonderdeel. Wel dient duidelijk te zijn welke soort groep wordt toegepast zodat de éénduidigheid gewaarborgd blijft.

De inspanning die geleverd moet worden voor de opzet en beheer van een groepsstructuur zal sterk afhankelijk zijn van de keuze. Zo kost het opzetten van een functionele groepsstructuur verhoudingsgewijs meer tijd dan van een activiteiten groepsstructuur.

Alhoewel dit afhankelijk is van de soort organisatie kan worden gezegd dat een functionele groep complexer is omdat meerdere activiteiten door één groep moeten kunnen worden uitgevoerd en daarvoor meerdere toegangsrechten moeten worden gedefinieerd. Meestal worden dezelfde activiteiten door meerdere groepen uitgevoerd en moeten dus in meerdere groepen worden gedefinieerd. Hieruit volgt dat toegangsrechten tot één resource veelal in meerdere functionele groepen moeten worden gedefinieerd wat een meer spaghettiachtige structuur geeft. Bij verandering zoals een naamswijziging, verwijdering, toevoeging, etc. van een resource moeten de toegangsrechten daarom bij meerdere functionele groepen worden aangepast. Indien van activiteitengroepen gebruik wordt gemaakt zullen de toegangsrechten meestal in één enkele groep moeten worden aangepast. Verder kan worden gezegd dat functies, afhankelijk van de organisatie, in een veranderende (dynamische) omgeving veelal inhoudelijk veranderen terwijl de uitgevoerde activiteiten nagenoeg stabiel zijn. Activiteit moet hoe dan ook worden uitgevoerd afgezien van de opzet van een organisatie.

Echter, het gebruikersbeheer van een activiteiten groepsstructuur kost, zonder ondersteunende beheermiddelen, verhoudingsgewijs meer tijd dan van het opzetten van een functionele groepsstructuur. Onder gebruikersbeheer wordt verstaan het definiëren en verwijderen van userids en het toevoegen en verwijderen van userids aan groepen. De volgende tabel geeft de relatieve inspanningen aan voor de opzet en beheer van de groepenstructuur en het gebruikersbeheer.

	Opzet groepsstructuur	**Gebruikersbeheer en beheer groepsstructuur in een statische organisatiestructuur**	**Gebruikersbeheer en beheer groepsstructuur in een dynamische organisatiestructuur**
Functionele groepen	Veel inspanning	Weinig inspanning	Zeer veel inspanning
Activiteiten groepen	Weinig inspanning	Weinig inspanning	Veel inspanning

Figuur 4. Inspanningsmatrix groepsstructuur

4.5 User management applicatie

Voor het definiëren van gebruikers en het toekennen van meerdere activiteiten op basis van activiteitengroepen is het raadzaam een ondersteunende applicatie (tool) te implementeren. Deze toepassing moet de functiescheiding waarborgen door uitsluitend die activiteiten voor een gebruiker te definiëren (connecten) die door één persoon of één afdeling gelijktijdig mogen

worden uitgevoerd. Zo dient deze toepassing de functiescheiding te waarborgen door de juiste combinatie van activiteiten voor een gebruiker te definiëren voor o.a.:

- Functioneel beheer
- Technisch beheer
- Operationeel beheer
- Test, Ontwikkeling, Acceptatie en Productie voor applicatiebeheer/ontwikkeling
- Test, Acceptatie en Productie voor z/OS-, subsysteem- en tool-beheer
- Bedrijfsfuncties

Dit soort applicaties zijn standaard in de markt te verkrijgen en hebben vaak een user interface die door niet-RACF-kenners kan worden toegepast. Door dit soort applicaties kan de inspanning, voor het gebruikersbeheer van activiteitengroepen, worden teruggebracht tot dezelfde inspanning als nodig bij functionelegroepen. Door deze opzet, gebruikmakend van activiteitengroepen en een ondersteunende applicatie (tool), heeft men en het voordeel van de functionele groepen (één handeling per gebruiker voor de toekenning van autorisaties) en van de eenvoudige opzet en onderhoud van activiteitengroepen. De volgende tabel geeft de relatieve inspanningen aan voor de opzet en beheer van de groepenstructuur en het gebruikersbeheer.

	Opzet groepsstructuur	**Gebruikersbeheer en beheer groepsstructuur in een statische organisatiestructuur**	**Gebruikersbeheer en beheer groepsstructuur in een dynamische organisatiestructuur**
Functionele groepen	Veel inspanning	Weinig inspanning	Zeer veel inspanning
Activiteiten groepen met ondersteunende applicatie	Weinig inspanning	Weinig inspanning	Weinig inspanning

Figuur 5. Inspanningsmatrix met User management applicatie

4.6 Autorisatie Matrix gebaseerd op activiteiten groepen

Voor het op een gestructureerde wijze uitgeven van toegangsrechten en voor een systematische opzet van de toegangsrechten dient de organisatie vooraf een mechanisme (matrix) gedefinieerd te hebben met de rollen en activiteiten die door de verschillende medewerkers moeten worden uitgevoerd. Zo'n matrix wordt een Autorisatie Matrix genoemd.

Hieronder een voorbeeld van de fase 1 Autorisatie Matrix. In deze matrix zijn een aantal verschillende rollen en virtuele activiteiten gedefinieerd. De activiteiten in de matrix hoeven niet uitsluitend aan computer systemen te zijn gerelateerd maar kunnen activiteiten inhouden zoals het volgen van overlegvergaderingen, toegang tot een gebouw of ruimte, het mogen uitgeven van geld, enz. Bij het definiëren van de Autorisatie Matrix dient er rekening mee te worden gehouden dat activiteiten de noodzakelijke functiescheiding niet doorbreken.

Rol / **Activiteit**	**Klerk**	**lokale Coördinator**	**Kas Controleur**	**Security Auditor**	**Senior Operator**	**Etc.**
ACTIVITEIT 1	X					
ACTIVITEIT 2	X	X				
ACTIVITEIT 3	X		X			
ACTIVITEIT 4					X	
ACTIVITEIT 5				X		
ACTIVITEIT 6					X	
ACTIVITEIT 7	X	X	X	X	X	X
ACTIVITEIT 8				X	X	

Figuur 6. Autorisatie matrix fase 1

De fase 1 Autorisatie Matrix definieert in principe alle activiteiten die noodzakelijk zijn voor de dagelijkse uitvoering van de bedrijfsprocessen en is daarom direct een inventarisatie hiervan. Indien activiteiten moeten worden uitgevoerd door computersystemen dienen deze door de Security administrator te worden gedefinieerd in het systeem. Hiervoor dient een vertaling van activiteiten naar beveiligingsdefinities te worden gemaakt.

Voor de meeste computersystemen kan de toegang tot systeemmiddelen zoals, datasets, files, printers, transacties, databases, enz. via gebruikers- of groepsautorisatie worden gegeven. Wij raden echter aan groepsautorisaties te gebruiken en in principe nooit autorisaties per individuele gebruiker te geven. Deze stelling vereist dat gebruikers lid zijn van één of meerdere groepen en

dat deze groepen de noodzakelijke autorisatie op de systeemmiddelen krijgen. Het uitgangspunt is hierbij dat één groep de toegangsrechten implementeert voor één activiteit. De implementatie van de relatie RACF-groep en activiteit wordt in de fase 2 Autorisatie Matrix weergegeven.

Groep / Activiteit	GINQ12	GMNG4	GCICS39	GSECADM1	GOPRT1	GOPRS3	Etc.
ACTIVITEIT 1	X						
ACTIVITEIT 2		X					
ACTIVITEIT 3			X				
ACTIVITEIT 4					X		
ACTIVITEIT 5							
ACTIVITEIT 6						X	
ACTIVITEIT 7				X			
ACTIVITEIT 8							

Figuur 7. Autorisatie matrix fase 2

Nadat de fase 2 Autorisatie Matrix is gedefinieerd moeten de RACF-groepen worden geautoriseerd (permit) voor toegang tot de systeemmiddelen. De groep moet de juiste toegangsrechten (aanmaken, lezen, wijzigen en verwijderen) tot de systeemmiddelen krijgen om de betreffende activiteit correct op het computer systeem te kunnen uitvoeren. Hieronder enige voorbeelden van PERMITs voor specifieke systeemmiddelen:

Voor activiteit 1:

```
PERMIT    PRODTRN*  CLASS(TCICSTRN)  ID(GINQ12)   ACCESS(READ)
```

Voor activiteit 2:

```
PERMIT    PTRNMNG*  CLASS(TCICSTRN)  ID(GMNG4)   ACCESS(READ)
```

Voor activiteit 3:

```
PERMIT    PTRNUPD*  CLASS(TCICSTRN)  ID(GCICS39)  ACCESS(READ)
```

Voor activiteit 4:

```
PERMIT    CUST94  CLASS(TAPEVOL)   ID(GOPRT1)     ACCESS(ALTER)
```

Voor activiteit 6:

```
PERMIT    OPSUB1.SUBMIT  CLASS(SURROGAT)
```

```
                ID(GOPRS3) ACCESS(READ)
PERMIT      'OPER.PROD.JCLLIB'          ID(GOPRS3) ACCESS(READ)
PERMIT      'OPER.PROD.OUTPUT*.**'      ID(GOPRS3) ACCESS(UPDATE)

PERMIT      CICSCMD1 CLASS(GCICSTRN) ID(GOPRS3) ACCESS(READ)
```

Voor activiteit 7:

```
PERMIT      ICHDSM00 CLASS(PROGRAM)
                ID(GSECADM1) ACCESS(EXECUTE)
PERMIT      'SYS1.SMFDUMP*.**   ID(GSECADM1) ACCESS(READ)
```

Nadat de activiteitengroepen toegang tot de betreffende systeemmiddelen zijn gegeven door middel van permits dienen de gebruikers lid te worden (geconnect) van de activiteitengroep(en) om hun taken op de systemen te kunnen uitvoeren. Bijvoorbeeld een Senior Operator moet aan de groep GOPRT1 en GOPRS3 worden geconnect om zijn/haar taken in het systeem te kunnen uitvoeren. Om inzichtelijk te maken welke gebruikers (medewerkers) welke rollen uitvoeren dient de fase 3 Autorisatie Matrix te worden gedefinieerd. Een voorbeeld is hieronder gegeven:

Rol / Medewerkers		**Klerk**	**Locatie Coördinator**	**Kas Controleur**	**Security Auditor**	**Senior Operator**	**Etc.**
Name	User ID						
F. Engel	OENGF01				X		
P. Mienes	OMIEP02					X	
T. Krens	OKRET01	X					
L. van Rij	ORIJL00			X			
Etc.	Etc.						X

Figuur 8. Autorisatiematrix fase 3

Voor het toekennen van toegangsrechten moeten gebruikers deel uitmaken van één of meerdere activiteitengroepen. Dit gebeurt in RACF met CONNECT commando's waarmee de gebruiker lid wordt gemaakt van een activiteitengroep en daardoor de juiste autorisatie krijgt. In een later hoofdstuk wordt het CONNECT commando behandeld. Ter illustratie wordt hieronder een voorbeeld gegeven gerelateerd aan de Autorisatiematrix fase 3:

```
CONNECT  XENGF01   GROUP(GSECA1)  OWNER(HRMCENTR)

CONNECT  XMIEP01   GROUP(GOPRS3)  OWNER(HRMCENTR)
CONNECT  XMIEP01   GROUP(GOPRT1)  OWNER(HRMCENTR)

CONNECT  XKRET01   GROUP(GCICS39)  OWNER(HRMCENTR)

CONNECT  XRIJL01   GROUP(GINQ12)  OWNER(HRMCENTR)

CONNECT  XRIJL01   GROUP(GMNG4)  OWNER(HRMCENTR)
```

4.6.1 Conclusie

Autorisatie Matrices zijn in alle drie de fasen systeem onafhankelijk en kunnen daarom worden toegepast voor een beveiligingsimplementatie in elk willekeurig systeem, zelfs al verschilt de technische architectuur.

4.7 Functiescheiding

Zoals beschreven in het hoofdstuk Inleiding, paragraaf Beveiligingsfunctionarissen, zijn er drie verschillende beveiligingsfuncties aanwezig. Namelijk:

- Security officer, verantwoordelijk voor de aanpak en keuzes van de beveiliging en beveiligingsimplementatie conform het geldende ICT-beveiligingsbeleid;
- Security administrator, zorgt voor de dagelijkse verwerking van autorisatieaanvragen en voor de definities in de RACF-database conform de normen en richtlijnen opgesteld door de Security officer. Deeltaken kunnen worden gedelegeerd naar decentrale Security administrators;
- Security auditor, zorgt voor de dagelijkse controle op de beveiligingsimplementatie en -beheer door middel van monitoring en vervaardigde rapportages. Deze functie kan worden gedelegeerd naar decentrale Security auditors.

Verder moet in de beveiligingsomgeving met bijvoorbeeld de volgende functies rekening worden gehouden:

- Systeem en Applicatie programmeurs, voor software wijzigingen en installaties;
- Operators, voor het dagelijks beheer van het systeem.

In de beveiligingsomgeving moet tussen de verschillende taken (activiteiten) een functiescheiding aanwezig zijn. Bijvoorbeeld: een Security auditor mag niet in staat zijn toegangsrechten te veranderen en de Security administrator mag niet in staat zijn auditing parameters te veranderen. Verder, de Systeem Programmeur en Operator mogen geen toegangsrechten en auditing parameters kunnen veranderen. Een voorbeeld van scheiding tussen verschillende functies is in de onderstaande tabel weergegeven. De tabel definieert medewerkersfuncties die wel of niet binnen één persoon of één groep mogen worden verenigd om de vereiste functiescheiding te implementeren.

	Security officer	**Security administrator**	**Security auditor**	**Systeem programmeur**	**Operator**
Security officer taken	Toegestaan	Toegestaan	Toegestaan	Toegestaan	Toegestaan
Security administrator taken	Toegestaan*	Toegestaan	Niet toegestaan	Niet toegestaan	Niet toegestaan
Security auditor taken	Toegestaan*	Niet toegestaan	Toegestaan	Niet toegestaan	Niet toegestaan
Systeem programmeur taken	Toegestaan*	Niet toegestaan	Niet toegestaan	Toegestaan	Niet toegestaan
Operator taken	Toegestaan*	Niet toegestaan	Niet toegestaan	Niet toegestaan	Toegestaan

Figuur 4. Functiescheidingsmatrix

Opmerking: De asterisk (*) in de kolom van Security officer geeft aan dat hooguit één van deze taken extra mogen worden uitgevoerd. Voorbeeld: de Security officer mag naast zijn/haar eigen werkzaamheden de taken uitvoeren van een Systeem programmeur, maar dan niet die van een Security administrator.

4.8 Gebruikers

4.8.1 Naamgevingconventie

Alle gebruikersidentificaties dienen aan een vooraf vastgestelde naamgevingconventie te voldoen. Indien een gebruiker jobs moet submitten is het aan te bevelen dat het userid maximaal 7 karakters lang wordt gemaakt. Voor gebruikers die gebruik willen maken van TSO is een lengte van maximaal 7 karakters noodzakelijk. Hierdoor kan de gebruiker de jobnaam laten beginnen met het userid met toevoeging van een willekeurig karakter. Door gebruik van verschillende toegevoegde karakters kunnen meerdere jobs van één gebruiker parallel worden uitgevoerd. Een naamgevingconventie voorbeeld:

- voor medewerkers in vaste dienst die eindgebruikers zijn begint het userid met een U (Users), van inhuurmedewerkers die eindgebruikers zijn met een X (eXtern), voor medewerkers in vaste dienst van het rekencentrum met een O (Operaties), voor inhuurmedewerkers van het rekencentrum met een Y (tja je moet iets)
- daarna volgen de eerste 3 letters van zijn/haar achternaam
- vervolgens de eerste letter van zijn/haar initialen
- afsluitend met een volgnummer van 2 cijfers

Een voorbeeld:

ADDUSER ***XKRET01***

De eigennaam van de gebruiker dient helder en eenduidig met het userid gerelateerd te zijn. De implementatie:

ADDUSER *XKRET01* **NAME**(*'T. Krens'*)

Userids die voor een started task worden gebruikt dienen zoveel mogelijk herkenbaar te zijn. Een naamgevingconventie voorbeeld waarbij het userid begint met een 'S' van started en eindigt op een 'U' van user:

ADDUSER ***SFTPU***

Een andere mogelijkheid voor een naamgevingconventie is om gebruik te maken van volledig geanonimiseerde userids. Bij deze userids is er geen enkel verband tussen het userid en de naam van de gebruiker. Als userid wordt een volgnummer gekozen dat willekeurig aan een gebruiker wordt toegekend, of bijvoorbeeld het personeelsnummer. Hierbij is het wel wenselijk dat zichtbaar is of het een medewerker in vaste dienst betreft (volgnummer begint bijvoorbeeld met een U) of een inhuurmedewerker (volgnummer begint bijvoorbeeld met een X):

ADDUSER ***U347895*** voor een medewerker in vaste dienst
ADDUSER ***X873471*** voor een inhuurmedewerker

4.8.2 Gebruikersbeheer

De gebruikersprofielen dienen een (gedelegeerd) eigenaar te hebben die het beheer kan voeren over de eigenschappen van het gebruikersprofiel. De implementatie:

ADDUSER *XKRET01* NAME(*'T. Krens'*) **OWNER**(*HRMCENTR*)

Indien een gebruiker op tijdelijke basis toegang heeft tot het systeem of indien het dienstverband/contract met de medewerker afloopt dient een automatische blokkering op het zijn/haar userid te worden aangebracht. Er dient te worden gezorgd dat deze datum beschikbaar blijft, ook al raakt het userid geblokkeerd. Bij een resume van het userid dient de datum weer te worden gedefinieerd. Een voorbeeld:

ADDUSER *XKRET01* NAME(*'T. Krens'*) OWNER(*HRMCENTR*)
REVOKE(*12/31/00*)

AANVULLENDE MAATREGEL. De mogelijkheid tot logon dient per gebruiker te worden beperkt tot zijn/haar gebruikelijke werktijden met enige marge. Hieronder een voorbeeld waarbij de gebruiker van maandag tot en met vrijdag tussen 8.00 uur en 18.00 uur mag aanloggen. Indien de gebruiker niet aflogt blijft de sessie actief ook al is dit buiten de gedefinieerde tijden. Deze maatregel zorgt wel voor extra beheerinspanningen in geval van bijvoorbeeld overwerk. De implementatie:

ADDUSER *XKRET01* NAME(*'T. Krens'*) OWNER(*HRMCENTR*)
REVOKE(*12/31/00*) **WHEN**(DAYS(*WEEKDAYS*) TIME(*0800:1800*))

4.8.3 Speciale attributen

Voor de gewenste functiescheiding mag een gebruiker slechts één van de twee privileges SPECIAL of AUDITOR als attribuut hebben op groep- of systeemniveau. Privileges zijn hiermee 'mutual exclusive' zowel op groeps- als op systeemniveau.
Een voorbeeld: Indien een gebruiker system-SPECIAL heeft dan mag deze gebruiker geen group-AUDITOR hebben. Als iemand group-SPECIAL heeft dan mag deze gebruiker geen system-AUDITOR hebben. Een voorbeeld:

ADDUSER *XKRET01* NAME(*'T. Krens'*) OWNER(*HRMCENTR*) **SPECIAL**

```
ADDUSER  XRIJL01   NAME('L. van Rij')  OWNER(HRMCENTR)  AUDITOR
```

Binnen een installatie mag maar een gelimiteerd aantal (max. 5) gebruikers met het SPECIAL attribuut gedefinieerd zijn. Deze gebruikers zijn:

- Security Administrators (max. 4)
- Calamiteiten envelop (max. 1)

Binnen een installatie mag maar een gelimiteerd aantal (max. 2) gebruikers met het AUDITOR attribuut gedefinieerd zijn. Deze gebruikers zijn:

- Security Auditors (max. 2)
- ICT auditor (max. 1 op tijdelijke basis, tijdens een onderzoeksfase)

Het privilege OPERATIONS is niet toegestaan (en niet meer noodzakelijk in een modern systeem met de juiste faciliteiten) en zal, indien aanwezig moeten worden verwijderd. De implementatie:

```
ALTUSER   XOUD007   NAME('W. Vergeten')  OWNER(HRMCENTR)
          NOOPERATIONS
```

4.8.4 Decentraal beheer

Binnen een installatie mogen meerdere gebruikers zijn met group-SPECIAL, maar binnen een groep dient een gelimiteerd aantal (max. 2) gebruikers met group-SPECIAL gedefinieerd te zijn. Deze gebruikers zijn de:

- Decentrale Security Administrators (max. 2)

Voor definities zie hoofdstuk ***Connects***

Binnen een installatie mogen meerdere gebruikers zijn met group-AUDITOR, maar binnen een groep dient een gelimiteerd aantal (max. 2) gebruikers met group-AUDITOR gedefinieerd te zijn. Deze gebruikers zijn de:

- Decentrale Security Auditors (max. 2)

Voor definities zie hoofdstuk ***Connects***

Het group-OPERATIONS attribuut is niet toegestaan (en niet meer noodzakelijk in een modern systeem met de juiste faciliteiten) en zal, indien aanwezig moeten worden verwijderd. De implementatie:

Voor definities zie hoofdstuk ***Connects***

Indien een groep gebruikers, b.v. de HRM-afdeling, verantwoordelijk is voor het toevoegen van gebruikers en het uitgeven van autorisaties aan die gebruikers voor de uitvoering van hun functie (role based access control) dient deze groep gebruikers uitsluitend de autorisatie te hebben voor:

- het toevoegen van gebruikersprofielen en
- het koppelen van gebruikers aan de noodzakelijke functie groep. De implementatie:

```
ADDUSER  OHRMP01   DFLTGRP(HRMCENTR)  CLAUTH(USER)
CONNECT  OHRMP01   GROUP(PCICS001)  OWNER(HRMCENTR)
         AUTHORITY(JOIN)
CONNECT  OHRMP01   GROUP(PCICS002)  OWNER(HRMCENTR)
         AUTHORITY(JOIN)
```

Indien beveiligingstaken gedelegeerd worden, dient de gedelegeerde gebruiker niet meer dan de minimale technische mogelijkheden te hebben voor het uitvoeren van zijn/haar taken (rol). Een voorbeeld hiervan is het toevoegen van tapes aan de TAPEVOL class voor verwerking door de afdeling Operations. Zie ook voorgaande norm voor het toevoegen van gebruikers. De implementatie:

ADDUSER *OBRIO01* DFLTGRP(*OPERGRP1*) **CLAUTH(TAPEVOL)**

4.8.5 Gebruikerssegmenten

Segmenten voor gebruikers- en groepsprofielen mogen uitsluitend worden gedefinieerd als hiervoor een technische noodzakelijkheid aanwezig en mag alleen worden toegepast voor die userids en groepen die er werkelijk gebruik van mogen maken. Een voorbeeld voor het Work attribute en TSO-segment:

ADDUSER *XKRET01* NAME('*T. Krens*') OWNER(*HRMCENTR*)
WORKATTR(WAACCNT(*ACTB14*)
WAADDR1(*Diamant*) WAADDR2() WAADDR3() WAADDR4()
WABLDG(*VH4*) WADEPT(*Delivery*) WANAME('*M.Y. Self*')
WAROOM(*E121*))

TSO(ACCTNUM(*IRMB6*) COMMAND(*ISPF*) DEST(*AMST1*)
HOLDCLASS(*H*) JOBCLASS(*B*) MAXSIZE(*4000*)
MSGCLASS(*M*) PROC(*TSOPROCA*) SIZE(*4000*)
SYSOUTCLASS(*P*) UNIT(*TSOALLOC*))

Zie de hierna volgende hoofdstukken voor het OMVS (UNIX System Services), CICS, DCE, DFP, LNOTES, DNS, NETVIEW en OPERPARM segment.

4.9 Started tasks en started jobs

Started tasks (STC) dienen onder een userid te draaien die niet voor logon kan worden gebruikt. Voorbeeld van een userid voor een CICS started task.

ADDUSER *CICS5* DFLTGRP(*STCGROUP*) OWNER(*SECADM*) **NOPASSWORD**

De Started class dient te worden geactiveerd als een GENERIC class. De implementatie:

SETROPTS GENERIC(**STARTED**)
SETROPTS CLASSACT(**STARTED**)
Started tasks dienen uitsluitend dat privilege te hebben dat noodzakelijk is voor een correcte werking. Een aantal voorbeelden:

Voor IMS:

RDEFINE STARTED *IMS.** **STDATA**(USER(*IMSA*) GROUP(*STCGROUP*))

Voor JES2:

RDEFINE STARTED *JES2.** **STDATA**(USER(*JES2*) GROUP(*STCGROUP*)
TRUSTED(**YES**))

SETROPTS RACLIST(**STARTED**) REFRESH

Voor started tasks moet in de general resource class STARTED een generieke definitie worden gemaakt waarbij een systeem boodschap wordt gegenereerd als deze definitie wordt gebruikt. Deze definitie, zoals hieronder beschreven, gebruikt RACF als geen specifieke definitie voor de started task aanwezig is. De implementatie:

```
RDEFINE STARTED  **  STDATA(USER(=MEMBER) GROUP(STCGROUP)
TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

4.10 Surrogate userids

Shared userids mogen niet worden gebruikt. Het gebruik van een userid is dus specifiek voorbehouden aan diegene die in het bezit is van het bijbehorende password. RACF geeft echter de mogelijkheid om in batchjobs de identificatie en daardoor de autorisaties van een userid (het executing userid) te laten gebruiken door een daartoe geautoriseerd userid (het surrogate userid). Elke gebruiker die geautoriseerd is voor het gebruik van een executing userid en hiervan gebruik wil maken dient zich eerst met zijn/haar eigen userid op het systeem te identificeren en te authenticeren. Hierna kunnen werkzaamheden (jobs) worden uitgevoerd onder het 'executing userid' met de betreffende rechten. Een job kan onder een ander 'executing' userid worden uitgevoerd door bijvoorbeeld in de job control language op te geven USER=OPERSUB1. Door het opgeven van PASSWORD= in de jobkaart kan de job worden gestart onder dat userid. Echter, dit is een ongewenste situatie omdat hiervoor een 'clear text passwords' in de jobkaart opgenomen moet worden. In plaats hiervan dient gebruik te worden gemaakt van de general resource class SURROGAT waarin een gebruiker wordt gedefinieerd die van een 'executing userid' gebruiken mag maken.

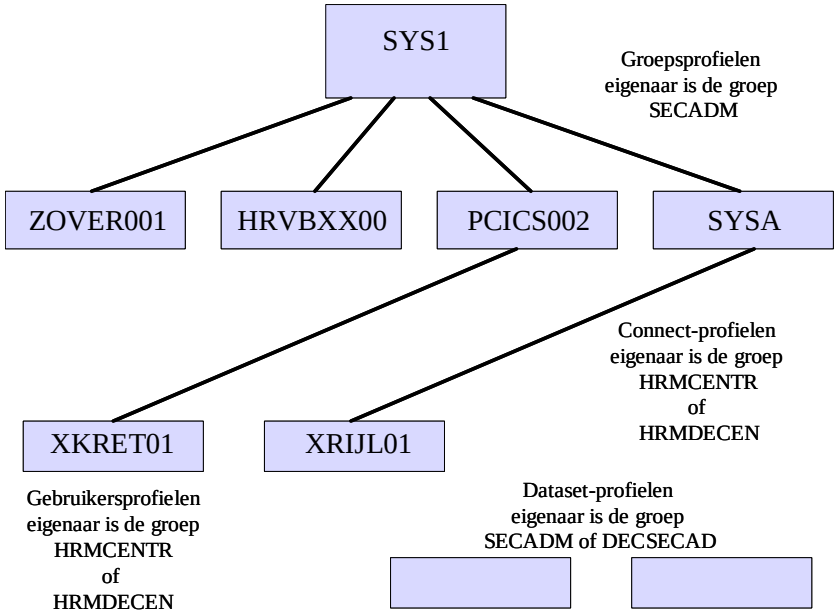
Het 'executing userid' kan gezien worden als een vorm van een 'shared userid'. Indien het 'executing userid' uitsluitend wordt gebruikt voor operationele taken en het niet noodzakelijk is dat hiermee moet worden ingelogd dient het wachtwoord door Security administration te zijn gedefinieerd en mag het niet in welke vorm dan ook worden bewaard.

Door deze aanpak blijft de navolgbaarheid van het gebruik van een 'shared userid' naar een individuele gebruiker gewaarborgd. Wel dient uitsluitend een selecte groep gebruikers en alleen indien noodzakelijk voor de uitoefening van hun taken (rol) geautoriseerd te zijn om zo'n 'controlled shared userid' te gebruiken. Tevens mag d.m.v. het gebruik van deze 'controlled shared userids' geen privileges zoals SPECIAL, OPERATIONS (OPERATIONS mag nooit worden toegepast, zie paragraaf Speciale attributen) of AUDITOR worden verkregen. Aangezien niemand het wachtwoord van het 'executing' userid mag kennen moet het betreffende userid een wachtwoord hebben dat niet behoeft te worden veranderd. Een voorbeeld voor een 'executing' userid te gebruiken in de SURROGAT class:

```
ADDGROUP  OPERSURR  SUPGROUP(SYS1)  OWNER(SECADM)

ADDUSER  OPERSUB1  DFLTGRP(OPERSURR)  OWNER(SECADM)
                                RESTRICTED

ALTUSER  OPERSUB1  PASSWORD(GEHEIM)  NOEXPIRED

PASSWORD  OPERSUB1  NOINTERVAL


PERMIT    'SYS1.PROCLIB.TEST'  ID(OPERSUB1)  ACCESS(UPDATE)

RDEFINE   SURROGAT OPERSUB1.SUBMIT UACC(NONE)
          OWNER(OPERGRP1)
PERMIT    OPERSUB1.SUBMIT CLASS(SURROGAT)
          ID(XMIEP01) ACCESS(READ)
```

```
SETROPTS CLASSACT(SURROGAT)
```

4.11 Connects



Figuur 3. Eigenaarschap van RACF resources

Gebruikers mogen uitsluitend aan die groepen worden toegevoegd die noodzakelijk zijn voor de uitvoering van zijn of haar functie. De connects aan één of meerdere groepen en daarmee het kunnen uitvoeren van één of meerdere rollen mag de functiescheiding niet doorbreken. Deze vereiste functiescheiding dient in een autorisatiematrix te zijn vastgelegd. Een voorbeeld:

```
CONNECT XMIEP01 GROUP(PCICS001)
CONNECT XMIEP01 GROUP(PCICS006)
```

Connect-profielen mogen uitsluitend door de Personeelsafdeling (HRM, P&O, PZ, etc.) worden beheerd en zij zijn daardoor impliciet eigenaar. Een voorbeeld:

```
CONNECT XMIEP01 GROUP(PCICS001) OWNER(HRMCENTR)
```

RACF dwingt af dat een gebruiker minimaal aan één groep is gekoppeld. Als de gebruiker aan meer dan één groep is gekoppeld, moet worden aangegeven welke groep als uitgangspunt (default-groep) voor autorisatie-controle wordt genomen. Deze koppeling wordt in het gebruikersprofiel gemaakt. De implementatie:

```
ADDUSER XKRET01 DFLTGRP(GSCHR0)
```

Als de ‘current connect’ groep van een gebruiker, de groep waarmee de gebruiker is aangelogd of een job heeft gesubmit, niet de gewenste autorisatie oplevert, dan mogen andere groepen waar de gebruiker deel vanuit maakt worden gebruik om de vereiste autorisatie tot het object te krijgen. Om deze autorisatie-checking met de andere groepen te laten uitvoeren moet de GRPLIST parameter worden geactiveerd. De autorisatie-checking wordt dan gebaseerd op alle autorisaties van elke group waaraan een gebruiker is toegevoegd.

De zoekbewerking ‘Grouplist access checking’ langs de groepen van een gebruiker is bijvoorbeeld noodzakelijk als activiteiten groepen worden toegepast. Bij de uitzondering waar een gebruiker lid is van uitsluitend één groep, wat niet praktisch is, behoeft deze faciliteit niet te

worden geactiveerd. Dit is echter een systeembrede keuze en geldt daarom voor alle gebruikers. De keuze is daarmee afhankelijk van de gekozen groepstructuur.

De implementatie voor het activeren van grouplist access checking:

```
SETROPTS  GRPLIST
```

Indien een gebruiker een privilege attribuut heeft, zoals SPECIAL of AUDITOR op userid-niveau (system-SPECIAL of system-AUDITOR), mogen deze privileges (group-SPECIAL of group-AUDITOR) niet op een koppeling met een groep worden gedefinieerd. Een gebruiker mag dus maar één privilege hebben op systeem of groep. Privileges zijn hiermee 'mutual exclusive' zowel op systeem- als op groepsniveau. Een voorbeeld:

```
ADDUSER  XKRET01   NAME('T. Krens')  OWNER(HRMCENTR)  SPECIAL
ADDUSER  XMIEP01   NAME('P. Mienes')  OWNER(HRMCENTR)

CONNECT  XKRET01  GROUP(PCICS001)  OWNER(HRMCENTR)
CONNECT  XMIEP01  GROUP(PCICS001)  OWNER(HRMCENTR)  AUDITOR
```

Het privilege group-OPERATIONS is niet toegestaan (en niet meer noodzakelijk in een modern systeem met de juiste faciliteiten) en zal, indien aanwezig moeten worden verwijderd. De implementatie:

```
CONNECT  XOUD007  GROUP(PCICS001)  NOOPERATIONS
```

Indien een gebruiker op tijdelijk basis een rol uitvoert dient een automatische blokkering op de rol te zijn aangebracht. De implementatie:

```
CONNECT  XMIEP01  GROUP(PCICS001)  OWNER(HRMCENTR)
REVOKE(07/01/01)
```

Gebruikers mogen groeps-recources uitsluitend gebruiken, niet aanmaken of verwijderen. Uitsluitend de eigenaar van de groep mag in staat zijn groep-resources aan te maken, te verwijderen of de beveiligingsdefinities van een groep aanpassen. Dit geldt ook voor de default-groep die in het gebruikersprofiel wordt gedefinieerd. De implementatie:

```
ADDGROUP  PCICS001  SUPGROUP(SYS1)  OWNER(SYS1)
            DATA('Security Administratie CICS001')

CONNECT  XMIEP01  GROUP(PCICS001)  OWNER(HRMCENTR)
           AUTHORITY(USE)

Of voor een nieuwe gebruiker:
ADDUSER  XKRET01  NAME('T. Krens')  OWNER(HRMCENTR)
           DFLTGRP(PCICS001)  AUTHORITY(USE)
```

Indien een gebruiker het beheer over een groep voert (gedelegeerd beheer) dienen de default-instellingen zodanig te zijn ingesteld dat deze geen inbreuk hebben op de bestaande functiescheidingen (rollen). Dit geldt ook voor de default-groep die in het gebruikersprofiel wordt gedefinieerd. De implementatie:

```
CONNECT  XENGF01  GROUP(PCICS001)  OWNER(HRMCENTR)
           AUTHORITY(JOIN)  UACC(NONE)  NOGRPACC

Of voor een nieuwe gebruiker:
ADDUSER  XENGF01  NAME('F. Engel')  OWNER(HRMCENTR)
           DFLTGRP(PCICS002)
           AUTHORITY(JOIN)  UACC(NONE)  NOGRPACC
```

4.12 Wachtwoorden en userid inactiviteit

Wachtwoorden dienen aan minimaal de volgende eisen te voldoen:

- minimaal 6 posities
- uiterlijk na 60 dagen te worden gewijzigd
- minstens één letter en één cijfer te bevatten
- laatst gebruikte 13 wachtwoorden moeten niet kunnen worden toegepast
- na 3 achtereenvolgende foutief ingevoerde wachtwoorden dient het userid te worden geblokkeerd. De implementatie:

```
SETROPTS INITSTATS
SETROPTS PASSWORD ( HISTORY(13) INTERVAL(60) REVOKE(3))
SETROPTS PASSWORD ( RULE 1 (LENGTH(6:8)  ALPHANUM)
```

De gebruiker dient 10 dagen voordat het wachtwoord verloopt hiervan automatisch een melding te krijgen. De implementatie:

```
SETROPTS PASSWORD(WARNING(10))
```

Een userid dient, als een gebruiker deze gedurende 90 dagen niet gebruikt, automatisch te worden gedeactiveerd. De implementatie:

```
SETROPTS INACTIVE(90)
```

AANVULLENDE MAATREGEL. Om het niveau van beveiliging te verhogen kunnen andere waarden worden geïmplementeerd. Namelijk:

Item	Basisniveau	AANVULLENDE MAATREGEL
Password change SETROPTS PASSWORD (INTERVAL(xx))	**60 dagen**	**30 dagen**
Userid inactivity SETROPTS INACTIVE(xx)	**90 dagen**	**60 dagen**

Voor emergency redenen kan een userid met uitgebreide toegangsrechten aanwezig zijn dat in een enveloppe op een beveiligde plaats wordt opgeborgen, bijvoorbeeld in een kluis gecontroleerd door bewakingsbeambten. Dit emergency userid kan in een operationele omgeving gebruikt moeten worden om, bij afwezigheid van personen met hoge bevoegdheden, operationele problemen op te lossen. Het gebruik van een emergency userid dient door management te worden goedgekeurd. Het gebruikte userid mag de system-SPECIAL privilege hebben en een wachtwoord dat, om verwarring te voorkomen, niet direct behoeft te worden aangepast. Na gebruik dient Security administratie het wachtwoord direct te veranderen. De implementatie:

```
ALTUSER   UFEWK01   SPECIAL   PASSWORD(JFYYHDGL)
                    NOEXPIRED
PASSWORD  USER(UFEWK01)  NOINTERVAL
```

Het gebruik van een emergency userid moet worden gelogd, zodat achteraf vastgesteld kan worden wanneer dit userid is gebruikt. De implementatie:

```
ALTUSER   UFEWK01  DATA('emergency user')  UAUDIT
```

In alle andere gevallen mogen de parameters **NOEXPIRED** en **NOINTERVAL** niet worden gebruikt en dient het periodiek wijzigen van wachtwoorden te worden afgedwongen door de juiste waarden bij implementatie van interval en revoke. De keywoorden zoals in de volgende commando's zijn toegepast zijn wel toegestaan en zijn de default-waarden. De implementatie:

```
ALTUSER   XMIEP01   PASSWORD(SECRET)   EXPIRED

PASSWORD   USER(XKRET01)   INTERVAL(60)
```

Bij userids met priveleged attributes, zoals de system- en group-SPECIAL en AUDITOR, moet worden afgedwongen dat deze minimaal elke 30 dagen hun wachtwoord veranderen. De implementatie:

```
PASSWORD   USER(XMIEP01)   INTERVAL(30)
```

Wanneer een wachtwoord wordt gereset dan mag geen gebruik worden gemaakt van het default wachtwoord (de naam van de default groep). De persoon die het wachtwoord reset moet een willekeurig en per keer een ander wachtwoord kiezen. Een voorbeeld:

```
ALTUSER   XMIEP01   PASSWORD(HLPWDR4)

of

PASSWORD   USER(XMIEP01)   PASSWORD(oude wachtwoord   HLPWDR4)
```

Helpdesk medewerkers mogen van de gebruikers uitsluitend het userprofiel bekijken, het wachtwoord resetten en indien de gebruiker is gedeactiveerd (revoked) deze weer activeren (resume). Bij deze handelingen dienen voor de organisatie specifieke richtlijnen te worden gevolgd. De volgende implementatie kan worden toegepast indien de organisatie over een centrale Helpdesk beschikt. De implementatie:

```
SETROPTS CLASSACT(FACILITY)

RDEFINE   FACILITY   IRR.LISTUSER   UACC(NONE)
PERMIT    IRR.LISTUSER   CLASS(FACILITY)   ID(HELPDESK)
          ACCESS(READ)

RDEFINE   FACILITY   IRR.PASSWORD.RESET   UACC(NONE)
PERMIT    IRR.PASSWORD.RESET   CLASS(FACILITY)   ID(HELPDESK)
          ACCESS(READ)
```

Opmerking:
Voor een decentrale Helpdesk is het niet aan te bevelen deze implementatie te gebruiken. Toegang tot het 'IRR.LISTUSER' profiel in de FACILITY class geeft de mogelijkheid tot het bekijken van alle user profielen en de toegang tot het 'IRR.PASSWORD' profiel tot het resetten en activeren van alle wachtwoorden binnen het gehele systeem (organisatie).

5 Protecting datasets, DASD en tapes

5.1 Introductie

Alle software zoals besturingssysteem, subsystemen, faciliteiten, applicaties, etc. en gegevens worden opgeslagen in datasets. Datasets zijn fysiek opgeslagen op bijvoorbeeld harde schijven (DASD), tapes, optical disk, etc. Omdat alle software en gegevens in datasets is opgeslagen begint de bescherming van het systeem en de gegevens bij de juiste afscherming van de datasets. Dit hoofdstuk beschrijft op welke wijze deze datasets kunnen worden beschermd.

5.2 Status

Dit hoofdstuk is gereed voor commentaar.

5.3 Administratieve faciliteiten

Alle toegang tot resources dient door het beveiligingspakket te worden geautoriseerd. De implementatie:

SETROPTS **PROTECTALL(FAILURES)**

Voor het effectief en efficiënt beheren van datasetprofielen dienen de onderstaande RACF faciliteiten te worden geactiveerd:

- activering van het gebruik van Enhanced Generic profielen (EGN). De implementatie:

  SETROPTS **EGN**

- activering van Generic command processing voor de class dataset. De implementatie:

  SETROPTS **GENCMD(DATASET)**

- activering van Generic profile processing voor de class dataset. De implementatie:

  SETROPTS **GENERIC(DATASET)**

- deactivering van de aanmaak van Discrete profielen. De implementatie:

  SETROPTS **NOADSP**

- een voorbeeld voor het definiëren van een generic profiel:

  ADDSD ***'SYS1.PRODAPPL.**'*** OWNER(*SECADM*)

  of

  ADDSD ***'SYS1.TESTAPPL.LOAD'*** **GENERIC** OWNER(*SECADM*)

- De actieve in-storage GENERIC-datasetprofielen moeten door de nieuw gedefinieerde profielen worden vervangen. De implementatie:

  SETROPTS **GENERIC(DATASET) REFRESH**

- gebruik van standaard RACF-variabelen in profielen. Een voorbeeld van het gebruik van een variabele in de Global Acces table voor de gebruiker's eigen datasets:

```
RDEFINE   GLOBAL  DATASET
RALTER    GLOBAL  DATASET  ADDMEM('&RACUID.**'/ALTER)
```

```
SETROPTS  GLOBAL(DATASET)  REFRESH
```

- gebruik van installatie gedefinieerde RACF-variabelen in profielen. Indien RACF-variabelen worden toegepast dient de argumentatie van de variabele-namen keuze te worden vastgelegd in een eenduidige structuur. Een voorbeeld van het gebruik van een RACF-variabele:

```
SETROPTS CLASSACT(RACFVARS)

RDEFINE   RACFVARS  &PAYTAPE  UACC(NONE)
          ADDMEM(TAPE12  TAPE05)

SETROPTS RACLIST(RACFVARS)  REFRESH

RDEFINE   TAPEVOL  &PAYTAPE  UACC(NONE)
PERMIT    &PAYTAPE  CLASS(TAPEVOL)  ID(PGRP23)  ACCESS(ALTER)
```

5.4 Dataset profielen

Toegangsrechten dienen op het need-to-know principe gebaseerd te zijn. Een voorbeeld van een user catalog:

```
ADDSD  'SYS1.UCAT212*'  OWNER(SECADM)  UACC(NONE)
PERMIT 'SYS1.UCAT212*'  ID(PCICS001)  ACCESS(UPDATE)
```

Voor de master catalog geldt ook een UACC van NONE. Voor gebruikers, die de master catalog moeten benaderen, om bijvoorbeeld een user catalog te raadplegen, mogen maximaal READ hebben. Gebruikers die bijvoorbeeld uitsluitend van CICS of IMS applicaties gebruik maken mogen geen toegang tot de master catalog hebben. Een voorbeeld voor de master catalog:

```
ADDSD  'SYS1.MCAT*'  OWNER(SECADM)  UACC(NONE)
PERMIT 'SYS1.MCAT*'  ID(UGRPTSO1)  ACCESS(READ)
```

Het uitgangspunt voor het definiëren van dataset-protectie moet er opgericht zijn om zo min mogelijk profielen te definiëren. Een voorbeeld:

```
ADDSD  'SYS1.**'  UACC(NONE)  OWNER(SECADM)
```

Specifiekere generic profielen mogen uitsluitend worden gedefinieerd indien dit noodzakelijk is voor het specifieker groeperen (deelverzameling) van datasets, voor de implementatie van de vereiste functiescheiding. Een voorbeeld:

```
ADDSD  'SYS1.PRODAP01.**'  UACC(NONE)  OWNER(SECADM)
```

en/of

```
ADDSD  'SYS1.PRODAP01.LOAD.**'  UACC(NONE)  OWNER(SECADM)

ADDSD  'SYS1.PRODAP02.**'  UACC(NONE)  OWNER(SECADM)
```

en/of

```
ADDSD  'SYS1.PRODAP02.LOAD.**'  UACC(NONE)  OWNER(SECADM)
```

Gebruikers mogen niet instaat zijn meer restrictievere generic profielen aan te kunnen maken dan de eigenaar van het originele generic profiel. De implementatie:

```
SETROPTS  GENERICOWNER
```

Bij de toegangsrechten dienen uitsluitend de minimale rechten, die nodig zijn om een goede werking te garanderen, te worden gedefinieerd. Een voorbeeld:

```
ADDSD  'SYS1.CICS2*.LOAD*'  OWNER(SECADM)  UACC(NONE)
PERMIT 'SYS1.CICS2*.LOAD*'  ID(PCICS001)   ACCESS(EXECUTE)
```

Gebruikers dienen niet automatisch toegangsrechten op een resource te krijgen als zij een RACF-profiel aanmaken. De implementatie:

```
SETROPTS NOADDCREATOR
```

Toegangsrechten voor datasets dienen uitsluitend met groepsautorisatie te worden uitgegeven. Een voorbeeld:

```
ADDSD  'SYS1.CICS2*.DB*'  OWNER(SECADM)  UACC(NONE)
PERMIT 'SYS1.CICS2*.DB*'  ID(UGRPA002)   ACCESS(READ)
PERMIT 'SYS1.CICS5*.DB*'  ID(UGRPB005)   ACCESS(READ)
```

5.4.1 Sluitprofielen

Voor sommige resource classes geeft RACF een return code van 4 af in plaats van 8 als een profiel niet wordt gevonden. De betreffende resource manager software die de return code van 4 ontvangt beslist dan zelf of de gebruiker toegang krijgt of niet. Welke return code bij niet aanwezig zijn van een RACF-profiel wordt afgegeven wordt in de Class Descriptor Table (CDT) voor elke resource class aangegeven.
Om nu RACF in alle gevallen een return code van 8 te laten geven als een profiel niet wordt gevonden moet een sluitprofiel ('**') worden gedefinieerd. Een voorbeeld van de implementatie:

```
RDEFINE  **  CLASS(TAPEVOL)  DATA('Sluitprofiel voor de TAPEVOL class')
             OWNER(SECADM)   UACC(NONE)
```

Opmerking: Bij de PROGRAM Class dient **UACC(READ)** op het sluitprofiel te worden gedefinieerd aangezien anders alle uit te voeren programma's moeten worden gedefinieerd wat in de praktijk niet haalbaar en voor de beveiliging niet noodzakelijk is.
Voor de default return code van de afzonderlijke resource classes zal de CDT moeten worden geraadpleegd en afhankelijk daarvan zullen sluitprofielen moeten worden gedefinieerd. Aangeraden wordt de FACILITY class niet van een sluitprofiel te voorzien aangezien een aantal systeemfuncties dan niet kunnen worden geactiveerd.

5.4.2 Decentraal beheer

Door het gebruik van GENERICOWNER, inclusief het gebruik van sluitprofielen, zoals hierboven aangegeven, geeft de basis voor decentraal beheer van toegangsrechten tot datasets. Indien centraal wordt toegestaan dat een decentrale Security administrator beheer voert over een groep van datasets kan deze een specifieker dataset profiel aan maken met het eigenaarschap voor de decentrale Security administrator. Een voorbeeld van de implementatie:

```
ADDSD  **  DATA('Sluitprofiel voor de DATASET class')  OWNER(SECADM)
           UACC(NONE)
```

en

```
ADDSD  'SYS1.**'  UACC(NONE)  OWNER(SECADM)
```

en/of

```
ADDSD  'SYS1.TEST.CICS.LOAD.**'  UACC(NONE)  OWNER(SECDEC4)
```

en/of

```
ADDSD  'SYS1.ACCP.**'  UACC(NONE)  OWNER(SECDEC2)
```

en/of

ADDSD ***'SYS1.TEST.APPL25.**'*** UACC(NONE) OWNER(*SECDEC9*)

5.5 Universal Access (UACC)

Als UACCs automatisch worden gegenereerd dienen deze uitsluitend NONE te zijn. De implementatie:

ADDUSER (*XMIEP01*) **UACC(NONE)**
CONNECT USER(*XMIEP01*) GROUP(*PCICS001*) **UACC(NONE)**

De Universal Access Control (UACC) mag uitsluitend anders zijn dan NONE indien toegang voor alle gebruikers in het gehele systeem noodzakelijk is en het profiel uitsluitend gegevens beschermd waarvan publieke openbaarheid geen vertrouwelijkheidsprobleem geeft. Een voorbeeld:

ADDSD ***'ALL1.PUBLIC.**'*** OWNER(*SECADM*) **UACC(READ)**
PERMIT *'ALL1.PUBLIC.**'* ID(*UGRPF002*) ACCESS(UPDATE)

5.6 Erase on Scratch

Indien een dataset uit het systeem wordt verwijderd dienen de gegevens die de dataset bevat eerst te worden overschreven zodat volgende gebruikers van de opslagruimte geen gebruik van deze gegevens kunnen maken. Deze faciliteit is een eis om het beveiligingsniveau C2 te implementeren en is gesteld door de Department of Defense of America (DoD). De implementatie:

SETROPTS **ERASE(ALL)**

5.7 Tape protectie

Het gebruik van tapes volumes dient geautoriseerd plaats te vinden. Een implementatie voorbeeld:

SETROPTS CLASSACT(**TAPEVOL**)

RDEFINE TAPEVOL ***TABC01*** UACC(**NONE**)
PERMIT TABC01 CLASS(TAPEVOL) ID(***XMIEP01***) ACCESS(**READ**)

RDEFINE TAPEVOL ***TXYZ01*** UACC(**NONE**)
PERMIT TXYZ01 CLASS(TAPEVOL) ID(***XMIEP01***) ACCESS(**UPDATE**)

AANVULLENDE MAATREGEL. Alle tape datasets dienen als DASD datasets te worden beschermd door RACF-profielen. De implementatie:

SETROPTS **CLASSACT(TAPEVOL)**
SETROPTS **TAPEDSN**

RDEFINE TAPEVOL *TABC02* **TVTOC** UACC(NONE)
ADDSD ***'LEEN.**'*** UACC(NONE)
PERMIT *'LEEN.**'* ID(*PCICS01*) ACCESS(**READ**)

5.8 Global Access Table

Zoveel als mogelijk is dienen profielen in het computergeheugen te worden geplaatst d.m.v. RACF faciliteiten. Een implementatie voorbeeld:

```
SETROPTS GLOBAL(DATASET)

RDEFINE   GLOBAL  DATASET
RALTER    GLOBAL  DATASET  ADDMEM('SYS1.BRODCAST'/UPDATE)

RALTER    GLOBAL  DATASET  ADDMEM('SYS1.MASTER.**'/READ)

RALTER    GLOBAL  DATASET  ADDMEM('SYS1.HELP'/READ)

RALTER    GLOBAL  DATASET  ADDMEM('&RACGPID.**'/READ)

RALTER    GLOBAL  DATASET  ADDMEM('&RACUID.**'/ALTER)

SETROPTS GLOBAL(DATASET)  REFRESH
```

De inhoud van in-storage autorisatietabellen, zoals de Global Access Table (GAC), dienen altijd synchroon met de profielen in de RACF-database te worden gehouden. Bij gesynchroniseerde profielen zullen gebruikers bij het deactiveren van de GAC table dezelfde autorisaties behouden. Een voorbeeld gerelateerd aan de GAC table hierboven:

```
ADDSD   'SYS1.BRODCAST'   UACC(UPDATE)
ADDSD   'SYS1.MASTER.**'  UACC(READ)
ADDSD   'SYS1.HELP'       UACC(READ)

ADDSD   'UGRPF002.**'     UACC(NONE)
PERMIT  'UGRPF002.**'  ID(UGRPF002)  ACCESS(READ)
```

Indien specifieke gebruikers een lagere autorisatie hebben dan de Universal ACCess dient het profiel niet in de Global Access Table (GAC) te worden opgenomen. Een voorbeeld waarbij het profiel ‘UGRPF003.**’ niet in de GAC table mag worden opgenomen:

```
ADDSD   'UGRPF003.**'  UACC(READ)
PERMIT  'UGRPF003.**'  ID(UGRPF002)  ACCESS(NONE)
```

5.9 Program control

Programma-modulen die informatie (b.v. program logic of control informatie) bevatten waarbij het ongewenst is als dit openbaar wordt, dienen uitsluitend door de gebruiker te kunnen worden uitgevoerd en niet op enige wijze te kunnen worden gekopieerd. Een voorbeeld:

```
ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)
PERMIT    'PROD1.APPL*.LOAD'  ID(UGRP006)  ACCESS(EXECUTE)
```

AANVULLENDE MAATREGEL. Programma’s die niet samen met andere onbekende programma’s mogen worden geladen, omdat de programmacode hierdoor benaderbaar kan worden, dienen alle programma’s door het beveiligingsmechanisme te worden gecontroleerd. Een voorbeeld:

```
SETROPTS WHEN(PROGRAM)

ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)
PERMIT    'PROD1.APPL*.LOAD'  ID(UGRP006)  ACCESS(EXECUTE)

RDEFINE   PROGRAM  MYDBM*
          ADDMEM('PROD1.APPL*.LOAD'//NOPADCHK)  UACC(NONE)
```

PERMIT *MYDBM** CLASS(PROGRAM) ID(*UGRP006*) ACCESS(**EXECUTE**)

SETROPTS WHEN(PROGRAM) **REFRESH**

Opmerking: Indien een load module een tabel of andere informatie bevat en daarom door een programma wordt gelezen moet de betreffende load library met READ worden gepermit. Verder zijn er programma's die een open doen met READ en daarom een violation veroorzaken. Ook hierbij geldt dat de library dan met READ moet worden gepermit. In beide gevallen zal niet het beoogde resultaat worden verkregen zoals met EXECUTE.

AANVULLENDE MAATREGEL. Datasets die uitsluitend via een specifiek programma mogen worden benaderd, omdat deze programma's bijvoorbeeld bepaalde controles (application controls) bevatten, dienen voor gebruikers niet direct maar uitsluitend via het specifieke programma te kunnen worden benaderd. Een implementatie voorbeeld:

SETROPTS **WHEN(PROGRAM)**

ADDSD *'PROD1.APPL.LOAD*'* UACC(**EXECUTE**)

RDEFINE PROGRAM ***LMODA23***
ADDMEM(*'PROD1.APPL15.LOAD'***//PADCHK)** UACC(EXECUTE)

SETROPTS WHEN(PROGRAM) **REFRESH**

PERMIT *'PROD1.SALES.DATAB*'* ID(***UGRPA006***) ACCESS(**UPDATE**)
WHEN(PROGRAM(*LMODA23*))

6 Gebruikers- en data-categorisering

6.1 Introductie

Organisaties dienen hun gebruikers en gegevens te categoriseren zodat bekend is op welke wijze deze behandeld en beschermd dienen te worden. RACF in combinatie met het besturingssysteem ondersteunt de mogelijkheid om de gecategoriseerde gebruikers en gegevens te definiëren. Voor deze indeling zijn beveiligingscategorieën en -levels geïmplementeerd conform het B1-beveiligingsniveau zoals beschreven in het 'Orange Book' van het Department of Defense of America (DoD).

AANVULLENDE MAATREGEL. De mogelijkheden voor de implementatie van het B1 beveiligingsniveau met behulp van RACF zijn hieronder beschreven. Het te behalen beveiligingsniveau, conform de definitie in deze RACF-standaard, is een wezenlijke verhoging van het basisniveau buiten de *AANVULLENDE MAATREGELEN* genoemd in de andere hoofdstukken.

6.2 Status

Dit hoofdstuk is gereed voor commentaar..

6.3 Gebruikerscategorisering

Voor het verkrijgen van een hoger beveiligingsniveau kunnen beveiligingslevels en -categorieën worden toegepast. Hiervoor moeten de volgende stappen worden ondernomen en toegepast.

Indien de keuze uitsluitend voor categorieën is dient men de volgende normen te hanteren.

Het gebruik van categorieën dient aan RACF bekend te worden gemaakt. In de general resource class SECDATA moet een profiel voor beveiligingscategorieën (CATEGORY) worden gedefinieerd. Daarna dienen de gewenste categorienamen aan het CATEGORY profiel te worden gekoppeld. De categorienamen dienen heldere en zinvolle te zijn. Een implementatie voorbeeld:

```
RDEFINE   SECDATA  CATEGORY  UACC(NONE)

RALTER    SECDATA  CATEGORY  ADDMEM(POSTK)
RALTER    SECDATA  CATEGORY  ADDMEM(INKOOP)
RALTER    SECDATA  CATEGORY  ADDMEM(PROD)
RALTER    SECDATA  CATEGORY  ADDMEM(VERKOOP)

SETROPTS CLASSACT(SECDATA)
```

Gebruikers dienen in één of meerdere categorieën te krijgen. Een voorbeeld:

```
ADDUSER  XKRET01   NAME('T. Krens')  ADDCATEGORY(POSTK)
ADDUSER  XRIJL01   NAME('L. van Rij')        ADDCATEGORY(VERKOOP)
ADDUSER  XMIEP01   NAME('P. Mienes')ADDCATEGORY(PROD)
ADDUSER  XENGF01   NAME('F. Engel')  ADDCATEGORY(INKOOP)
```

Voor gebruik van beveiligingscategorieën dienen de resources eveneens van één of meerdere beveiligingscategorieën te worden voorzien. Een voorbeeld:

```
ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)
```

```
ADDCATEGORY(PROD)
```

6.4 Gebruikerslevels

Naast het gebruik van user-categorieën kunnen ook beveiligingslevels worden toegepast.

Indien de keuze uitsluitend voor beveiligingslevels is dient men de volgende normen te hanteren.

Het gebruik van beveiligingslevels dient aan RACF bekend te worden gemaakt. In de general resource class SECDATA moet een profiel voor beveiligingslevels (SECLEVEL) worden gedefinieerd. Daarna dienen de gewenste beveiligingslevelnamen aan het SECLEVEL profiel te worden gekoppeld. De levelnamen dienen heldere en zinvolle te zijn. De numerieke waarde van de levels dienen overeenkomstig de perceptie van de levelnamen te worden gekozen. De implementatie:

```
RDEFINE   SECDATA  SECLEVEL  UACC(NONE)

RALTER    SECDATA  SECLEVEL  ADDMEM(PUBLIEK/5)
RALTER    SECDATA  SECLEVEL  ADDMEM(INTERN/20)
RALTER    SECDATA  SECLEVEL  ADDMEM(GEHEIM/50)
RALTER    SECDATA  SECLEVEL  ADDMEM(TOPSECRET/150)

SETROPTS CLASSACT(SECDATA)
```

Gebruikers dienen een beveiligingslevel te krijgen. De implementatie:

```
ADDUSER  XKRET01   NAME('T. Krens')   SECLEVEL(INTERN)
ADDUSER  XRIJL01   NAME('L. van Rij')  SECLEVEL(TOPSECRET)
ADDUSER  XMIEP01   NAME('P. Mienes')   SECLEVEL(GEHEIM)
ADDUSER  XENGF01   NAME('F. Engel')    SECLEVEL(GEHEIM)
```

Voor gebruik van beveiligingslevels dienen de resources eveneens van één beveiligingslevel te worden voorzien. De implementatie:

```
ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)
          SECLEVEL(GEHEIM)
```

6.5 Gebruikerscategorisering en -levels

Beveiligingscategorieën beveiligingslevels kunnen gelijktijdig worden toegepast. Hierdoor moeten gebruikers niet alleen de juiste categorie hebben maar tevens een voldoende hoog beveiligingslevel hebben voor het kunnen benaderen van de resources.

Indien de keuze beveiligingscategorieën en beveiligingslevels is dient men de volgende normen te hanteren.

Het gebruik van beveiligingscategorieën en beveiligingslevels dient aan RACF bekend te worden gemaakt. In de general resource class SECDATA moet een profiel voor beveiligingscategorieën (CATEGORY) en beveiligingslevels (SECLEVEL) worden gedefinieerd. Daarna dienen de gewenste beveiligingscategorienamen aan het CATEGORY profiel en de beveiligingslevelnamen met een gekozen numeriek level aan het SECLEVEL profiel te worden gekoppeld. De categorienamen en levelnamen dienen heldere en zinvolle te zijn. De numerieke waarde van de levels dienen overeenkomstig de perceptie van de levelnamen te worden gekozen. Een voorbeeld:

```
RDEFINE   SECDATA  CATEGORY  UACC(NONE)

RALTER    SECDATA  CATEGORY  ADDMEM(POSTK)
RALTER    SECDATA  CATEGORY  ADDMEM(INKOOP)
RALTER    SECDATA  CATEGORY  ADDMEM(PROD)
RALTER    SECDATA  CATEGORY  ADDMEM(VERKOOP)

RDEFINE   SECDATA  SECLEVEL  UACC(NONE)

RALTER    SECDATA  SECLEVEL  ADDMEM(PUBLIEK/5)
RALTER    SECDATA  SECLEVEL  ADDMEM(INTERN/20)
RALTER    SECDATA  SECLEVEL  ADDMEM(GEHEIM/50)
RALTER    SECDATA  SECLEVEL  ADDMEM(TOPSECRET/150)

SETROPTS CLASSACT(SECDATA)
```

Gebruikers dienen één of meer beveiligingscategorieën en één beveiligingslevel te krijgen. De implementatie:

```
ADDUSER  XKRET01   NAME('T. Krens')
         ADDCATEGORY(POSTK)  SECLEVEL(INTERN)
ADDUSER  XRIJL01   NAME('L. van Rij')
         ADDCATEGORY(VERKOOP)  SECLEVEL(TOPSECRET)
ADDUSER  XMIEP01   NAME('P. Mienes')
         ADDCATEGORY(PROD)  SECLEVEL(GEHEIM)
ADDUSER  XENGF01   NAME('F. Engel')
         ADDCATEGORY(INKOOP PROD)  SECLEVEL(GEHEIM)
```

Voor gebruik van beveiligingscategorieën en beveiligingslevels dienen de resources eveneens van één of meerdere beveiligingscategorieën en één beveiligingslevel te worden voorzien. De implementatie:

```
ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)
          ADDCATEGORY(PROD)  SECLEVEL(GEHEIM)
```

6.6 Beveiligingslabels

Indien van beveiligingscategorieën en beveiligingslevels gebruik wordt gemaakt kan worden overwogen om beveiligingslabels toe te passen. Met beveiligingslabels worden één of meerdere beveiligingscategorieën en één beveiligingslevel gecombineerd tot één beveiligingslabel. Indien beveiligingscategorieën en beveiligingslevels moeten worden aangepast wordt het beheer bij het gebruikt van beveiligingslabels sterk vereenvoudigd.

Indien de keuze voor beveiligingslabels is dient men de volgende normen te hanteren.

Het gebruik van beveiligingscategorieën, beveiligingslevels en beveiligingslabels dient aan RACF bekend te worden gemaakt. In de general resource class SECDATA moet een profiel voor beveiligingscategorieën (CATEGORY), beveiligingslevels (SECLEVEL) en beveiligingslabels (SECLABEL) worden gedefinieerd. Daarna dient de gewenste beveiligingslabelnaam aan het SECLABEL profiel te worden gekoppeld met daarbij behorende één of meerdere beveiligingscategorieën en één beveiligingslevel. Een beveiligingslabel heeft één beveiligingslevel met daarbij géén, één of meerdere beveiligingscategorieën.

De categorienamen, levelnamen en labelnamen dienen heldere en zinvolle te zijn. Het is aan te raden de numerieke waarde van de levels overeenkomstig de perceptie van de levelnamen te kiezen. De implementatie:

```
RDEFINE   SECDATA  CATEGORY  UACC(NONE)

RALTER    SECDATA  CATEGORY  ADDMEM(POSTK)
RALTER    SECDATA  CATEGORY  ADDMEM(INKOOP)
RALTER    SECDATA  CATEGORY  ADDMEM(PROD)
RALTER    SECDATA  CATEGORY  ADDMEM(VERKOOP)

RDEFINE   SECDATA  SECLEVEL  UACC(NONE)

RALTER    SECDATA  SECLEVEL  ADDMEM(PUBLIEK/10)
RALTER    SECDATA  SECLEVEL  ADDMEM(INTERN/20)
RALTER    SECDATA  SECLEVEL  ADDMEM(GEHEIM/50)
RALTER    SECDATA  SECLEVEL  ADDMEM(TOPSECRET/150)

SETROPTS CLASSACT(SECDATA)

RDEFINE   SECLABEL  HRSAFD
          SECLEVEL(GEHEIM)  ADDCATEGORY(INKOOP PROD)

SETROPTS CLASSACT(SECLABEL)  RACLIST(SECLABEL)
```

Gebruikers dienen één beveiligingslabel te krijgen dat als default label voor toegangscontrole dient. Tevens dient de gebruiker hiervoor te worden geautoriseerd. De implementatie:

```
PERMIT    HRSAFD   CLASS(SECLABEL)  ACCESS(READ)
          ID(MITGRP)

SETROPTS CLASSACT(SECLABEL)  REFRESH

ALTUSER   XENGF01  NAME('F. Engel')  SECLABEL(HRSAFD)
```

Indien het noodzakelijk is dat de gebruiker bevoegd moet zijn meerdere beveiligingslabels te gebruiken dient de gebruiker voor deze beveiligingslabels te worden geautoriseerd. De gebruiker kan daarna de beveiligingslabels aangeven in de logon, indien door de applicatie ondersteund, of in de JCL jobkaart. De implementatie:

```
PERMIT    (HRPAFD HRQAFD)  CLASS(SECLABEL)  ACCESS(READ)
          ID(MITGRP)

SETROPTS CLASSACT(SECLABEL)  REFRESH
```

Voor gebruik van beveiligingslabels dient elke resource van één beveiligingslabels te worden voorzien. De implementatie:

```
ADDSD     'PROD1.APPL*.LOAD'  UACC(NONE)  SECLABEL(HRSAFD)
```

6.7 Standaard labels

Het gebruik van de standaard beveiligingslabels SYSHIGH, SYSLOW en SYSNONE zijn uitsluitend toegestaan voor beveiligingsmedewerkers (SECADM). Het beveiligingslabel SYSNONE mag uitsluitend voor beheerwerkzaamheden door operationele medewerkers op tijdelijke basis worden gedefinieerd. De implementatie:

```
PERMIT    SYSHIGH  CLASS(SECLABEL)  ACCESS(READ)
          ID(BSITGRP)

SETROPTS CLASSACT(SECLABEL)  REFRESH

ALTUSER   SECADM01  NAME('O.N. Veilig')  SECLABEL(SYSHIGH)
```

6.8 Beheer en processing beveiligingslabels

Indien beveiligingslabels worden toegepast mogen gebruikers niet de mogelijkheid hebben om zelf beveiligingslabels te definiëren. Dit recht moet voorbehouden zijn aan de beveiligingsmedewerkers. De implementatie:

SETROPTS **SECLABELCONTROL**

AANVULLENDE MAATREGEL. Indien in een resource-profiel geen beveiligingslabel is gedefinieerd mag geen toegang worden verleend. De implementatie:

SETROPTS **MLACTIVE(FAILURES)**

AANVULLENDE MAATREGEL. Uitsluitend gebruikers met de juiste beveiligingslabels mogen toegang tot resources krijgen. De implementatie:

SETROPTS **NOCOMPATMODE**

AANVULLENDE MAATREGEL. Gegevens mogen niet door de gebruikers naar een lager level kunnen worden gekopieerd. De implementatie:

SETROPTS **MLS(FAILURES)**

AANVULLENDE MAATREGEL. Tijdens de normale werking van het systeem mogen geen beveiligingslabels kunnen worden aangepast. Onder normale werking wordt verstaan dat er geen uitzonderlijke situaties aanwezig zijn waarvoor het noodzakelijk is dat security labels moeten worden aangepast. De implementatie:

SETROPTS **MLSTABLE**

7 Audit controls en logging

7.1 Introductie

Om de controleerbaarheid van de gegevensverwerking te kunnen garanderen is het noodzakelijk logging records aan te maken. Verder is het van belang dat een evenwicht wordt gevonden tussen de noodzakelijke logging records en het gebruik van systeemmiddelen voor het aanmaken, vastleggen en archiveren van deze records. In dit hoofdstuk is aangegeven wat de minimale implementatie is.

7.2 Status

Dit hoofdstuk moet nog verder worden uitgewerkt.

7.3 System-wide audit settings

Het misbruik van RACF-commando's moet worden gedetecteerd en worden vastgelegd. De implementatie:

SETROPTS **CMDVIOL**

Alle wijzigingen op RACF-profielen moeten worden gelogd om een eenduidige vastlegging van autorisatiewijzigingen te krijgen. De implementatie:

SETROPTS **AUDIT**(*)

Alle toegangsaanvragen tot datasets dienen in normale operationele omstandigheden te worden gelogd conform de definities in de dataset-profielen. De implementatie:

SETROPTS LOGOPTIONS(**DEFAULT(DATASET)**)

Al het gebruik van operator-commando's en tapes dienen altijd te worden gelogd. De implementatie:

SETROPTS LOGOPTIONS(**ALWAYS(OPERCMDS TAPEVOL)**)

Datasets dienen met de originele naam, dus niet met een alias, te worden gelogd. De implementatie:

SETROPTS **REALDSN**

AANVULLENDE MAATREGEL. Voor controle van het aantal keren dat toegang tot een resource is gegeven, dient voor security auditing bij beveiligingsincidenten, statistische gegevens te worden bijgehouden. De implementatie:

SETROPTS **STATISTICS(DATASET)**

7.4 Group audit settings

Geen groep settings beschikbaar.

7.5 User audit settings

Alle RACF commando's, die een gebruiker met het attribuut SPECIAL uitvoert, dienen te worden gelogd. De implementatie:

```
SETROPTS SAUDIT
```

Het gebruik van resources door gebruikers met het SPECIAL attribuut moet altijd worden gelogd. Hiervoor wordt op gebruikersniveau auditing geactiveerd. Indien gewaarborgd is dat gebruikers met het SPECIAL-attribuut altijd het auditing krijgen, zoals hieronder aangegeven, is de SETROPTS SAUDIT definitie niet strik noodzakelijk. Een voorbeeld van het activeren van auditing op gebruikersniveau:

```
ALTUSER  XMIEP01  DATA('P. Mienes')  UAUDIT
```

7.6 Resource audit settings

Ongeautoriseerde toegangsaanvragen tot datasets, die gegevens bevatten die de integriteit van het systeem kunnen beïnvloeden, dienen voortdurend te worden gelogd. De implementatie:

```
ADDSD  'SYS1.RACF*'  AUDIT(FAILURES(READ))
```

Indien de dataset gekwalificeerd is als zeer belangrijk voor de werking van het gehele systeem dient de Security administrator de logging van de datasets te activeren. De implementatie:

```
ALTDSD    'SYS1.PARM*'    GLOBALAUDIT(ALL(READ))

of

ALTDSD    'PRODA.CICS.DBFILE*'
          GLOBALAUDIT(FAILURES(READ)  SUCCESS(UPDATE))
```

Indien de eigenaar van de dataset het noodzakelijk vindt dat auditing geactiveerd moet zijn kan deze worden geactiveerd indien dit niet de performance van het systeem negatief beïnvloedt. De implementatie:

```
ALTDSD    'OPERGRP.PARM*'   AUDIT(ALL(READ))

of

ALTDSD    'TESTA.CICS.DBFILE*'
          AUDIT(FAILURES(READ)  SUCCESS(UPDATE))
```

Indien de class GLOBAL, voor het definiëren van de Global Access Table, wordt gebruikt zal de toegang tot de resource, toegestaan door de global-profielen, niet worden vastgelegd. Een voorbeeld:

```
SETROPTS GLOBAL(DATASET)

RALTER    GLOBAL  DATASET  ADDMEM('SYS1.HELP'/READ)
```

Voor tape volumes met vertrouwelijke gegevens dient de Security administrator alle toegang te laten vastleggen. De implementatie:

```
RALTER    TAPEVOL  ABX387  GLOBALAUDIT(ALL(READ))
```

AANVULLENDE MAATREGEL. Alle ongeautoriseerde toegangsaanvragen tot datasets dienen voortdurend te worden gelogd. De implementatie:

```
SETROPTS LOGOPTIONS(FAILURES(DATASET))
```

7.7 Monitoring

Bij misbruik van resources, zoals datasets en tapes, dient de Security auditor en Security administrator direct te worden gewaarschuwd zodat direct maatregelen kunnen worden getroffen. De implementatie:

ADDSD *'SYS1.PARM*'* NOTIFY(*SECADM2*)

RDEFINE TAPEVOL *ABX387* NOTIFY(*SECADM2*)

AANVULLENDE MAATREGEL. Alle beveiligingsgerichte systeemberichten dienen naar een apart console te worden gerouteerd waarbij uitsluitend de Security auditor toegang toe heeft. Het routeren van beveiligingsgerichte systeemberichten is een z/OS console facility.

Aangezien het aantal beveiligingsgerichte systeemberichten dermate hoog kan zijn dat het niet meer met een console is te beoordelen of er onrechtmatige handelingen plaatsvinden, is het aan te raden een meer geautomatiseerde monitor te implementeren. De monitor dient filtering van de systeemberichten uit te voeren en te analyse of de verschillende typeberichten buiten de verwachte trend vallen. Gesteld kan worden dat hiervoor nog geen standaard en afdoende monitoren op de markt zijn.

7.8 Logging to SMF and archivering

Alle beveiligingsgerichte logginggegevens dienen tenminste 15 maanden te worden bewaard.

De Security auditor dient hiervoor een procedure te implementeren.

7.9 Reporting and reviewing

Alle parameters en definities die de beveiligingsnormen en -standaarden implementeren dienen periodiek te worden gecontroleerd door reviewing van rapportage. De rapportage dient een gegarandeerde integere weergave te zijn van de werkelijke implementatie. Een aantal voorbeelden van RACF hulpmiddelen voor het vervaardigen van rapportage:

- RACF commando's:
  - RACF LIST commando's, voor het verkrijgen van actuele (online) informatie, zoals:
  - SETROPTS LIST voor het listen van RACF opties (controle, minimaal 1 keer per week);
  - LIST USER (LU) voor het rapporteren van gebruikersgegevens (controle, op adhoc basis);
  - LIST GROUP (LG) voor het rapporteren van groepsgegevens (controle, op adhoc basis);
  - LIST DATASET (LD) voor het rapporteren van dataset gegevens (controle, op adhoc basis);
  - LIST general resources (RLIST) voor het rapporteren van andere middelen dan datasets (controle, op adhoc basis);
  - SEARCH voor het combineren van zoekargumenten over verschillende profielen (gebruik, op adhoc basis).
- Data Security Monitor (DSMON) voor het rapporteren van een aantal RACF basis instellingen (controle, minimaal 1 keer per week);
- RACF Report Writer (RACFRW) voor het selecteren en rapporteren van SMF data (controle van de belangrijkste resources en gebruikers, elke dag). De RACF Report

Writer wordt niet meer door de leverancier geactualiseerd waardoor nieuwe faciliteiten niet worden gerapporteerd;

- RACF Database unload utility IRRDBU00 voor het kopiëren van de RACF database inhoud naar een andere database manager voor analyses en rapportages (gebruik, op adhoc basis);
- RACF SMF Unload Utility IRRADU00 voor het verder verwerken van SMF records door een database manager (gebruik, op adhoc basis);
- RACF Cross Reference utility IRRUT100 voor het rapporteren van informatie van een zelfde gebruiker en groepen (gebruik, op adhoc basis).

In de rapportages moeten minimaal de volgende gegevens in de vorm van afwijkingen van vastgestelde normen zijn opgenomen:

- het aantal malen dat is geprobeerd toegang te krijgen tot een systeem;
- het aantal malen dat toegang is verleend;
- het aantal malen dat toegang is geweigerd;
- het aantal malen dat een bepaalde gebruiker toegang werd geweigerd;
- de ongebruikelijke tijden waarop gebruikers actief waren;
- het aantal vervallen wachtwoorden/userids;
- het aantal vervallen autorisaties;
- het aantal nieuw toegekende wachtwoorden/userids;
- het aantal nieuw toegekende autorisaties;
- het aantal actieve userids op het moment van rapportage;
- het aantal actieve autorisaties op het moment van rapportage;
- het aantal malen dat userids, en eventueel wachtwoorden of autorisaties, werden teruggetrokken in verband met slordig of onoordeelkundig gebruik (bijvoorbeeld vaak verkeerd inloggen);
- gegevens over gebruikers die slordig of onoordeelkundig gebruik maken van toegangsmogelijkheden, eventueel met het advies hun verdere toegang te ontzeggen;
- eventuele beveiligingstendensen die uit de logging zijn verkregen.

7.10 B1 security audit settings

Bij gebruik van B1 beveiligingsfaciliteiten dient auditing hiervoor te worden geactiveerd vanaf minimaal die niveaus die vertrouwelijkheid van gegevens in gevaar kunnen brengen. De implementatie:

```
SETROPTS SECLEVELAUDIT(CONFIDENTIAL)
```

Indien beveiligingslabels worden toegepast dient het misbruik van beveiligingslabels te worden gedetecteerd en vastgelegd. De implementatie:

```
SETROPTS SECLABELAUDIT

RALTER   SECLABEL  BSIT  AUDIT(FAILURES(READ))
```

8 RACF Remote Sharing Facility (RRSF)

8.1 Introductie

Met de RACF Remote Sharing Facility (RRSF) kunnen (selectief) wachtwoorden en RACF-commando's tussen verschillende RACF-systemen worden uitgewisseld. Hierdoor is het mogelijk om met RRSF een single-point-of-administration te verwezenlijken en de informatie in verschillende RACF-databases gelijk te houden. Een RACF-subsysteem communiceert hiervoor met één of meer andere RACF-subsystemen. In dit geval wordt gesproken over RRSF-nodes. Voor communicatie tussen RRSF-nodes wordt het Advanced Program to Program Communications (APPC) protocol toegepast waarvan het noodzakelijke netwerkinterface onderdeel is van het APPC/MVS-subsysteem. Voor netwerkcommunicatie wordt de Virtual Telecommunication Access Methode (VTAM) gebruikt. Voor veilige communicatie van wachtwoorden en RACF-commando's tussen RRSF-nodes wordt deze door het RACF-subsysteem versleuteld.

8.2 Virtual Telecommunication Access Methode (VTAM)

8.2.1 Introductie

RRSF gebruikt het APPC-protocol (Advanced Program to Program Communication) voor communicatie tussen de verschillende RRSF-nodes. APPC is ontwikkeld binnen het SNA-protocol maar is daarbij volledig toepasbaar bij het gebruik van TCP/IP-netwerken. Binnen VTAM, de z/OS implementatie van SNA, kan het SNA-protocol geschikt worden gemaakt voor transport over TCP/IP.

8.2.2 Status

Gereed voor commentaar.

8.2.3 VTAM definities

De voor APPC/MVS gebruikte netwerkverbinding, een Logical Unit (LU), moet binnen VTAM worden gedefinieerd. De VTAM-definitie moet zijn opgenomen in een SYS1.VTAMLST member. Een voorbeeld van de VTAM-definities op systeem *PRODA1*:

```
LURRSFA1  APPL  ACBNAME=LURRSFA1
```

Voor het gebruik van VTAM zal APPC/MVS een VTAM ACB (Access Control Block) moeten kunnen openen. Omdat het RACF-subsysteem APF-authorised is is het niet noodzakelijk deze via RACF toegang tot de VTAM ACB te verlenen. Echter andere applicaties mogen geen toegang kunnen krijgen tot het VTAM ACB dat voor RRSF bestemd is en dient daarom door RACF te worden afgeschermd. Een voorbeeld:

```
SETROPTS GENERIC(VTAMAPPL)
SETROPTS CLASSACT(VTAMAPPL)
SETROPTS AUDIT(VTAMAPPL)

RDEFINE   VTAMAPPL  LURRSFA1  OWNER(SECADM)  UACC(NONE)

SETROPTS RACLIST(VTAMAPPL)  REFRESH
```

Om de noodzakelijke APPC-interface te activeren moet in de LU definities de parameter APPC=YES worden aangebracht. De implementatie:

```
LURRSFA1   APPL   ACBNAME=LURRSFA1,
                  APPC=YES
```

Alhoewel niet van belang voor de beveiliging maar wel voor de juiste werking is er voor RRSF nog een logmode- en logtab-parameter in de LU-definitie noodzakelijk. Deze zijn:

```
LURRSFA1   APPL   ACBNAME=LURRSFA1,
                  APPC=YES,
                  MODETAB=MTAPPC1,
                  DLOGMODE=IRRMODE
```

8.2.4 Netwerk access control

VTAM heeft verschillende beveiligingsmechanismen. Hieronder zijn deze beschreven en het is van het grootste belang deze te implementeren omdat geen enkel ongeautoriseerd systeem zich als een RRSF-node kan voordoen en van de RRSF-server gebruik kan maken.

8.2.4.1 LU-LU verificatie

Om de RRSF-server te beschermen tegen ongeoorloofd gebruik door andere remote systemen moet de lokale LU uitsluitend kunnen worden benaderd door bekende remote-LUs. Dit is mogelijk door de verificatie te laten uitvoeren door een authenticatie mechanisme. Het authenticatie mechanisme tussen LUs is gebaseerd op het verifiëren van het aanwezig zijn van een gelijke sleutel op de lokale en remote LU. Dit mechanisme wordt LU-LU verificatie genoemd. Hieronder een voorbeeld hoe de implementatie voor twee systemen (RRSF-nodes) binnen RACF wordt gedefinieerd:

Voor de systemen *PRODA1* en *PRODR2* die een netwerksessie willen aangaan moet aan beide kanten in RACF dezelfde sessiesleutel worden gedefinieerd. Een voorbeeld:

Op *PRODA1* met lokale LU-naam *LURRSFA1*:

```
SETROPTS GENERIC(APPCLU)
SETROPTS CLASSACT(APPCLU)
SETROPTS AUDIT(APPCLU)

RDEFINE    APPCLU  NLLVRTK.LURRSFA1.LURRSFR2
                     SESSION(SESSKEY(DE5A3A3B))
                     OWNER(SECADM)  UACC(NONE)
```

Op *PRODR2* met lokale LU-naam *LURRSFR2*:

```
SETROPTS GENERIC(APPCLU)
SETROPTS CLASSACT(APPCLU)
SETROPTS AUDIT(APPCLU)

RDEFINE    APPCLU  NLLVRTK.LURRSFR2.LURRSFA1
                     SESSION(SESSKEY(DE5A3A3B))
                     OWNER(SECADM)  UACC(NONE)
```

Voor LU-LU-verificatie gebruikt VTAM, om performance redenen, in-storage beveiligingsprofielen. Het is daarom noodzakelijk de beveiligingsprofielen in VTAM voor de betreffende applicatie te vervangen (Refresh) nadat deze RACF zijn aangemaakt of aangepast. Het vervangen moet op beide systemen met het volgende VTAM-operatorcommando worden uitgevoerd:

Op systeem PRODA1:

```
MODIFY     NET,PROFILES,ID=LURRSFA1
```

Op systeem PRODR2:
MODIFY *NET*,PROFILES,ID=*LURRSFR2*

8.2.4.2 *Local LU toegang*

De voor RRSF gedefinieerde lokale LU mag uitsluitend door de RRSF-server worden gebruikt. Deze LU dient daarom voor gebruik door andere lokale applicaties te worden afgeschermd. Er hoeft geen specifieke toegang tot deze lokale LU voor het userid waaronder RRSF wordt uitgevoerd te worden gedefinieerd omdat de started task *RACF*, waaronder RRSF wordt uitgevoerd, als trusted started task is gedefinieerd. De implementatie op alle systemen die in het RRSF-netwerk zijn opgenomen:

```
SETROPTS GENERIC(APPL)
SETROPTS CLASSACT(APPL)
SETROPTS AUDIT(APPL)

RDEFINE   APPL  **   OWNER(SECADM)  UACC(NONE)

SETROPTS RACLIST(APPL)  REFRESH
```

8.2.4.3 *Remote LU toegang*

De lokale LU, gedefinieerd voor de RRSF-server, mag uitsluitend gebruikt worden door remote RRSF-servers. De userids van de remote RRSF-servers (het userid van het remote RACF-subsysteem) moet voor de lokale LU worden geautoriseerd. De implementatie voor systeem *PRODA1* en *PRODR2*:

Op systeem *PRODA1*,
voor het userid van het remote RACF-subsysteem (*RACF* op *PRODR2*):

```
SETROPTS GENERIC(APPCPORT)
SETROPTS CLASSACT(APPCPORT)
SETROPTS AUDIT(APPCPORT)

RDEFINE   APPCPORT  LURRSFR2  OWNER(SECADM)  UACC(NONE)
PERMIT    LURRSFR2  CLASS(APPCPORT)  ACCESS(READ)  ID(RACF)

SETROPTS RACLIST(APPCPORT)  REFRESH
```

Op systeem *PRODR2*,
voor het userid van het remote RACF-subsysteem (*RACF* op *PRODA1*):

```
SETROPTS GENERIC(APPCPORT)
SETROPTS CLASSACT(APPCPORT)
SETROPTS AUDIT(APPCPORT)

RDEFINE   APPCPORT  LURRSFA1  OWNER(SECADM)  UACC(NONE)
PERMIT    LURRSFA1  CLASS(APPCPORT)  ACCESS(READ)  ID(RACF)

SETROPTS RACLIST(APPCPORT)  REFRESH
```

8.2.4.4 *Conversation Security*

Voor het opzetten van een conversatie over een netwerksessie, geïnitieerd door een remote systeem moet identificatie en authenticatie van de remote-gebruiker worden uitgevoerd. De remote RRSF-nodes hebben echter niet de mogelijkheid om userid en wachtwoord te geven maar zijn wel vertrouwde (trusted) subsystemen. Voor de remote systemen kan daarom worden aangenomen dat identificatie en authenticatie reeds is uitgevoerd (Already Verified). VTAM op

het lokale systeem hoeft daarom geen verificatie meer uit te voeren. De VTAM APPL parameters:

```
LURRSFA1   APPL     ACBNAME=LURRSFA1,
                    SECACPT=ALREADYV
```

Het type beveiliging voor het opzetten van een conversatie met de LU moet 'Already Verified' zijn omdat de remote RRSF geen wachtwoord meegeeft maar wel een betrouwbare (trusted) LU is. De VTAM parameter SECACPT=ALREADYV is niet echt noodzakelijk omdat de RACF APPCLU definitie deze overruled waardoor RACF de controle heeft. De implementatie waarbij RACF de lokale LU *LURRSFA1* toestaat om een 'Already Verified' conversatie te accepteren met remote LU *LURRSFR2*. Beiden behoren tot hetzelfde netwerk *NLLVRTK*:

```
RDEFINE   APPCLU   NLLVRTK.LURRSFA1.LURRSFR2
                   SESSION(CONVSEC(ALREADYV)  SESSKEY(DE5A3A3B))
```

VTAM moet de conversatiecontrole uitvoeren. Dit moet in de LU-definitie worden aangegeven. De implementatie:

```
RACFRRSF   APPL     ACBNAME=RACFRRSF,
                    SECACPT=ALREADYV,
                    VERIFY=REQUIRED
```

8.3 Advanced Program to Program Commnunication / MVS (APPC/MVS)

8.3.1 Introductie

Voor gebruik van het APPC-protocol voor RRSF is het nodig om de APPC/MVS address space te activeren. APPC/MVS is een VTAM-applicatie die ondersteuning geeft aan de communicatie tussen VTAM en RRSF. RRSF is onderdeel van het RACF-subsysteem en gebruikt de mogelijkheid van APPC/MVS om zich als server te definiëren. Bij activering van het RACF-subsysteem kan met behulp van parameters of operatorcommando's de functionaliteit van de RRSF-server worden geactiveerd.

8.3.2 Installatie

8.3.2.1 Started Task

Voor communicatie tussen RRSF-nodes moet de APPC/MVS address space als started task worden geactiveerd. De implementatie:

```
ADDUSER  APPCMVS  DFLTGRP(STCGROUP)  OWNER(SECADM)
                  NOPASSWORD

SETROPTS GENERIC(STARTED)
SETROPTS CLASSACT(STARTED)

RDEFINE   STARTED  APPCMVS.*  STDATA(USER(APPCMVS)
                               GROUP(STCGROUP)  TRUSTED(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

8.3.2.2 Dataset protectie

De APPC/MVS programma datasets moeten met een afdoende datasetprofiel worden beveiligd. Een voorbeeld:

```
ADDSD     'SYS1.APPC.*'  OWNER(SECADM)  UACC(NONE)
```

```
PERMIT    'SYS1.APPC.*'  ID(STCGROUP)  ACCESS(READ)
```

Alhoewel de RRSF-server geen gebruik maakt van TP-profielen moet toch een TP-profile dataset worden gealloceerd om daarin een DB-token te definiëren. De VSAM-files voor de TP-profile dataset moet met een meer specifiek RACF-datasetprofiel worden beschermd zodat later uitsluitend de Security administrator toegang hiertoe kan krijgen. In een TP-profile dataset kan gevoelige informatie staan. Bij verwijdering van de VSAM-cluster uit het systeem moet dan ook de inhoud worden gewist. Een voorbeeld:

```
ADDSD  'SYS1.RACFRRSF.TPROFILE'  OWNER(SECADM)  UACC(NONE)
                                 ERASE
```

8.3.2.3 *Netwerk definities*

Aan APPC/MVS moet bekend worden gemaakt welke LU deze moet gebruiken. Welke LU APPC/MVS gebruikt wordt in een SYS1.PARMLIB APPCPMxx member gedefinieerd. Een voorbeeld van de APPC/MVS definities op systeem *PRODA1*:

```
LUADD      ACBNAME(LURRSFA1)
```

Voor RRSF wordt niet de standaard APPC/MVS scheduler (ASCH) toegepast maar is een specifieke RRSF-server ontwikkeld. De RRSF-server programmatuur is onderdeel van het RACF-subsysteem. Bij APPC/MVS moet worden aangegeven dat niet de ASCH-scheduler wordt gebruikt. APPC/MVS ondersteund in dit geval uitsluitend het transport en queuing van de transactieaanvragen voor de RRSF-server. De implementatie:

```
LUADD      ACBNAME(LURRSFA1)
NOSCHED
```

De VSAM-cluster voor TP-profielen moet aan APPC/MVS bekend worden gemaakt. Een voorbeeld voor de definitie van een VSAM-cluster voor TP-profielen in een SYS1.PARMLIB APPCPMxx member:

```
LUADD      ACBNAME(LURRSFA1)
NOSCHED
TPDATA(SYS1.RACFRRSF.TPROFILE)
```

8.3.3 APPC/MVS Administration security

Om het gebruik van de APPC/MVS en de RRSF-server te beperken tot uitsluitend het RACF-subsysteem moeten alle aanwezige beveiligingsmechanismen worden gedefinieerd om een afdoende bescherming te garanderen.

8.3.3.1 *Administration utility control*

Met behulp van de applicatie ATBSDFMU moet een DB-token in de TP-profiledatabase worden gedefinieerd. Het is daarbij noodzakelijk om de juiste applicatie te gebruiken wanneer een DB-token wordt gedefinieerd. Het gebruik van het juiste programma en entry points voor het beheer van de TP-profiledatabase wordt met de general resource class PROGRAM gecontroleerd. De implementatie:

```
RDEFINE   PROGRAM  ATBINMIG  ADDMEM('SYS1.MIGLIB'/******/PADCHK)
                                OWNER(SECADM)  UACC(NONE)
RDEFINE   PROGRAM  ATBSDEPE  ADDMEM('SYS1.MIGLIB'/******/PADCHK)
                                OWNER(SECADM)  UACC(NONE)
RDEFINE   PROGRAM  ATBSDFMU  ADDMEM('SYS1.MIGLIB'/******/PADCHK)
                                OWNER(SECADM)  UACC(NONE)
RDEFINE   PROGRAM  ATBSDFCS  ADDMEM('SYS1.MIGLIB'/******/PADCHK)
                                OWNER(SECADM)  UACC(NONE)
```

```
RDEFINE   PROGRAM  ATBSDFM1  ADDMEM('SYS1.MIGLIB'/******/PADCHK)
                                OWNER(SECADM)  UACC(NONE)
RDEFINE   PROGRAM  *  ADDMEM('SYS1.LINKLIB'/******/NOPADCHK)
                                OWNER(SECADM)  UACC(NONE)
PERMIT    ATBSDFMU  CLASS(PROGRAM)  ID(SECADM)  ACCESS(READ)

SETROPTS  RACLIST(PROGRAM)  REFRESH
```

Uitsluitend de Security administrator mag toegang hebben tot TP-profiledatabase. Deze mag uitsluitend worden benaderd met de PROGRAM controlled applicatie ATBSDFMU. De implementatie:

```
PERMIT    'SYS1.RACFRRSF.TPROFILE'  ID(SECADM)
                WHEN(PROGRAM(ATBSDFMU))  ACCESS(UPDATE)
```

8.3.3.2 *Database token*

In de TP-profiledatabase moet een DB-token worden geplaatst met behulp van de APPC/MVS administratie applicatie ATBSDFMU. Een voorbeeld:

```
//XRIJL011  JOB  (1223,AMS),XRIJL01,REGION=4096K,CLASS=B,
//               MSGLEVEL=(1,1),NOTIFY=XRIJL01,MSGCLASS=P
//STEP1     EXEC PGM=ATBSDFMU
//SYSPRINT  DD  SYSOUT=*
//SYSSDOUT DD  SYSOUT=*
//SYSSDLIB  DD  DSN=SYS1.RACFRRSF.TPROFILE,DISP=SHR
//SYSIN      DD *
      DBMODIFY
      DBTOKEN(RRSFDBTK)
/*
```

Het DB-token is de belangrijkste identifier om de IRRRACF-transactie te beschermen. Het definiëren wijzigen of verwijderen van het DB-token in de TP-profile database moet met het APPCMVS.DBTOKEN-profiel in de general resource class FACILITY worden beveiligd. Uitsluitend Security administrators mogen DB-tokens definiëren, wijzigen of verwijderen. De implementatie:

```
RDEFINE   FACILITY   APPCMVS.DBTOKEN
                     OWNER(SECADM) UACC(NONE)
PERMIT    APPCMVS.DBTOKEN  CLASS(FACILITY)
                     ID(SECADM)  ACCESS(UPDATE)
```

8.3.3.3 *Transactie Profielen*

Alhoewel de IRRRACF-transactie niet in de TP-profile database is gedefinieerd wordt de autorisatiecontrole op het gebruik van de standaard RRSF-transactie IRRRACF door remote RRSF-servers wel op de TP-profile database (DB-token) uitgevoerd. Om remote RRSF-servers de IRRRACF-transactie te kunnen laten uitvoeren moeten deze het TP-beveiligingsprofiel voor de IRRRACF-transactie kunnen lezen. Hiervoor moet de IRRRACF-transactie voor remote RRSF-servers (remote RACF-subsysteemuserids) in de general resource class APPCTP worden geautoriseerd. De implementatie:

```
SETROPTS CLASSACT(APPCTP)
SETROPTS GENERIC(APPCTP)
RDEFINE   APPCTP  RRSFDBTK.SYS1.IRRRACF
                     OWNER(SECADM) UACC(NONE)
PERMIT    RRSFDBTK.SYS1.IRRRACF  CLASS(APPCTP)  ACCESS(READ)
                     ID(RACF)  userid van het remote RACF-subsysteem
```

```
SETROPTS RACLIST(APPCTP)   REFRESH
```

8.3.3.4 Server access to LU

Uitsluitend de juiste lokale RRSF-server mag als geautoriseerde RRSF-transactieserver optreden en de IRRRACF-transactie uitvoeren. Om de RRSF-server als APPC-server te definiëren moet deze in de general resource class APPCSERV voor de betreffende transactie worden gedefinieerd. De IRRRACF-transactie wordt geïdentificeerd met het DB-token van de TP-profile database en de transactienaam. De TP-profile database is met de LUADD parameter gekoppeld aan de voor RRSF gedefinieerde LU. Zo kan APPC/MVS een aanvraag voor de IRRRACF-transactie naar de geautoriseerde RRSF-server communiceren. De implementatie:

```
SETROPTS   CLASSACT(APPCSERV)
SETROPTS   GENERIC(APPCSERV)

RDEFINE    APPCSERV   RRSFDBTK.IRRRACF
                      OWNER(SECADM) UACC(NONE)
PERMIT     RRSFDBTK.IRRRACF   CLASS(APPCSERV)   ACCESS(READ)
                      ID(RACF)   userid van het lokale RACF-subsysteem

SETROPTS   RACLIST(APPCSERV)   REFRESH
```

8.4 RACF Remote Sharing Facility (RRSF)

8.4.1 Introductie

Elk z/OS-systeem of Sysplex heeft zijn eigen RACF-omgeving met een aparte RACF database voor gebruikersprofielen en beveiligingsprofielen. Door het gebruik van de RACF Remote Sharing Facility (RRSF) is het mogelijk gebruikers- en beveiligingsprofielen in de verschillende RACF-omgevingen vanuit één systeem over netwerken te beheren. RRSF synchroniseert, waar dit is aangegeven, de specifieke gebruikers- en beveiligingsprofielen. Voor een gebruiker, die op meerdere systemen of Sysplexen een userid heeft, is het niet meer noodzakelijk meerdere keren zijn of haar wachtwoord op de verschillende systemen te wijzigen. Met RRSF hoeft het wachtwoord maar op één systeem worden veranderd waarna het transparant voor de gebruiker door RRSF met de andere RACF-omgevingen worden gesynchroniseerd.

Voor de onderlinge communicatie tussen de RACF-systemen wordt het APPC (Advanced Program to Program Communication) communicatieprotocol gebruikt. APPC is een generiek communicatieprotocol voor verschillende toepassingen. De inrichting van APPC is daarom in een apart hoofdstuk beschreven. Voor de toepassing van APPC voor de RRSF zijn in dit hoofdstuk de specifieke aspecten opgenomen.

8.4.2 Installatie

Voordat wordt overgegaan op activering van RRSF, in het bijzonder de automatische wachtwoord synchronisatie en commando routing, zullen de beveiligingsprofielen handmatig moeten worden gesynchroniseerd voor de gewenste eindsituatie. Daarna kan RRSF worden geactiveerd om de beveiligingsprofielen synchroon te houden. Voor het synchroniseren heeft de leverancier 2 programma's. De eerste (IRRDBU00) is een RACF-database unload programma waarbij de RACF-profielen worden gelezen en voor verdere verwerking in een dataset geplaatst. De tweede (DBSYNC) is een REXX-programma waarmee de verschillende RACF-database unloads kunnen worden vergeleken en commando's worden gegenereerd om de verschillen op te

heffen. Na synchronisatie kunnen naar behoeften de verschillende RRSF-functies worden geactiveerd.

8.4.2.1 Netwerkcommunicatie

Alle communicatie tussen RRSF-nodes en opslag in de workdatasets binnen de RRSF-nodes wordt versleuteld met het Commercial Data Masking Facility (CDMF), een 40 DEA-key masking algoritme met vaste sleutel. CMDF geeft een minimale bescherming voor vertrouwelijkheid. Indien bijvoorbeeld een workdataset voor lezen toegankelijk is is de informatie niet direct interpreteerbaar maar met hulpmiddelen is de informatie relatief eenvoudig te ontsleutelen. Indien RRSF-communicatie over (onveilige) open netwerken plaatsvindt moet een sterker encryptie algoritme worden toegepast. Gedacht kan worden aan de mogelijkheden van VTAM sessie-encryptie.

8.4.2.2 Started Task

Voor het toepassen van RRSF moet het RACF-subsysteem worden geactiveerd. De beschrijving hiervan is te vinden in het hoofdstuk ‘RACF Subsysteem’

8.4.2.3 Workdatasets

Voor het tijdelijk vastleggen van de inkomende en uitgaande commando’s gebruikt RRSF VSAM-clusters als workdatasets. De naamgeving van de workdatasets is van belang voor de beveiliging hiervan. De datasetnaamgeving moet met het TARGET-commando worden geïnitieerd. Om een duidelijk beeld te krijgen waarvoor de datasetprofielen worden gebruikt moet de highlevel qualifier en de workdataset qualifier worden gedefinieerd. Een voorbeeld voor systeem *PRODA1*:

```
TARGET    NODE(RRSFA1)   LOCAL
          DESCRIPTION('RRSF node PRODA1')
          PROTOCOL(APPC(LUNAME(LURRSFA1)))   OPERATIVE
          PREFIX(SYS1)   WDSQUAL(RRSFA1)

TARGET    NODE(RRSFR2)
          DESCRIPTION('RRSF remote node PRODR2')
          PROTOCOL(APPC(LUNAME(LURRSFR2)))   OPERATIVE
          PREFIX(SYS1)   WDSQUAL(RRSFR2)
```

Een datasetprofiel moet elke workdatasets beveiligen en geen enkele gebruiker mag toegang hiertoe hebben. Het RACF-subsysteem, die de workdatasets dynamisch alloceerd en gebruikt, hoeft geen rechten op de dataset te hebben omdat het RACF-subsysteem als ‘trusted’ started task is gedefinieerd. Een voorbeeld:

```
ADDSD     'SYS1.RRSFA1.**'   OWNER(SECADM)   UACC(NONE)
```

Voor remote RRSF-systemen wordt zowel de lokale als de remote LU-naam een qualifier van de workdatasets. Hiermee wordt een workdataset voor elke remote RRSF gealloceerd. Met het gebruik van WDSQUAL-parameter wordt echter de remote LU-naam niet gebruikt maar de WDSQUAL-parameter behorende bij de remote RRSF-node. Een voorbeeld:

```
ADDSD     'SYS1.LURRSFA1.RRSFR2.**'   OWNER(SECADM)   UACC(NONE)
```

De suffixen van alle workdatasets eindigden op .INMSG en op .OUTMSG een generic datasetprofiel is voldoende om beide te beschermen.

8.4.2.4 Beheerprogramma IRRBRW00

De workdataset worden gebruikt om inkomende en uitgaande commando's vast te leggen. Indien problemen optreden met RRSF-communicatie of commandoverwerking zal de inhoud van de workdatasets voor diagnose moet worden gelezen. Voor het lezen van de VSAM-workdatasets kan het IRRBRW00 programma worden gebruik. Het IRRBRW00 programma kan de VSAM-records bewerken tot leesbare records. Het gebruik van IRRBRW00 moet echter worden beschermd tegen ongeautoriseerd gebruikt omdat anders de inhoud van beveiligingsgevoelige informatie door ongeautoriseerde personen kan worden gelezen. Uitsluitend een Security administrator mag IRRBRW00 gebruiken om de workdatasets te lezen. De implementatie:

```
RDEFINE   RRSFDATA  IRRBRW00  OWNER(SECADM)  UACC(NONE)
PERMIT    IRRBRW00  CLASS(RRSFDATA)
                        ID(SECADM)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

De Security administrator moet naast het gebruik van het IRRBRW00 programma ook toegang hebben tot de betreffende workdataset(s). Een voorbeeld:

```
PERMIT    'SYS1.RRSFA1.**'  ID(SECADM)  ACCESS(READ)
PERMIT    'SYS1.LURRSFA1.RRSFR2.**'  ID(SECADM)  ACCESS(READ)
```

8.4.2.5 RACF Operatorcommando's

Bij gebruik van RRSF dienen minimaal de sluitprofielen voor de RACF operatorcommando's TARGET in de general resource class OPERCMDS aanwezig te zijn. Zie ook het hoofdstuk 'RACF Operator Commands'. De voorbeelden:

```
RDEFINE   OPERCMDS  RACF.TARGET.**
                    OWNER(SECADM)  UACC(NONE)
PERMIT    RACF.TARGET.**  CLASS(OPERCMDS)
                    ID(SECADM)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  RACF.TARGET.WORKSPACE
                    OWNER(SECADM)  UACC(NONE)
PERMIT    RACF.TARGET.WORKSPACE  CLASS(OPERCMDS)
                    ID(OPRGRP)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  RACF.TARGET.LIST
                    OWNER(SECADM)  UACC(NONE)
PERMIT    RACF.TARGET.LIST  CLASS(OPERCMDS)
                    ID(OPRGRP)  ACCESS(UPDATE)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

8.4.3 Configuratie

De configuratie van de RRSF-nodes worden gedefinieerd in de RACF-parameterdataset. De inhoud van deze RACF-parameterdataset en de actieve RRSF-configuratie is de verantwoordelijkheid van de Security administrator. De verdere beschrijving en beveiliging is te vinden in het hoofdstuk 'RACF parameters'.

In een IRROPTxx member van de RACF-parameterdataset moeten de RRSF-configuratieparameters worden gedefinieerd. Hieronder is een voorbeeld gegeven van de configuratieparameters van systemen in een 2 nodes RRSF-configuratie:

Definities op systeem *PRODA1* voor het lokale systeem:

```
TARGET    NODE(RRSFA1)  LOCAL
          DESCRIPTION('RRSF node van systeem PRODA1')
          PROTOCOL(APPC(LUNAME(LURRSFA1)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFA1)
Definities op systeem PRODA1 voor het remote systeem:
TARGET    NODE(RRSFR2)
          DESCRIPTION('RRSF node van systeem PRODR2')
          PROTOCOL(APPC(LUNAME(LURRSFR2)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFR2)

Definities op systeem PRODR1 voor het lokale systeem:
TARGET    NODE(RRSFA1)
          DESCRIPTION('RRSF node van systeem PRODA1')
          PROTOCOL(APPC(LUNAME(LURRSFA1)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFA1)
Definities op systeem PRODR1 voor het remote systeem:
TARGET    NODE(RRSFR2)  LOCAL
          DESCRIPTION('RRSF node van systeem PRODR2')
          PROTOCOL(APPC(LUNAME(LURRSFR2)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFR2)
```

In een shared RACF-database-omgeving, waarbij meerdere systemen RACF-gegevens uit dezelfde database gebruiken, moet één systeem de overhand (MAIN) hebben. Door het gezamenlijk delen van de RACF-database behoren beide systemen, in het hieronder beschreven voorbeeld, tot dezelfde RRSF-node. Het voorbeeld:

```
TARGET    NODE(RRSFR3)  LOCAL  SYSNAME(SYSA)  MAIN
          DESCRIPTION('RRSF node van systeem PRODR2')
          PROTOCOL(APPC(LUNAME(LURRSFR2)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFR2)

TARGET    NODE(RRSFR3)  LOCAL  SYSNAME(SYSB)
          DESCRIPTION('RRSF node van systeem PRODR3')
          PROTOCOL(APPC(LUNAME(LURRSFR3)))  OPERATIVE
          PREFIX(SYS1)  WDSQUAL(RRSFR3)
```

Als RRSF uitsluitend wordt gebruikt voor bijvoorbeeld wachtwoord synchronisatie tussen verschillende userids op hetzelfde systeem, dan is er geen noodzaak VTAM-parameters te definiëren. Wel is het nodig RRSF als local-node met workdataset parameters te definiëren. Een voorbeeld:

```
TARGET    NODE(RRSFR1)  LOCAL
          DESCRIPTION('RRSF node van systeem PRODR1')
          PREFIX(SYS1)  WDSQUAL(RRSFD1)
```

8.4.4 Synchronisatie

8.4.4.1 Uitgangspunten

Beveiligingsniveau

Het uitwisselen van RACF-commando's door middel van RRSF dient uitsluitend uitgevoerd te kunnen worden tussen systemen met hetzelfde beveiligingsniveau of naar systemen met een lager beveiligingsniveau. Zo mogen bijvoorbeeld geen RACF-commando's of wachtwoord synchronisatie worden toegestaan vanuit een test- naar een productiesysteem. In het algemeen, een systeem met een lager beveiligingsniveau mag geen beheer kunnen voeren naar een systeem met een hoger beveiligingsniveau.

Automatisch versus specifieke synchronisatie

Met RRSF is het mogelijk om RRSF-complex breed wachtwoorden automatische tussen gelijke userids te synchroniseren en automatische RACF-commando's naar alle gelijke userids op de aangesloten RRSF-nodes te routeren. Indien deze aanpak te ruim is kan worden gekozen voor specifieke wachtwoord synchronisatie en commando routing. Met specifieke aanpak moet elke mogelijkheid van synchronisatie en routing met een RACLINK-commando worden gedefinieerd. In de praktijk gaat de voorkeur uit naar een RRSF-complex brede wachtwoord synchronisatie en commando routing eventueel beperkt voor bepaalde gebruikers of gebruikersgroepen. Een strikte voorwaarde voor RRSF-complex brede synchronisatie en routering is dat de betreffende systemen op dezelfde manier zijn gestandaardiseerd, in het bijzonder de naamgevingsconventie voor userids, groepen en beveiligingsprofielen. Een tweede voorwaarde is dat de aangesloten systemen hetzelfde beveiligingsniveau hebben.

Bij systemen die niet gestandaardiseerd zijn zal specifieke synchronisatie en routing moeten plaatsvinden. Reden hiervoor is dat bijvoorbeeld gelijke userids over verschillende systemen niet altijd tot dezelfde persoon behoren of dat gelijke beveiligingsprofielen niet altijd dezelfde resource beschermen.

8.4.5 Automatische synchronisatie

Met het RACF SET-commando moet automatisch wachtwoord en commando routing voor alle gebruikers worden geactiveerd. Het SET-commando is een RACF-subsysteemcommando en moet in het default IRROPT00 member van de RACF parameter library worden geplaatst. Door het plaatsen van het SET-commando in het default member krijgt men de meeste zekerheid dat de gekozen RRSF functionaliteit, automatische wachtwoord synchronisatie en commando routing, bij het opstarten van het RACF-subsysteem altijd wordt geactiveerd.

8.4.6 Automatische Wachtwoord synchronisatie

Voor activering van automatisch wachtwoord synchronisatie moet het SET AUTOPWD commando in IRROPT00 worden geplaatst. De implementatie:

```
SET  AUTOPWD
```

Met het SET AUTOPWD-commando wordt systeem breed automatische wachtwoord synchronisatie geactiveerd. De Security administrator zal echter op de hoogte moeten blijven of de wachtwoord synchronisatie effectief blijft functioneren. Een verstoring zal daarom moeten worden gesignaleerd naar het userid van de Security administrator. De implementatie:

```
SET  AUTOPWD  NOTIFY(FAIL(XKRET01)  OUTPUT(FAIL(XKRET01))
```

Door het definiëren van het SET-commando, met de juiste parameters, in een IRROPTxx member is het niet nodig om deze voor het SET-commando te autoriseren aangezien het RACF-subsysteem, die het SET-commando uitvoert, als trusted started task is gedefinieerd en daarom alle resources mag benaderen. Naast het RACF-subsysteem en de centrale Security administrator mag niemand anders het SET-operatorcommando uitvoeren. De implementatie:

```
RDEFINE   OPERCMDS  RACF.SET.AUTOPWD  UACC(NONE)
PERMIT    RACF.SET.AUTOPWD  CLASS(OPERCMDS)
                            ID(SECADM)  ACCESS(UPDATE)
```

Een voorbeeld van het RACF SET-operatorcommando waarbij het teken '#' als RACF subsysteem karakter is toegekend. Uitsluitend de Security administrator mag dit operator commando uitvoeren:

```
#SET  AUTOPWD
```

Het toestaan van automatische wachtwoordsynchronisatie heeft twee invalshoeken. Het wachtwoord wordt van iedere gebruiker gesynchroniseerd met eventueel een klein aantal gebruikers waarvoor dit niet moet gelden of voor geen enkel gebruiker geldt automatische wachtwoordsynchronisatie met een aantal uitzonderingen waarvoor het wel wordt toegestaan. Hieronder de twee mogelijkheden verder uitgewerkt.

8.4.6.1 Meerderheid gebruikers met automatische wachtwoordsynchronisatie

In een RRSF-omgeving waarbij de aangesloten systemen op dezelfde manier zijn gestandaardiseerd is het aannemelijk dat alle wachtwoorden tussen gelijke userids worden gesynchroniseerd. Een beperkt aantal gebruikers kan echter om praktische redenen hiervoor worden uitgesloten. Een voorbeeld:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.USER.PWSYNC  UACC(READ)
PERMIT    AUTODIRECT.*.USER.PWSYNC  CLASS(RRSFDATA)
                ID(XRIJL01)  ACCESS(NONE)
```

8.4.6.2 Minderheid gebruikers met automatische wachtwoordsynchronisatie

Indien in een RRSF-omgeving wordt besloten een beperkt aantal gebruikers hun wachtwoord over meerdere systemen automatisch te laten synchroniseren moet dit uitsluitend voor deze gebruikers worden geactiveerd. Overwogen dient te worden of bij deze keuze, de hierna beschreven, specifieke wachtwoord synchronisatie niet een betere optie is. Een voorbeeld:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.USER.PWSYNC  UACC(NONE)
PERMIT    AUTODIRECT.*.USER.PWSYNC  CLASS(RRSFDATA)
                ID(VERKOOP)  ACCESS(READ)
```

8.4.6.3 Beperking wachtwoord synchronisatie per RRSF-node

In een RRSF-complex met meerdere nodes is het niet altijd wenselijk wachtwoorden tussen alle aangesloten systemen te synchroniseren ook al zijn de userids gelijk. Dit kan worden beperkt door uitsluitend de RRSF-nodes te autoriseren waarmee wachtwoord synchronisatie moet plaatsvinden. Een voorbeeld waarbij vanuit RRSF-node *RRSFA1* uitsluitend wachtwoord synchronisatie mag plaatsvinden naar RRSF-node *RRSFR2*:

```
Commando uitgevoerd op RRSF-node RRSFA1:
RDEFINE   RRSFDATA  AUTODIRECT.RRSFR2.USER.PWSYNC
                OWNER(SECADM)  UACC(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Een voorbeeld waarbij wachtwoord synchronisatie van uitsluitend voor één groep op *RRSFA1* naar één remote RRSF-node *RRSFR2* plaatsvindt:

```
RDEFINE   RRSFDATA  AUTODIRECT.RRSFR2.USER.PWSYNC
                OWNER(SECADM)  UACC(NONE)
PERMIT    AUTODIRECT.RRSFR2.USER.PWSYNC  CLASS(RRSFDATA)
                ID(VERKOOP)  ACCESS(READ)
```

8.4.7 Automatische Commando routing

Voor het activeren van automatisch commando routing moet het SET AUTODIRECT commando in IRROPT00 worden geplaatst. De implementatie:

```
SET  AUTODIRECT
```

Met het SET AUTODIRECT-commando wordt de systeem brede automatische commando routing geactiveerd. De Security administrator zal echter op de hoogte moeten blijven of de

commando routing effectief blijft functioneren. Een verstoring zal daarom moeten worden gesignaleerd naar het userid van de Security administrator. De implementatie:

```
SET   AUTODIRECT   NOTIFY(FAIL(XKRET01)   OUTPUT(FAIL(XKRET01))
```

Door het definiëren van het SET-commando in een IRROPTxx member is het niet nodig het SET-commando voor het RACF-subsysteem te autoriseren aangezien het userid waaronder het wordt uitgevoerd als trusted started task is gedefinieerd. Naast het RACF-subsysteem en de centrale Security administrator mag niemand anders het SET-operatorcommando uitvoeren. De implementatie:

```
RDEFINE    OPERCMDS   RACF.SET.AUTODIRECT   UACC(NONE)
PERMIT     RACF.SET.AUTODIRECT   CLASS(OPERCMDS)
                                 ID(SECADM)   ACCESS(UPDATE)
```

Voorbeeld van het RACF SET-operatorcommando waarbij het teken '#' als RACF subsysteem prefixkarakter is toegekend. Uitsluitend de Security administrator mag dit operator commando uitvoeren:

```
#SET   AUTODIRECT
```

Het toestaan van automatische commando routing heeft twee invalshoeken. De commando's worden van iedere gebruiker gerouteerd met eventueel een klein aantal gebruikers waarvoor dit niet moet gelden of voor geen enkele gebruiker geldt automatische commando routing met een aantal uitzonderingen. Hieronder de twee mogelijkheden verder uitgewerkt.

8.4.7.1 Meerderheid gebruikers met automatische commando routing

Indien in een RRSF-omgeving het merendeel van de gebruikers commando's over de systemen mag routeren en voor een beperkt aantal gebruikers dit niet wordt toegestaan moet deze gebruikers worden uitgesloten. Een voorbeeld:

```
RDEFINE    RRSFDATA   AUTODIRECT.**   OWNER(SECADM)   UACC(READ)
PERMIT     AUTODIRECT.**    CLASS(RRSFDATA)
                 ID(INKOOP)   ACCESS(NONE)
PERMIT     AUTODIRECT.**    CLASS(RRSFDATA)
                 ID(XKRET01)   ACCESS(NONE)
```

8.4.7.2 Minderheid gebruikers met automatische commando routing

Indien in een RRSF-omgeving wordt besloten om voor een beperkt aantal gebruikers commando routing over meerdere systemen mogelijk te maken moet dit uitsluitend voor deze gebruikers worden geactiveerd. Overwogen dient te worden of bij deze keuze, de hierna beschreven specifieke commando routing niet een betere optie is. Een voorbeeld:

```
RDEFINE    RRSFDATA   AUTODIRECT.**   UACC(NONE)
PERMIT     AUTODIRECT.**  CLASS(RRSFDATA)
                 ID(VERKOOP)   ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)   REFRESH
```

8.4.7.3 Beperking commando routing per RRSF-node

In een RRSF-complex met meerdere nodes is het niet altijd wenselijk commando routing naar alle aangesloten systemen uit te voeren. Commando routing moet daarom worden beperkt tot de gewenste RRSF-nodes. Een voorbeeld van een commando, uitgevoerd op *RRSFA1*, waarbij door alle gebruikers uitsluitend commando routing mag plaatsvinden van RRSF-node *RRSFA1* naar *RRSFR2*:

```
RDEFINE    RRSFDATA   AUTODIRECT.RRSFR2.**
```

```
                                        OWNER(SECADM)  UACC(READ)
```

Een voorbeeld van een commando, uitgevoerd op *RRSFA1*, waarbij door uitsluitend de gebruikersgroep *VERKOOP* commando routing mag uitvoeren van RRSF-node *RRSFA1* naar *RRSFR2*:

```
RDEFINE   RRSFDATA  AUTODIRECT.RRSFR2.**
                    OWNER(SECADM)  UACC(NONE)
PERMIT    AUTODIRECT.RRSFR2.**  CLASS(RRSFDATA)
                    ID(VERKOOP)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

8.4.7.4 Routering per resource class

Indien automatische commando routering wordt geactiveerd is het noodzakelijk te bepalen of een gedeelte van de general resource classes of mogelijk alle hiervoor in aanmerking komen. Elke general resource class kan afzonderlijk worden gedefinieerd om in aanmerking te komen voor commando routing. Een voorbeeld waarbij de PERMITS op CICS-transacties, gegeven door de gebruikersgroep *VERKOOP*, naar RRSF-node *RRSFR2* worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.RRSFR2.TCICSTRN.PERMIT
                    OWNER(SECADM)  UACC(NONE)
PERMIT    AUTODIRECT.RRSFR2.TCICSTRN.PERMIT  CLASS(RRSFDATA)
                    ID(VERKOOP)  ACCESS(READ)
```

Een voorbeeld waarbij alle CONNECTS naar alle RRSF-nodes worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.USER.CONNECT
                    OWNER(SECADM)  UACC(READ)
```

Een voorbeeld waarbij alle ADDSD naar alle RRSF-nodes worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.DATASET.ADDSD
                    OWNER(SECADM)  UACC(READ)
```

Een voorbeeld waarbij alle PERMITS op datasetprofielen naar alle RRSF-nodes worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.DATASET.PERMIT
                    OWNER(SECADM)  UACC(READ)
```

Een voorbeeld waarbij het SETROPTS-commando naar alle RRSF-nodes worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.RACF.SETROPTS
                    OWNER(SECADM)  UACC(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

8.4.8 Specifieke synchronisatie

Specifieke synchronisatie wordt door middel van het RACLINK-commando gedefinieerd. Met het RACLINK-commando wordt een specifieke relatie (ID association) gelegd tussen twee of meerdere userids. Na het definiëren van een specifieke relatie kan RRSF tussen deze geassocieerde userids de wachtwoorden synchroniseren en commando's routeren. Of wachtwoord synchronisatie en commando routing moet plaatsvinden moet afzonderlijk in het RACLINK-commando worden aangegeven en daarnaast worden geautoriseerd. De relaties kunnen tussen userids binnen hetzelfde systeem of tussen gelijke en ongelijke userids, op remote systemen worden gedefinieerd. De relaties kunnen, indien voor het RACLINK-commando geautoriseerd, door de gebruikers van de verschillende userids of door een Security administrator worden gedefinieerd.

8.4.8.1 Specifieke commando routing

Voor activering van specifieke commando routing, waarna RACF-commando's tussen de geassocieerde userids kunnen worden gerouteerd, moeten relaties tussen betreffende userids met behulp van het RACLINK-commando zijn gedefinieerd. Deze relaties kunnen door een Security administrator of door de gebruikers van de userids worden gedefinieerd. Een voorbeeld waarbij een relatie tussen userid *XRIJL01* en userid *XKRET01* wordt opgezet.

RACLINK ID(*XRIJL01*) **DEFINE(*XKRET01*)**

Het userid *XKRET01* of een Security administrator moet nog wel de relatie goedkeuren voordat dit effectief wordt. Daarvoor moet het volgende RACLINK-commando door *XKRET01* voor goedkeuring worden gegeven.

RACLINK ID(*XKRET01*) **APPROVE**(*XRIJL01*)

Of als de relatie niet op prijs wordt gesteld kan deze ongedaan worden gemaakt:

RACLINK ID(*XKRET01*) **UNDEFINE**(*XRIJL01*)

Userid *XRIJL01* wil een relatie opzetten met userid *XKRET01* waarbij de opzet is om RACF-commando's die door *XRIJL01* wordt gegeven ook door het andere userid *XKRET01* moet worden uitgevoerd. Het is niet de bedoeling dat commando's door userid *XKRET01* worden uitgevoerd ook onder userid *XRIJL01* worden uitgevoerd. Een voorbeeld:

RACLINK ID(*XRIJL01*) DEFINE(*XKRET01*) **MANAGED**

Het userid *XKRET01* moet nog wel de relatie goedkeuren voordat dit effectief wordt. Daarvoor moet het RACLINK-commando door *XKRET01* voor goedkeuring worden gegeven.

RACLINK ID(*XKRET01*) APPROVE(*XRIJL01*)

Of als de relatie niet op prijs wordt gesteld kan deze ongedaan worden gemaakt:

RACLINK ID(*XKRET01*) UNDEFINE(*XRIJL01*)

Opmerking: bij een MANAGED relatie wordt het wachtwoord niet gesynchroniseerd.

Userid *XRIJL01* wil een relatie opzetten met userid *XKRET02* op een ander systeem (*RRSFR2*) waarbij de opzet is om RACF-commando's die door *XRIJL01* op systeem *RRSFA1* worden gegeven ook door het andere userid *XKRET02* wordt uitgevoerd. Andersom is niet de bedoeling (MANAGED). Een voorbeeld:

RACLINK ID(*XRIJL01*) DEFINE(***PRODA1.****XKRET02*) MANAGED

Het userid *XKRET01* moet nog wel de relatie goedkeuren voordat dit effectief wordt. Daarvoor moet het *RACLINK* commando door *XKRET01* voor goedkeuring worden gegeven.

RACLINK ID(*XKRET02*) APPROVE(***PRODR2.****XRIJL01*)

Of als dit niet op prijs wordt gesteld en de relatie niet wordt aangegaan:

RACLINK ID(*XKRET02*) UNDEFINE(***PRODR2.****XRIJL01*)

Het definiëren van relaties tussen userids kan ook naar individuele gebruikers of gebruikersgroepen worden gedelegeerd. Voor gebruik van het RACLINK-commando moeten de individuele gebruikers of gebruikersgroepen worden geautoriseerd. Een voorbeeld:

```
RDEFINE   RRSFDATA  RACLINK.DEFINE.*
                        OWNER(SECADM)  UACC(NONE)
PERMIT    RACLINK.DEFINE.*  CLASS(RRSFDATA)
                        ID(VERKOOP)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Of het RACLINK-commando is voor iedereen toegestaan met beperking van een aantal personen.

```
RDEFINE   RRSFDATA  RACLINK.DEFINE.*
                        OWNER(SECADM)  UACC(READ)
PERMIT    RACLINK.DEFINE.*  CLASS(RRSFDATA)
                        ID(INKOOP)  ACCESS(NONE)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Met het RACLINK-commando wordt de mogelijkheid gemaakt om RACF-commando's vanuit TSO naar de userids, waarvoor een specifieke relaties is gedefinieerd, te routeren. Het feitelijke routeren moet met behulp van de AT-parameter in de betreffende RACF-commando's worden aangegeven. De mogelijkheid voor een gebruiker om de AT-parameter te kunnen gebruiker moet worden gedefinieerd. Een voorbeeld waarvoor gebruikersgroep *GROEPR* hiervoor is geautoriseerd:

```
RDEFINE   RRSFDATA  DIRECT.*  OWNER(SECADM)  UACC(NONE)
PERMIT    DIRECT.*  CLASS(RRSFDATA)
                        ID(GROEPR)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Het gebruik van specifieke commando routing heeft zoals aangegeven een aantal stappen. Als eerste moet een gebruiker geautoriseerd zijn voor het gebruik van het RACLINK-commando, daarna moet de gebruiker een relatie aanbrengen, verder moet de gebruiker worden geautoriseerd om de relatie te mogen gebruiken waarna deze een RACF-commando met de AT-parameter kan uitvoeren. Een voorbeeld van het gebruik van de AT-parameter:

```
PERMIT    'XRIJL01.PROGRAM.LOAD'  ID(XKRET01)  ACCESS(READ)
                                  AT(RRSFA1)
```

8.4.8.2 Specifieke wachtwoord synchronisatie

Voor het activeren van specifieke wachtwoord synchronisatie moet het SET PWSYNC commando in IRROPT00 worden geplaatst. De implementatie:

```
SET  PWSYNC
```

Met het SET PWSYNC-commando wordt de specifieke wachtwoord synchronisatie geactiveerd. De Security administrator zal echter op de hoogte moeten blijven of de specifieke wachtwoord synchronisatie effectief blijft functioneren. Een verstoring zal daarom moeten worden gesignaleerd naar het userid van de Security administrator. De implementatie:

```
SET  PWSYNC  NOTIFY(FAIL(XKRET01)  OUTPUT(FAIL(XKRET01))
```

Door het definiëren van het SET-commando in een IRROPTxx member is het niet nodig het SET-commando voor het RACF-subsysteem te autoriseren aangezien het userid waaronder het wordt uitgevoerd als trusted is gedefinieerd. Naast het RACF-subsysteem en de Security administrator mag niemand anders het SET-operatorcommando uitvoeren. De implementatie:

```
RDEFINE   OPERCMDS  RACF.SET.PWSYNC
                        OWNER(SECADM)  UACC(NONE)
```

```
PERMIT      RACF.SET.PWSYNC  CLASS(OPERCMDS)
                             ID(SECADM)  ACCESS(UPDATE)
```

Voor activering van specifieke wachtwoord synchronisatie tussen userids moeten relaties met behulp van het RACLINK-commando door een Security administrator worden gedefinieerd. Een voorbeeld waarbij userid *XRIJL01* een relatie wil opzetten met userid *XKRET01* waarbij de opzet is om naar beide kanten van de relatie wachtwoord synchronisatie toe te staan. Een voorbeeld:

```
RACLINK    ID(XRIJL01)  DEFINE(XKRET01)  PEER(PWSYNC)
```

Het userid *XKRET01* moet nog wel de relatie goedkeuren voordat deze effectief wordt. Daarvoor moet het *RACLINK*-commando door *XKRET01* voor goedkeuring worden gegeven.

```
RACLINK    ID(XKRET01)  APPROVE(XRIJL01)
```

Of als dit niet op prijs wordt gesteld en de relatie niet wordt aangegaan:

```
RACLINK    ID(XKRET01)  UNDEFINE(XRIJL01)
```

Het definiëren van relaties tussen userids kan ook naar individuele gebruikers of gebruikersgroepen worden gedelegeerd. Voor gebruik van het RACLINK-commando met de mogelijkheid voor wachtwoord synchronisatie moeten de individuele gebruikers of gebruikersgroepen worden geautoriseerd. Een voorbeeld waarbij de gebruikersgroep *VERKOOP* de specifieke relaties mag definiëren:

```
RDEFINE    RRSFDATA  RACLINK.PWSYNC.*
                     OWNER(SECADM) UACC(NONE)
PERMIT     RACLINK.PWSYNC.*  CLASS(RRSFDATA)
                     ID(VERKOOP)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Of het RACLINK-commando is voor iedereen toegestaan met beperking van een aantal personen.

```
RDEFINE    RRSFDATA  RACLINK.PWSYNC.*
                     OWNER(SECADM) UACC(READ)
PERMIT     RACLINK.PWSYNC.*  CLASS(RRSFDATA)
                     ID(INKOOP)  ACCESS(NONE)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

Nadat het RACLINK-commando voor wachtwoord synchronisatie is gedefinieerd moet de gebruiker nog wel hiervoor geautoriseerd. Een voorbeeld:

```
RDEFINE    RRSFDATA  PWSYNC  OWNER(SECADM)  UACC(NONE)
PERMIT     PWSYNC  CLASS(RRSFDATA)
           ID(VERKOOP)  ACCESS(READ)

SETROPTS RACLIST(RRSFDATA)  REFRESH
```

8.4.9 Gecombineerde Automatische en Specifieke synchronisatie

Voor een optimale beheerbaarheid van een RRSF-omgeving is het streven dat de aangesloten systemen volledig met dezelfde standaard zijn geïmplementeerd en dat de automatische wachtwoord synchronisatie en automatische commando routing is geactiveerd. Echter in deze omgeving kan het ook noodzakelijk zijn om specifieke wachtwoord synchronisatie en automatische commando routing te activeren.

Een praktisch voorbeeld hiervan is het synchroniseren van wachtwoorden tussen twee verschillende userids van dezelfde gebruiker. Als voorbeeld een gebruiker met userid *XRIJL01* heeft ook userid *XRIJL02* waarvan het wachtwoord tussen de twee userids moet worden gesynchroniseerd. De beide userids zijn zowel op RRSF-node *RRSFA1* als op *RRSFR2* gedefinieerd en automatische wachtwoord synchronisatie is geactiveerd, de voorgestelde optimale RRSF-implementatie. Voor het synchroniseren van het wachtwoord tussen de 2 verschillende userids moet echter een specifieke relatie met een RACLINK PEER(PWSYNC) worden gedefinieerd. Indien daarna het wachtwoord van *XRIJL01* op RRSF-node *RRSFA1* wordt veranderd zal het wachtwoord van *XRIJL02* op *RRSFA1* met specifieke wachtwoord synchronisatie worden aangepast en zal het userid *XRIJL01* op RRSF-node *RRSFR2* met automatische wachtwoord synchronisatie worden aangepast. Het userid *XRIJL02* op RRSF-node *RRSFR2* zal niet worden aangepast omdat het wachtwoord van userid *XRIJL02* op *RRSFA1* wordt aangepast doormiddel van een specifieke relatie (RACLINK) en daarbij niet met een automatische wachtwoord synchronisatie wordt doorgegeven. Om toch ook het wachtwoord van userid *XRIJL02* op *RRSFR2* te synchroniseren moet ook op *RRSFR2* een specifieke relatie tussen de userids *XRIJL01* en *XRIJL02* worden gedefinieerd.

In het algemeen kan worden gezegd dat een specifieke relaties tussen verschillende userids op verschillende systemen op gelijke wijze moet worden gedefinieerd. Als automatische wachtwoord synchronisatie is geactiveerd mogen geen specifieke relaties tussen gelijke userids worden gedefinieerd. Het gevolg zou zijn dat dubbele wachtwoord synchronisatie ontstaat.

8.4.10 Adhoc commando routing

Om profielen op een adhoc-basis, veelal voor reparatiewerkzaamheden, vanuit een (centrale) RRSF-node op remote RRSF-nodes te definiëren is de ONLYAT-parameter voor de betreffende RACF-commando's aanwezig. Het gebruik van de ONLYAT-parameter is voorbehouden aan de Security administrator (met het SPECIAL attribuut) en hoeft daarom niet verder te worden geautoriseerd. Een relatie tussen de verschillende userids is niet nodig wel moet het target userid de autorisatie of privileges hebben om het commando te kunnen uitvoeren. Een voorbeeld waarbij een Security administrator een PERMIT geeft voor een remote RRSF-node, uitgevoerd onder hetzelfde userid als van de Security administrator:

```
PERMIT    SYS1.LINKLIB  ID(OPRGRP)  ACCESS(READ)
          ONLYAT(RRSFR2)
```

Als de Security administrator onder een ander userid het commando laat uitvoeren moet er een specifieke relatie gedefinieerd zijn en het userid waaronder het commando wordt uitgevoerd moet de SPECIAL attribuut hebben. Een voorbeeld voor een PERMIT-commando uit te voeren op RRSF-node *RRSFR2*:

```
PERMIT    SYS1.LINKLIB  ID(OPRGRP)  ACCESS(READ)
          ONLYAT(RRSFR2.XKRET01)
```

8.4.11 Applicatie commando routing

RACF-profielen die applicaties definiëren door middel van bijvoorbeeld de RACROUTE macro interface kunnen eveneens worden gerouteerd. Voor het activeren van automatisch commando routing voor commando's uitgevoerd door applicaties moet het SET AUTOAPPL commando in IRROPT00 worden geplaatst. De implementatie:

```
SET  AUTOAPPL
```

Met het SET AUTOAPPL-commando wordt de systeem brede automatische commando routing geactiveerd. De Security administrator zal echter op de hoogte moeten blijven of de commando

routing effectief blijft functioneren. Een verstoring zal daarom moeten worden gesignaleerd naar het userid van de Security administrator. De implementatie:

```
SET  AUTOAPPL  NOTIFY(FAIL(XKRET01)  OUTPUT(FAIL(XKRET01))
```

Door het definiëren van het SET-commando in een IRROPTxx member is het niet nodig het SET-commando voor het RACF-subsysteem te autoriseren aangezien het userid waaronder het wordt uitgevoerd als trusted started task is gedefinieerd. Naast het RACF-subsysteem en de Security administrator mag niemand anders het SET-operatorcommando uitvoeren. De implementatie:

```
RDEFINE   OPERCMDS  RACF.SET.AUTOAPPL
                    OWNER(SECADM) UACC(NONE)
PERMIT    RACF.SET.AUTODIRECT  CLASS(OPERCMDS)
                    ID(SECADM)  ACCESS(UPDATE)
```

Voorbeeld van het RACF SET-operatorcommando waarbij het teken '#' als RACF-subsysteem prefixkarakter is toegekend. Uitsluitend de Security administrator mag dit operator commando uitvoeren:

```
#SET  AUTOAPPL
```

Het routeren van commando's door applicaties uitgevoerd wordt met AUTODIRECT-profielen die afsluiten met de qualifier .APPL gecontroleerd. Een voorbeeld waarbij alle RACF-definities, die applicaties via de RACROUTE maken, naar alle aangesloten RRSF-nodes worden gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.*.APPL
                    OWNER(SECADM) UACC(READ)
```

Een voorbeeld waarbij de RACF-definities, gemaakt door een applicatie die wordt uitgevoerd onder het userid *STCGROUP*, naar alle aangesloten RRSF-nodes wordt gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.*.*.APPL  UACC(NONE)
PERMIT    AUTODIRECT.*.*.APPL  CLASS(RRSFDATA)
                    ID(STCGROUP)  ACCESS(READ)
```

Een voorbeeld waarbij RACF-definities, door een applicatie met een RACROUTE macro gegenereerd, uitsluitend voor de TCICSTRN general resource class naar één RRSF-node (*RRSFR2*) wordt gerouteerd:

```
RDEFINE   RRSFDATA  AUTODIRECT.RRSFR2.TCICSTRN.APPL
                    UACC(NONE)
PERMIT    AUTODIRECT.RRSFR2.TCICSTRN.APPL  CLASS(RRSFDATA)
                    ID(CICSADM)  ACCESS(READ)
```

Aangezien de DATASET general resource class 2 wezenlijk verschillende profielen bevat, voor DASD datasets en voor TAPE datasets, zijn hiervoor verschillende profielen in de RRSFDATA class gedefinieerd. Een voorbeeld voor routering van DASD datasetprofielen naar uitsluitend de RRSF-node *RRSFR2*:

```
RDEFINE   RRSFDATA  AUTODASD.RRSFR2.DATASET.APPL
                    OWNER(SECADM)  UACC(NONE)
PERMIT    AUTODASD.RRSFR2.DATASET.APPL  CLASS(RRSFDATA)
                    ID(SMSSTC)  ACCESS(READ)
```

Een voorbeeld voor routering van TAPE datasetprofielen naar uitsluitend *RRSFR2*:

```
RDEFINE   RRSFDATA  AUTOTAPE.RRSFR2.DATASET.APPL
                    UACC(NONE)
PERMIT    AUTOTAPE.RRSFR2.DATASET.APPL  CLASS(RRSFDATA)
                    ID(HSMSTC)  ACCESS(READ)
```

9 z/OS faciliteiten

9.1 Introductie

Dit hoofdstuk geeft de mogelijkheden aan welke faciliteiten, die een integraal onderdeel van het z/OS platform zijn, door middel van RACF kunnen worden beschermd.

9.2 Status

Dit hoofdstuk is gereed voor commentaar.

9.3 Consoles

Het gebruik van consoles dient door RACF te worden gecontroleerd. De consoles dienen in z/OS en RACF met een herkenbare naam te worden gedefinieerd. De implementatie:

```
RDEFINE   CONSOLE  MST1  UACC(NONE)

PERMIT    MST1  CLASS(CONSOLE)  ID(OPRGRP JONES)  ACCESS(READ)

SETROPTS CLASSACT(CONSOLE)
```

9.4 Operator commands

Alle operator-commando's dienen door het beveiligingssysteem te worden gecontroleerd. De implementatie:

```
RDEFINE   OPERCMDS  **  UACC(NONE)
```

De mogelijkheid voor het uitvoeren van de commando's voor de verschillende systemen, subsystemen en faciliteiten dienen volgens de geldende autorisatiematrix te zijn geautoriseerd. Hieronder een aantal implementatievoorbeelden:

```
SETROPTS GENERIC(OPERCMDS)

RDEFINE   OPERCMDS  MVS.DISPLAY.JOB.**  UACC(NONE)
PERMIT    MVS.DISPLAY.JOB.**  CLASS(OPERCMDS)
          ID(OPRGRP  OPRJUN)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  MVS.CANCEL.JOB.**  UACC(NONE)
PERMIT    MVS.CANCEL.JOB.**  CLASS(OPERCMDS)
          ID(OPRGRP)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  JES.**  UACC(NONE)
PERMIT    JES.**  CLASS(OPERCMDS)
          ID(OPRGRP)  ACCESS(UPDATE)

SETROPTS CLASSACT(OPERCMDS)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

Uitsluitend door MVS en JES2 herkende operator commando's worden uitgevoerd. Onbekende MVS- en JES2-commando's worden door een speciaal RACF-profile (UNKNOWN) ondervangen. Elk gebruik van het profiel moet worden gelogd aangezien het gebruik van het profiel een verkeerd commando gebruik aangeeft. De implementatie:

```
RDEFINE   OPERCMDS  MVS.UNKNOWN   UACC(NONE)  AUDIT(ALL)
RDEFINE   OPERCMDS  JES2.UNKNOWN  UACC(NONE)  AUDIT(ALL)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

Opmerking:
Niet alle subsystemen geven het userid van de gebruiker door voor command-autorisatie, maar het userid van de betreffende started task. Bijvoorbeeld bij Netview, waarbij Netview een gebruiker zelf autoriseert voor console commando's, wordt de autorisatie check door RACF uitgevoerd met behulp van het Netview Started Task userid. Een andere optie van Netview is dat deze het userid van de gebruiker (aanvrager) door geeft zodat RACF command autorisatie kan controleren. Uitsluitend subsystemen die het userid van de gebruiker doorgeven mogen console commando's uitvoeren.

9.4.1 RACF Operator Commands

Indien het RACF-subsysteem (*RACF* is hier gekozen als subsysteem naam in de IEFSSNxx parmlib member) actief is zal voor operationele aspecten met dit subsysteem moeten worden gecommuniceerd. Dit kan door specifieke RACF-subsysteem operator commando's. Deze commando's dienen voor het gebruik door Operators of Security administrators te worden geautoriseerd. Minimaal dienen sluitprofielen voor DISPLAY, RESTART, SET, SIGNOFF en STOP aanwezig te zijn. Enige voorbeelden:

```
RDEFINE   OPERCMDS  RACF.DISPLAY.**  UACC(NONE)
PERMIT    RACF.DISPLAY.**  CLASS(OPERCMDS)
          ID(SECADM OPRGRP)  ACCESS(READ)

RDEFINE   OPERCMDS  RACF.RESTART.**  UACC(NONE)
PERMIT    RACF.RESTART.**  CLASS(OPERCMDS)
          ID(SECADM OPRGRP)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  RACF.SET.**  UACC(NONE)
PERMIT    RACF.SET.**  CLASS(OPERCMDS)
          ID(SECADM OPRGRP)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  RACF.SIGNOFF  UACC(NONE)
PERMIT    RACF.SIGNOFF  CLASS(OPERCMDS)
          ID(SECADM OPRGRP)  ACCESS(UPDATE)

RDEFINE   OPERCMDS  RACF.STOP  UACC(NONE)
PERMIT    RACF.STOP  CLASS(OPERCMDS)
          ID(SECADM OPRGRP)  ACCESS(UPDATE)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

Niet alleen de RACF-subsysteemcommando's moeten worden beschermd voor ongeautoriseerd gebruik maar ook de RACF-commando's zelf. De uitvoering van RACF-commando's door het RACF-subsysteem zal als eerste in een Role Based Access Control matrix moeten worden gedefinieerd. Alle RACF-commando's moeten worden beveiligd met de general resource class OPERCMDS. Enige voorbeelden:

```
RDEFINE   OPERCMDS  RACF.LISTUSER  UACC(NONE)
PERMIT    RACF.LISTUSER  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(READ)

RDEFINE   OPERCMDS  RACF.LISTGRP  UACC(NONE)
PERMIT    RACF.LISTGRP  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(READ)
```

```
RDEFINE   OPERCMDS  RACF.ALTUSER  UACC(NONE)
PERMIT    RACF.ALTUSER  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(READ)

RDEFINE   OPERCMDS  RACF.SETROPTS  UACC(NONE)
PERMIT    RACF.SETROPTS  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(UPDATE)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

Voor het gebruik van RACF-commando's dient een sluitprofiel te worden gedefinieerd zodat niet gedefinieerde RACF-commando's worden beschermd. De implementatie:

```
RDEFINE   OPERCMDS  RACF.**  UACC(NONE)
PERMIT    RACF.**  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(UPDATE)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

Uitsluitend door het RACF-subsysteem herkende operator commando's worden uitgevoerd. Onbekende RACF-commando's worden door een speciaal RACF-profile (UNKNOWN) ondervangen. Elk gebruik van een RACF-commando dat niet als zodanig wordt herkend zal worden ondervangen met het UNKNOWN-profiel waardoor alle commandinformatie wordt gelogd. De implementatie:

```
RDEFINE   OPERCMDS  RACF.UNKNOWN  UACC(NONE)  AUDIT(ALL)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

9.4.2 RACF Security administration Commands

Indien het RACF-subsysteem (*RACF* is hier de gekozen subsysteemnaam) actief is kan met behulp van MVS operator commando's RACF Security administrator commando's worden gegeven. Uitsluitend een Security administrator mag RACF-commando's geven en daarom moeten deze voor andere worden afgeschermd. De implementatie:

```
RDEFINE   OPERCMDS  RACF.**  UACC(NONE)
PERMIT    RACF.**  CLASS(OPERCMDS)
          ID(SECADM)  ACCESS(UPDATE)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

9.5 Dump datasets (Program dumps)

Het maken van een programmadump waarvan het programma execute controlled of program controlled is dient geautoriseerd te worden. De implementatie:

```
RDEFINE   FACILITY  IEAABD.DMPAUTH  UACC(NONE)

PERMIT    IEAABD.DMPAUTH  CLASS(FACILITY)
          ID(TESTGRP)  ACCESS(READ)

SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST(FACILITY)  REFRESH
```

Het maken van een programmadump van programma's die APF geautoriseerd zijn dient gecontroleerd te worden. De implementatie:

```
RDEFINE   FACILITY  IEAABD.DMPAKEY  UACC(NONE)
```

```
PERMIT     IEAABD.DMPAKEY   CLASS(FACILITY)
           ID(SYSPRO)   ACCESS(READ)

SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST(FACILITY)   REFRESH
```

Programmadumps mogen uitsluitend voor geautoriseerde gebruikers beschikbaar te zijn. De implementatie:

```
ADDSD      'SYS1.DUMP%%'   UACC(NONE)

PERMIT     'SYS1.DUMP%%'   ID(SYSPRO1)   ACCESS(READ)
```

9.6 Allocatie van devices (Device Allocation)

Geen standaard noodzakelijk.

9.7 Library Lookaside facility (LLA)

Geen standaard noodzakelijk.

9.8 Virtual Lookaside Facility (VLF)

Geen standaard noodzakelijk.

9.9 DATA Lookaside Facility (DLF)

Het gebruik van DLF objecten moet zijn geautoriseerd. De implementatie:

```
RDEFINE    DLFCLASS   PRODA.DATA   DLFDATA(JOBNAMES(TRNS**))
           UACC(NONE)

PERMIT     'SALES.DATA'   CLASS(DLFCLASS)
           ID(SALES)   ACCESS(READ)

SETROPTS CLASSACT(DLFCLASS)
```

9.10 Parallel Sysplex policies

Nog nader in te vullen.

10 Job Entry Subsystem (JES2)

10.1 Introductie

Het Job Entry Subsystem JES2 is het z/OS primaire subsysteem voor job scheduling. Het verzorgt de volgende basisfuncties:

- Leest de Job Control Language (JCL) en legt de job-informatie vast op SPOOL (JCL en SYSIN), Input fase;
- Interpreteert de JCL en converteert deze voor gebruik door z/OS, Conversie fase;
- Scheduled de job voor verwerking in een initiator, Scheduling fase;
- Leest de SYSIN voor de job en legt de gegenereerde output (SYSOUT) vast op SPOOL, Execution fase;
- Categoriseert de output naar output type, Output fase;
- Stuurt printers en andere devices aan, Hardcopy fase;
- Verwijderd de job informatie van SPOOL als alle werkzaamheden zijn verricht, Purge fase.

10.2 Status

Dit hoofdstuk is gereed voor commentaar.

10.3 Installatie

Gebruik van JES2-exits is niet toegestaan tenzij aangetoond kan worden dat dit het beveiligingsniveau verhoogt. Als exits worden toegepast dienen deze als USERMOD door middel van SMP/E te worden geïnstalleerd.

10.3.1 Dataset protectie

De JES2 programma en parameter libraries en de SPOOL datasets mogen uitsluitend voor JES2 toegankelijk zijn. Aangezien JES2 als een trusted started task wordt uitgevoerd zijn geen toegangsrechten voor JES2 noodzakelijk. Een voorbeeld:

```
ADDSD    'SYS1.LINKLIB'    GENERIC  UACC(NONE)
ADDSD    'SYS1.LPALIB'     GENERIC  UACC(NONE)
ADDSD    'SYS1.PARMLIB'    GENERIC  UACC(NONE)
ADDSD    'SYS1.JESPARM'    GENERIC  UACC(NONE)

ADDSD    'SYS1.HASPACE'    GENERIC  UACC(NONE)
```

10.3.2 Started Task

De JES2 started task moet als 'trusted' worden gedefinieerd zodat alle resources binnen het systeem kunnen worden benaderd. Verder moet JES2 onder een protected userid worden uitgevoerd. De implementatie:

```
ADDUSER  JES2 DFLTGRP(STCGROUP) OWNER(SECADM) NOPASSWORD

SETROPTS GENERIC(STARTED)
```

```
SETROPTS CLASSACT(STARTED)

RDEFINE   STARTED  JES2.*  OWNER(SECADM)  STDATA(USER(JES2)
          GROUP(STCGROUP)  PRIVILEGED(NO)  TRUSTED(YES)
          TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

10.4 SPOOL protectie

10.4.1 SYSIN en SYSOUT

SPOOL datasets zoals SYSIN en SYSOUT van gebruikers dienen voor gebruik te worden geautoriseerd. Door het uitsluitend activeren van de generic resource class JESSPOOL beschermt RACF alle SPOOL datasets. Indien de SPOOL datasets uitsluitend mogen worden benaderd door de gebruikers die deze hebben aangemaakt dient de volgende definitie te worden gemaakt.

```
SETROPTS CLASSACT(JESSPOOL)
SETROPTS GENERIC(JESSPOOL)
```

Indien andere dan de gebruiker die een SPOOL dataset heeft aangemaakt deze moet kunnen benaderen dient een zo'n specifiek mogelijk profiel te worden aangemaakt. De implementatie waarbij de gehele administratie de job input mag zien van een financiële verwerking:

```
RDEFINE   JESSPOOL  PRODC.UFINA00.JOB123.*.*.INPUT
          OWNER(SECADM)  UACC(NONE)
PERMIT    PRODC.UFINA00.JOB123.*.*.INPUT
          ID(FINADM1)  ACCESS(READ)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

Indien besloten wordt dat gebruikers andere toegang tot hun eigen SPOOL datasets mogen geven kan worden overwogen het eigenaarschap van het SPOOL-profiel aan de gebruiker te geven. De implementatie:

```
RDEFINE   JESSPOOL  PRODC.&RACUID.**
          OWNER(&RACUID)  UACC(READ)    Voor alle gebruikers

RDEFINE   JESSPOOL  PRODC.OENGF01.**
          OWNER(OENGF01)  UACC(NONE)    Voor één specifieke gebruiker
PERMIT    PRODC.OENGF01.**  ID(OPRGRP2)  ACCESS(READ)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

Een voorbeeld waarbij de operatorgroep 2 de job output mag zien van een financiële verwerking:

```
RDEFINE   JESSPOOL  PRODC.UFINA00.JOB123.*.*.OUTPUT
          OWNER(SECADM)  UACC(NONE)
PERMIT    PRODC.UFINA00.JOB123.*.*.OUTPUT
          ID(OPRGRP2)  ACCESS(READ)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

10.4.2 SYSLOG

The SYSLOG bevat alle dagelijkse systeem gebeurtenissen en is daarom één van de belangrijke informatiebronnen voor de navolgbaarheid van gebruikersactiviteiten. De SYSLOG is een SPOOL dataset waarvan het gebruik dient te worden geautoriseerd. Uitsluitend die personen die

de SYSLOG voor de uitvoering van hun werkzaamheden nodig hebben mogen hier toegang toe hebben. De implementatie:

```
RDEFINE   JESSPOOL  PRODA.+MASTER+.SYSLOG.*.*.?  UACC(NONE)
PERMIT    PRODA.+MASTER+.SYSLOG.*.*.?  CLASS(JESSPOOL)
          ID(ORIJL01)  ACCESS(READ)
PERMIT    PRODA.+MASTER+.SYSLOG.*.*.?  CLASS(JESSPOOL)
          ID(OARCH2)  ACCESS(ALTER)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

10.4.3 JESNEWS

De gegevens in de JESNEWS dataset worden gebruikt als informatie in elke joblog voor elke gebruiker. Gebruikers dienen daarom leesrechten op deze dataset te hebben. De implementatie:

```
RDEFINE   JESSPOOL  PRODA.*.$JESNEWS.*.*.JESNEWS  UACC(READ)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

Indien dit profiel wordt toegepast dient deze voor performance redenen in de Global Access Table te worden opgenomen. De implementatie:

```
RALTER    GLOBAL  JESSPOOL
          ADDMEM('PRODA.*.$JESNEWS.*.*.JESNEWS'/READ)

SETROPTS  GLOBAL(JESSPOOL)  REFRESH
```

Het kunnen wijzigen van de JESNEWS berichten voor de eindgebruiker dient te worden geautoriseerd. De implementatie:

```
RDEFINE   OPERCMDS  JES2.UPDATE.JESNEWS
          OWNER(SECADM)  UACC(NONE)
PERMIT    JES2.UPDATE.JESNEWS  ID(OPERGRP)  ACCESS(ALTER)

SETROPTS RACLIST(OPERCMDS)  REFRESH
```

10.4.4 Trace data

Trace datasets worden gebruikt om JES2 trace informatie op te slaan. Deze datasets kunnen daarom vertrouwelijke informatie zoals userids met wachtwoorden bevatten. Het kunnen benaderen van trace datasets dient te worden geautoriseerd en uitsluitend een beperkte groep personen, zoals Systeem programmeurs voor het onderzoeken van problemen, mag toegang hebben indien dit voor hun werkzaamheden noodzakelijk is. De implementatie:

```
RDEFINE   JESSPOOL  PRODA.JES.*.*.*.JESTRACE  UACC(NONE)
PERMIT    PRODA.JES.*.*.*.JESTRACE  CLASS(JESSPOOL)
          ID(OPROB13)  ACCESS(ALTER)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

10.4.5 SPOOL offload en reload

Voor het kopiëren van SPOOL datasets naar een ander medium zoals schijf of tape, kan de installatie de offload writer toepassen. Het gebruik van de writer moet worden geautoriseerd. De implementatie voor de SYSOUT Transmittor (ST):

```
RDEFINE   WRITER  JES2.LOCAL.OFF1.ST  UACC(NONE)

PERMIT    JES2.LOCAL.OFF1.ST  CLASS(WRITER)
          ID(OPERGRP)  ACCESS(READ)
```

De implementatie voor de Job Receiver (JR) en Sysout Receiver (SR):

```
RDEFINE   WRITER  OFF1.JR  UACC(NONE)

PERMIT    OFF1.JR  CLASS(WRITER)  ID(OPERGRP)  ACCESS(READ)

RDEFINE   WRITER  OFF1.SR  UACC(NONE)

PERMIT    OFF1.SR  CLASS(WRITER)  ID(OPERGRP)  ACCESS(READ)
```

De SPOOL offload transmittor kan voor het opslaan van de SPOOL-gegevens, datasets op tape of op schijf gebruiken. Deze datasets dienen beschermd te worden zodat ongeautoriseerde personen de gegevens niet kunnen lezen of schrijven. De groep die toegangsrechten krijgt dient bijvoorbeeld ALTER rechten te krijgen zodat tijdens uitvoering van de SPOOL offload de allocatie kan worden aangepast naar de hoeveelheid SPOOL-gegevens. De implementatie:

```
ADDSD     'SYS1.SPOOL.OFFLOAD*'  OWNER(SECADM)  UACC(NONE)
PERMIT    'SYS1.SPOOL.OFFLOAD*'  ID(OPERGRP)  ACCESS(ALTER)
```

10.5 Reader protectie

Internal readers (INTRDR)

Started task readers (STCINRDR)

TSO readers (TSUINRDR)

10.6 Writer en SAPI control

Writers zijn over het algemeen home made programma's uitgevoerd als een started task, waarmee SPOOL datasets naar een specifiek device kunnen worden geschreven (gekopieerd). Indien writers als started task worden geactiveerd mag niet het trusted attribute worden toegekend. De implementatie:

```
ADDUSER  XWTR  DFLTGRP(STCGROUP)  OWNER(SECADM)
         NOPASSWORD

RDEFINE   STARTED  XWTR.*  OWNER(SECADM)  STDATA(USER(XWTR)
          GROUP(STCGROUP)  PRIVILEGED(NO)  TRUSTED(NO)
          TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

De writer dient toegang te worden gegeven voor elke SPOOL dataset of groep van datasets die moet worden gelezen. De implementatie:

```
RDEFINE   JESSPOOL  PRODC.UFINA00.JOB123.*.*.OUTPUT
          OWNER(SECADM)  UACC(NONE)
PERMIT    PRODC.UFINA00.JOB123.*.*.OUTPUT
          ID(XWTR)  ACCESS(ALTER)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

Als een programma de SYSOUT Application Programming Interface (SAPI) gebruikt dient als een writer deze als een started task zonder het trusted attribute te worden uitgevoerd en toegang te krijgen tot de betreffende SPOOL profielen. De toegangsrechten mogen dan maximaal UPDATE zijn. De implementatie:

```
RDEFINE   JESSPOOL  PRODC.UFINA00.JOB123.*.*.OUTPUT
          OWNER(SECADM)  UACC(NONE)
PERMIT    PRODC.UFINA00.JOB123.*.*.OUTPUT
          ID(XSAPI)  ACCESS(UPDATE)

SETROPTS RACLIST(JESSPOOL)  REFRESH
```

10.7 Job Control

10.7.1 Job Input control

De devices waar vandaan een job kan worden ingevoerd kunnen worden geautoriseerd. Het is aan te raden voordat de JESINPUT faciliteit wordt gebruikt een sluitprofiel te maken zodat de niet gedefinieerde inputs geen jobs kunnen submitten. De implementatie:

```
SETROPTS GENERIC(JESINPUT)

RDEFINE   JESINPUT  **  OWNER(SECADM)  UACC(READ)

RDEFINE   JESINPUT  NJERDR1  OWNER(SECADM)  UACC(NONE)
PERMIT    NJERDR1  CLASS(JESINPUT)  ID(OPERGRP3)  ACCESS(READ)

SETROPTS CLASSACT(JESINPUT)
SETROPTS RACLIST(JESINPUT)  REFRESH
```

10.7.2 Job Submission

Alle batch jobs moeten onder RACF worden uitgevoerd. De implementatie:

```
SETROPTS JES(BATCHALLRACF)
```

Indien batch jobs gesubmit worden met een execution batch monitor dienen deze onder RACF te worden uitgevoerd. De implementatie:

```
SETROPTS JES(XBMALLRACF)
```

Het submitten van jobs moet worden geautoriseerd. Voor eindgebruikers dient afgedwongen te worden dat de jobnaam met het executing userid begint (vermeld in het JOB statement of via userid propagation). De implementatie:

```
SETROPTS GENERIC(JESJOBS)

RDEFINE   JESJOBS  **  UACC(NONE)

RDEFINE   JESJOBS  SUBMIT.&RACLNDE.&RACUID*.&RACUID
          OWNER(SECADM)  UACC(NONE)
PERMIT    SUBMIT.&RACLNDE.&RACUID*.&RACUID  CLASS(JESJOBS)
          ID(TSOSUB1)  ACCESS(READ)

SETROPTS CLASSACT(JESJOBS)
SETROPTS RACLIST(JESJOBS)  REFRESH
```

De profiel-qualifiers betekenen achtereenvolgens: SUBMIT.localnodeid.jobname.userid

- SUBMIT Het commando (job submission) dat wordt beschermd met dit profiel;
- &RACLNDE De tweede qualifier definieert de lokale JES2-nodenaam. Hier is de RACF-variabele &RACLNDE gebruikt waar voor RACF de lokale JES2-nodenaam invult. Het is ook mogelijk om een vaste naam in te vullen, wat echter voor het

beveiligingsbeheer wordt afgeraden omdat dan voor elke nodenaam één of meerdere profielen moeten worden gemaakt;

- &RACUID* De derde qualifier definieert de te beschermen jobnaam of jobnamen. Hier is de RACF-variabele &RACUID gebruikt waar voor RACF het userid invult welke de gebruiker is zijn/haar JOB statement heeft gebruikt. Dit geeft aan dat een gebruiker uitsluitend jobs mag submitten waarvan de jobnaam met zijn/haar eigen userid begint eventueel aangevuld met een extra karakter zodat meerdere jobs parallel kunnen worden uitgevoerd;
- &RACUID De vierde qualifier definieert het userid waaronder de voorgaande jobnamen mogen worden uitgevoerd. Hier is de RACF variabele &RACUID gebruikt waarvoor RACF het userid invult welke de gebruiker in zijn/haar JOB statement heeft gebruikt. De gebruikers die toegang hebben op dit profiel mogen jobs submitten die met het executing userid beginnen.

Indien gebruikers een job onder een ander userid of jobnaam willen submitten, dan toegestaan met het voorgaande profiel, dient dit te worden geautoriseerd. Een voorbeeld waarbij de operatorsgroep 2 (OPERGRP2) jobs mogen submitten als deze worden uitgevoerd onder userid CICS12 en de jobnaam met CICST begint:

```
RDEFINE   JESJOBS  SUBMIT.&RACLNDE.CICST*.CICST12
          OWNER(SECADM)  UACC(NONE)
PERMIT    SUBMIT.&RACLNDE.CICST*.CICST12  CLASS(JESJOBS)
          ID(OPERGRP2)  ACCESS(READ)
```

Het gebruik van variabelen in de JES gerelateerde profielen is sterk aan te raden. Het gebruik van RACF-variabelen vereenvoudigt het beveiligingsbeheer sterk. Variabelen moeten in de general resource class RACFVARS worden gedefinieerd. Een voorbeeld van de hierboven gebruikte variabelen &RACLNDE:

```
RDEFINE   RACFVARS  &RACLNDE  OWNER(SECADM)
          ADDMEM(PRODA)

SETROPTS CLASSACT(RACFVARS)
```

Een ander voordeel met het gebruik van variabelen is dat verantwoordelijkheden beter kunnen worden gedelegeerd. Door het toekennen van het eigenaarschap aan een variabele kunnen decentrale afdelingen deze variabelen beheren en daardoor het beheer voeren over bijvoorbeeld de jobnamen. De implementatie:

```
RDEFINE   RACFVARS  &TSTCICS  OWNER(DECSEC1)
          ADDMEM(CICST13  CICSX26  CICSX63)

PERMIT    SUBMIT.&RACLNDE.&TSTCICS.CICST12  CLASS(JESJOBS)
          ID(TSTGRP2)  ACCESS(READ)

RDEFINE   RACFVARS  &ACCCICS  OWNER(DECSEC2)
          ADDMEM(CICSA13  CICSA26  CICSA63)

PERMIT    SUBMIT.&RACLNDE.&ACCCICS.CICST19  CLASS(JESJOBS)
          ID(TSTGRP2)  ACCESS(READ)

SETROPTS RACLIST(JESJOBS  RACFVARS)  REFRESH
```

Als gebruikers een job onder het userid van een andere gebruiker mogen uitvoeren (executing userid), zoals in de JESJOBS profielen kan worden gedefinieerd, moet het wachtwoord van het executing userid nog wel worden opgegeven. Het executing userid en het bijbehorende

wachtwoord moet dan in de JOB-kaart worden gedefinieerd. Het in leesbaar moeten definiëren van het wachtwoord geeft op zich een security violation. RACF biedt de mogelijkheid voor het uitvoeren van een job onder een ander userid zonder het wachtwoord van dat andere userid te verstrekken mits gedefinieerd in de general resource class SURROGAT. Het gebruik van surrogate userids in deze situatie heeft de voorkeur. De implementatie:

```
RDEFINE   SURROGAT  CICST12.SUBMIT  OWNER(SECADM)  UACC(NONE)

PERMIT    CICST12.SUBMIT  CLASS(SURROGAT)
          ID(TSTGRP2)  ACCESS(READ)

SETROPTS CLASSACT(SURROGAT)
SETROPTS RACLIST(SURROGAT)  REFRESH
```

Userids die uitsluitend voor batch jobs worden gebruikt, zoals voor operationele taken en voor CICS-jobs, dienen als een protected userid te worden gedefinieerd. De implementatie:

```
ADDUSER  CICST12  DFLTGRP(CICSTEST)  OWNER(SECADM)
         NOPASSWORD
```

Als jobs geen gerelateerd userid hebben wordt het default userid '++++++++' (8 plustekens) toegewezen. Het is aan te raden hiervoor een helder te herkennen userid te definiëren dat als default userid wordt gebruikt. Voor het default userid is geen RACF gebruikersprofiel te definiëren zodat hiermee toegang tot systeem resources nooit mogelijk is. De implementatie van het default userid:

```
SETROPTS  JES(UNDEFINEDUSER(UJESUND))
```

10.7.3 Job Cancellation

Het cancelen van eigen jobs of waarvoor gebruikers doormiddel van een SUBMIT. profiel in de JESJOBS class zijn geautoriseerd is automatisch toegestaan. Uitsluitend indien een gebruiker wordt toegestaan een job te cancelen waarvoor de gebruiker geen submit autorisatie had, dient men een CANCEL profiel in de JESJOBS aan te maken. Hieronder een voorbeeld waarbij een gebruiker de jobs van een andere gebruiker mag cancellen.

```
RDEFINE   JESJOBS  CANCEL.&RACLNDE.*.XMIEP01  UACC(NONE)
PERMIT    CANCEL.&RACLNDE.*.XMIEP01  CLASS(JESJOBS)
          ID(XKRET00)  ACCESS(ALTER)
```

De implementatie van operatorgroep 2 die CICS testsystemen, die als een job zijn gestart, kunnen cancellen.

```
RDEFINE   JESJOBS  CANCEL.&RACLNDE.CICS*.CICST*  UACC(NONE)
PERMIT    CANCEL.&RACLNDE.CICS*.CICST*  CLASS(JESJOBS)
          ID(OPERGRP2)  ACCESS(ALTER)

SETROPTS CLASSACT(JESJOBS)
```

10.8 Network Job Entry (NJE)

Al het NJE-verkeer dient te worden gecontroleerd en geautoriseerd. Hiervoor dient minimaal een sluitprofiel te worden gedefinieerd waardoor geen ongeautoriseerd NJE-verkeer kan plaatsvinden. De implementatie:

```
RDEFINE   NODES  **  OWNER(SECADM)  UACC(NONE)
SETROPTS CLASSACT(NODES)
```

Met de profielen in de general resource class NODES kunnen verschillende controles worden afgedwongen. Meer specifieke profielen kunnen worden gedefinieerd voor bijvoorbeeld de

autorisatie van een gebruiker of groep die een job of sysout in het systeem wil invoeren. Een voorbeeld waarbij operatorgroep 2 jobs vanaf de systemen kunnen submitten die in de variabele &RACLNDE zijn opgenomen. In dit voorbeeld kunnen de operators jobs submitten vanuit andere systemen zonder dat het wachtwoord wordt geverifieerd (UPDATE). De submitting systemen moeten alle bewezen betrouwbaar (trusted) te zijn. De implementatie:

```
RDEFINE   RACFVARS  &OPERGRP2
          ADDMEM(OPER301  OPER304  OPER309)

RDEFINE   NODES  &RACLNDE.USERJ.&OPERGRP2  UACC(UPDATE)
```

NJE-verkeer vanuit een ander systeem (inbound) dient te worden geautoriseerd. Het externe systeem dient als RACF-gebruiker met een user-profiel te worden gedefinieerd en in de FACILITY class te worden geautoriseerd. De implementatie van NJE-verkeer van de gebruiker XENGF01 vanuit een vertrouwd systeem:

```
ADDUSER  EXTNOD3  DATA('NJE node userid voor HRM-systeem')
         OWNER(SECADM)  DFLTGRP(NJEGRP)
         PASSWORD(nje-wachtwoord)

RDEFINE   FACILITY  NJE.EXTNOD3

RDEFINE   NODES  EXTNOD3.RUSER.XENGF01  UACC(UPDATE)
```

Indien toegestaan wordt dat een gebruiker uit een ander niet vertrouwd systeem (untrusted), NJE met het systeem mag uitvoeren (job submission en sysout), dient het userid/wachtwoord combinatie te worden geverifieerd. De implementatie:

```
RDEFINE   NODES  JNODEL.USER*.XMIEP01  UACC(READ)
```

Indien ongedefinieerde gebruikers via NJE toegang tot het systeem hebben, bijvoorbeeld als het systeem een intermediate node voor SYSOUT verkeer is krijgt het default userid '????????' (8 vraagtekens) toegewezen. Het is aan te raden hiervoor een ander helder te herkennen userid te definiëren dat als default userid wordt gebruikt. Voor het default userid is geen RACF gebruikersprofiel te definiëren zodat hiermee toegang tot systeem resources nooit mogelijk is. De implementatie:

```
SETROPTS  JES(NJEUSERID(UNJEUND))
```

Het default userid dient geen toegang tot resources te hebben die de integriteit en vertrouwelijkheid van de gegevens in het systeem in gevaar kunnen brengen. Indien het mogelijk moet zijn dat het default userid toegang tot specifieke resources moet krijgen dient het userid te worden vertaald naar een ander userid dat onder RACF is gedefinieerd. Voor het vertaalde default userid kan dan noodzakelijke autorisatie worden gedefinieerd. De implementatie van een NJE default userid vertaling:

```
ADDUSER  UNJEDFTL   DATA('NJE default user')  OWNER(SECADM)

RDEFINE   NODES  *.USER*.UNJEUND  ADDMEM(UNJEDFTL)
          UACC(UPDATE)

PERMIT    'REKLAME.FOLDERS'  ID(UNJEDFTL)  ACCESS(READ)
```

Het opstarten van NJE-verkeer (outbound) met een ander systeem dient te worden geautoriseerd. De implementatie:

```
RDEFINE   WRITER  JES2.NJE.EXTNOD5  UACC(NONE)

PERMIT    JES2.NJE.EXTNOD5  CLASS(WRITER)
          ID(OPERGRP)  ACCESS(READ)
```

```
SETROPTS CLASSACT(WRITER)
```

10.9 Remote Job Entry (RJE)

Al het RJE-verkeer (inbound) dient te worden geautoriseerd. Het RJE-systeem dient als RACF-gebruiker met user-profiel te worden gedefinieerd en in de FACILITY class te worden geautoriseerd als RJE-systeem. De gebruiker dient zich altijd te identificeren en te authenticeren. De implementatie:

```
ADDUSER  XRJE803   DATA('RJE node userid voor Home-systeem')
         OWNER(SECADM)   DFLTGRP(RJEGRP)
         PASSWORD(initieel-wachtwoord)

RDEFINE    FACILITY   RJE.XRJE803
```

Het opstarten van RJE-verkeer (outbound) met een ander systeem dient te worden geautoriseerd. De implementatie:

```
RDEFINE    WRITER   JES2.RJE.EXTPC53   UACC(NONE)

PERMIT     JES2.NJE.EXTPC53   CLASS(WRITER)
           ID(OPERGRP)   ACCESS(READ)

SETROPTS CLASSACT(WRITER)
```

10.10 Devices

Het gebruik van bijvoorbeeld printers moet worden geautoriseerd. De implementatie voor een lokale printer:

```
RDEFINE    WRITER   JES2.LOCAL.PAGEPRT1   UACC(NONE)
PERMIT     JES2.LOCAL.PAGEPRT1   ID(UGRP385)   ACCESS(READ)
```

De implementatie voor een NJE printer:

```
RDEFINE    WRITER   JES2.NJE.PRT14   UACC(NONE)
PERMIT     JES2.NJE.PRT14   ID(UGRP845)   ACCESS(READ)
```

De implementatie voor een RJE printer:

```
RDEFINE    WRITER   JES2.RJE.PRT475   UACC(NONE)
PERMIT     JES2.RJE.PRT475   ID(UGRP921)   ACCESS(READ)
```

10.11 JES Commando's

Voor het beheer van JES2 kunnen commando's worden gegeven vanaf het z/OS operator console of vanuit een job waarin JES2-commando's zijn opgenomen. Een job kan op het lokale systeem of van een ander systeem worden gesubmit en door middel van NJE of RJE worden ingevoerd. Het inbrengen van commando's dient in alle gevallen te worden geautoriseerd. De implementatie:

```
RDEFINE    OPERCMDS   JES2.DISPLAY.*   OWNER(SECADM)   UACC(NONE)

PERMIT     JES2.DISPLAY.*   CLASS(OPERCMDS)
           ID(XENGF01)   ACCESS(READ)
```

AANVULLENDE MAATREGEL. Voor een hoger beveiligingsniveau is aan te raden om commando's uitsluitend via gedefinieerde systeem of TSO consoles te autoriseren. De implementatie:

```
PERMIT      JES2.DISPLAY.*  CLASS(OPERCMDS)
            ID(OPERGRP)  ACCESS(READ)
            WHEN(CONSOLE(MSTR2))
```

10.12 B1-beveiliging

Het JES2 subsysteem ondersteunt volledig het B1-beveiligingsniveau. Voor de implementatie kan gebruik worden gemaakt van beveiligingslevels, beveiligingscategorieën en beveiligingslabels zoals beschreven in hoofdstuk 'Gebruikers- en data-categorisering'. Het gebruik van B1-beveiliging geeft een ***AANVULLENDE MAATREGEL.***

Het gebruik van B1-beveiliging voor JES2 dient nog te worden uitgewerkt.

11 System Managed Storage (SMS)

11.1 Introductie

11.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

12 Time Sharing Option (TSO)

12.1 Introductie

12.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

13 Interactive System Productivity Facility (ISPF)

13.1 Introductie

13.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

14 System Display and Search facility (SDSF)

14.1 Introductie

14.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

15 Virtual Telecommunication Access Method (VTAM)

15.1 Introductie

Inclusief APPC/MVS

15.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

16 Operations planning and Control (OPC)

16.1 Introductie

16.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

17 Integrated Cryptographic Service Facility (ICSF)

17.1 Introductie

ICSF heeft 72 verschillende methodes (services) voor het versleutelen van gegevens en het aanmaken van o.a. beveiligingscertificaten, pincodes en softwaresleutels. Het gebruik van een service en autorisatie van het gebruik van certificaten, pincodes en sleutels kan binnen RACF worden geregeld. Dit hoofdstuk beschrijft de minimale RACF implementatie voor ICSF.

17.2 Status

Dit hoofdstuk is gereed voor commentaar.

17.3 ICSF Services

Voor een efficiënt beheer van de ICSF services dient met RACF, generic commando's en generic profielen te kunnen worden gebruikt. De implementatie:

```
SETROPTS  GENCMD(CSFSERV)
SETROPTS  GENERIC(CSFSERV)
```

Het gebruik van ICSF services moet worden geautoriseerd. Hiervoor moeten alle, reeds voorgedefinieerde, services eerst worden geblokkeerd waarna per services een profiel moet worden gedefinieerd waarop gebruikersgroepen geautoriseerd kunnen worden. De implementatie:

```
RDEFINE   CSFSERV   **   UACC(NONE)

RDEFINE   CSFSERV  CSFECO  UACC(NONE)
RDEFINE   CSFSERV  CSFENC  UACC(NONE)
SETROPTS CLASSACT(CSFSERV)
```

Gebruikersgroepen (functionele en/of administratieve groepen) moeten voor het gebruik van de services worden geautoriseerd. De implementatie:

```
PERMIT    CSFECO  CLASS(CSFSERV)  ACCESS(READ)  ID(HRMCENTR)

SETROPTS RACLIST(CSFSERV)  REFRESH
```

17.4 ICSF Keys

Voor een efficiënt ICSF sleutelbeheer dienen generic commando's te kunnen worden gegeven en moet RACF generic profielen processen. De implementatie:

```
SETROPTS  GENCMD(CSFKEYS)
SETROPTS  GENERIC(CSFKEYS)
```

Indien ICSF sleutels (keys) worden gebruikt moet dit worden geautoriseerd. Elk te gebruiken sleutel moet eerst worden gedefinieerd waarna deze voor gebruik geautoriseerd kunnen worden. De implementatie:

```
RDEFINE   CSFKEYS  C1223A45  UACC(NONE)
RDEFINE   CSFKEYS  K123*      UACC(NONE)
SETROPTS RACLIST(CSFKEYS)  REFRESH
```

Gebruikersgroepen (bijvoorbeeld functionele groepen of activiteitengroepen) moeten voor het gebruik van sleutels worden geautoriseerd. De implementatie:

```
PERMIT    C1223A45  CLASS(CSFKEYS)  ACCESS(READ)  ID(HRMCENTR)
PERMIT    K123*     CLASS(CSFKEYS)  ACCESS(READ)  ID(SECADM)

SETROPTS RACLIST(CSFKEYS)  REFRESH
```

Indien sleutels een ongelijksoortige naam hebben zodat geen generic profiel in de CSFKEYS general resource class kan worden gedefinieerd moet voor een efficiënt ICSF sleutelbeheer de generic resource class **G**CSFKEYS worden gebruikt. De implementatie:

```
RDEFINE   GCSFKEYS  V123  UACC(NONE)
          ADDMEM(C145782  F376484  K848635)

PERMIT    V123  CLASS(GCSFKEYS)  ACCESS(READ)  ID(BSIT)

SETROPTS CLASSACT(GCSFKEYS)
```

17.5 ICSF logging en monitoring

Het ongeautoriseerd gebruik van ICSF services dient direct te worden gedetecteerd. De implementatie:

```
RDEFINE   CSFSERV  **  UACC(NONE)  NOTIFY(SECADM)
```

18 Customer Information Control System (CICS)

18.1 Introductie

Het Customer Information Control System (CICS) is een transactie manager voor de meeste andere gangbare besturingssystemen waaronder z/OS. Een CICS (CICS region) wordt gebruikt voor het starten van transacties die gezamenlijk een applicatie vormen. Hiervoor kan één CICS region worden gebruikt of een cluster van CICS regions. Bij een cluster (CICSplex) kan voor workload balancing elke CICS region de zelfde functionaliteit hebben. Bij een cluster kan voor eenvoudiger beheer elke CICS region ook een andere functionaliteit hebben. Zo kan een CICS region worden ingericht om uitsluitend de terminals af te handelen terwijl de andere de transacties uitvoert.

Voor alle CICS regions geldt dat de autorisatierechten voor CICS resources in een extern beveiligingspakket dienen te worden ondergebracht. De implementatie van CICS, CICS commando's en transacties dienen ervoor zorg te dragen dat de beveiliging van CICS en de door CICS beheerde resources niet kan worden doorbroken. De applicaties dienen de voorgeschreven beveiligingsrichtlijnen te implementeren.

18.2 Status

Dit hoofdstuk is gereed voor commentaar.

18.3 Installatie

18.3.1 CICS load libraries

Uitsluitend CICS region userids en gebruikers die uit hoofde van hun taken en bevoegdheden toegang tot CICS libraries moeten hebben mogen hiervoor zijn geautoriseerd. Uitsluitend de noodzakelijke toegangsrechten dienen te worden gegeven. Enkele voorbeelden:

```
ADDSD    'PROD1.CICS.SDFHLOAD'  UACC(NONE)
PERMIT   'PROD1.CICS.SDFHLOAD'  ID(SCICS16)  ACCESS(EXECUTE)

ADDSD    'PROD1.CICS.SDFHLPA'  OWNER(SECADM)  UACC(NONE)
         Bij de LPA library zijn permits niet noodzakelijk en niet toegestaan.

ADDSD    'PROD1.CICS.SDFHLINK'  OWNER(SECADM)  UACC(NONE)
         Bij de Linklib library zijn permits niet noodzakelijk en niet toegestaan.

ADDSD    'PROD1.CICS.SDFHAUTH'  OWNER(SECADM)  UACC(NONE)
PERMIT   'PROD1.CICS.SDFHAUTH'  ID(SCICS16)  ACCESS(EXECUTE)
```

18.3.2 Decentraal beheer

Voor decentraal beheer van CICS dient de decentrale Security administrator eigenaar van de betreffende CICS gerelateerde dataset profielen te zijn: De implementatie:

```
ADDSD    'PROD1.CICS16.TEMPSTOR'  OWNER(SECDEC3)  UACC(NONE)
PERMIT   'PROD1.CICS16.TEMPSTOR'  ID(SCICS16)  ACCESS(CONTROL)

ADDSD    'PROD1.CICS16.RESTART'  OWNER(SECDEC3)  UACC(NONE)
PERMIT   'PROD1.CICS16.RESTART'  ID(SCICS16)  ACCESS(UPDATE)
```

```
ADDSD    'PROD1.CICS16.APPLC74.LOAD1'  OWNER(SECDEC3)
         UACC(NONE)
PERMIT   'PROD1.CICS16.APPLC74.LOAD1'  ID(SCICS16)  ACCESS(READ)

ADDSD    'PROD1.CICS16.APPLC74.VSAMDB'  OWNER(SECDEC3)
         UACC(NONE)
PERMIT   'PROD1.CICS16.APPLC74.VSAMDB'  ID(SCICS16)
         ACCESS(UPDATE)
```

AANVULLENDE MAATREGEL. Uitsluitend gecontroleerde programma's mogen worden geladen. De hieronder gebruikte optie NOPADCHK is uitsluitend geoorloofd indien alle transacties worden vertrouwd. RACF voert bij NOPADCHK geen controle uit of de userid/group-programma combinatie in de conditional access list van reeds geopende data sets is gedefinieerd.

De implementatie:

```
SETROPTS WHEN(PROGRAM)

RDEFINE  PROGRAM DFH*  UACC(EXECUTE)
         ADDMEM('PROD1.CICS.SDFHLOAD'//NOPADCHK
                'PROD1.CICS.SDFHLINK'//NOPADCHK
                'PROD1.CICS.SDFHAUTH'//NOPADCHK)

SETROPTS WHEN(PROGRAM)  REFRESH
```

Het uitvoeren van het CICS initialisatie-programma 'DFHSIT' en dus het opstarten van elk CICS-systeem, moet worden geautoriseerd. De implementatie:

```
RDEFINE  PROGRAM DFHSIT  UACC(NONE)
         ADDMEM('PROD1.CICS.SDFHAUTH'//NOPADCHK)
PERMIT   DFHSIT CLASS(PROGRAM)  ID(OPRGRP)  ACCESS(READ)

SETROPTS WHEN(PROGRAM)  REFRESH
```

18.3.3 Initialisatie parameters

De volgende CICS-initialisatieparameters zijn als uitgangspunt genomen voor de implementatie van de CICS-RACF-standaard. Indien bij een RACF-standaard een andere instelling noodzakelijk is wordt dit bij de betreffende norm aangegeven.

Alle CICS-geheugengebieden moeten zoveel als mogelijk worden beschermd tegen ongeautoriseerd gebruik. Over het algemeen is scheiding van geheugengebieden een architectuur mechanisme maar het wordt hier als uitgangspunt genomen omdat dit de gehele beveiliging beïnvloedt. De implementatie van CICS-initialisatieparameters:

```
DFHSIT   CMDPROT=YES,RENTPGM=PROTECT,
         STGPROT=YES,TRANISO=YES
```

Alle CICS regions dienen gebruik te maken van de external security interface. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT   SEC=YES
```

Indien meerdere CICS regions in het systeem aanwezig zijn, of verwacht wordt dat deze in een later stadium worden geactiveerd, dan dienen alle CICS regions uniek te worden geïdentificeerd. In het RACF resource-profiel moet dan het CICS region-userid worden opgenomen. Hiervoor

dient de CICS-initialisatieparameter SECPRFX=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      SECPRFX=YES
```

Een voorbeeld van een prefixed RACF transactie- en permit-profiel (deze profielen zullen later in meer detail worden behandeld):

```
RDEFINE   TCICSTRN  CICS23.PRODTRN*   UACC(NONE)
                    OWNER(SECADM)
PERMIT    CICS23.PRODTRN*   CLASS(TCICSTRN)
          ID(PRODU1)   ACCESS(READ)
```

Met CICS mogen uitsluitend standaard beveiligingsmechanismen worden toegepast. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      ESMEXITS=NOINSTLN
```

AANVULLENDE MAATREGEL. Alle CICS resources dienen voor gebruik te worden geautoriseerd. De implementatie van de CICS-initialisatieparameters en RACF definities worden afzonderlijk in de volgende paragrafen behandeld.

18.4 Userids

18.4.1 Started Task userid

De CICS started task (CICS region) dient onder een userid te draaien die niet voor logon kan worden gebruikt (protected userid). De implementatie:

```
ADDUSER  SCICSA1   DFLTGRP(SCICSGRP)   OWNER(SECADM)
         NOPASSWORD
```

Elke productie CICS dient als een started task of started job te worden geactiveerd. Het gebruik van het ‘Trusted’ en ‘Privileged’ attribuut is voor het started task userid niet toegestaan. De implementatie:

```
RDEFINE   STARTED  (CICSPROC.SCICSA12)   STDATA(USER(SCICSA1)
          PRIVILEGED(NO)   TRUSTED(NO)   TRACE(YES))
```

CICS regions die tot hetzelfde CICSPlex behoren of voor dezelfde doeleinden worden gebruikt en waarbij geen verschil in beveiliging gewenst is, moeten hetzelfde userid toegewezen krijgen. CICS regions die voor verschillende doeleinden worden gebruikt moeten wel onder verschillende userids worden gestart. De implementatie:

```
RDEFINE   STARTED  (CICSPROC.CICSA*)   STDATA(USER(SCICSA1))
RDEFINE   STARTED  (CICSPROC.CICSB*)   STDATA(USER(SCICSA2))
```

Het userid van de CICS started task mag niet worden gebruikt voor het uitvoeren van transacties of jobs van eindgebruikers (userid propagation) en dient daarom altijd te worden afgeschermd voor propagatie. De implementatie:

```
RDEFINE   PROPCNTL   SCICSA1  SCICSA2

SETROPTS  CLASSACT(PROPCNTL)
SETROPTS  RACLIST(PROPCNTL)
```

18.4.2 Post-initialization process userid

Tijdens het opstarten en afsluiten van CICS kunnen programma's worden opgestart om eventuele door de installatie gewenste handelingen uit te voeren. De namen van de programma's die bij het opstarten worden gebruikt moeten worden vermeld in de PLTPI tabel. De programma-namen die tijdens het afsluiten moeten worden uitgevoerd moeten in de PLTSD tabel worden geplaatst. Voor het uitvoeren van deze programma's gebruikt CICS het userid dat bij de PLTPIUSR parameter in de SIT wordt gedefinieerd. Indien de PLTPIUSR parameter niet wordt gedefinieerd zal het CICS region userid worden gebruikt. Wij raden hierom aan de PLTPIUSR parameter altijd te wijzigen naar een installatie specifiek userid, zodat deze aan de installatie specifieke naamgevingconventies voldoet. Dit userid moet daarna binnen RACF worden gedefinieerd en de toegangsrechten worden toegekend die nodig zijn voor de uitvoering van de opstart- en afsluitprogramma's. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      PLTPISEC=ALL,PLTPIUSR=UCICSPLT
```

Een voorbeeld voor het RACF userid genoemd in de PLTPIUSR parameter:

```
ADDUSER  UCICSPLT  OWNER(SECADM)
```

18.4.3 Default userid

CICS-toegang kan door middel van identificatie en authenticatie (userid/wachtwoord combinatie) worden geregeld. Een installatie kan eisen dat alle toegang door middel van een persoonlijk userid wordt geregeld of dat voor algemeen gebruik, zoals voor het raadplegen van telefoonnummers en/of de locatie van werkplekken van medewerkers, toegang wordt verkregen zonder userid/wachtwoord te hoeven geven.

CICS heeft een default userid dat aan niet gedefinieerde gebruikers wordt toegewezen om toegang tot CICS (sign-on) te krijgen. Het CICS default userid 'CICSUSER' moet binnen de geldende naamgevingconventies naar een installatie specifiek userid worden veranderd. Dit kan worden gedaan door de CICS-parameter 'DFLTUSER' aan te passen. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      DFTLUSER=UDFTLA1
```

Indien het default userid wordt toegepast dient in RACF het CICS default userid als een protected userid te worden gedefinieerd zodat voorkomen wordt dat met behulp van andere toepassingen hiermee kan worden aangelogd. De implementatie:

```
ADDUSER  UDFTLA1  DATA('CICS Default userid voor productie CICS24')
         OWNER(SECADM)  NOPASSWORD
```

Indien geen gebruik van het CICS default userid wordt gemaakt, dient dit userid onder RACF te worden revoked. Het toepassen van een revoked CICS default userid geeft de mogelijkheid om te detecteren en te loggen of ongeautoriseerd toegang tot CICS wordt gezocht. Een voorbeeld:

```
ALTUSER  UDFTLA1  DATA('CICS Default userid voor productie CICS24')
         REVOKE
```

Het default userid dient een CICS-segment te hebben dat voldoet voor gebruikers die geen CICS sign-on behoeven uit te voeren. De implementatie:

```
ALTUSER  UDFTLA1  DATA(' CICS Default userid voor productie CICS24')
         OWNER(SECADM)  NOPASSWORD
      CICS(TIMEOUT(00:15)  XRFSOFF(NOFORCE))
```

18.4.4 Eindgebruiker userid

Voor het uitvoeren van bedrijfskritische en bedrijfsgevoelige CICS-transacties dient te worden afgedwongen dat alle eindgebruikers een sign-on (userid/wachtwoord) moeten uitvoeren. Het eindgebruikers (sign-on) userid dient door de local en remote CICS regions te worden gebruikt voor transactie en andere resource autorisatie. Verder dient het sign-on userid te worden gebruikt voor de 'USER=' parameter van eventueel gebruikte JCL-skeleton (job submission).

Het uitvoeren van de sign-on kan met de standaard CESN transactie worden uitgevoerd die met de GMTRAN systeem parameter wordt gedefinieerd. De sign-on kan ook vanuit een applicatie transactie met het EXEC CICS SIGNON commando worden uitgevoerd. De implementatie van de CICS initialisatieparameter voor de GMTRAN transactie ('Good Morning' transactie):

```
DFHSIT      GMTRAN=CESN
```

De gebruiker dient erop gewezen te worden dat de (CICS-)applicatie uitsluitend met toestemming van het management mag worden gebruikt. Een voorbeeldtekst voor het sign-on bericht:

```
DFHSIT      GMTEXT='Het gebruik van deze applicatie is uitsluitend toegestaan
                    met goedkeuring van het management'
```

18.4.5 Transient Data trigger transaction userid

Aan een Transient Data queue dient met de USERID operand van de DFHDCT macro (Destination Control Table) een userid worden toegekend. Indien geen sign-on userid aanwezig is zal de transactie onder het dit Transient Data queue userid worden gestart en daardoor de autorisaties krijgen van dit userid. Indien geen sign-on userid en Transient Data queue userid aanwezig is zal het CICS default userid (DFLTUSER) worden gebruikt voor autorisatie controle. Een voorbeeld van het in CICS toekennen van een DCT userid:

```
DFHDCT      USERID=UCICSDCT
```

Een voorbeeld voor de RACF userid:

```
ADDUSER  UCICSDCT  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
```

18.4.6 Userid Automatic Transaction Initiation

Wanneer een transactie het EXEC CICS SET commando met de optie ATIUSERID gebruikt voor een Automatic Transaction Initiation dan zal de CICS-user, waarvoor dit commando wordt uitgevoerd, surrogate rechten moeten hebben op het met de ATIUSERID optie gedefinieerde userid. Een voorbeeld voor het EXEC CICS SET ATIUSERID('*UCICSTRG*') userid:

```
ADDUSER  UCICSTRG  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
RDEFINE  SURROGAT  UCICSTRG.DFHINSTL  UACC(NONE)
         OWNER(SECADM)
PERMIT   UCICSTRG. DFHINSTL  CLASS(SURROGAT)
         ID(XMIEP01)  ACCESS(READ)
```

18.4.7 Surrogate userids

Met het surrogate userid mechanisme kunnen gebruikers jobs submitten onder een ander eventueel gezamenlijk userid ('executing userid'). De gebruikers moeten dan als een 'surrogate userid' van dit 'executing userid' worden gedefinieerd. Zie voor een verdere beschrijving van de paragraaf 'surrogate userids' in het hoofdstuk 'Gebruikers'.

18.4.7.1 Surrogate checking

Tijdens het opstarten van CICS worden de definities voor het surrogate mechanisme gecontroleerd. Hiervoor dient het CICS region userid als surrogate voor alle te controleren executing userids te worden gedefinieerd. Als deze niet gedefinieerd zijn zal de betreffende CICS niet worden geïnitialiseerd. Hierdoor wordt voorkomen dat ongeautoriseerde CICS regions van de betreffende executing userids gebruik kunnen maken. Het surrogate user checking proces moet door CICS worden afgedwongen. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      XUSER=YES
```

Hieronder de voorbeelden van de executing userid waarvoor het CICS region userid moet worden geautoriseerd:

Een voorbeeld voor het surrogate userid voor started transactions:

```
ADDUSER  UCICSTRT  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
RDEFINE  SURROGAT  UCICSTRT.DFHSTART  UACC(NONE)
OWNER(SECADM)
PERMIT   UCICSTRT. DFHSTART  CLASS(SURROGAT)
ID(SCICS85)  ACCESS(READ)
```

Een voorbeeld voor het PLT userid:

```
ADDUSER  UCICSPLT  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
RDEFINE  SURROGAT  UCICSPLT.DFHINSTL  UACC(NONE)
OWNER(SECADM)
PERMIT   UCICSPLT. DFHINSTL  CLASS(SURROGAT)
ID(SCICS85)  ACCESS(READ)
```

Een voorbeeld voor het trigger level userid:

```
ADDUSER  UCICSTRG  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
RDEFINE  SURROGAT  UCICSTRG.DFHINSTL  UACC(NONE)
OWNER(SECADM)
PERMIT   UCICSTRG. DFHINSTL  CLASS(SURROGAT)
ID(SCICS85)  ACCESS(READ)
```

Een voorbeeld voor het CICS default userid (DFLTUSER):

```
ADDUSER  UCICSDFT  DFLTGRP(GCICS85)  OWNER(SECADM)
                   NOPASSWORD
RDEFINE  SURROGAT  UCICSDFT.DFHINSTL  UACC(NONE)
OWNER(SECADM)
PERMIT   UCICSDFT. DFHINSTL  CLASS(SURROGAT)
ID(SCICS85)  ACCESS(READ)
```

18.5 Sign-on control

18.5.1 User control

Het is niet toegestaan dat gebruikers gelijktijdig meerdere sessies met hetzelfde CICS-userid hebben. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT      SNSCOPE=1
```

18.5.2 Applicaties control

Voor het netwerkprogramma VTAM wordt een CICS region als 'applicatie' beschouwd. Gebruikers mogen uitsluitend voor die CICS-regions zijn geautoriseerd die zij uit hoofde van hun taken en verantwoordelijkheden mogen benaderen. Een voorbeeld van de CICS-initialisatieparameter:

```
DFHSIT      APPLID=CICS47

Een voorbeeld van de RACF commando's:

RDEFINE   APPL   CICS47  UACC(NONE)  NOTIFY(SECADM)
PERMIT    CICS47  CLASS(APPL)  ID(UGRP34)  ACCESS(READ)

SETROPTS CLASSACT(APPL)
SETROPTS RACLIST(APPL)
```

Opmerking: voor alle CICS regions dient een APPL profiel met UACC(NONE) te worden gedefinieerd. Overwogen kan worden een sluitprofiel ('RDEFINE APPL ** UACC(NONE)') te definiëren. Let hierbij op dat dit dan voor alle applicaties geldt!

18.5.3 VTAM control

Het gebruik van een application-identificatie (APPLID) door een CICS region moet worden geautoriseerd, zodat andere taken of gebruikers zich niet kunnen voordoen als een CICS region. Een voorbeeld van de CICS-initialisatieparameter:

```
DFHSIT      APPLID=CICS47
```

Een voorbeeld van de RACF commando's:

```
RDEFINE   VTAMAPPL  APPL47  UACC(NONE)  NOTIFY(SECADM)
PERMIT    APPL47  CLASS(VTAMAPPL)  ID(SCICS16)  ACCESS(READ)

SETROPTS CLASSACT(VTAMAPPL)
SETROPTS RACLIST(VTAMAPPL)
```

18.5.4 Terminal/Console Security

CICS Preset Terminal Security, Normal of Automatic voor display terminals en consoles is niet toegestaan.

AANVULLENDE MAATREGEL. Voor het werken met zeer vertrouwelijke gegevens dienen gebruikers uitsluitend via een voorgedefinieerde terminal toegang tot CICS krijgen. De implementatie:

```
PERMIT    FIN*  CLASS(TCICSTRN)  ID(UVEIL88)  ACCESS(READ)
          WHEN(TERMINAL(AC12ERT5))
          WHEN(CONSOLE(*))
```

18.6 Attached transactions

Alle CICS applicatie-transacties geïnitieerd vanaf een terminal dienen voor gebruik te worden geautoriseerd. De implementatie van de CICS initialisatieparameter:

```
DFHSIT      XTRAN=YES
```

Voor het gebruik van CICS-transacties dient de TCICSTRN general resource class te worden geactiveerd en een sluitprofiel te worden gedefinieerd. De implementatie:

```
SETROPTS GENERIC(TCICSTRN)

RDEFINE   TCICSTRN  **  UACC(NONE)  OWNER(SECADM)  Sluitprofiel
```

De naamgeving van CICS-transacties dient zoveel mogelijk gekozen te worden dat groepering door middel van een generiek profile in de CICS-transactie-class (TCICSTRN) mogelijk is. De implementatie:

```
RDEFINE   TCICSTRN  CICS23.PRODTRN*  OWNER(SECADM)
                    UACC(NONE)
PERMIT    CICS23.PRODTRN*  CLASS(TCICSTRN)
          ID(PRODU1)  ACCESS(READ)

SETROPTS CLASSACT(TCICSTRN)        CICS normal transactions
```

Indien de naamgeving niet toereikend is om door middel van generieke profielen in de TCICSTRN resource class onder te brengen kan worden gekozen voor het groeperen van de transacties in de CICS-transactiegroepclass (GCICSTRN). De implementatie:

```
RDEFINE   GCICSTRN  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE   GCICSTRN  CICS23.CUSTHNDL  OWNER(SECADM)
          UACC(NONE)  ADDMEM(CUSTRN1  INVNTRN2  INKTRN5)
PERMIT    CICS23.CUSTHNDL  CLASS(GCICSTRN)  ACCESS(READ)
          ID(VERKOOP)

SETROPTS CLASSACT(GCICSTRN)        CICS normal transactions group class
```

18.6.1 Decentraal beheer

Indien de CICS-beveiligingstaken door specifieke Security administrators wordt uitgevoerd kan het beheren van CICS-profielen worden gedelegeerd. Hiervoor dient de gedelegeerd Security administrator het eigenaarschap van de betreffende transactieprofielen te krijgen waardoor het geven van permits kan worden gedelegeerd. De implementatie:

```
RDEFINE   TCICSTRN  CICS95.LOAN*  OWNER(UJAMJ01)  UACC(NONE)

en/of

RDEFINE   GCICSTRN  CICS96.INSURANC  OWNER(UJAMJ01)
UACC(NONE)  ADDMEM(CARTRN5  FIRETRN2  HOUSTRN4)
```

18.7 Started transacties

AANVULLENDE MAATREGEL. De door andere CICS-transacties gestarte transacties (started transactions) dienen voor gebruik te worden geautoriseerd. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XPCT=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT    RESSEC=YES,XPCT=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(ACICSPCT)

RDEFINE   ACICSPCT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

```
RDEFINE    ACICSPCT  CICS23.COMMTRN*  UACC(NONE)
OWNER(SECADM)
PERMIT     CICS23.COMMTRN*  CLASS(ACICSPCT)
           ID(VERKOOP)  ACCESS(READ)

SETROPTS CLASSACT(ACICSPCT)  CICS started transactions

CICS started transactions groups class
RDEFINE    BCICSPCT **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

Het gebruik van een started transactie die onder de autorisatie van een ander userid dan het aanroepende userid (normal transactie userid) moeten draaien, dient hiervoor te worden geautoriseerd. Voorbeeld XMIEP01 is het aanroepende userid terwijl de started transactie onder het CICSDA2 userid draait (EXEC CICS START TRANSID('*TRNBAC*') USERID('*CICSDA2*')). Zie ook de paragraaf 'surrogate userid' Een voorbeeld:

```
RDEFINE    SURROGAT  CICSDA2.DFHSTART  UACC(NONE)
           OWNER(SECADM)

PERMIT     CICSDA2.DFHSTART  CLASS(SURROGAT)  ID(XMIEP01)
           ACCESS(READ)
```

18.8 Transient Data queues

AANVULLENDE MAATREGEL. Indien Transient Data queues worden toegepast voor vertrouwelijke gegevens dienen deze te worden beschermd voor lezen en/of wijzigingen door andere gebruikers. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XDCT=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT     RESSEC=YES,XDCT=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(DCICSDCT)

RDEFINE    DCICSDCT **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE    DCICSDCT  CICS23.DQ23  OWNER(SECADM)  UACC(NONE)
PERMIT     CICS23.DQ23  CLASS(DCICSDCT)
           ID(VERKOOP)  ACCESS(UPDATE)

SETROPTS CLASSACT(DCICSDCT)        CICS transient data queue

CICS transient data queue groups class
RDEFINE    ECICSDCT **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

CICS heeft voor eigen gebruik een aantal Transient Data queues die dienen te worden beschermd. De implementatie:

```
RDEFINE    ECICSDCT  CICSQUEUES  OWNER(SECADM)  UACC(NONE)
           ADDMEM(CPLI,CSSL,CCSO)  NOTIFY(SECADM)
```

18.9 File Control

18.9.1 SMSVSAM Subsystem control

Voor het sharen van VSAM files heeft CICS een faciliteit die de noodzakelijke integriteit van de gegevens verzorgt. De SMSVSAM faciliteit is een subsysteem ondergebracht in een z/OS Address Space.

Voor het openen van de shared VSAM files dient de CICS region toegang tot het SMSVSAM subsysteem te hebben. De implementatie:

```
RDEFINE   SUBSYSNM  APPL47  UACC(NONE)  NOTIFY(SECADM)
PERMIT    APPL47  CLASS(SUBSYSNM)
          ID(SCICS16)  ACCESS(READ)

SETROPTS CLASSACT(SUBSYSNM)
```

18.9.2 Dataset control

Als gebruikers datasets willen benaderen worden deze in de File Control Table van CICS geplaatst. De datasets (maximum 44 karakters) krijgen dan een logische naam (filenaam) van maximaal 8 karakters. De datasets worden binnen CICS met de logische naam benaderd en CICS voert autorisatie aanvragen op de filenaam uit indien file control is geactiveerd (zie de volgende paragraaf). Voordat gebruikers een dataset kunnen benaderen moet CICS deze openen waarvoor CICS de juiste autorisatie moet hebben. De implementatie:

```
ADDSD     'PROD1.CICS16.APPLC74.INPUT59'  OWNER(SECADM)
          UACC(NONE)
PERMIT    'PROD1.CICS16.APPLC74.INPUT59'
          ID(SCICS16)  ACCESS(ALTER)
```

18.9.3 File control

AANVULLENDE MAATREGEL. Indien Files worden toegepast voor vertrouwelijke gegevens dienen deze te worden beschermd voor lezen, wijzigen en verwijderen door andere gebruikers. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XFCT=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT    RESSEC=YES,XFCT=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(FCICSFCT)

RDEFINE   FCICSFCT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel


RDEFINE   FCICSFCT  CICS23.INPUT59  OWNER(SECADM)  UACC(NONE)
PERMIT    CICS23.INPUT59  CLASS(FCICSFCT)
          ID(VERKOOP)  ACCESS(UPDATE)

SETROPTS CLASSACT(FCICSFCT)  CICS file control

CICS file control groups class
RDEFINE   HCICSFCT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

18.10 Journal Control

De primairy CICS system log dient uitsluitend door CICS region te kunnen worden benaderd en voor gebruik door gebruikerstransacties te worden afgeschermd. De implementatie:

```
RDEFINE   JCICSJCT  DFHLOG  OWNER(SECADM)  UACC(NONE)
          NOTIFY(SECADM)

SETROPTS CLASSACT(JCICSJCT)  CICS Journal control
```

AANVULLENDE MAATREGEL. Indien Journals worden toegepast voor vertrouwelijke gegevens dienen deze te worden beschermd voor lezen, wijzigingen en verwijderen door andere gebruikers. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XJCT=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT     RESSEC=YES,XJCT=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(JCICSJCT)

RDEFINE   JCICSJCT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE   JCICSJCT  CICS23.JOURN23  OWNER(SECADM)  UACC(NONE)
PERMIT    CICS23.JOURN23  CLASS(FCICSJCT)
          ID(VERKOOP)  ACCESS(UPDATE)

SETROPTS CLASSACT(JCICSJCT)  CICS Journal control

CICS Journal control groups class
RDEFINE   KCICSJCT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

18.11 Program Properties

CICS transacties kunnen andere programma's (applicatie programma's) aanroepen met de LINK, LOAD en XCTL services. Het aanroepen van deze programma's wordt niet met de attached- of started transactions security geautoriseerd aangezien dit z/OS services zijn die direct een z/OS gecontroleerd programma opstarten. Welke gebruikers een LINK, LOAD of XCTL service mogen uitvoeren moet worden geautoriseerd. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XPPT=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT     RESSEC=YES,XPPT=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(MCICSPPT)

RDEFINE   MCICSPPT  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE   MCICSPPT  CICS23.CARTRN2  OWNER(SECADM)  UACC(NONE)
PERMIT    CICS23.CARTRN2  CLASS(MCICSPPT)
          ID(VERKOOP)  ACCESS(UPDATE)

SETROPTS CLASSACT(MCICSPPT)        CICS Program Properties control

CICS Program Properties control groups class
```

```
RDEFINE   NCICSPPT **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

18.12 Tempory Storage control

18.12.1 TS Server control

Voor het gebruik van Tempory Storage pools (TS pools) in een Parallel Sysplex-omgeving dient binnen elke z/OS een CICS region te worden geactiveerd die de TS pools beheert. Deze CICS region wordt dan de TS Server genoemd.

Het gebruik van Coupling Facility geheugenstructuren dient te worden geautoriseerd waarbij uitsluitend de TS server toegang krijgt. De Implementatie:

```
RDEFINE   FACILITY  IXLSTR.DFHXQLS_TSPRODA16  UACC(NONE)
PERMIT    IXLSTR.DFHXQLS_TSPRODA16  CLASS(FACILITY)
          ID(STSSRV)  ACCESS(ALTER)
```

Voor het beheer (control) van Tempory Storage pools door de TS server dient deze te worden geautoriseerd voor de TS pools. De implementatie:

```
RDEFINE   FACILITY  DFHXQ.TSPAPPL47  UACC(NONE)
PERMIT    DFHXQ.TSPAPPL47  CLASS(FACILITY)  ID(STSSRV)
          ACCESS(CONTROL)
```

Voor het gebruik van Tempory Storage pools beheerd door de TS server dienen CICS regions, van waaruit de aanvragen komen, hiervoor te worden geautoriseerd. De implementatie:

```
RDEFINE   FACILITY  DFHXQ.TSPAPPL47  UACC(NONE)
PERMIT    DFHXQ.TSPAPPL47  CLASS(FACILITY)
          ID(SCICS16)  ACCESS(UPDATE)
```

Indien geen passend Tempory Storage pool profiel aanwezig is voor een toegangsaanvraag, verleent CICS toegang tot de TS pool. Er dient geen toegang tot TS pools verleend te worden indien er geen passend profiel aanwezig is, daarom dient een sluitprofiel te worden gedefinieerd. De implementatie:

```
RDEFINE   FACILITY  DFHXQ.*  UACC(NONE)
```

18.12.2 Tempory Storage control

AANVULLENDE MAATREGEL. Gebruikers dienen voor het gebruik van Temporary Storage te worden geautoriseerd. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES en XTST=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT    RESSEC=YES,XTST=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(SCICSTST)

RDEFINE   SCICSTST  **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE   SCICSTST  DFHXQLS_TSPRODA16  OWNER(SECADM)
          UACC(NONE)
PERMIT    DFHXQLS_TSPRODA16  CLASS(SCICSTST)  ID(VERKOOP)
          ACCESS(UPDATE)

SETROPTS CLASSACT(SCICSTST)  CICS Temporary Storage control
```

```
CICS Temporary Storage control groups class
RDEFINE    UCICSTST **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

18.13 Program Specification Blocks

AANVULLENDE MAATREGEL. DL/1 Program Specification Control Blocks, gebruikt voor IMS database toegang, dienen voor gebruik te worden geautoriseerd. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES, XPSB=YES en PSBCHK=YES te worden gedefinieerd. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT     RESSEC=YES,XPSB=YES,PSBCHK=YES
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(PCICSPSB)

RDEFINE    PCICSPSB **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel

RDEFINE    PCICSPSB CICS23.PSB397  OWNER(SECADM)  UACC(NONE)
PERMIT     CICS23.PSB397  CLASS(PCICSPSB)
           ID(VERKOOP)  ACCESS(READ)

SETROPTS CLASSACT(PCICSPSB)  CICS Program Specification Block control

CICS Program Specification Block control groups class
RDEFINE    QCICSPSB **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
```

18.14 CICS command control

Met de CEMT 'CICS supplied transacties' en met de EXEC CICS interface kunnen CICS-commando's worden gegeven. Al deze CICS-commando's dienen te worden geautoriseerd. Hiervoor dienen de CICS-initialisatieparameters RESSEC=YES, XCMD=YES en CMDSEC=ALWAYS te worden gedefinieerd. Aangezien CICS commando's geen eenduidige naam hebben die eenvoudig onder een generiek profiel zijn te plaatsen is het aan te bevelen de VCICSCMD groups class te gebruiken. De implementatie van de CICS-initialisatieparameters:

```
DFHSIT     RESSEC=YES,XCMD=YES,CMDSEC=ALWAYS
```

Een voorbeeld van de RACF commando's:

```
SETROPTS GENERIC(CCICSCMD)

RDEFINE    CCICSCMD **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
SETROPTS CLASSACT(CCICSCMD)  CICS Command control

CICS Command control groups class
RDEFINE    VCICSCMD **  OWNER(SECADM)  UACC(NONE)  Sluitprofiel
RDEFINE    VCICSCMD  SCICS39.OPERCMD  OWNER(SECADM)
                       UACC(NONE)  NOTIFY(SECADM)
                       ADDMEM(AUTINSTMODEL, AUTOINSTALL,
                     CONNECTION,  DSNAME, TRANSACTION,
                     TRANDUMPCODE, VTAM)
PERMIT     SCICS39.OPERCMD  CLASS(VCICSCMD)
           ID(OPERGRP2)  ACCESS(READ)  READ voor inquiry
```

```
PERMIT    SCICS39.OPERCMD   CLASS(VCICSCMD)
          ID(OPERGRP2)  ACCESS(UPDATE)  UPDATE voor Perform, Set,
          Create en Discard
```

Opmerking: voor de gehele lijst van CICS-commando's dient de documentatie van de leverancier te worden geraadpleegd en dient deze in de praktijk verder te worden aangepast en gelimiteerd naar de noodzakelijkheid voor de betreffende functies.

CICS Operators dienen uitsluitend die informatie te krijgen die voor de uitvoering van hun werkzaamheden noodzakelijk zijn. Een voorbeeld:

```
ADDUSER  XKRET01   NAME('T. Krens')  OWNER(HRMCENTR)
  CICS(OPCLASS(3)   OPIDENT(100)
       OPPRTY(250)   TIMEOUT(0030)
       XRFSOFF(NOFORCE))
```

18.15 z/OS command control

Om een MVS console voor CICS console te gebruiken dient op het MVS console een CICS sign-on te worden uitgevoerd. Dit gebeurt met het MVS modify commando. Een voorbeeld van het MVS-commando:

```
MODIFY    CICS48,CESN   USERID=OPER386,PS=geheim
```

Het gebruik van het MVS modify commando dient te worden geautoriseerd. Een voorbeeld:

```
RDEFINE    OPERCMDS   MVS.MODIFY   OWNER(SECADM)   UACC(NONE)

PERMIT     MVS.MODIFY   CLASS(OPERCMDS)   OWNER(SECADM)
                        ID(OPERGRP3)   ACCESS(READ)
```

18.16 CICS supplied transaction

Uitsluitend CICS region userids mogen geautoriseerd zijn voor de zogenaamde 'category 1 CICS supplied transacties'. De CICS region userids moeten via een groep worden geautoriseerd zoals in het onderstaande voorbeeld is weergegeven. Een voorbeeld:

```
RDEFINE    GCICSTRN  CICS85.CICSCAT1   OWNER(SECADM)
                     NOTIFY(SECADM)   UACC(NONE)
                     ADDMEM(CSPQ CDBD CBDO etc. etc.)
PERMIT     CICS85.CICSCAT1   CLASS(GCICSTRN)
                     ID(SGRPCICS)   ACCESS(READ)
```

Afhankelijk van de functie van de gebruikers mogen sign-on userids voor een selectie van de zogenaamde 'category 2 CICS supplied transacties' zijn geautoriseerd. De sign-on userids moeten via een groep worden geautoriseerd zoals in het onderstaande voorbeeld is weergegeven. Voor de voorbeelden van de betreffende transacties zie de documentatie van de leverancier. Hieronder voorbeelden voor de verschillende functionele gebruikers.

Een voorbeeld voor de CICS-systeemadministrators:

```
RDEFINE    GCICSTRN  SCICS85.CICSCAT2A   OWNER(SECADM)
                     NOTIFY(SECADM)   UACC(NONE)
                     ADDMEM(CBRC  CDBT  CEDA  CEMT  CETR)
PERMIT     SCICS85.CICSCAT2A   CLASS(GCICSTRN)
                     ID(GCICSADM)   ACCESS(READ)
```

Een voorbeeld voor de CICS-operators:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2D  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CEOT  CEST  CMSG  CWTO)
PERMIT    SCICS85.CICSCAT2D  CLASS(GCICSTRN)
                   ID(GOPERGR8)  ACCESS(READ)
```

Een voorbeeld voor de CICS-webgebruikers:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2W  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CWBA)
PERMIT    SCICS85.CICSCAT2W  CLASS(GCICSTRN)
                   ID(GWEBUSER)  ACCESS(READ)
```

Een voorbeeld voor de CICS-ontwikkelaar:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2O  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CEBR  CECI  CECS  CEDB  CEDF)
PERMIT    SCICS85.CICSCAT2O  CLASS(GCICSTRN)
                   ID(GCICSONT)  ACCESS(READ)
```

Een voorbeeld voor CICS-intercommunicatie:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2C  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CEHP  CEHS  CPMI  CRTE CSMI
                          CSM1  CSM2  CSM3  CSM5  CVMI)
PERMIT    SCICS85.CICSCAT2C  CLASS(GCICSTRN)
                   ID(GCICSCOM)  ACCESS(READ)
```

Een voorbeeld voor CICS-inquire:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2I  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CDBI  CEDC)
PERMIT    SCICS85.CICSCAT2I  CLASS(GCICSTRN)
                   ID(GCICSADM  GOPERGR8)  ACCESS(READ)
```

Een voorbeeld voor alle CICS sign-on users:

```
RDEFINE   GCICSTRN SCICS85.CICSCAT2U  OWNER(SECADM)
                   NOTIFY(SECADM)  UACC(NONE)
                   ADDMEM(CMAC  CRTX  CSGM)
PERMIT    SCICS85.CICSCAT2U  CLASS(GCICSTRN)
                   ID(GCICSADM  GOPERGR8)  ACCESS(READ)
```

Opmerking: de autorisatie voor ‘category 2 CICS supplied transacties’ dient in de praktijk verder te worden aangepast en gelimiteerd naar de noodzakelijkheid voor de betreffende functies.

18.17 Logging

CICS dient alle security gerelateerde gebeurtenissen vast te leggen. Hiervoor dient de z/OS LOGSTREAMER te worden geactiveerd en voor de CICS regions te worden geautoriseerd. Tevens dient de logstream voor een heldere identificatie, de naam van de CICS region userid te krijgen. De implementatie:

```
RDEFINE   LOGSTRM  SCICS23.**  UACC(NONE)
PERMIT    SCICS23.**  CLASS(LOGSTRM)  ID(SCICS23)  ACCESS(ALTER)
```

Als meerdere applicaties van dezelfde CICS region gebruik maken dient een specifieker generiek profiel te worden aangemaakt en geautoriseerd. De implementatie:

```
RDEFINE   LOGSTRM  SCICS23.QUERY.*  UACC(NONE)
PERMIT    SCICS23.QUERY.*  CLASS(LOGSTRM)
          ID(SCICS23)  ACCESS(ALTER)
```

Alleen geautoriseerde gebruikers mogen de logstreams lezen. De implementatie:

```
PERMIT    SCICS23.QUERY.*  CLASS(LOGSTRM)
          ID(GAPPL23)  ACCESS(READ)
```

Het beheer van de logstreams mag uitsluitend door geautoriseerde gebruikers worden uitgevoerd. De implementatie:

```
PERMIT    SCICS23.QUERY.*  CLASS(LOGSTRM)
          ID(OARCH)  ACCESS(UPDATE)
```

18.17.1 VTAM trace

AANVULLENDE MAATREGEL. Het kunnen uitlezen van gegevens van CICS-gebruikers door VTAM (trace data) is niet toegestaan. De implementatie van de CICS-initialisatieparameter:

```
DFHSIT    CONFDATA=HIDETC,CONFTXT=YES
```

18.18 CICS region communication Security

Voor het verdelen van de workload over verschillende CICS regions of het toekennen van bepaalde functies of gegevensbeheer aan een CICS region en toch voor de gebruiker één logische CICS-omgeving te maken kunnen CICS regions met elkaar communiceren. Voor communicatie tussen CICS-regions binnen één host (één z/OS, of één Sysplex) kan de Multi Region Option (MRO) worden toegepast.

Voor communicatie tussen CICS regions die over verschillende hosts zijn verdeeld zal over het netwerk moeten worden gecommuniceerd. Hiervoor wordt het SNA netwerk protocol LU 6.1 of LU 6.2 gebruikt. Het LU 6 protocol is ontwikkeld om applicaties met elkaar te kunnen laten communiceren. Voor communicatie tussen CICS-CICS, IMS-IMS en CICS-IMS is LU 6.1, een verbeterde versie ontwikkeld. Later is LU 6.2, ook Advance Program-to-Program Communication (APPC) genoemd, ontwikkeld wat al aangeeft dat dit protocol geavanceerder is dan de vorige versies. APPC is het universele program-to-program netwerk communicatie protocol geworden met vele standaard beveiligingsmechanismen. Zowel CICS als elk ander programma kan voor netwerkcommunicatie van APPC en de daarbij horende beveiligingsmechanismen gebruik maken.

Voor de communicatie tussen CICS regions zijn drie netwerkbeveiligingsmechanismen aanwezig, namelijk:

- Bind-time (of session) beveiliging, voorkomt ongeautoriseerde netwerksessies tussen CICS regions. Op LU-LU niveau, dus tussen twee netwerkverbindingen, moet een sessie geautoriseerd zijn. Met een LU 6.2 zijn meerdere parallel sessies mogelijk.
- Link beveiliging (of conversatie), autorisatie of een remote CICS over deze sessie transacties mag opstarten en resources mag benaderen. Over een LU-LU sessie kunnen meerdere CICS regions converseren.
- User beveiliging, autorisatie of een gebruiker vanuit een remote CICS region transacties mag uitvoeren en resources mag benaderen. Meerdere gebruikers kunnen gebruik maken van dezelfde link.

Bij communicatie tussen CICS regions dient het beveiligingsniveau van elke individuele CICS region niet door andere CICS regions te worden verlaagd. Bij communicatie met een lager of niet beveiligde CICS region dient altijd identificatie en authenticatie (userid/wachtwoord) plaats te vinden.

18.18.1 Multi Region Operation

Multi Region Operation geeft de mogelijkheid CICS-regions binnen één host (één z/OS of één Sysplex) te laten communiceren. Het communiceren tussen de CICS regions dient altijd via de MRO SVC te worden uitgevoerd. Deze SVC voert altijd security checking uit, ongeacht de instellingen van de betreffende CICS regions. Om MRO te kunnen toepassen dient het LU61 protocol te worden geactiveerd over Cross Memory services. Een voorbeeld voor de connectie naar CICSB op het CICSA systeem:

```
CEDA      DEFINE     CONNECTION(CICSB)
                     ACCESSMETHOD(XM)   PROTOCOL(LU61)
```

18.18.1.1 Bind-time security

Voor bind-time security dienen per CICS region een APPLID profiel in de FACILITY class te worden gedefinieerd. Een voorbeeld:

```
RDEFINE   FACILITY   DFHAPPL.CICSA   UACC(NONE)
RDEFINE   FACILITY   DFHAPPL.CICSB   UACC(NONE)
```

CICS regions dienen te worden geautoriseerd om gebruik te maken van een APPLID en zich daarmee te identificeren als een local MRO CICS region. Het CICS region userid dient hiervoor te worden geautoriseerd. Een voorbeeld:

```
PERMIT    DFHAPPL.CICSA   CLASS(FACILITY)
          ID(SCICSA)   ACCESS(UPDATE)
PERMIT    DFHAPPL.CICSB   CLASS(FACILITY)
          ID(SCICSB)   ACCESS(UPDATE)
```

Om CICS regions met elkaar te laten communiceren (MRO) dienen deze voor elkaars APPLID geautoriseerd te zijn. Een voorbeeld:

```
PERMIT    DFHAPPL.CICSA   CLASS(FACILITY)
          ID(SCICSB)   ACCESS(READ)
PERMIT    DFHAPPL.CICSB   CLASS(FACILITY)
          ID(SCICSA)   ACCESS(READ)
```

18.18.1.2 Link security

Ondanks dat over een MRO bind maar één link mogelijk is moet een link userid worden gedefinieerd. Dit userid moet in RACF zijn gedefinieerd en wordt gebruikt voor het geven van toegangsrechten als het remote CICS-user userid niet wordt ontvangen. Als een link userid niet gedefinieerd is zal hiervoor in de plaats het remote CICS region userid worden gebruikt, wat niet is toegestaan. Een voorbeeld van de definitie van een link userid van de CICS definities:

```
CEDA  DEFINE   CONNECTION(CICL)   SECURITYNAME(UCICR05)
```

Bij het gebruik van de USERID parameter in de SESSIONS definitie zal deze als link userid worden gebruikt in plaats van het userid van de SECURITYNAME parameter in de CONNECTION definitie. Een voorbeeld van de CICS definitie:

```
CEDA  DEFINE   SESSIONS(SESS01)   USERID(UCICT06)
```

18.18.1.3 User security

De remote CICS region dient altijd het betreffende CICS-user userid voor identificatie te versturen en daardoor autorisatie checking, door de ontvangende local CICS region, mogelijk te maken. Het link userid mag niet als default worden gebruikt. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **USEDFLTUSER**(**NO**)

Bij een untrusted remote CICS region dient altijd het betreffende CICS-user userid/password combinatie voor identificatie/authenticatie door de local CICS region ontvangen te worden. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**VERIFY**)

Bij een trusted remote CICS region is het niet nodig het password te vragen en kan worden volstaan met uitsluitend het userid (identificatie). De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**IDENTIFY**)

Bij een trusted remote CICS region is het niet nodig bij elke transactie het password te vragen en kan worden volstaan met uitsluitend het userid (identificatie). Overwogen kan worden uitsluitend bij de eerste transactie aanvraag het password te leveren en gedurende de volgende transactie aanvragen aan te nemen dat het CICS-user userid volstaat. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**PERSISTENT**)

18.18.2 InterSystem Communication met LU 6.1

Een InterSystem Communication (ISC) verbinding gebaseerd op LU 6.1, mag uitsluitend worden toegepast indien een LU 6.2 technisch niet mogelijk is, bijvoorbeeld tussen CICS-IMS. Voor alle andere verbindingen waarbij netwerk connecties noodzakelijk zijn moet LU 6.2 worden toegepast. Om het LU 6.1 protocol te kunnen toepassen dient LU61 te worden geactiveerd. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*IMS5*)
ACCESSMETHOD(**VTAM**) **PROTOCOL**(**LU61**)

18.18.2.1 Bind security

Bind-time security bij een LU 6.1 verbinding wordt technisch niet ondersteund.

18.18.2.2 Link security

De identificatie en dus de autorisatie bij een LU 6.1 verbinding kan uitsluitend worden gebaseerd op het link userid. Dit userid moet binnen CICS worden gedefinieerd. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*IMS5*) **SECURITYNAME**(*UWERT05*)

18.18.2.3 User security

User security bij een LU 6.1 verbinding wordt technisch niet ondersteund. Het userid dat voor identificatie en daarmee voor autorisatie gebruikt wordt is altijd het link userid.

18.18.3 InterSystem Communication met LU 6.2

Om het InterSystem Communication (ISC) met het LU 6.2 protocol te kunnen toepassen dient het APPC protocol te worden geactiveerd. De implementatie van de CICS definitie:

```
CEDA  DEFINE  CONNECTION(CIC5)
              ACCESSMETHOD(VTAM)  PROTOCOL(APPC)
```

Verder dient CICS de beveiliging met het APPC protocol te ondersteunen. De implementatie van de CICS initialisatie parameter:

```
XAPPC=YES
```

18.18.3.1 Bindtime security

Bind-time security dient bij het LU 6.2 protocol te worden geactiveerd. De implementatie van de CICS definitie:

```
CEDA  DEFINE  CONNECTION(CIC5)  BINDSECURITY(YES)
```

Ter controle of de remote CICS-region de local CICS region mag benaderen is een netwerknaam nodig. De implementatie van de CICS definitie:

```
CEDA  DEFINE  CONNECTION(CIC5)  NETNAME(CICSREM)
```

De remote CICS region dient binnen RACF te worden gedefinieerd voor het gebruik van de betreffende LU. Meerdere remote CICS regions kunnen gebruik maken van een LU 6.2. Iedere CICS region dient dan ook in een APPCLU profiel te worden gedefinieerd. Voor het gebruik van de LU dient dezelfde session key (SESSKEY) in het beveiligingssysteem van de remote CICS region te worden gedefinieerd. Een voorbeeld:

```
RDEFINE  APPCLU  NETREMOT.CICSLOC.CICSREM  UACC(NONE)
                 SESSION(SESSKEY(946739576))  AUDIT(ALL(READ))
```

Opmerking: de netwerk identificatie, hier genoemd NETREMOT, de netwerknaam waarin de remote CICS region zich bevindt, is de naam gedefinieerd met de VTAM NETID-parameter in het SYS1.VTAMLST(ATCSTRxx) member.

18.18.3.2 Link security met LU6.2:

Met een LU 6.2 bind zijn meerdere links mogelijk en daarom moet een link userid het onderscheid maken van welke link (remote CICS region) de aanvraag voor een transactie komt. Dit link userid moet in RACF zijn gedefinieerd en wordt gebruikt voor het geven van toegangsrechten als het remote CICS-user userid niet wordt ontvangen. Als een link userid niet gedefinieerd is zal hiervoor in de plaats het remote CICS region userid worden gebruikt, wat niet is toegestaan. Een voorbeeld van de definitie van een link userid van de CICS definities:

```
CEDA  DEFINE  CONNECTION(CICL)  SECURITYNAME(UCICR05)
```

Met een LU 6.2 verbinding kunnen meerdere sessies over een link actief zijn. Het is daarom aan te raden een userid te verbinden aan de sessie met de CICS region die men wil autoriseren. Hiervoor dient de USERID parameter in de SESSIONS definitie te worden gebruikt. Bij het gebruikt van de USERID parameter in de SESSIONS definitie zal deze als link userid worden gebruikt in plaats van het userid van de SECURITYNAME parameter in de CONNECTION definitie. Een voorbeeld van de CICS definitie:

```
CEDA  DEFINE  SESSIONS(SESS01)  USERID(UCICT06)
```

18.18.3.3 User beveiliging

De remote CICS region dient altijd het betreffende CICS-user userid voor identificatie te versturen en daardoor autorisatie checking, door de ontvangende local CICS region, mogelijk te maken. Het link userid mag niet als default worden gebruikt. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **USEDFLTUSER**(**NO**)

Bij een untrusted remote CICS region dient altijd het betreffende CICS-user userid/password combinatie voor identificatie/authenticatie door de local CICS region ontvangen te worden. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**VERIFY**)

Bij een trusted remote CICS region is het niet nodig het password te vragen en kan worden volstaan met uitsluitend het userid (identificatie). De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**IDENTIFY**)

Bij een trusted remote CICS region is het niet nodig bij elke transactie het password te vragen en kan worden volstaan met uitsluitend het userid (identificatie). Overwogen kan worden uitsluitend bij de eerste transactie aanvraag het password te leveren en gedurende de volgende transactie aanvragen aan te nemen dat het CICS-user userid volstaat. De implementatie van de CICS definitie:

CEDA DEFINE CONNECTION(*CICL*) **ATTACHSEC**(**PERSISTENT**)

18.19 Web support

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

18.19.1 HTML

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

18.19.2 Web support transactions

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

18.19.3 TCP/IP

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

18.19.4 Secure Socket Layer

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

18.20 CICS Beveiligingslevels en –categorieën

Dit onderwerp dient nog te worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

19 CICSPlex System Manager

19.1 Introductie

19.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

20 Message Queuing Series (MQSeries)

20.1 Introductie

20.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

21 Information Management System (IMS)

21.1 Introductie

Het Informatie Management System (IMS) is een Transactie Management (IMS TM) en Database Management (IMS DB) systeem geschikt voor zeer grote transactie volumes. In de praktijk kunnen dit miljoenen transacties per uur, per systeem zijn. Het IMS-systeem is hierbij uitstekend geschikt voor vooral grote organisaties, zoals banken, verzekeringen, beurzen, verkooporganisaties, etc. waar een centrale verwerking van gegevens noodzakelijk is om de integriteit van de gegevens te waarborgen. Opgemerkt dient te worden dat IMS uitsluitend voor IBM mainframes is ontwikkeld.

Met uitzonderlijke dank aan de werkgroepleden voor hun bijdrage in de ontwikkeling van dit hoofdstuk:

- Frank Engel, KPMG Information Risk Management
- Peter Mienes, KPMG Information Risk Management
- Leen van Rij, KPMG Information Risk Management

21.2 Status

De inhoud is mede gebaseerd op informatie uit IBM manuals en Redbooks, en dient nog (in een testomgeving) te worden gevalideerd. Het document is gebaseerd op IMS V7R1 en z/OS V1R1. Dit hoofdstuk moet nog verder worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

21.3 Installatie

21.3.1 IMS macro's

IMS wordt opgebouwd (gegenereerd) uit een set van macro's de zogenaamde SYSGEN macro's. Voor de beveiliging kunnen in een drietal macro's parameters worden opgegeven. Deze macro's zijn de SECURITY, COMM en IMSGEN macro. Alle beveiligingsparameters opgenomen in de SECURITY macro bepalen het uiteindelijke beveiligingsniveau van IMS waarbij gelijkluidende parameters in de COMM of IMSGEN macro worden genegeerd.

Uitsluitend parameters in de SECURITY macro mogen worden gebruikt om het beveiligingsniveau te implementeren. Een voorbeeld:

```
SECURITY  TYPE=( RACFAGN,RACFTERM,RACFCOM),
          SECLVL=(FORCTRAN,FORCSIGN)
```

21.3.2 IMS exits

Het gebruik van IMS-exits is niet toegestaan tenzij er dringende redenen voor zijn en het gebruik expliciet is toegelicht en vastgelegd.

21.3.3 IMS Libraries

Toegang tot IMS-libraries dient te zijn beveiligd. Uitsluitend de noodzakelijke toegangsrechten dienen te worden gegeven. Een voorbeeld:

```
ADDSD    'IMS.PROD1. RESLIB*'  UACC(NONE)   OWNER(SECADM)
PERMIT   'IMS.PROD1. RESLIB'   ID(SIMSCNTL)  ACCESS(EXECUTE)
```

Toegang tot overige IMS-systeembestanden dient eveneens te zijn beperkt tot user-id's die de toegang nodig hebben. Een voorbeeld:

```
ADDSD    'IMS.PROD1.PROCLIB'  UACC(NONE)   OWNER(SECADM)
PERMIT   'IMS.PROD1.PROCLIB'  ID(SIMSCNTL)  ACCESS(READ)
```

21.3.4 IMS Started Tasks

De IMS started tasks (STCs) moeten onder een user-id draaien die niet voor logon kan worden gebruikt (protected user-id). Een voorbeeld:

```
ADDUSER  SIMSCNTL  DFLTGRP(STCGROUP)  OWNER(SECADM)
         NOPASSWORD
```

Elke IMS control region moet als een started task of started job worden geactiveerd. Het gebruik van het 'Trusted' en 'Privileged' attribuut is hierbij niet toegestaan. Een voorbeeld:

```
RDEFINE  STARTED  IMSA.*  OWNER(SECADM)
         STDATA(USER(SIMSCNTL)  GROUP(STCGROUP)
         PRIVILEGED(NO)  TRUSTED(NO))
```

21.3.5 Security Maintenance Utility

Het gebruik van de Security Maintenance Utility (SMU), tenzij anders niet mogelijk, is voor het definiëren van userids, passwords en autorisaties niet toegestaan.

21.4 Common Queue Structures (CQS) Sysplex

Indien de Common Queue Structures CQS in een (parallel) sysplex-omgeving wordt toegepast moet deze voor gebruik door een IMS-systeem worden geautoriseerd. De naam van de sysplex-structure moet zijn opgenomen in de parameter STRNAME en OVFLWSTR in de PROCLIB members CQSSLxx voor de local queue, en/of in CQSSGxx voor de global queue en in het CQS-initialisatie-member CQSIPxx. In het voorbeeld hieronder is de naam IMSMSGQ1 aangenomen. Een voorbeeld:

```
STRNAME= IMSMSGQ1
```

Een voorbeeld van de RACF definities die nodig zijn voor de Common Queue Structures:

```
RDEFINE  FACILITY  CQSSTR. IMSMSGQ1  OWNER(SECADM)
                   UACC(NONE)
PERMIT   CQSSTR. IMSMSGQ1  CLASS(FACILITY)
                   ID(SIMSCNTL)  ACCESS(UPDATE)
```

De toegang tot de XCF-structuren die door het CQS-subsysteem worden gebruikt, worden met een CQSSTR.*structuurnaam* profiel in de RACF FACILITY class beschermd. Echter als geheel andere faciliteiten de generieke IXL-services gebruiken zijn deze CQS-structuren hier niet voor beschermd. Alhoewel het risico van misbruik klein is omdat IXL-services uitsluitend door APF geautoriseerde gebruikers kunnen worden toegepast is het toch wenselijk de structuren voor de IXL-services te beschermen. Een voorbeeld:

```
RDEFINE  FACILITY  IXLSTR.IMSMSGQ1  OWNER(SECADM)
                   UACC(NONE)

PERMIT   IXLSTR.IMSMSGQ1  CLASS(FACILITY)  ID(CQSA)
```

ACCESS(**UPDATE**)

21.5 Master Terminal

Nagegaan moet nog worden of het mogelijk is om ook voor de MTO een sign-on af te dwingen, of in elk geval: hoe de MTO functioneert zodra command security is geactiveerd, en hoe afgedwongen kan worden dat vanaf de MTO geen transacties kunnen worden gegeven.

Er is geen betrouwbare informatie gevonden, in de geraadpleegde literatuur genoemd in de laatste paragraaf, over de afscherming van het Master Terminal gebruik. Zo is niet duidelijk of er een beveiligingsmechanisme is om de Master Terminal Operator (MTO) te laten aanloggen (identificatie en authenticatie) en daarna de uitvoering van IMS commando's selectief te kunnen autoriseren. Zie voor commando autorisatie paragraaf 'IMS commands control'. Daar er onvoldoende informatie beschikbaar is zal het verder uitzoeken van de beveiligingsmechanismen voor het Master Terminal in samenwerking met de IMS-deskundigen van het B/CICT moeten plaatsvinden. Wel kan worden gezegd dat het aanloggen door MTOs een aanvullende maatregel is. De fysieke toegangbeveiliging moet altijd afdoende geregeld zijn zodat ongeautoriseerde personen geen toegang tot de Master Terminal hebben.

21.6 IMS commands control

Het gebruik van alle kritieke IMS-commando's moet worden geautoriseerd. Voor het activeren van commandobeveiliging onder RACF moet de volgende parameter in de IMS SECURITY-macro worden gedefinieerd. De implementatie:

```
SECURITY TYPE=(,,RACFCOM)
```

Voor het gebruik van IMS-commando's dient de CIMS general resource class te worden geactiveerd of zoals in het hierna beschreven voorbeeld de DIMS general resource class. In de DIMS class kunnen commando's die moeilijk met een wildcard (generic profile) zijn te definiëren worden gegroepeerd. De IMS-commando's behoeven slechts in één van de IMS command classes (CIMS en DIMS) te worden gespecificeerd. Een voorbeeld:

```
RDEFINE   DIMS   ALLUSERS   OWNER(SECADM)   UACC(READ)
          ADDMEM(BRO CAN DIS END EXC EXI FOR HOL LOG LOO
                 RCL RCO RDI REL RES RML SET SIG TES)

RDEFINE   DIMS   IMSOPER    OWNER(SECADM)   UACC(NONE)
          ADDMEM(ACT ALL ASS CHA CHE CLS COM CQ* DBD DBR
                 DEL DEQ ERE IAM IDL LOC MOD MON MSA MSV
                 NRE OPN PST PUR QUI RM* RST RTA SEC SMC
                 SSR STA STO SWI TRA UNL VUN)
PERMIT    IMSOPER   CLASS(DIMS)   ID(OPERGRP)   ACCESS(READ)

PERMIT    STO    CLASS(CIMS)   ID(CICS12)   ACCESS(READ)

SETROPTS CLASSACT(CIMS  DIMS)
```

AANVULLENDE MAATREGEL. Door IMS kan herverificatie van het password worden afgedwongen bij het uitvoeren van commando's, ondanks dat initieel identificatie en authenticatie heeft plaatsgevonden. De implementatie van de IMS-parameter in de IMS-procedure (als EXEC parm of statement in DFSPBxxx):

```
RVFY=Y
```

In RACF moet in het betreffende commando-profiel worden aangegeven welke IMS-commando('s) herverificatie moeten uitvoeren. Een voorbeeld:

```
RDEFINE   CIMS   STA   OWNER(SECADM)   UACC(NONE)
                       APPLDATA('REVERIFY')
```

21.7 Commando's van MCS of E-MCS Consoles

Vanuit MCS of E-MCS consoles kunnen IMS commando's worden gegeven. Het gebruik hiervan is beschreven in het hoofdstuk 'z/OS faciliteiten' van de PI RACF standaard.

AANVULLENDE MAATREGEL. Vanaf een Multiple Console Support (MCS) of een Extended Multiple Console Support (E-MCS) console kunnen IMS commando's worden gegeven. Deze commando's moeten worden beveiligd door in de DFSPBxxx member het volgende op te nemen:

```
CMDMCS=R              voor DB/DC en DBCTL
```

Deze functionaliteit vereist wel dat een command recognition character voor IMS is gereserveerd en dat consoles zijn aangelogd.

Zie verder de paragraaf over IMS commands control.

21.8 Automated Operations (AO) Programs die een DL/I ICMD Call doen

Nog te ontwikkelen.

21.9 Scheiding van IMS-systemen

Voor het binnen één besturingssysteem opereren van verschillende IMS-systemen kunnen binnen RACF verschillende general resource classes worden gebruikt. Met de verschillende classes kunnen verschillende RACF-autorisaties worden verleend. De standaard class-namen eindigen op IMS, bijvoorbeeld TIMS voor transactie beveiliging. Deze naam kan worden aangepast naar bijvoorbeeld de class TEST. Binnen RACF dient deze class te worden gedefinieerd en geactiveerd in de Class Descriptor Table (CDT). IMS moet worden aangepast om de betreffende RACF general resource classes aan te spreken. Dit kan worden bereikt door de SECURITY-macro aan te passen. Een voorbeeld:

```
IMSCTRL   SECURITY(RCLASS=IMS)      standaard implementatie

of

IMSCTRL   SECURITY(RCLASS=EST)      dit wordt de RACF
                                    general resource class TEST
                                    voor bijvoorbeeld het testsysteem.
```

21.10 Database data sets

IMS datasets, die een database bevatten, moeten met een dataset profiel beschermd zijn. Alleen IMS started tasks en batch user-id's mogen update autorisaties op deze datasets hebben. Toegang tot de databasegegevens mag uitsluitend via IMS worden gegeven. Een voorbeeld:

```
ADDSD     'IMS.DB.PROD.*'   OWNER(SECADM)   UACC(NONE)
PERMIT    'IMS.DB.PROD.*'   ID(SIMSDB)     ACCESS(CONTROL)
```

21.11 Database gegevens

AANVULLENDE MAATREGEL. In de IMS-database opgeslagen gegevens moeten voor gebruik worden geautoriseerd. Voor het autoriseren van de verschillende databases worden de RACF general resource classes PIMS en QIMS gebruikt. Een voorbeeld van de RACF commando's:

```
RDEFINE   PIMS  IMSDB1  OWNER(SECADM)  UACC(NONE)

PERMIT    IMSA  CLASS(PIMS)  ID(VERKOOP)  ACCESS(READ)

SETROPTS CLASSACT(PIMS)          IMS databases

IMS database groups class
RDEFINE   QIMS  DBGRPA7  ADDMEM(IMSDB1  IMSDB3)
                OWNER(SECADM)  UACC(NONE)

SETROPTS CLASSACT(QIMS)
```

AANVULLENDE MAATREGEL. Voor het autoriseren van de verschillende databasesegmenten worden de RACF general resource classes SIMS en UIMS gebruikt. Een voorbeeld van de RACF commando's:

```
RDEFINE   SIMS  SEGA594  OWNER(SECADM)  UACC(NONE)

PERMIT    SEGA594  CLASS(SIMS)
                   ID(VERKOOP)  ACCESS(READ)

SETROPTS CLASSACT(SIMS)          IMS databases segmenten

IMS database segment groups class
RDEFINE   UIMS  SEGGRP9  ADDMEM(SEGC946  SEGH946  SEGK054)
                OWNER(SECADM)  UACC(NONE)

SETROPTS CLASSACT(UIMS)
```

AANVULLENDE MAATREGEL. Voor het autoriseren van de verschillende velden in de database-segmenten worden de RACF general resource classes FIMS en HIMS gebruikt. Een voorbeeld van de RACF commando's:

```
RDEFINE   FIMS  FLDF957  OWNER(SECADM)  UACC(NONE)

PERMIT    FLDF957  CLASS(FIMS)
                   ID(VERKOOP)  ACCESS(READ)

SETROPTS CLASSACT(FIMS)          IMS databases segmentvelden

IMS database segmentvelden groups class
RDEFINE   UIMS  FLDGRP4  ADDMEM(FLDK094  FLD046  FLD748  FLD299)
                OWNER(SECADM)  UACC(NONE)

SETROPTS CLASSACT(UIMS)
```

21.12 Gebruikerstoegang IMS-systeem

Toegang tot IMS dient beperkt te blijven tot uitsluitend die personen die dit nodig hebben voor de uitvoering van hun werkzaamheden. Het gebruik van IMS wordt geregeld door de IMS control region als een RACF-controlled applicatie te beschermen. RACF gebruikt hierbij als applicatienaam de IMS-naam zoals gedefinieerd in de IMSCTRL-macro. Een voorbeeld:

```
IMSCTRL    IMSID=IMSA
```

Om een gebruiker toegang tot IMS te geven dient in RACF een APPL-definitie te worden gemaakt. Een voorbeeld:

```
RDEFINE    APPL   IMSA   UACC(NONE) )  OWNER(SECADM)
PERMIT     IMSA   CLASS(APPL)   ID(XMIEP01)   ACCESS(READ)

SETROPTS CLASSACT(APPL)
```

21.13 Gebruikers Identificatie en Authenticatie

IMS-autorisatie door alleen terminal access control mag niet worden toegepast. In de praktijk worden terminal namen dynamisch toegekend waardoor het onmogelijk wordt deze te gebruiken voor autorisatiedoeleinden. Gebruikers dienen zich daarom te identificeren en te authenticeren door middel van de persoonlijke userid/wachtwoord combinatie. Hiermee wordt tevens voorkomen dat zonder identificatie en authenticatie activiteiten op de IMS master terminal kunnen worden uitgevoerd. Om sign-on van gebruikers af te dwingen dient in de SMU te worden aangegeven dat vanaf elke terminal een sign-on verplicht is. De implementatie:

```
)(SIGN
 STERM ALL
```

Voor het aanroepen van RACF, voor identificatie en authenticatie, moet de SECURITY-macro als volgt worden gedefinieerd. De implementatie:

```
SECURITY  TYPE=(,RACFTERM,),SECLVL=(,FORCSIGN)
```

Bij de sign-on mag het wachtwoord niet op het scherm worden getoond; een methode om dit te realiseren is gebruik te maken van het bij IMS meegeleverde sign-on screen. Op homemade schermen dient het invoerveld voor een password zo worden geformatteerd dat deze niet zichtbaar is.

21.13.1 PassTickets

AANVULLENDE MAATREGEL. Om er voor te zorgen dat een IMS password niet leesbaar over het netwerk wordt verstuurd kunnen IMS PassTickets toepassen. Een Passticket is een alternatief voor een RACF password en kan, nadat deze is gegenereerd, gedurende 10 minuten één maal door de gebruiker voor het aanloggen worden gebruikt. Zie voor een beschrijving en gebruik het hoofdstuk ‘PassTickets’.

21.14 Terminal Identificatie en Autorisatie

Terminal identificatie en autorisatie is geen standaard maatregel, omdat de standaard autorisaties moeten worden bepaald op basis van het user-id.

Indien in RACF de TERMINAL class reeds voor andere doeleinde is geactiveerd, moet ervoor gezorgd worden dat ook de terminals van de IMS-gebruikers geautoriseerd zijn. Dit kan, afhankelijk van de overige toepassingen van de TERMINAL class, bijvoorbeeld door de SETROPTS TERMINAL universal access te specificeren. Hierdoor kunnen de terminals die niet in de TERMINAL class zijn gedefinieerd toegang krijgen tot applicaties waaronder IMS. De implementatie:

```
SETROPTS        CLASSACT(TERMINAL)   TERMINAL(READ)
```

AANVULLENDE MAATREGEL. Indien vaste netwerk-adressen van toepassing zijn kan worden overwogen om de toegang per werkstation te autoriseren.

Indien een terminal/werkstation uitsluitend door een specifieke gebruiker mag worden gebruikt moet dit in een profiel in de TERMINAL of GTERMINL general resource class worden gedefinieerd. Een voorbeeld voor de TERMINAL class:

```
RDEFINE    TERMINAL  TERM1234  UACC(NONE)
PERMIT     CLASS(TERMINAL)  TERM1234  ID(XMIEP01)  ACCESS(READ)
SETROPTS CLASSACT(TERMINAL)
```

AANVULLENDE MAATREGEL. Indien een gebruiker IMS alleen vanaf een bepaald werkstation mag gebruiken moet dit als conditie worden opgenomen. Een voorbeeld:

```
RDEFINE    APPL  IMSA   OWNER(SECADM)   UACC(NONE)
PERMIT     CLASS(APPL)  IMSA  ID(XMIEP01)  ACCESS(READ)
           WHEN(TERMINAL(TERM1234))
SETROPTS CLASSACT(APPL)
```

21.15 Extended Terminal Option

Bij het gebruik van terminals/werkstations door middel van de Extended Terminal Option (ETO) is het niet nodig deze in IMS te definiëren. Elke in het netwerk gedefinieerde terminal/werkstation kan toegang tot het IMS-systeem zoeken. Om deze reden moet elk gebruiker zich identificeren en authenticeren met userid en password. Zie voor de normen het gedeelten “Gebruikers Identificatie en Authenticatie” en “Terminal Identificatie en Autorisatie”.

21.16 Transaction security

Alle kritieke IMS-transacties moeten met RACF zijn beveiligd. Voor het activeren van transactiebeveiliging onder RACF moet binnen IMS de volgende parameters in de SECURITY-macro worden gedefinieerd. De implementatie:

```
SECURITY  TYPE=(,RACFTERM,),
          SECLVL=(FORCTRAN,)
```

Voor het gebruik van IMS-transacties dient de TIMS general resource class te worden geactiveerd en een sluitprofiel te worden gedefinieerd. De implementatie:

```
SETROPTS  GENERIC(TIMS)
SETROPTS  CLASSACT(TIMS)        IMS normal transactions
RDEFINE    TIMS  **  UACC(NONE)  OWNER(SECADM)  Sluitprofiel

SETROPTS  RACLIST(TIMS)  REFRESH
```

De naamgeving van IMS-transacties dient zoveel mogelijk gekozen te worden dat groepering door middel van een generiek profiel in de IMS-transactieclass (TIMS) mogelijk is. Voorbeeld:

```
RDEFINE    TIMS   TRN1*  OWNER(IMS)   UACC(NONE)

PERMIT     TRN1*  CLASS(TIMS)
           ID(XMIEP01)  ACCESS(READ)
```

Indien de naamgeving niet toereikend is om door middel van generieke profielen in de TIMS resource class onder te brengen, kan worden gekozen voor het groeperen van de transacties in de IMS-transactiegroepclass (GIMS).

```
SETROPTS CLASSACT(GIMS)        IMS normal transactions group class
RDEFINE    GIMS  CUSTHNDL   UACC(NONE)  OWNER(SECADM)
           ADDMEM(CUSTRN2  INVNTRN0 INKTRN2)
PERMIT     CUSTHNDL   CLASS(GIMS)  ACCESS(READ)  ID(XRIJL01)

SETROPTS CLASSACT(GIMS)        IMS normal transactions group class
```

AANVULLENDE MAATREGEL. Als de organisatie voor een transactie bijzondere/extra authenticatie wenst is dit mogelijk door de gebruiker elke keer voor het gebruik van deze specifieke transactie opnieuw zijn/haar password te laten geven. De implementatie van de IMS-parameter in de IMS-procedure (als EXEC parm of statement in DFSPBxxx):

```
RVFY=Y
```

Daarnaast moet de betreffende transactie binnen RACF worden gedefinieerd als REVERIFY. Een voorbeeld:

```
RDEFINE    TIMS        TOPSEC   UACC(NONE)  OWNER(SECADM)
                                              APPLDATA('REVERIFY')
PERMIT     TOPSEC      CLASS(TIMS)  ACCESS(READ)  ID(XRIJL01)
```

21.17 PSB access security

Program Specification Blocks (PSBs) worden niet afzonderlijk geautoriseerd maar door middel van de AGN-beveiliging. Zie hiervoor de paragraaf Application Group Names. Uitzondering hierop zijn de PSB-autorisaties die voortkomen uit CPI-C calls. Deze kunnen gelijk AGNs worden beveiligd in de RACF AIMS general resource class.

21.18 Application Group Names

AANVULLENDE MAATREGEL. Het gebruik van IMS resources door middel van de IMS dependent regions (z/OS started tasks of jobs) moet worden geautoriseerd. Autorisatie kan plaatsvinden door toepassing van de applicatiegroepering faciliteit, Application Group Names (AGNs). Het activeren van AGN-beveiliging moet in de SECURITY-macro worden aangegeven. De implementatie:

```
SECURITY   TYPE=(RACFAGN,,)
```

Verder moeten de IMS resources met behulp van de SMU in Application Group Names worden samengesteld en van een unieke AGN-naam worden voorzien. Een voorbeeld van een SMU-definitie voor de samenstelling van de applicatiegroep TEST002:

```
)( AGN      TEST002
  AGPSB     PSB11
  AGPSB     PSB33
  AGPSB     PAYROLL
  AGTRAN    PAYTRAN
```

Hierna dienen RACF-profielen gedefinieerd te worden om toegang tot de AGNs te autoriseren. Een voorbeeld van de RACF general resource class profielen:

```
RDEFINE    AIMS  TEST002   OWNER(SECADM)  UACC(NONE)
PERMIT     TEST002   CLASS(AIMS)  ID(IMSBATCH  XKRET01)
                     ACCESS(READ)
```

```
SETROPTS CLASSACT(AIMS)
```

21.19 BMP/MPP authenticatie

Bij het starten van een dependent region vindt authenticatie plaats op basis van een user-id/password combinatie. Het password mag niet in de JCL zijn opgenomen. Evenmin mag het password in een aparte dataset (IMSJOBS DD statement) worden opgeslagen.

21.20 CICS-DBCTL

Indien CICS gebruik wil maken van IMS/DB moet de CICS toegang krijgen tot IMS en daarnaast toegang krijgen tot PSBs welke de toegestane activiteiten voor die CICS bevatten. Zie voor normen “InterSystem Communications met LU 6.1” en “Program Specification Blocks” in het CICS-hoofdstuk van de PI-standaard voor RACF en de paragraaf “PSB access security” in dit document.

21.21 Advanced Program to Program Communication

Zie hoofdstuk ‘Virtual Telecommunication Access Method (VTAM)’ (Nog te ontwikkelen).

21.22 Open Transaction Manager Access (OTMA) Interface

De OTMA interface is een transaction based connectionless client-server protocol voor IMS. De OTMA interface gebruikt XCF als primair communicatiemiddel naar de IMS control region. De OTMA interface moet zich dus in hetzelfde Sysplex bevinden als waar het IMS-systeem zich in bevindt. Als toepassingen OTMA gebruiken moet de beveiliging met RACF worden geregeld. Als eerste moet de OTMA-beveiliging binnen IMS volledig worden geactiveerd. De OTMA-beveiliging wordt per default bij een cold start van IMS volledig geactiveerd. Als OTMA-beveiliging is uitgeschakeld moet dit met het SECURE commando worden geactiveerd. De implementatie:

```
/SECURE  OTMA  FULL
```

De applicaties/subsystemen die gebruik willen maken van IMS via het OTMA-protocol moeten hiervoor binnen RACF worden geautoriseerd. De IMSXCF.xcf-groupname.xcf-membername (membername = applicatie/subsysteem) profielen in de FACILITY class moeten hiervoor worden gedefinieerd en de clients-userids (userids waaronder de applicaties/subsystemen draaien) moeten hierop worden geautoriseerd. Twee voorbeelden:

```
Voorbeeld 1:
RDEFINE   FACILITY  IMSXCF.XCFIMSP.SUBP12  UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.SUBP12  CLASS(FACILITY)
          ACCESS(READ)  ID(SMQSP12)

Voorbeeld 2:
RDEFINE   FACILITY  IMSXCF.XCFIMSP.DCERPC2  UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.DCERPC2  CLASS(FACILITY)
          ACCESS(READ)  ID(SDCERPC2)
```

21.23 Open Transaction Manager Access Callable Interface (OTMA C/I)

Voor het gebruik van de OTMA door applicaties die niet tot het sysplex behoren, zoals andere z/OS-systemen of niet-z/OS-systemen en voor ongeautoriseerde programma's, wordt een high-level Callable interface geleverd. Uitsluitend geautoriseerde gebruikers mogen van de OTMA C/I-interface gebruik maken. Een voorbeeld:

```
RDEFINE   FACILITY   IMSXCF.OTMACI   OWNER(SECADM)   UACC(NONE)
PERMIT    IMSXCF.OTMACI   CLASS(FACILITY)
          ACCESS(READ)   ID(VERKOOP)
```

21.24 IMS Connect

IMS Connect is de TCP/IP-interface voor IMS OTMA. Met IMS Connect kunnen werkstations met behulp van een TCP/IP-netwerk aan IMS worden gekoppeld.

IMS Connect moet onder een user-id draaien die niet voor logon kan worden gebruikt (protected userid). Een voorbeeld:

```
ADDUSER   SIMSCONP   DFLTGRP(STCGROUP)   OWNER(SECADM)
          NOPASSWORD
```

IMS Connect dient als een started task of als job te worden geactiveerd. Het gebruik van de 'Trusted' en 'Privileged' attribuut is hierbij niet toegestaan. Een voorbeeld:

```
RDEFINE   STARTED   IMSCON.*   OWNER(SECADM)
          STDATA(USER(SIMSCONP)   GROUP(STCGROUP)
          PRIVILEGED(NO)   TRUSTED(NO))
```

IMS Connect is binnen een Sysplex een cliënt van OTMA en moet als member hiervoor worden geautoriseerd. De cliënt moet in de FACILITY general resource class van RACF worden gedefinieerd. Hiervoor moet een profiel worden gedefinieerd die de cliënt (IMS Connect) vertegenwoordigt en het user-id waaronder IMS Connect wordt uitgevoerd moet hiervoor worden geautoriseerd. Een voorbeeld:

```
RDEFINE   FACILITY   IMSXCF.XCFIMSP.IMSCON   UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.IMSCON   CLASS(FACILITY)
          ACCESS(READ)   ID(SIMSCONP)
```

Indien het gewenst is dat ongedefinieerde gebruikers (user-id's) toegang krijgen tot de OTMA interface kan IMS Connect een default user-id meegegeven waarmee RACF autorisaties kan uitvoeren. In het TCP/IP-statement van IMS Connect initialisatieparameters kan het te gebruiken default user-id worden opgegeven. Een voorbeeld:

```
TCPIP(RACFID=IMSCDFLT)
```

Het door IMS Connect gebruikte default user-id mag uitsluitend zeer beperkte autorisaties hebben zoals elke willekeurige gebruiker deze zou mogen hebben. Uitsluitend die autorisaties die specifiek voor het default user-id zijn gedefinieerd mogen worden gebruikt. De implementatie:

```
ADDUSER   IMSCDFLT   DFLTGRP(IMSCONGR)   OWNER(SECADM)
          RESTRICTED
```

Identificatie en authenticatie van elke client van IMS Connect moet worden uitgevoerd. Dit moet in het HWS initialisatie-statement in de HWSCFG configuratiefile worden aangegeven met de RACF-parameter. De RACF parameter moet op "Y" ingesteld worden om het client user-id en password (van de eindgebruiker) door RACF te laten verifiëren. De implementatie:

```
HWS(,RACF=Y,)
```

De IMS Connect beveiliging wordt geactiveerd met de hiervoor beschreven HWS-parameter. Als deze niet bij initialisatie van IMS Connect is geactiveerd moet deze met een IMS Connect commando worden geactiveerd. De implementatie:

```
SETRACF  ON
```

21.25 MQSeries IMS Bridge

De 'MQSeries IMS Bridge' is de MQSeries-interface voor IMS OTMA. MQSeries IMS Bridge is een client van OTMA binnen een sysplex en moet als member hiervoor worden geautoriseerd. De client moet in de FACILITY general resource class van RACF worden gedefinieerd. Hiervoor moet een profiel worden gedefinieerd dat de client (MQSeries IMS Bridge) vertegenwoordigt en waartoe het user-id waaronder MQSeries IMS Bridge wordt uitgevoerd toegang heeft. Een voorbeeld:

```
RDEFINE   FACILITY  IMSXCF.XCFIMSP.MQSP12  UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.MQSP12  CLASS(FACILITY)
          ACCESS(READ)  ID(SMQSP12)
```

Afhankelijk van de betrouwbaarheid van de aangesloten MQSeries (IMS-clients) kan worden bepaald met welke mate van beveiliging rond de MQSeries IMS interface kan worden volstaan. De voorbeelden:

```
Indien de aangesloten MQSeries niet wordt vertrouwd moet identificatie en
authenticatie bij elke aanvraag worden geverifieerd:
RDEFINE   FACILITY  IMSXCF.XCFIMSP.MQSP12  UACC(NONE)
          OWNER(SECADM)
Indien de aangesloten MQSeries wordt vertrouwd kan om performance redenen
worden volstaan om identificatie en authenticatie van de gebruiker eenmalig door
IMS te laten voeren:
RDEFINE   FACILITY  IMSXCF.XCFIMSP.MQSP12  UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.MQSP12  CLASS(FACILITY)
          ACCESS(READ)  ID(SMQSP12)

Indien de aangesloten MQSeries wordt vertrouwd kan om performance redenen
aan MQSeries worden overgelaten om de identificatie en authenticatie uit te voeren
en uitsluitend het user-id van de gebruiker aan IMS door te geven:
RDEFINE   FACILITY  IMSXCF.XCFIMSP.MQSP12  UACC(NONE)
          OWNER(SECADM)
PERMIT    IMSXCF.XCFIMSP.MQSP12  CLASS(FACILITY)
          ACCESS(UPDATE)  ID(SMQSP12)
```

Het gebruik van ALTER access, waarbij IMS geen identificatie en authenticatie door MQSeries verlangt, mag niet worden toegepast.

21.26 IMS Command violation

Wanneer een command violation door een gebruiker of een operator wordt veroorzaak moet deze worden gelogd. Het loggen moet worden geactiveerd met de SECURITY macro. De implementatie:

```
SECURITY  SECCNT=1
```

22 Database 2 (DB2)

Dit DB2-hoofdstuk is conceptversie en moet nog aan de praktijk worden getoetst. De lezer wordt van harte uitgenodigd om eventuele tekortkomingen of aanvullingen door te geven. De lezer kan hiervoor contact opnemen via e-mailadres: mailto:VanRij.Leen@kmpg.nl.

De DB2 standaard gaat er vanuit zoveel mogelijk resource autorisaties via RACF door groepsautorisatie toe te kennen. Autorisatie verkregen door privileges van andere gebruikers is zoveel mogelijk vermeden. Een voorbeeld waarbij dit niet kan worden geïmplementeerd is bij applicatieplannen en packages waarbij de eigenaar en niet de uitvoerder, die de betreffende SQL-statements moet uitvoeren, de autorisaties op alle objecten moet hebben.

Met uitzonderlijke dank aan de werkgroepleden voor hun bijdrage in de ontwikkeling van dit hoofdstuk:

- Frank Engel, KPMG Information Risk Management
- Lorenz Kenter, Kenter AutoVision
- Theo Krens, KPMG Information Risk Management
- Peter Mienes, KPMG Information Risk Management
- John Rudolf, LogicaCMG
- Leen van Rij, KPMG Information Risk Management

22.1 Introductie

Het DataBase 2 (DB2) systeem is een relationeel database management systeem (RDBMS). DB2 wordt geleverd voor verschillende typen platformen waaronder z/OS. Binnen z/OS wordt DB2 meestal in combinatie met IMS, CICS en batchverwerking toegepast. Verder biedt DB2 faciliteiten zoals een gebruikersinterfaces, beheerprogrammatuur en debugging hulpmiddelen. Deze faciliteiten worden verder niet in deze standaard beschreven aangezien het gebruik impliciet gekoppeld is aan de autorisaties op de verschillende objecten.

DB2 met z/OS kan worden gekarakteriseerd als het RDBMS dat geschikt is voor grote tot zeer grote omgevingen met hoge transactie volumes. Zo kan DB2 als een high performance, high availability database server worden toegepast voor grootschalige ERP-implementaties. Voor het z/OS platform is DB2 een subsysteem dat gebruik maakt van het z/OS-subsysteeminterface. Hierdoor is DB2 een geïntegreerde database manager en kan onder andere op een eenvoudige manier van de z/OS faciliteiten gebruik maken.

In deze DB2-standaard is er vanuit gegaan dat specifieke toegang tot gebruikersgegevens in de DB2-database(s) door de gebruikers zelf wordt bepaald, daarom zijn hiervoor geen normen in deze standaard opgenomen.

Opmerking: Voor deze standaard is uitgegaan van DB2 versie 7 met z/OS versie 1.2.

22.1.1 Leeswijzer DB2 Hoofdstuk

Dit DB2 hoofdstuk is in 5 delen onderverdeeld. Als eerste worden de DB2-onderwerpen besproken waarover de organisatie als eerste een besluit moet nemen voordat aan implementatie van DB2-beveiliging met behulp van RACF kan worden gedacht. Daarna worden de RACF-definities beschreven die nodig zijn voor de implementatie van de DB2-subsysteemsoftware. Het

derde gedeelte beschrijft de RACF-definities noodzakelijk voor het beheer van de DB2-resources zoals table spaces, tables, plans, etc. Het vierde gedeelte is gericht op de RACF-definities nodig om de toegang tot DB2-gegevens voor eindgebruikers te regelen. Het laatste gedeelte beschrijft de RACF-definities nodig voor de DB2-communicatie faciliteiten.

22.1.2 Leeswijzer voorbeelden RACF-profielen

In dit hoofdstuk zijn verschillende RACF-profielen als voorbeeld weergegeven. RACF-profielen voor DB2 hebben veelal meerdere qualifiers van verschillende samenstelling en betekenis voor de beveiliging. De betekenis en samenstelling van de qualifiers zijn in de handboeken van de leverancier terug te vinden. Echter voor het kunnen interpreteren van de voorbeelden van de RACF-profielen zijn voor de qualifiers herkenbare variabelen gebruikt, zoals:

PERMIT *DB2P.DBP23.***DROP** CLASS(MDSNDB)
ID(*DB2DBA1*) ACCESS(**READ**)

Dit RACF-commando geeft de gebruikersgroep *DB2DBA1* (DB2 Database Administratorsgroep 1) de permissie om productie-database 23 (*DBP23*) op het productie DB2-subsysteem (*DB2P*) te droppen (DROP). Hieronder een beperkte opsomming van de variabelennamen die wij voor de voorbeelden hebben gekozen:

Resource	**Voorbeeld**	**Opmerking bij voorbeeld**
DB2-subsysteemnaam	*DB2P*	Qualifier voor het aangeven van een DB2-systeem in een productie-, acceptatie- of testomgeving. Dit voorbeeld geeft een productieomgeving aan.
Datasharing group	*DB2PSYS1*	Qualifier voor het aangeven van een DB2-datasharing group (DB2 in een Sysplex) in een productie-, acceptatie- of testomgeving. Dit voorbeeld geeft een productieomgeving aan.
Storage group	*STOGRP23*	Dit voorbeeld geeft groep 23 aan.
Database	*DBP23*	Dit voorbeeld geeft database 23 aan.
Table Space	*TABS34*	Dit voorbeeld geeft tabel space 34 aan.
Table Owner	*DB2TAB*	Dit voorbeeld geeft gebruiker DB2TAB als eigenaar aan.
Table	*TAB29*	Dit voorbeeld geeft tabel 29 aan.
Bufferpool	*BPL87*	Dit voorbeeld geeft bufferpool 87 aan.
Plan	*PLAN002*	Dit voorbeeld geeft plan 002 aan.
Package	*PACK001*	Dit voorbeeld geeft package 001 aan.

Collection	*COLL77*	Dit voorbeeld geeft Collectie 77 aan.
Schema	*SCHEMA12*	Dit voorbeeld geeft schema 12 aan.

Tabel 1. Variabele gebruikt in de voorbeelden

22.1.3 Uitgangspunt DB2 beveiliging

Het uitgangspunt voor de beveiliging van resources onder RACF is dat primair de gegevens door een RACF-profiel worden beschermd en de gebruiker met dit profiel toegang tot de gegevens kan krijgen. Het beveiligen van resources zoals een programma is van secondair belang en geldt dan ook vaak als een extra maatregel. Het primaire uitgangspunt van DB2-beveiliging is dat het gebruik van het aangeroepen programma wordt beveiligd en het programma de toegangsrechten tot de gegevens krijgt. Deze DB2-RACF-standaard implementeert de RACF-filosofie waarbij primair de gegevens (tabellen en views) worden beschermd en de gebruiker expliciete toegang op deze gegevens krijgt en niet het programma (plan of package). Het programma (plan of package) kan daarbij als extra maatregel worden beveiligd.

Opmerking: In de gehele RACF DB2-standaard is er voor gekozen om alle rechten die aan een beheerder of gebruiker worden gegeven met het meest restrictieve profiel te definiëren. In de praktijk blijkt dat bepaalde profielen een te grote bevoegdheid geven. Dit is echter met de huidige functionaliteiten van RACF-DB2 niet te vermijden. Wanneer hier sprake van is wordt dit in de beschrijving aangegeven.

22.1.4 Motivatie DB2 onder RACF

Voor het onder RACF brengen van DB2 is relatief veel inspanning noodzakelijk. Deze inspanning moet opwegen tegen de voordelen die hiermee kunnen worden behaald. De ontwikkelaars van dit hoofdstuk hebben de volgende voordelen onderkend:

- De mogelijkheid voor een afdoende functiescheiding tussen de verschillende rollen die binnen een database systeem kunnen worden onderkend kunnen met RACF verder worden geïmplementeerd;
- Beveiligingsdefinities kunnen vooraf in RACF worden gedefinieerd voordat resources zoals tabellen, views, etc. in DB2 worden gecreëerd en met gegevens worden geladen. Met het vooraf in RACF aanbrengen van de beveiligingsdefinities zal bedrijfsgevoelige informatie altijd beschermd zijn;
- Na verwijdering van DB2 resources zoals tabellen, views, tabel spaces, etc. blijven de beveiligingsdefinities in RACF bestaan en bij eventueel herdefiniëring van de resources zullen deze nog steeds beschermd zijn. Dit is in tegenstelling tot DB2 waarbij de beveiligingsdefinities opnieuw moeten worden gedefinieerd;
- Gelijk andere faciliteiten zoals CICS, Storage Management Subsysteem, Integrated Encryption Services Facility, etc. waarvoor uitgebreide kennis van en ervaring met RACF-beveiliging nodig is, wordt met DB2 onder RACF het beveiligingsbeheer onder één eenduidige rol gebracht. De rol van Security Administrator zelf kan met RACF centraal en decentraal worden ingevuld;
- De huidige rol van Database Administrator wordt qua beveiligingstaken gereduceerd. Alhoewel dit binnen het normale aandachtsgebied ligt van de Security Administrator worden hiermee zijn/haar taken verzwaard. Communicatie tussen Security Administrator en Database Administrator is hier een essentieel punt;

- De inspanning om over meerdere omgevingen heen (test, acceptatie, productie) de beveiligingsregels consistent te houden wordt verminderd.

22.1.5 DB2 onder RACF

DB2 is origineel ontworpen met een eigen, interne beveiliging. In de laatste versies is het mogelijk DB2-autorisaties in RACF onder te brengen (vanaf RACF release 2.4 met DB2 release 5). Voor een transparante overgang van interne (DB2) naar externe beveiliging (RACF) is de volgende oplossing geïmplementeerd:

1. is RACF actief en is er een RACF-profiel aanwezig die de betreffende resource beschrijft dan wordt de RACF-autorisatie toegepast. Hiermee is het mogelijk om met RACF-definities DB2-autorisaties te verlenen. RACF regelt met deze opzet de autorisaties en door DB2 wordt deze autorisatie gevolgd;
2. als RACF actief is en er is voor de betreffende autorisatiecontrole geen profiel in RACF aanwezig dan wordt DB2-beveiliging gebruikt en gebruikt DB2 zijn eigen autorisatiedefinities;
3. als RACF inactief is wordt de interne DB2-beveiliging gebruikt.

Een combinatie van interne en externe beveiliging mag uitsluitend gedurende een korte overgangsperiode bestaan. Bij het volledig gebruik van RACF dienen er geen autorisaties in DB2 aanwezig te zijn.

In dit DB2-hoofdstuk gaan we uit van externe beveiliging waar dit met RACF mogelijk is. Hierbij wordt niet alleen aangegeven hoe de externe beveiliging moet worden ingericht, maar ook welke DB2-instellingen noodzakelijk zijn om de externe beveiliging te laten functioneren.

22.1.6 Rollen

Voor het beheer van de DB2-omgeving kunnen we de volgende rollen onderscheiden:

- DB2 Systems Programming: verantwoordelijk voor het installeren en implementeren van DB2-subsysteemsoftware en -parameters. Verder is DB2 Systems Programming verantwoordelijk voor het onderkennen van zwakheden en het aandragen van preventieve oplossingen en voor het verhelpen van problemen in de DB2-subsysteemsoftware;
- Storage Administration: verantwoordelijk voor het beheer van de fysieke opslagmedia die gebruikt worden voor het opslaan van DB2 beheerde gegevens zoals databases;
- DB2 Systems Operation: verantwoordelijk voor het starten en stoppen van DB2 en zorgt ervoor dat DB2 juist blijft functioneren. Andere taken zijn het veiligstellen (backup) van de databases en het eventueel restoren van databases in geval van uitzonderlijke problemen. Dit komt overeen met de DB2 SYSOPR Administrative Authority;
- DB2 Systems Administration: verantwoordelijk voor het gehele beheer van DB2 zoals specifieke systeemtabellen, backups, logging etc. Dit komt overeen met de DB2 SYSADM Administrative Authority;
- Security Administration: verantwoordelijk voor het beheer van de autorisatiematrixes en de daaruit voortvloeiende beveiligingsinstellingen zoals de DB2- en RACF-parameters en -definities. Binnen DB2 komt dit overeen met de DB2 SYSCTRL Administrative Authority;
- Database Performance Analist: verantwoordelijk voor het optimaal inregelen en het gebruik van de systeem resources voor het behalen van de performance indicatoren afgesproken met de eindgebruikers;

- Database Administration (DBA): verantwoordelijk voor het ontwerp, inrichting en beheer van de applicatiedatabases. Dit komt overeen met de DB2 DBADM Administrative Authority;
- Application Programming: verantwoordelijk voor het ontwikkelen van DB2-applicatieprogrammatuur;
- DB2 Production Binder: verantwoordelijk voor het (RE)BINDen van DB2-applicatieprogrammatuur. Voor het uitvoeren van deze rol is de DB2 BIND privilege nodig;
- DB2 Package Administration: verantwoordelijk voor het beheer van de packages. Dit komt overeen met de DB2 PACKADM Administrative Authority;
- DB2 User Analist: verantwoordelijk voor de analyse van de databasegegevens voor applicatieontwikkeling. Hiervoor is vooral autorisatie op de DB2-systeemtabellen nodig;
- Information Center Consultant: verantwoordelijk voor het definiëren van queries naar aanleiding van eindgebruikersvragen voor management informatie of voor project specifieke doeleinden;
- Query Users: voeren SQL-queries uit om gegevens te lezen toe te voegen of te wijzigen;
- Program End User (Eindgebruikers): personen of systeemtaken die via DB2 of DB2-faciliteiten DB2-gegevens benaderen en daarvoor een programma met SQL-statements uitvoeren.

22.1.7 Primaire Autorisatie Matrix

Voor de inrichting van de autorisaties is uitgegaan van de verschillende rollen die in een DB2-omgeving kunnen worden onderkend. Voor de eenduidigheid zijn de hieronder genoemde rollen, met een kleine uitzondering, overeenkomstig de rollen in de DB2-handboeken van de leverancier. In de tabel zijn de volgende aanduidingen gegeven:

- **B**, geeft aan dat de betreffende rol uitsluitend **B**eheertaken op het aangegeven resource uitvoert;
- **S**, geeft aan dat de betreffende rol uitsluitend **S**ecurity-taken op de betreffende resources uitvoert;
- **G**, geeft aan dat de betreffende rol uitsluitend **G**ebruik maakt van de betreffende resources;
- **O,** geeft aan dat de betreffende rol deze resources in de **O**ntwikkelomgeving mag gebruiken;
- *, geeft aan dat de databasegegevens gezien de betreffende rol benaderbaar (moeten) zijn.

De hier beschreven rollen geven een indicatie voor de werkelijke implementatie. Afwijkingen hierop kunnen altijd worden geïmplementeerd mits de vereiste functiescheiding gewaarborgd blijft. In de standaard wordt bij de normen de hieronder beschreven rol aangegeven en bij het betreffende RACF-voorbeeldprofiel de bijbehorende mogelijke autorisaties.

Rol \ Resource	DB2 subsystem software	Storage / datasets	DB2 Storage groups	Commando's en Utilities	System Tables	Databases	Table Space	Tables	Indexen	Views	Plans	Packages	Collections	Schema's	Stored Procedures	UDF / UDT	Triggers	Database gegevens
DB2 Systems Programming	**B**			**B**														
Storage Administration		**B**	**B**															
DB2 Systems Operation				**G**														**B**
DB2 Systems Administration					**B**													
Database Performance Analist									**B**									
Security Administration	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	**S**	
DB2 Database Administration						**B**	**B**	**B**		**B**								**G***
DB2 Application Programming				**O**				**O**	**O**	**O**	**O**	**O**						**O**
DB2 Production Binder											**B**	**B**						
DB2 Package Administration													**B**	**B**	**B**	**B**	**B**	**G***
DB2 User Analist					**G**													
Information Center Consultant								**G**	**G**	**G**	**G**	**G**			**G**			**G**
Query Users								**G**	**G**	**G**	**G**	**G**			**G**			**G**
Program End User								**G**	**G**	**G**	**G**	**G**			**G**			**G**

Table 2. DB2-Beheer- en Gebruikersrollen in een productieomgeving

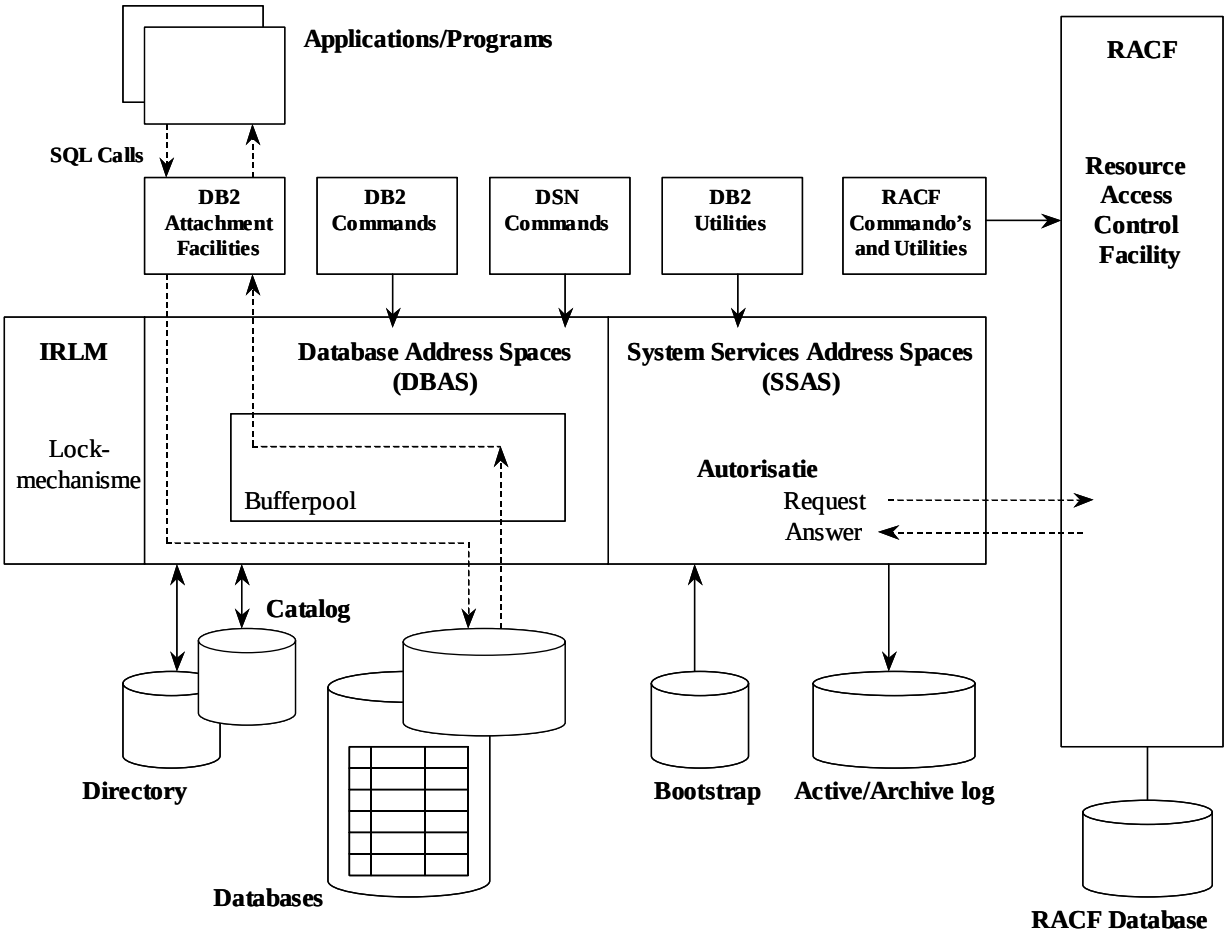
22.2 Status

Dit DB2-hoofdstuk is in ontwikkeling en om de kwaliteit te verhogen wordt de lezer van harte uitgenodigd commentaar hierop te geven.

22.3 DB2 Concept

De basis van DB2 bestaat uit 3 address spaces die de volgende faciliteiten leveren:

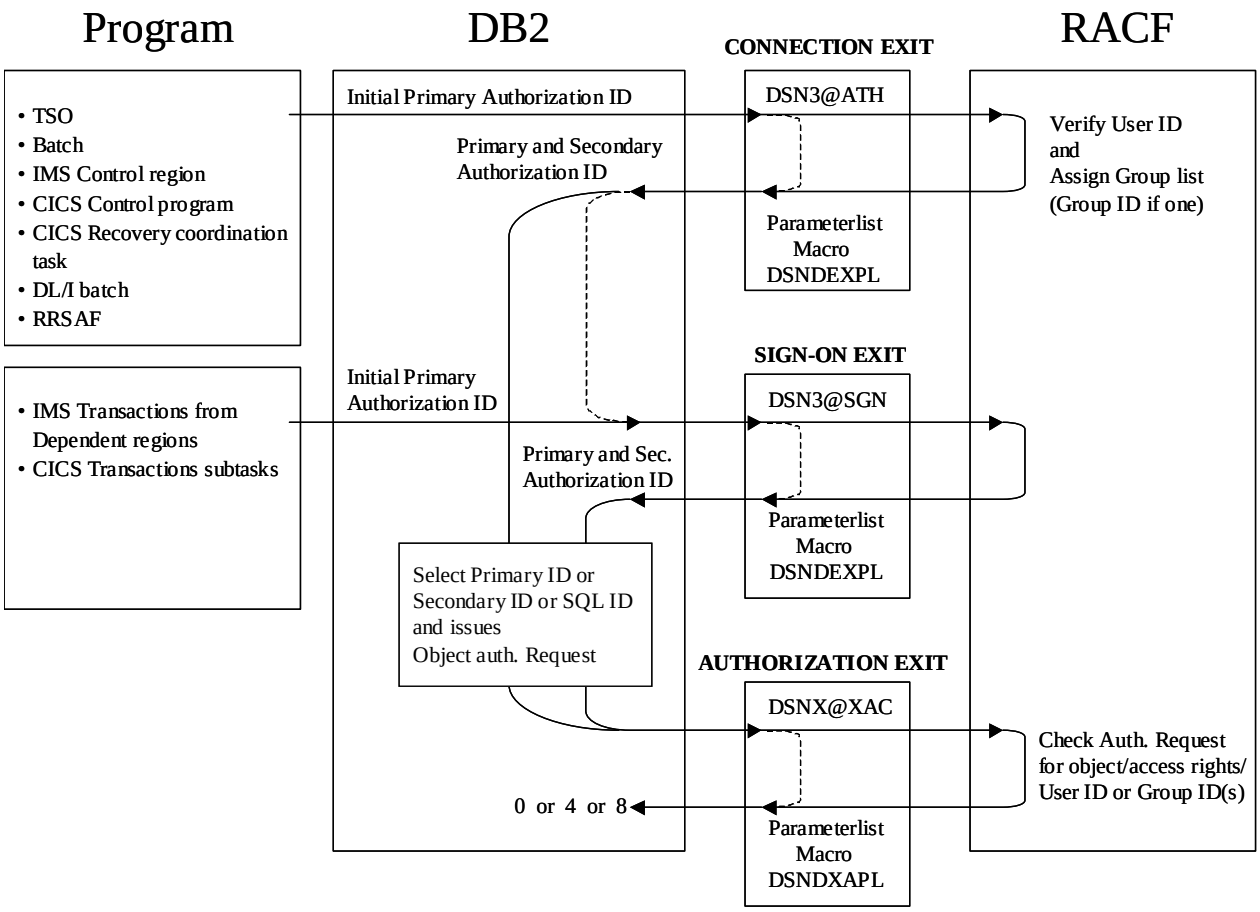
- De DataBase address space (DBAS) leest en schrijft (I/O) van en naar de fysieke databases en beheert de bufferpools;
- De System Services address space (SSAS) voor algemene beheertaken zoals:
  - het verzorgen van autorisaties;
  - opstarten en afsluiten van verbindingen met andere (sub)systemen;
  - wegschrijven van logrecords;
  - ondersteuning bij checkpointing en restarting.
- De Internal Resource Lock Manager (IRLM) voor locking van resources zoals tabel rows.



Figuur 1. DB2-concept

22.4 DB2 Access Control Flow Concept

De Identicatie, Authenticatie en Autorisatie calls van DB2 naar RACF gaan via verschillende DB2-exitroutines zoals hieronder gevisualiseerd. Voor gedetailleerde beschrijving van de mogelijkheden van de exits moet de geldende officiële leveranciersinformatie worden geraadpleegd.



Figuur 2. DB2-Autorisatieflow

22.5 Installatie

22.5.1 DB2 load libraries

Het DB2-subsystem heeft zonder daarvoor expliciet te zijn geautoriseerd (Programs Propertie Table RACF passthrough optie is gebruikt) toegang tot de load libraries. Individuele gebruikers die uit hoofde van hun taken en bevoegdheden toegang tot DB2-libraries moeten hebben dienen via Change Management / Security Management te worden geautoriseerd. Uitsluitend de noodzakelijke toegangsrechten dienen te worden gegeven. De implementatie voor de DB2-programma libraries:

```
ADDSD   'PROD1.DSN.SDSNLOAD'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))
ADDSD   'PROD1.DSN.SDSNLOD2'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))
ADDSD   'PROD1.DSN.SDSNLINK'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))
ADDSD   'PROD1.DSN.SDSNEXIT'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))  NOTIFY(SECADM)
ADDSD   'PROD1.DSN.SDSNDBRM'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))
```

De implementatie voor de Internal Resource Lock Manager (IRLM) programma libraries:

```
ADDSD   'PROD1.DSN.SDXRRESL'  OWNER(SECADM)
        UACC(NONE)  AUDIT(ALL(UPDATE))
```

Voor de gehele set DB2 gerelateerde libraries kan ook een generic-profiel worden toegepast. Een voorbeeld:

```
ADDSD     'PROD1.DSN.**'   OWNER(SECADM)
          UACC(NONE)  AUDIT(ALL(UPDATE))
```

22.5.2 DB2 data sets en databases

Naast de DB2-load libraries moeten voor het beheer van de gegevensbestanden DB2-autorisaties worden aangemaakt. Over het algemeen zal het beheer van deze databestanden door DB2 System Operations worden uitgevoerd. Uitsluitend gebruikers die uit hoofde van deze taken en bevoegdheden toegang tot DB2 data-bestanden moeten hebben mogen hiervoor zijn geautoriseerd. Uitsluitend de noodzakelijke toegangsrechten dienen te worden gegeven. Hieronder enkele voorbeelden voor dataset autorisatie:

De actieve logs worden door DB2 gebruikt voor het vastleggen van relevante gebeurtenissen die onder andere nodig zijn voor het herstellen van de database na een hersteloperatie (recovery). Een voorbeeld voor de actieve log datasets:

```
ADDSD     PROD1.DSN.LOGCOPY*'  OWNER(SECADM)  UACC(NONE)
```

De archive logs zijn de actieve logs die voor een langere periode moeten worden bewaard. Een voorbeeld voor de archive log datasets:

```
ADDSD     PROD1.DSN.ARCHLOG*'  OWNER(SECADM)  UACC(NONE)
```

De bootstrap dataset bevat cruciale informatie over de status van het DB2-subsysteem en wordt gebruikt om DB2 op de juiste manier op te opstarten en te initialiseren. Een voorbeeld voor de bootstrap dataset:

```
ADDSD     PROD1.DSN.BSDS*'  OWNER(SECADM)  UACC(NONE)
```

De DB2-directory is een database met DB2 interne besturings- en statusinformatie. Alhoewel deze informatie in een set tabellen is opgeslagen kan deze niet met SQL worden benaderd. Een voorbeeld voor de directory dataset:

```
ADDSD     PROD1.DSN.DIRECTORY.*'  OWNER(SECADM)  UACC(NONE)
```

De DB2-catalog is database met een set tabellen met informatie (beschrijving) over alle DB2-objecten. Aangezien de DB2-catalog standaard DB2-tabellen heeft kunnen deze worden gelezen en bewerkt met normale SQL-statements. Een voorbeeld voor de catalog dataset:

```
ADDSD     PROD1.DSN.CATALOG.*'  OWNER(SECADM)  UACC(NONE)
```

Voor table spaces en index spaces zijn dataset nodig. Aangezien er een grote gevarieerdheid aan deze datasets kunnen bestaan en afhankelijk is van de organisatie is hieronder één profiel beschreven. Een voorbeeld voor een table spaces en index spaces dataset:

```
ADDSD     PROD1.DSN.DSNDBC.*'  OWNER(SECADM)  UACC(NONE)
```

22.5.3 Started Task

Er moet voor één DB2-omgeving verschillende taken (started tasks) worden opgestart. Elke taak dient onder een userid te draaien die niet voor logon kan worden gebruikt (protected userid). Een voorbeeld van de groep voor alle DB2 started tasks en de verschillende userids:

```
ADDGROUP      DB2STC  SUPGROUP(SYS1)  DATA('DB2 hoofdgroep')
                      OWNER(SECADM)

ADDUSER  SDB2PROD  DFLTGRP(DB2STC)  OWNER(SECADM)
                   NOPASSWORD
```

Elk DB2-subsysteem dient als een started task te worden geactiveerd. Het gebruik van het 'Trusted' en 'Privileged' attribuut is voor de started tasks niet toegestaan.

De DB2-Subsysteem Manager SSAS (System Services address space) is een started task die als controler functioneert voor het gehele DB2-subsysteem en de intersystem-communicatie tussen verschillende DB2-gerelateerde started tasks coördineert. Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNMSTR.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

De Database Manager address space (DBAS) is een started task die alle DB2-databases benadert. Deze started task voert alle SQL-statements uit en communiceert de gevraagde gegevens aan de gebruikers. Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNDBM.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

De Distributed Database Manager is een started task voor communicatie met andere DB2-systemen op andere platformen. Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNDIST.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

De Stored Procedures address space is een started task die Stored Procedures uitvoert op aanvraag van een extern systeem. Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNSPAS.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

De DB2 WLM address space (optioneel) is een WLM-started task die stored procedures uitvoert onder controle van de Work Load Manager. Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNWLM.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

Voor locking van objecten gebruikt DB2 de Internal Resource Lock Manager (IRLM). Een voorbeeld van de started task:

```
RDEFINE STARTED (DSNIRLM.**) OWNER(SECADM)
        STDATA(USER(SDB2PROD) GROUP(DB2STC))
        PRIVILEGED(NO) TRUSTED(NO) TRACE(YES))
```

Als alternatief voor de specifieke definiëring van de started tasks kan gebruik worden gemaakt van de generieke definitie STDATA(USER(=MEMBER) GROUP(*STCGROUP*) TRACE(YES)) in de general resource class STARTED. Hierdoor behoeven uiteindelijk minder RACF-profielen te worden gedefinieerd. Wel dient de naam van het MEMBER, de started task naam, als gebruiker te worden gedefinieerd en, in dit voorbeeld, aan de groep STCGROUP te worden geconnect om te kunnen worden gebruikt.

22.5.4 User exits

Voor correcte DB2-autorisaties moet DB2 gebruik maken van de door de leverancier geleverde voorbeeld (sample) exits, dit zijn de:

- connection exit (sample assembler code DSN3@ATH), die voor de connectie met DB2 zorgt van een:

- TSO foreground en background sessies, inclusief online utilities;
- Aanvragen via de Call Attachment Facility (CAF);
- Batch programma's (JES2);
- Started tasks
- IMS Control region;
- CICS Control program;
- CICS Recovery coordination task;
- DL/I batch;
- RRS aanvragen via de RRS Attachment Facility (RRSAF).

- sign-on exit (sample assembler code DSN3@SGN), wordt aangeroepen bij
  - elke CICS transacties subtasking;
  - elke IMS transacties (MPP, BMP, and Fast Path);
  - IMS Control region;
  - CICS Recovery coordination task;
  - DL/I batch;
  - RRS aanvragen via de RRS Attachment Facility (RRSAF).
- autorisatie exit/module (sample assembler code IRR@XACS). In de volgende paragraaf wordt deze exit/module besproken.

Opmerking: Van de standaard DB2-exits (default exits) mag **geen** gebruik worden gemaakt en moeten worden vervangen door de hiervoor genoemde standaard voorbeeld-exits (sample exits). De standaard DB2-exits gebruiken niet de RACF-interface voor autorisatiecontrole en moeten daarom worden vervangen door de voorbeeld-exits waarbij de RACF-interface wordt aangeroepen. Door de implementatie van de voorbeeld-exits kan de DB2-beveiliging worden geïmplementeerd volgens deze PI-standaard. De standaard voorbeeld-exits kunnen worden gevonden in de SAMPLIB. Verder mogen geen exits worden gebruikt die het standaard DB2-RACF-beveiligingsmechanismen aanpassen.

22.5.5 DB2 autorisatiemodule

Door installatie van het autorisatiemodule DSNX@XAC (IRR@XACS) wordt ervoor gezorgd dat voor de beveiliging van de DB2-objecten RACF wordt aangeroepen voor objectautorisatie. Bij installatie van de autorisatiemodule moet worden gespecificeerd dat DB2 gebruik maakt van 'multi-system scope' ook wel 'classification model II' genoemd. Bij deze instelling maken alle DB2-subsystemen, bijvoorbeeld de productie, acceptatie en test DB2 onder één besturingssysteem, gebruik van dezelfde standaard RACF classes. De verschillende standaard RACF class-namen voor DB2 beginnen met MDSN (member). De classes beginnend met GDSN (group) worden, met de standaard instelling van multi-system scope, zoals hier toegepast, niet gebruikt. De volgende assembler macro settings dienen in de autorisatiemodule te worden gedefinieerd:

```
&CLASSOPT   SETC   '2'      resource namen (profielen) worden geprefixed met
                            de DB2 subsysteemnaam of met de naam van
                            datasharing-groep, nodig voor multi system scope
```

```
&CLASSNMT   SETC   'DSN'    noodzakelijk voor optie &CLASSOPT=2
                            (DSN is de default waarde)

&ERROROPT   SETC   '2'      bij een niet voorziene fout in de autorisatiemodule
                            wordt DB2 direct gestopt
```

22.5.6 Initialisatie parameters

DB2 biedt een ISPF-panelinterface waarbij onder andere beveiligingsgerelateerde parameters kunnen worden gedefinieerd. De basisbeveiligingsparameters kunnen op het initialisatie-panel DSNTIPP worden gespecificeerd. De implementatie van de parameters:

Door DB2 moet autorisatiecontrole worden uitgevoerd. De implementatie:

```
USE PROTECTION   ===>  YES
```

Voor de bescherming van de archive logs mag door DB2 geen discrete RACF-profiel worden aangemaakt.

```
ARCHIVE LOG RACF   ===>  NO
```

DB2 kent een parameter waarbij een userid kan worden gedefinieerd welke DB2 kan gebruiken als RACF niet beschikbaar is voor batchverwerking. Dit is een situatie waarbij DB2 niet verder mag functioneren en een userid is daarom niet noodzakelijk, echter DB2 vereist dat hiervoor een userid is ingevuld. Uitsluitend een userid dat geen rechten heeft mag hiervoor worden gedefinieerd en aan DB2 bekend worden gemaakt. Een voorbeeld:

```
ADDUSER  DB2UNKN  DFLTGRP(DB2GRP)  OWNER(SECADM)
                  NOPASSWORD  RESTRICTED  REVOKE

UNKNOWN AUTHID   ===>  DB2UNKN
```

Voor het gebruik van de Resource Limit Facility (governor) kan een userid worden gedefinieerd. Een voorbeeld:

```
ADDUSER  DB2RLFG  DFLTGRP(DB2GRP)  OWNER(SECADM)
                  NOPASSWORD  RESTRICTED

RESOURCE AUTHID   ===>  DB2RLFG
```

22.5.7 Initialisatie RACF classes

Voor het activeren van de DB2-RACF-interface moet voordat DB2 wordt gestart de betreffende RACF-classes worden geactiveerd. Een voorbeeld:

```
SETROPTS GENERIC(DSNADM)
SETROPTS GENCMD(DSNADM)
SETROPTS CLASSACT(DSNADM)
```

Nadat de RACF-profielen zijn gedefinieerd moeten deze voor gebruik door RACF worden geladen. Een voorbeeld:

```
SETROPTS RACLIST(DSNADM)  REFRESH
```

22.6 Resource namen

DB2 past verschillende RACF classes toe voor het autoriseren van de verschillende objecten. Bij een ‘multi-system scope’ DB2-omgeving, waar dit document vanuit gaat, worden de standaard RACF-classes toegepast. Elke RACF-class wordt dan gebruikt door de verschillende DB2-subsystemen (members) binnen één besturingssysteem en door een groep van samenwerkende

DB2-substemen (data sharing group) binnen één Sysplex. Het verschil van autorisaties tussen de verschillende DB2-subsystemen wordt gemaakt door de RACF-profielen te prefixen met de DB2-subsysteemnaam of in een Sysplex-omgeving met de DB2-data sharing groupname. De manier van autorisaties definieren voor de verschillende DB2-subsystemen (members) of data sharing groups is niet anders. Een voorbeeld RACF-profiel voor een DB2-member:

```
RDEFINE   DSNADM   DB2P.**   UACC(NONE)   OWNER(SECADM)
```

Een voorbeeld RACF-profiel voor een DB2-data sharing group:

```
RDEFINE   DSNADM   DB2PSYSP.**   UACC(NONE)   OWNER(SECADM)
```

22.6.1 Subsystem name

Voor het prefixen van de RACF-profielen wordt, zoals aangegeven, de DB2-subsysteem naam gebruikt zoals gedefinieerd in het SYS1.PARMLIB-member IEFSSNxx. En aan DB2 bekend gemaakt met behulp van het installatiepanel DSNTIPK. De implementatie van de parameters in het IEFSSNxx-member:

```
SUBSYS    SUBNAME(DB2P)
              INITRTN(DSN3INI)
          INITPARM('DSN3EPX,prefix<,scope<,group-attach>>')
```

De implementatie wordt uitgevoerd met behulp van de GROUP ATTACH-optie op het DB2-installatiepanel DSNTIPK:

```
GROUP ATTACH   ===> DB2P
```

22.6.2 Datasharing naam

Voor het prefixen van de RACF-profielen in een Sysplex-omgeving wordt, zoals aangegeven, de DB2-data sharing group-naam gebruikt. Deze naam wordt aan DB2 bekend gemaakt met behulp van het installatiepanel DSNTIPK. De implementatie wordt uitgevoerd met behulp van de GROUP NAME-optie op het DB2-installatiepanel DSNTIPK:

```
GROUP NAME      ===> DB2PSYS1
```

Opmerking: Een DB2 data sharing group bestaat uit één of meerdere members (DB2-subsystemen over één of meerdere besturingssystemen in een Sysplex). Een DB2-substeem moet dan als member aan de data sharing group worden gekoppeld waarbij het meest zinvolle is om de subsysteemnaam te gebruiken. De implementatie wordt uitgevoerd met behulp van de MEMBER NAME-optie op het DB2-installatiepanel DSNTIPK:

```
MEMBER NAME    ===> DB2P
```

22.6.3 Qualifiers

Een object zoals een tabelnaam bestaat uit twee qualifiers. De eerste qualifier is over het algemeen de eigenaar (OWNER) van de tabel en de tweede qualifier is de naam die de eigenaar aan de tabel heeft gegeven. Een overeenkomstig voorbeeld is een z/OS datasetnaam met een high level qualifier gelijk aan het userid van de gebruiker of het groepid van een RACF-groep. Een voorbeeld van een tabelnaam:

```
DB2TAB.TABNAW
```

22.7 Installatie Authorities

Voor installatie, implementatie en migratie van een DB2-subsysteem is het noodzakelijk dat gebruikers speciale authorities hebben. Met de hier bedoelde authorities (installation authorities)

is het niet nodig voor objecten geautoriseerd te zijn om beheerwerkzaamheden op het DB2-subsysteem en -objecten te kunnen uitvoeren. Deze installation authorities zoals hieronder beschreven kunnen uitsluitend worden gebruikt als het DB2-subsysteem met het -START DB2 ACCESS(MAINT) is opgestart. Het opstarten van DB2 met dit commando mag uitsluitend via Change Management worden uitgevoerd.

22.7.1 Installatie System-Administrator (Installation SYSADM)

De ‘installation system administrator’ authority wordt gebruikt voor userids (maximaal 2) noodzakelijk voor installatie en implementatie van een DB2-subsysteem. Een ‘installation system administrator’ wordt uitsluitend binnen DB2 gecontroleerd, er wordt geen aanroep naar RACF uitgevoerd. Met deze userids kunnen DB2-bevoegdheden worden toegekend, tabellen worden aangemaakt en verwijderd, data zoals catalog informatie worden gemigreerd, etc. Binnen DB2 moet worden aangegeven dat een userid de ‘installation system administrator’ authorities krijgt. Om fouten in definities te voorkomen moeten deze ‘installation system administrator’ userids per DB2-subsysteem verschillend zijn. Na implementatie van een DB2-subsysteem zijn deze vergaande authorities normaliter niet meer nodig en moeten de userids binnen RACF worden revoked. De implementatie:

```
ADDGROUP  DB2PADM   DATA('DB2 Productie SYSADM groep')
                    OWNER(SECADM)

ADDGROUP  DB2TADM   DATA('DB2 Test SYSADM groep')
                    OWNER(SECADM)

ADDUSER   DB2PADM1  DFLTGRP(DB2PADM)  OWNER(SECADM)
                    DATA('DB2 Installation System Administrator P-
                    omgeving')
ADDUSER   DB2TADM1  DFLTGRP(DB2TADM)  OWNER(SECADM)
                    DATA('DB2 Installation System Administrator T-
                    omgeving')
```

Met het ‘installation system administrator’ userid moet TSO kunnen worden gebruikt. De implementatie:

```
ALTUSER   DB2PADM1  TSO(COMMAND(ISPF)  PROC(TSOPROCA)
                        SIZE(4000) MAXSIZE(8000))
```

Het gebruik van het ‘installation system administrator’ userid, alsmede alle acties die ermee worden uitgevoerd, dient te worden gelogd. De implementatie:

```
ALTUSER   DB2PADM1  UAUDIT
```

De ‘installation system administrator’ userids moeten na definitie in RACF aan DB2 bekend worden gemaakt. De implementatie wordt uitgevoerd met behulp van de SYSTEM ADMIN-optie op het DB2-installatiepanel DSNTIPP:

```
SYSTEM ADMIN 1  ===>  DB2PADM1
SYSTEM ADMIN 2  ===>  DB2PADM2
```

Na implementatie van het DB2-subsysteem en voordat er gebruikersgegevens worden verwerkt, dient het ‘installation system administrator’ userid te worden geblokkeerd. De implementatie:

```
ALTUSER   DB2PADM1  REVOKE
ALTUSER   DB2PADM2  REVOKE
```

Het ‘installation system administrator’ userid kan in noodgevallen via Change Management worden gereactiveerd (RESUME).

22.7.2 Installatie System-Operator (Installation SYSOPR)

De ‘installation system operator’ authorities wordt gebruikt voor userids (maximaal 2) noodzakelijk voor onder andere de initiële opstart, uitvoeren van DB2-utilities voor het herstellen van databases, directory en catalog, etc. Binnen DB2 moet worden aangegeven dat een userid de ‘installation system operator’ authorities krijgt. Een ‘installation system operator’ wordt uitsluitend binnen DB2 gecontroleerd, er wordt geen aanroep naar RACF uitgevoerd. Om fouten in definities te voorkomen moeten deze ‘installation system operator’ userids per DB2-subsysteem verschillend te zijn. Na implementatie van een DB2-subsysteem zijn deze vergaande authorities normaliter niet meer nodig en kunnen de userids binnen RACF worden revoked. De implementatie:

```
ADDGROUP  DB2POPR  DATA('DB2 Productie SYSOPR groep')
                   OWNER(SECADM)

ADDGROUP  DB2TOPR  DATA('DB2 Test SYSOPR groep') OWNER(SECADM)

ADDUSER   DB2POPR1 DFLTGRP(DB2POPR)  OWNER(SECADM)
                   DATA('DB2 Installation System Operator P-omgeving')
ADDUSER   DB2TOPR1 DFLTGRP(DB2TOPR)  OWNER(SECADM)
                   DATA('DB2 Installation System Operator T-omgeving')
```

Met het ‘installation system operator’ userid moet TSO kunnen worden gebruikt. De implementatie:

```
ALTUSER   DB2POPR1  TSO(COMMAND(ISPF)  PROC(TSOPROCA)
                    SIZE(4000) MAXSIZE(8000))
```

Het gebruik van het ‘installation system operator’ userid, alsmede alle acties die ermee worden uitgevoerd, dient te worden gelogd. De implementatie:

```
ALTUSER   DB2POPR1  UAUDIT
```

De ‘installation system operator’ userids moeten na definitie in RACF aan DB2 bekend worden gemaakt. De implementatie wordt uitgevoerd met behulp van de SYSTEM OPERATOR-optie op het DB2-installatiepanel DSNTIPP:

```
SYSTEM OPERATOR 1  ===>  DB2POPR1
SYSTEM OPERATOR 2  ===>  DB2POPR2
```

Na implementatie van het DB2-subsysteem en voordat er gebruikersgegevens worden verwerkt, dient het ‘installation system operator’ userid te worden geblokkeerd. De implementatie:

```
ALTUSER   DB2POPR1  REVOKE
ALTUSER   DB2POPR2  REVOKE
```

Het ‘installation system operator’ userid kan in noodgevallen via Change Management worden gereactiveerd (RESUME).

22.8 Administrative Authorities

Voor het beheer en onderhoud van DB2-objecten is het in de praktijk noodzakelijk dat sommige gebruikers (beheerders) op tijdelijke basis administrative authorities moeten hebben. Met de hier bedoelde administrative authorities is het niet nodig specifieke objectautorisaties (privileges) te hebben om beheerwerkzaamheden op DB2-objecten te kunnen uitvoeren. Administrative authorities mogen uitsluitend voor een beperkte periode via het Change Management proces aan individuele gebruikers worden toegekend.

DB2 onderscheidt, buiten de 'installatie System Administrator' en 'installatie System Operator', 7 type administrative authorities met elk hun eigen beheermogelijkheden. Hierbij heeft het administrative authorities SYSADM de meeste rechten. Voor een effectieve functiescheiding is dit niet gewenst en zijn er meerdere minder krachtige administrative authorities aanwezig. Binnen deze administrative authorities is een splitsing gemaakt tussen het beheer van het DB2-subsysteem en de door het subsysteem beheerde databases. Voor het databasebeheer zijn, met in volgorde aflopende beheermogelijkheden, de administrative authorities DBADM, DBCTRL en DBMAINT. Voor het DB2-subsysteembeheer zijn aanwezig, met in volgorde aflopende beheermogelijkheden, de SYSADM, SYSCTRL en SYSOPR. Verder is voor het beheer van collections en packages de PACKADM administrative authorities. Voor specifiek informatie over de mogelijkheden van de 7 verschillende administrative authorities verwijzen wij naar de officiële documentatie van de leverancier. Hieronder is aangegeven wat de definities in RACF zijn om een userid een administrative authorities toe te kennen.

Het gebruik van de administrative authorities is niet toegestaan, tenzij in uitzonderlijke situaties van een emergency. Uitsluitend specifieke privileges dienen te worden toegepast.

Opmerking: RACF gebruikt het begrip 'privilege' voor SPECIAL, OPERATIONS en AUDITOR welke in de context van DB2 administrative authorities kunnen worden genoemd.

22.8.1 Limited administrative authorities

Administrative authorities mogen uitsluitend voor een beperkte tijd via Change Management worden uitgegeven en uitsluitend voor het betreffende resource waarover beheeractiviteiten moeten worden uitgevoerd. Voor het effectief limiteren van de administrative authorities is het daarom noodzakelijk deze in RACF te definiëren. De voorbeelden:

Database beheer

Een voorbeeld waarbij een beheerder DBADM administrative authorities heeft over alle databases onder een DB2 subsysteem:

```
ADDGROUP  DB2PDADM  DATA('DB2 Productie DBADM groep')
                              OWNER(SECADM)
ALTUSER   XMIEP01   UAUDIT
CONNECT   XMIEP01   GROUP(DB2PDADM)  OWNER(SECADM)

Voor een DB2-member (subsysteem naam):

SETROPTS CLASSACT(DSNADM)

RDEFINE   DSNADM  DB2P.*.DBADM  OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.*.DBADM  CLASS(DSNADM)
                              ID(DB2PDADM)  ACCESS(READ)

Of in een DB2 data sharing group omgeving (data sharing group name):

RDEFINE   DSNADM  DB2PSYSP.*.DBADM
                              OWNER(SECADM)  UACC(NONE)
PERMIT    DB2PSYSP.*.DBADM  CLASS(DSNADM)
                              ID(DB2PDADM)  ACCESS(READ)
```

Een voorbeeld waarbij een beheerder DBADM administrative authorities heeft over één database:

```
ADDGROUP      DB2PDAB3  DATA('DB2 Productie DBADM groep B3')
```

```
                                        OWNER(SECADM)
ALTUSER   XRIJL01     UAUDIT
CONNECT   XRIJL01     GROUP(DB2PDAB3)   OWNER(SECADM)

RDEFINE   DSNADM   DB2P.SALES_DB3.DBADM
                        OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SALES_DB3.DBADM  CLASS(DSNADM)
                        ID(DB2PDAB3)  ACCESS(READ)
```

Een voorbeeld waarbij een beheerder DBCTRL administrative authorities heeft over één database:

```
ADDGROUP  DB2PDC3  DATA('DB2 Productie DBCTRL groep 3')
                        OWNER(SECADM)
ALTUSER   XSPEJ01    UAUDIT
CONNECT   XSPEJ01    GROUP(DB2PDC3)   OWNER(SECADM)

RDEFINE   DSNADM   DB2P.SALES_DB3.DBCTRL
                        OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SALES_DB3.DBCTRL  CLASS(DSNADM)
                        ID(DB2PDC3)  ACCESS(READ)
```

Een voorbeeld waarbij een beheerder DBMAINT administrative authorities heeft over één database:

```
ADDGROUP  DB2PDM3  DATA('DB2 Productie DBMAINT groep 3')
                        OWNER(SECADM)
ALTUSER   XPERO01    UAUDIT
CONNECT   XPERO01    GROUP(DB2PDM3)   OWNER(SECADM)

RDEFINE   DSNADM   DB2P.SALES_DB3.DBMAINT
                        OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SALES_DB3.DBMAINT   CLASS(DSNADM)
                        ID(DB2PDM3)  ACCESS(READ)
```

Collection en Package beheer

Een voorbeeld waarbij een beheerder PACKADM administrative authorities heeft over één collection:

```
ADDGROUP  DB2PPK23  DATA('DB2 Productie PACKADM groep 23')
                        OWNER(SECADM)
ALTUSER   XENGF01    UAUDIT
CONNECT   XENGF01    GROUP(DB2PPK23)   OWNER(SECADM)

RDEFINE   DSNADM   DB2P.COL_SERV23A.PACKADM
                        OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.COL_SERV23A.PACKADM   CLASS(DSNADM)
                        ID(DB2PPK23)  ACCESS(READ)
```

DB2 Subsysteem beheer

Een voorbeeld waarbij een beheerder SYSADM administrative authorities heeft:

```
ADDGROUP       DB2PSADM   DATA('DB2 Productie SYSADM groep')
                        OWNER(SECADM)
ALTUSER   XMINA01    UAUDIT
CONNECT   XMINA01    GROUP(DB2PSADM)   OWNER(SECADM)

RDEFINE   DSNADM   DB2P.SYSADM  OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SYSADM  CLASS(DSNADM)
```

```
                                    ID(DB2PSADM)  ACCESS(READ)
```

Een voorbeeld waarbij een beheerder SYSCTRL administrative authorities heeft:

```
ADDGROUP  DB2PSCTL  DATA('DB2 Productie SYSCTRL groep 5')
                                    OWNER(SECADM)
ALTUSER   XKRET01   UAUDIT
CONNECT   XKRET01   GROUP(DB2PSCTL)  OWNER(SECADM)

RDEFINE   DSNADM  DB2P.SYSCTRL
                                    OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SYSCTRL  CLASS(DSNADM)
                                    ID(DB2PSCTL)  ACCESS(READ)
```

Een voorbeeld waarbij een beheerder SYSOPR administrative authorities heeft:

```
ADDGROUP      DB2PSOPR  DATA('DB2 Productie SYSOPR groep 1')
                                    OWNER(SECADM)
ALTUSER   XOPED01   UAUDIT
CONNECT   XOPED01   GROUP(DB2PSOPR)  OWNER(SECADM)

RDEFINE   DSNADM  DB2P.SYSOPR  OWNER(SECADM)  UACC(NONE)
PERMIT    DB2P.SYSOPR  CLASS(DSNADM)
                                    ID(DB2PSOPR)  ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde administrative authorities door DB2 wordt geautoriseerd moet in de RACF general resource class DSNADM een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE   DSNADM  **  OWNER(SECADM)  UACC(NONE)
```

22.9 Subsysteem beheer

Voor het beheer van het DB2-subsysteem moet in de praktijk operator commando's worden gegeven die alle databases en daardoor alle informatie die door DB2 wordt beheerd treft. Deze handelingen zijn onder andere het starten en stoppen van het DB2-subsysteem, het archiveren van de loggegevens, etc. Uitsluitend Operators met deze specifieke taken mogen hiervoor zijn geautoriseerd. Een voorbeelden voor het monitoren van systeem informatie van databases, buffer pools, locations, logs, threads, traces en archives:

```
SETROPTS CLASSACT(MDSNSM)

RDEFINE   MDSNSM  DB2P.DISPLAY
                                    OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de groep *DB2POPR0* de systeeminformatie onder één DB2-subsysteem kan benaderen:

```
PERMIT    DB2P.DISPLAY  CLASS(MDSNSM)
                                    ID(DB2POPR0)  ACCESS(READ)
```

Voor het delegeren van monitoring van systeeminformatie is het mogelijk hiervoor een meer fijnmazigere verdeling van autorisaties te definiëren. Een voorbeeld:

```
RDEFINE   MDSNSM  DB2P.DBP23.DISPLAY
                                    OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de groep *DB2POPR8* de systeeminformatie onder één DB2-subsysteem kan benaderen:

```
PERMIT    DB2P.DBP23.DISPLAY  CLASS(MDSNSM)
                              ID(DB2POPR8)  ACCESS(READ)
```

DB2 biedt mogelijkheden om trace-gegevens te lezen. Dit mag uitsluitend worden uitgevoerd onder verantwoordelijkheid van Security Management. Operations zal hier wel voor moeten worden geautoriseerd. Een voorbeeld:

```
RDEFINE   MDSNSM  DB2P.MONITOR2
                  OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de groep *DB2POPR6* de DB2-trace gegevens kan lezen:

```
PERMIT    DB2P.MONITOR2  CLASS(MDSNSM)
                         ID(DB2POPR6)  ACCESS(READ)
```

Andere autorisaties mogelijkheden voor operations van het DB2-subsysteem zijn:

```
RDEFINE   MDSNSM  DB2P.MONITOR1  OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.MONITOR2  OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.TRACE     OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.RECOVER   OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.BSDS      OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.STOPALL   OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.ARCHIVE   OWNER(SECADM)  UACC(NONE)
```

Het gebruik van de BINDADD en BINDAGENT is niet toegestaan. De voorbeelden:

```
RDEFINE   MDSNSM  DB2P.BINDADD   OWNER(SECADM)  UACC(NONE)

RDEFINE   MDSNSM  DB2P.*.BINDAGENT  OWNER(SECADM)  UACC(NONE)
```

Het gebruik van de volgende autorisaties is niet toegestaan aangezien deze met meer specifiekere RACF-profielen kunnen worden gedefinieerd. De voorbeelden:

```
RDEFINE   MDSNSM  DB2P.CREATETAB
          OWNER(SECADM)  UACC(NONE)
RDEFINE   MDSNSM  DB2P.CREATETMTAB
          OWNER(SECADM)  UACC(NONE)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde privilege door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNSM een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE   MDSNSM  **   OWNER(SECADM)  UACC(NONE)
```

22.10 Speciale autorisaties

22.10.1 PUBLIC*

Het gebruik van het userid PUBLIC* als eigenaar (OWNER) voor DB2-objecten geeft alle gebruikers in het DB2-subsysteem de autorisatie om het betreffende object te benaderen of uit te voeren. Bij het definieren van het PUBLIC* userid wordt de DB2-autorisatiemodule niet aangeroepen en daardoor kan er door RACF geen controle worden uitgeoefend. Het gebruik van PUBLIC* is daarom niet toegestaan en het RACF-profiel dat het object beschermd dient in dit geval met UACC(READ) te worden gedefinieerd. Een voorbeeld:

```
RDEFINE   MDSNPN  DB2P.PLAN99.EXECUTE
                  OWNER(SECADM)  UACC(READ)
```

22.10.2 Eigenaarschap (Ownership)

Bij het creëren (BINDen) van een DB2-applicatieplan of -package wordt hiervoor binnen DB2 impliciet (de uitvoerder van het BIND-commando) of expliciet (met de OWNER-parameter) een eigenaar voor het object gedefinieerd. Het toebedelen van het object-eigenaarschap geeft de eigenaar impliciete rechten op het package of applicatieplan. De DB2-autorisatie exit en dus RACF worden niet aangeroepen indien de eigenaar het betreffende applicatieplan of package benadert. Omdat deze autorisatie techniek, het toekennen van het eigenaarschap binnen DB2, gelijk is aan het toekennen van autorisatie op het applicatieplan of package, moet door Security Management worden geverifieerd of de juiste eigenaar is toegekend. Een voorbeeld:

```
DSN  BIND  PLAN(PLAN1)   MEMBER(PROG13)
           LIRARY(DB2DEV.ACCP.LOAD)
           OWNER(DB2PLAN)  QUALIFIER(DB2PLAN)
```

Voor het beheer van de RACF-profielen, bijvoorbeeld het profiel dat het plan in dit voorbeeld beschermd, moet de eigenaar de Security Administration zijn (centraal of decentraal). Een voorbeeld in relatie met de voorgaande definitie:

```
RDEFINE    MDSNPL   DB2P.PLAN1.EXECUTE
                              OWNER(SECADM)  UACC(NONE)
```

Security Administration (centraal of decentraal) kan door middel van het eigenaarschap, zonder extra privileges de rechten definiëren voor het gebruik van het in dit voorbeeld genoemde plan. Een voorbeeld hoe de groep *VERKOOP* het *PLAN1* kan gebruiken:

```
PERMIT     DB2P.PLAN1.EXECUTE   CLASS(MDSNPL)
                              ID(VERKOOP)  ACCESS(READ)
```

Zo dienen er voor de meeste typen objecten een apart groepid te worden gedefinieerd, welke als eigenaar voor de DB2-objecten kan worden gebruikt. De groepids voor applicatieplannen of packages mogen geen verdere autorisaties hebben dan voor de resources die door de hierin opgenomen SQL-statements moeten worden benaderd. Enige voorbeelden:

```
ADDGROUP    DB2POWN   DATA('DB2 Ownerholder groep')
            OWNER(SECADM) SUPGROUP(DB2)
ADDGROUP    DB2ACCES  DATA('DB2 Owner DB2 Access')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2ADM    DATA('DB2 Owner Admin Authorities')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2SPRIV  DATA('DB2 Owner System Privileges')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2STOGR  DATA('DB2 Owner Storage Groups')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2BUFPL  DATA('DB2 Owner Buffer Pools)
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2TABSP  DATA('DB2 Owner Table Spaces')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2DB     DATA('DB2 Owner Databases')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2COLL   DATA('DB2 Owner Collections')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2PACK   DATA('DB2 Owner Packages')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2PLAN   DATA('DB2 Owner Plans')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2TAB    DATA('DB2 Owner Tables')
            OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP    DB2INDEX  DATA('DB2 Owner Indexes')
```

```
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2VIEW    DATA('DB2 Owner Views')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2ALIAS   DATA('DB2 Owner Aliases')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2SYN     DATA('DB2 Owner Synoniems')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2SPROC   DATA('DB2 Owner Stored Procedures')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2UDEF    DATA('DB2 Owner User def Funct')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2UTYP    DATA('DB2 Owner User Dist Types')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2TRIG    DATA('DB2 Owner Triggers')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2SCHMA   DATA('DB2 Owner Schemas')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2PROF    DATA('DB2 Owner Sluitprofielen')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
ADDGROUP        DB2AMS     DATA('DB2 Owner Remote Access')
                OWNER(SECADM)  SUPGROUP(DB2POWN)
```

Opmerking: Als alternatief kan worden besloten om uitsluitend één eigenaar te gebruiken voor alle typen resources. Hierdoor wordt het beheer verminderd maar differentiatie bij toekenning van rechten is dan niet mogelijk. Bij deze opzet zal de naam van de eigenaar (OWNER) uiteindelijk in elke DB2-definitie hetzelfde zijn. Een voorbeeld voor een owner:

```
ADDGROUP        DB2POWN   DATA('DB2 Productie Owner groep')
                OWNER(SECADM) SUPGROUP(DB2)
```

22.10.3 GRANT

Met de 'WITH GRANT' optie in SQL-statements kunnen gebruikers autorisaties overdragen aan andere gebruikers. Dit is tegen de filosofie van RACF en autorisaties verleend via de GRANT worden niet door de DB2-autorisatiemodule en RACF gecontroleerd. SQL-statements met de 'WITH GRANT' optie mogen dan ook niet worden gebruikt.

22.10.4 Any Table

Het gebruik van het 'Any Table' privilege is niet door RACF ondersteund en zal, indien noodzakelijk voor de uitvoering van een gebruikersrol, met verschillende meer gedetailleerde RACF-profielen moeten worden gedefinieerd.

22.11 Expliciete Autorisaties voor resourcebeheer

Buiten de administratieve autoriteiten kunnen binnen RACF ook specifieke autorisaties worden toegekend. Het is hiermee mogelijk om op detailniveau gebruikers specifieke rechten toe te kennen. Deze rechten worden door middel van RACF-profielen in diverse general resource classes gedefinieerd. De toekenning van autorisaties moet ten grondslag liggen aan een autorisatiematrix waarin de rollen en de bijbehorende autorisaties zijn beschreven.

Opmerking: Binnen DB2 wordt de term 'privilege' gebruikt voor toegang tot resources waar wij de voorkeur hebben om over autorisaties te praten hierdoor wordt een eenduidigheid van terminologie binnen de gehele PI RACF-standaard bewerkstelligd.

22.11.1 Storage groep

Een storage groep is een DB2-definitie waaraan een set DASD-volumes kan worden toegewezen. In de storage groep en dus op de toegewezen DASD kunnen de DB2-gegevens fysiek worden opgeslagen. Uitsluitend Storage Administration mag geautoriseerd zijn om storage groepen te kunnen beheren. Een voorbeeld:

```
RDEFINE    MDSNSM   DB2P.CREATESG
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2SG* de storage groepen kan beheren:

```
PERMIT     DB2P.CREATESG   CLASS(MDSNSM)
                        ID(DB2SG)   ACCESS(READ)
```

Voor het beheer van storage pools heeft de Storage Administration autorisaties nodig. Informatie over het gebruik van storage pools kan met behulp van de STOSPACE utility worden verkregen. Uitsluitend Storage Administration mag geautoriseerd zijn om het STOSPACE utility te kunnen gebruiken. Een voorbeeld:

```
RDEFINE    MDSNSM   DB2P.STOSPACE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep DB2SG de storage groepen kan monitoren:

```
PERMIT     DB2P.STOSPACE   CLASS(MDSNSM)
                        ID(DB2SG)   ACCESS(READ)
```

Naast het beheren, zoals het aanmaken en monitoren, van een storage group is het noodzakelijk om het gebruik hiervan voor het definiëren van databases te autoriseren. Uitsluitend Database Administration mag een storage group gebruiken voor het definiëren van databases. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNSG)

RDEFINE    MDSNSG   DB2P.STOGRP23.USE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* de storage groepen kan gebruiken:

```
PERMIT     DB2P.STOGRP23.USE   CLASS(MDSNSG)
                        ID(DB2DBA1)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde storage groep door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNSG een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNSG   **    OWNER(SECADM)   UACC(NONE)
```

22.11.2 Database

Een database is een verzameling table spaces en index spaces met daarin tabellen en indexen. Uitsluitend Database Administration mag via Change Management geautoriseerd zijn om databases te kunnen aanmaken en te kunnen verwijderen. Het nadeel met create database (CREATEDBC) is dat de persoon die de database aanmaakt direct DBCTRL privileges heeft over de database. De eigenaar van de database moet daarom een groepid zijn dat geen verdere autorisaties heeft. Voor het benaderen van databasegegevens moeten specifieke autorisaties worden gedefinieerd welke met DBADM impliciet worden gedefinieerd. Het gebruik van de CREATEDBA autorisatie (in de MDSNSM RACF general resource class) wat DBADM

privileges aan de eigenaar van de database geeft, is daarom niet toegestaan. Een tweetal voorbeelden van autorisatie voor het creëren en voor het droppen van een database:

Voor het creëren van een database:

```
RDEFINE    MDSNSM   DB2P.CREATEDBC
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* de databases kan aanmaken:

```
PERMIT     DB2P. CREATEDBC   CLASS(MDSNSM)
                        ID(DB2DBA1)   ACCESS(READ)
```

Voor het droppen (verwijderen) van een database:

```
SETROPTS CLASSACT(MDSNDB)

RDEFINE    MDSNDB   DB2P.DBP23.DROP
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* de databases kan droppen:

```
PERMIT     DB2P.DBP23.DROP   CLASS(MDSNDB)
                        ID(DB2DBA1)   ACCESS(READ)
```

Voor het operationeel beheer van de databases mag uitsluitend DB2 Systems Operation worden geautoriseerd. Een voorbeeld voor een image copy, merge copy, modify recovery of een queisce bewerking:

```
RDEFINE    MDSNDB   DB2P.DBP23.IMAGCOPY
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2POPR3* hiervoor geautoriseerd kan worden:

```
PERMIT     DB2P.DBP23.IMAGCOPY   CLASS(MDSNDB)
                        ID(DB2POPR3)   ACCESS(READ)
```

Andere autorisatie mogelijkheden voor database operations zijn onder andere:

```
RDEFINE    MDSNDB   DB2P.DBP23.LOAD          Load a table
RDEFINE    MDSNDB   DB2P.DBP23.STARTDB       Starten database
RDEFINE    MDSNDB   DB2P.DBP23.STOPDB        Stoppen database
RDEFINE    MDSNDB   DB2P.DBP23.REORG         Reorganisatie Db
RDEFINE    MDSNDB   DB2P.DBP23.RECOVERDB     Recovery Db
RDEFINE    MDSNDB   DB2P.DBP23.REPAIR        Repair Db
RDEFINE    MDSNDB   DB2P.DBP23.DISPLAYDB     Display Db status
RDEFINE    MDSNDB   DB2P.DBP23.STATS         Display statitics
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde database door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNDB een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNDB   **    OWNER(SECADM)   UACC(NONE)
```

22.11.3 Table space

Een table space is een logische ruimte voor het vastleggen van één of meerdere tabellen en is vastgelegd in één of meerdere datasets verdeeld over één of meerdere DASD. Uitsluitend een Database Administration mag geautoriseerd zijn om table spaces te kunnen beheren. Een voorbeeld voor het beheer van een table space:

```
RDEFINE    MDSNDB   DB2P.DBP23.CREATETS
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* tabel spaces in een database (*DBP84*) kan aanmaken:

```
PERMIT     DB2P.DBP23.CREATETS   CLASS(MDSNDB)
                        ID(DB2DBA1)   ACCESS(READ)
```

Naast het beheren (het kunnen aanmaken) van een table space is het noodzakelijk om het gebruik van een table space te autoriseren. Uitsluitend Database Administration mag een table space gebruiken voor het definiëren van een tabel. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNTS)

RDEFINE    MDSNTS   DB2P. DBP23.TBS34.USE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* gebruik kan maken van een tabel space:

```
PERMIT     DB2P.DBP84.TBS34.USE   CLASS(MDSNTS)
                        ID(DB2DBA1)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde tabel space door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNTS een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNTS   **    OWNER(SECADM)   UACC(NONE)
```

22.11.4 Table

Een tabel is het basisgegevensmodel in een DB2-database. Deze bestaat uit kolommen (column) en regels (row of record). Uitsluitend Database Administration mag worden geautoriseerd voor het beheer van een tabel(len) in de database. Een voorbeeld voor het creëren van een tabel:

```
RDEFINE    MDSNDB   DB2P.DBP23.CREATETAB
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA3* een tabel in een database (*DBP23*):

```
PERMIT     DB2P.DBP23.CREATETAB   CLASS(MDSNDB)
                        ID(DB2DBA3)   A0CCESS(READ)
```

Tijdelijke tabellen

Naast de permanente tabellen kunnen ook tijdelijke (temporary) tabellen worden aangemaakt. Uitsluitend een geautoriseerde gebruiker mag een tijdelijke tabel in de database aanmaken. Een voorbeeld:

```
RDEFINE    MDSNDB   DB2P.DBP23.CREATETMTAB
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *VERKOOP* tijdelijke tabellen in een database (*DBP23*) kan aanmaken:

```
PERMIT     DB2P.DBP23.CREATETMTAB   CLASS(MDSNDB)
                        ID(VERKOOP)   ACCESS(READ)
```

Beheer Tabeldefinities

De opbouw van een tabel verschilt per tabel. Met DB2-definities worden bijvoorbeeld nieuwe kolommen aangemaakt. Uitsluitend Database Administration mag deze definities uitvoeren. Een voorbeeld:

SQL ALTER TABLE *TAB29.RDAM* (*COL1, COL2, COLX*)

Een voorbeeld voor beheer van tabeldefinities:

Voor het aanpassen van tabel (TBP29) definities:

SETROPTS CLASSACT(**MDSNTB**)

RDEFINE MDSNTB *DB2P.DB2TAB.TBP29.***ALTER**
OWNER(*SECADM*) UACC(NONE)

Een voorbeeld hoe de groep *DB2DBA1* tabel (TBP29) definities kan aanpassen:

PERMIT *DB2P.DB2TAB.TBP29.***ALTER** CLASS(MDSNTB)
ID(*DB2DBA1*) ACCESS(**READ**)

Voor het aanpassen van tabel (TBP29) referenties voor referential integrity:

RDEFINE MDSNTB *DB2P.DB2TAB.TBP29.* **REFERENCES**
OWNER(*SECADM*) UACC(NONE)

Een voorbeeld hoe de groep *DB2DBA1* tabel (TBP29) referenties kan aanpassen:

PERMIT *DB2P.DB2TAB.TBP29.* **REFERENCES** CLASS(MDSNTB)
ID(*DB2DBA1*) ACCESS(**READ**)

Voor het aanpassen van tabel (TBP29) referenties per kolom of set van kolommen:

RDEFINE MDSNTB *DB2P.DB2TAB.TBP29.COLNAW.***REFERENCES**
OWNER(*SECADM*) UACC(NONE)

Een voorbeeld hoe de groep *DB2DBA1* tabel (TBP29) kolom of een set van kolom referenties kan aanpassen:

PERMIT *DB2P.DB2TAB.TBP29.COLNAW.***REFERENCES** CLASS(MDSNTB)
ID(*DB2DBA1*) ACCESS(**READ**)

Om te voorkomen dat een niet specifiek in RACF gedefinieerde tabel door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNTB een sluitprofiel worden gedefinieerd. De implementatie:

RDEFINE MDSNTB ****** OWNER(*SECADM*) UACC(NONE)

22.11.5 Index (space)

Een index is een set van verwijzingen (pointers) in een tabel om gegevens op een andere en daardoor snellere manier te benaderen. Een index space is een logische ruimte voor het vastleggen van één index en is vastgelegd in één datasets. Voor het creëren van een index op een tabel is de INDEX-autorisatie noodzakelijk. Uitsluitend een Database performance analist mag geautoriseerd zijn om index (spaces) te kunnen beheren. Een voorbeeld voor het beheer van een index (space):

Voor het definiëren van een index op een tabel (*TBP29*):

```
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.INDEX
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBPRF* een index op tabel (TBP29) kan definiëren:

```
PERMIT     DB2P. DB2TAB.TBP29.INDEX   CLASS(MDSNTB)
                        ID(DB2DBPRF)   ACCESS(READ)
```

22.11.6 View

Een view geeft de mogelijkheid om tabelgegevens op een andere manier te representeren. Een view kan gegevens presenteren die uit meerdere tabellen of andere views afkomstig zijn. Verder kan met een view een selectie van gegevens worden uitgevoerd zodat de gebruikers maar een deel van de gegevens kunnen zien/bewerken. De informatie hoe een view is opgebouwd is in de DB2-catalog opgeslagen. Uitsluitend Dababase Administration mag een view aanmaken. Binnen DB2-RACF is er geen fijnmazige autorisatie mogelijk voor het aanmaken van een view en moet Database Administration voor deze activiteit SYSCTRL krijgen. Dit is het doorbreken van het beveiligingsconcept zoals in de standaard is opgebouwd. Voor het creëren van een view is het SYSCTRL privilege voor Database Administration noodzakelijk en moet daarom via Change Management worden toegestaan. Een voorbeeld:

```
RDEFINE    DSNADM   DB2P.SYSCTRL
                        OWNER(SECADM)   UACC(NONE)
PERMIT     DB2P.SYSCTRL   CLASS(DSNADM)
                        ID(DB2DBA1)   ACCESS(READ)
```

22.11.7 DB2 Synoniem en Alias

Met een DB2-synoniem kan een alternatieve naam worden gegeven aan tabellen of views die zich in het lokale systeem bevinden. Voor een alternatieve tabel- of view-naam die door een remote DB2-systeem is te benaderen kan een DB2-alias worden gedefinieerd. De alternatieve naam (synoniem of alias) kan in de SQL-statements worden gebruikt als tabel- of view-naam. Uitsluitend Database Administration mag een synoniem of alias aanmaken. Binnen DB2/RACF kunnen geen specifieke autorisatie voor het aanmaken van een synoniem worden gedefinieerd. Voor het creëren van een synoniem heeft de beheerder daarom minimaal SYSADM bevoegdheden nodig. Uitsluitend voor de periode noodzakelijk voor het aanmaken van een synoniem moet Database Administration via Change Management de SYSADM privileges krijgen. Een voorbeeld voor het kunnen definieren van een synoniem:

```
PERMIT     DB2P.SYSADM   CLASS(DSNADM)   ID(DB2DBA1)
                ACCESS(READ)
```

Voor het definiëren van een alias zijn, alhoewel beperkt, meer specifiekere autorisaties mogelijk. Uitsluitend een Database Administration mag een alias aanmaken. Een voorbeeld voor een alias:

```
RDEFINE    MDSNSM   DB2P.CREATEALIAS
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBA1* aliases kan beheren:

```
PERMIT     DB2P.CREATEALIAS   CLASS(MDSNSM)
                        ID(DB2DBA1)   ACCESS(READ)
```

22.11.8 Buffer pool

Door het reduceren van de lees/schrijf-bewerkingen naar DASD met buffer pools kan de DB2-performance worden verbeterd. Met buffer pools worden gegevens tussen de applicatie en de

database in het werkgeheugen gebufferd. Gegevens die reeds van schijf zijn gelezen worden dan voor een volgende leesbewerking in een buffer pool bewaard. Uitsluitend een Database performance analist mag geautoriseerd zijn om buffer pools te kunnen beheren. Een voorbeeld voor een autorisatie van het buffer pool gebruik:

```
SETROPTS CLASSACT(MDSNBP)

RDEFINE    MDSNBP   DB2P.BPL87.USE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *DB2DBPRF* gebruik kan maken van een tabel space:

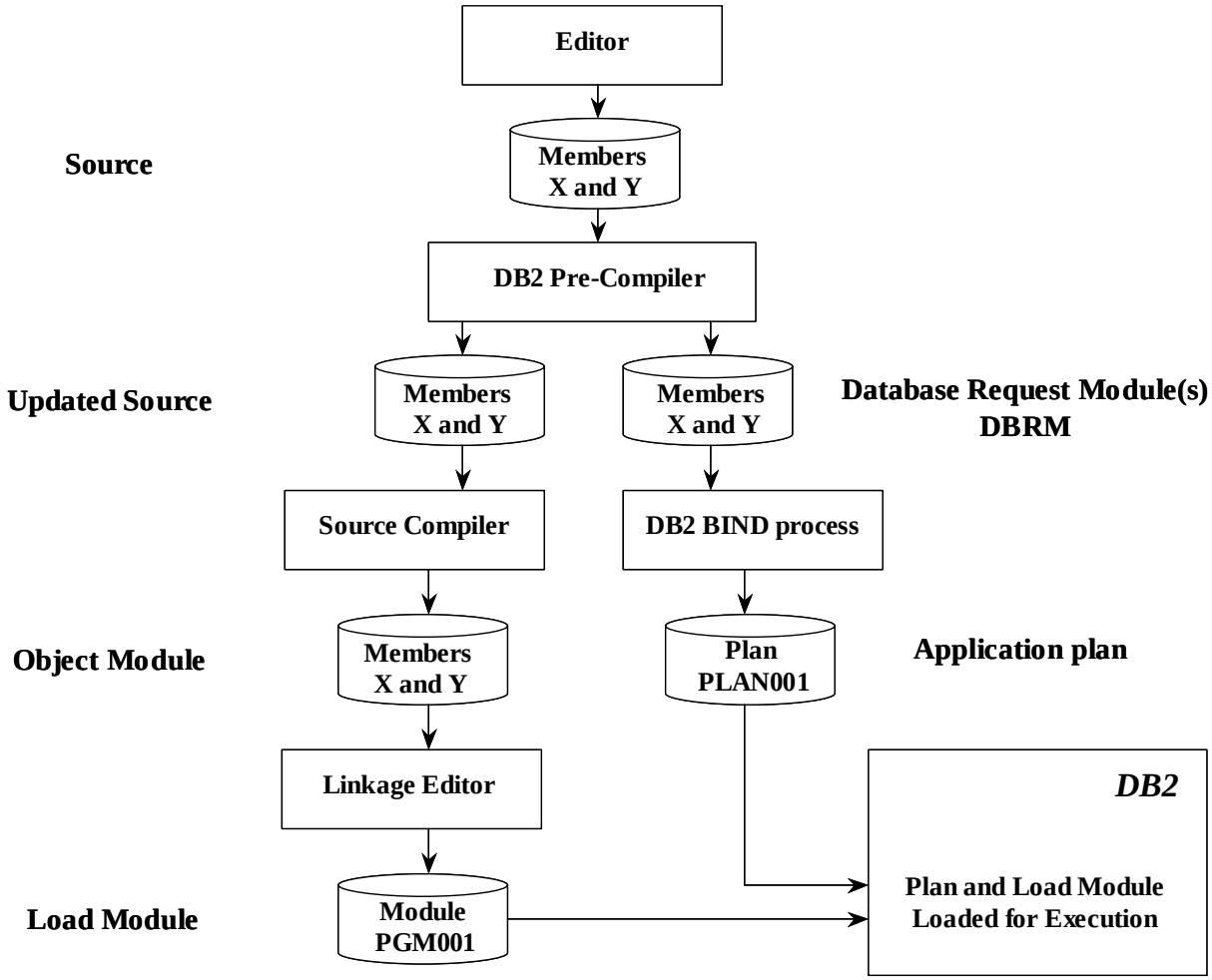
```
PERMIT     DB2P.BPL87.USE   CLASS(MDSNBP)
                        ID(DB2DBPRF)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde buffer pool door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNBP een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNBP   **    OWNER(SECADM)   UACC(NONE)
```

22.12 Programma beheer

Om een uitvoerbaar DB2 programma te vervaardigen wordt de applicatie-source met daarin SQL-statements als eerste door de DB2-precompiler verwerkt waarbij de SQL-statements en de programmacode worden gescheiden. De SQL-statements worden apart in een Database Request Module (DBRM) geplaatst. Zowel de DBRM als de programma-logic krijgen voor unieke identificatie van de DB2 pre-compiler een timestamp. Een DBRM wordt met het BIND-commando vertaald (compiled) en binnen DB2, met dezelfde naam als de DBRM, als plan of package geregistreerd. Op uitvoeringstijd wordt door een programma het package-id en de bijbehorende timestamp aan DB2 gegeven zodat deze de juiste SQL-statements (in een package of plan) kan laden en uitvoeren. Hieronder een schematische weergave van het DB2-compilatie proces.



Figuur 2. DB2-Autorisatieflow

De uitkomst van een BIND kan een ‘applicatieplan’ ook wel ‘access-plan’ genoemd of een ‘package’ zijn. Hieronder zijn de verschillende beveiligingsaspecten beschreven die met het BIND-proces samenhangen.

22.12.1 DB2 BIND-concepts

Het BIND PACKAGE-commando vertaald één DBRM tot één package en krijgt daarbij de DBRM-naam toegewezen. In een package staan alle SQL-statements en de bijbehorende toegangspaden tot de gegevens die door de SQL-statements zullen worden benaderd. Een voorbeeld van het binden van een package:

```
DSN  BIND  PACKAGE(PACK001)  MEMBER(PROG68)
                              LIRARY(DB2DEV.ACCP.LOAD)
```

Het BIND PLAN-commando kan op twee manieren worden toegepast, namelijk het creëert een applicatieplan direct vanuit één of meerdere DBRMs (MEMBER parameter) of het creëert een applicatieplan vanuit een package-list (PKLIST-parameter). Voor het gebruik van een package-list moeten de packages vooraf door het BIND PACKAGE-commando zijn geBIND en kunnen de packages met de PKLIST(*packages*) parameter in het applicatieplan worden geregistreerd. Bij het gebruik van een package-list kan het BIND PACKAGE-commando opnieuw voor een DBRM/package wordt uitgevoerd zonder dat de BIND PLAN opnieuw behoeft te worden uitgevoerd voor de applicatieplannen waarin de packages zijn geregistreerd. Een package kan

met de package-list optie in meerdere applicatieplannen worden geregistreerd. Indien de DBRMs direct met het BIND PLAN als member zijn opgenomen moet bij wijziging van een member deze BIND PLAN opnieuw worden uitgevoerd. Een voorbeeld van een DBRM die direct in het plan wordt opgenomen:

```
DSN  BIND  PLAN(PLAN002)  MEMBER(PROG68)
                          LIBRARY(DB2DEV.ACCP.LOAD)
                          OWNER(DB2PLAN)
```

Een voorbeeld voor het registreren van packages in een applicatieplan:

```
DSN  BIND  PLAN(PLAN003)  PKLIST(SALES.PROG69,SALES.PACK75)
                          OWNER(DB2PLAN)
```

Binnen de BIND PACKAGE en BIND PLAN worden de volgende typen BINDs onderkend:

Static BIND
Een static BIND is een proces waar alle SQL-statements in de DBRM tijdens de uitvoering van het BIND-commando worden gevalideerd en daarna in een applicatieplan worden opgenomen. Het BIND-commando valideert onder andere of de eigenaar van het applicatieplan geautoriseerd is om alle SQL-statements in het DBRM te mogen uitvoeren. Vooraf worden alle SQL-statements gevalideerd en kunnen dan niet meer worden veranderd zonder een nieuwe BIND uit te voeren. Deze optie heeft vanwege beveiligingsredenen de voorkeur boven andere opties die hieronder worden beschreven. Een static BIND wordt gerealiseerd met het DB2-commando:

```
DSN  BIND  PLAN(PLAN003)  VALIDATE(BIND)  OWNER(DB2PLAN)
```

Incremental BIND
Als vooraf de autorisaties voor alle SQL-statements niet kan worden geverifieerd, omdat bijvoorbeeld tabellen niet aanwezig zijn, kan worden gekozen voor de BIND-optie waarbij op uitvoeringstijd de autorisaties worden geverifieerd. Deze optie heeft vanwege beveiligingsredenen niet de voorkeur omdat vooraf de juiste autorisaties niet zijn bepaald. Een incremental BIND wordt gerealiseerd met het DB2-commando:

```
DSN  BIND  PLAN(PLAN003)  VALIDATE(RUN)  OWNER(DB2PLAN)
```

Automatic BIND
Als een gebruiker EXECUTE-rechten heeft op een ongeldig applicatieplan of package zal DB2 automatisch eerst proberen het applicatieplan of package opnieuw teBINDen. In werkelijkheid wordt een automatische REBIND uitgevoerd van het reeds aanwezig applicatieplan of package. Dit kan gebeuren doordat een object als bijvoorbeeld een index is verwijderd. Deze mogelijkheid is geen optie omdat dit automatisch door DB2 wordt uitgevoerd. Veranderingen, waardoor het applicatieplan of package ongeldig is geworden, worden hiermee direct geactiveerd. De REBIND zorgt ervoor dat de autorisaties van de eigenaar van het plan geverifieerd worden zoals bij een static BIND en gebruikt daarbij dezelfde opties als bij de originele BIND waren toegepast.

Dynamic BIND
SQL-statements kunnen dynamisch tijdens de uitvoering van een applicatie worden gedefinieerd en uitgevoerd. De SQL-statements kunnen om die reden elke willekeurige vorm krijgen die praktisch mogelijk is. Dit is in tegenstelling tot statische SQL-statements welke tijdens een handmatig geïnitieerd BIND-proces vooraf worden gedefinieerd en gefixeerd zoals hiervoor in de paragraaf static, incremental en automatic BIND is beschreven. Bij uitvoering van dynamic SQL-statements moet de BIND dynamisch (tijdens de uitvoering van het SQL-statement) plaatsvinden. Het applicatieplan (accessplan) met daarin de autorisatie controle(s) wordt dan automatisch gecreëerd.

Er zijn 4 typen dynamic SQL, namelijk:

- interactive SQL, waarbij de gebruiker de SQL-statements via SPUFI invoert. Dit is echt dynamic SQL;
- Dynamic SQL via een ODBC functie, waarbij DB2 de SQL-statements als argumenten ontvangt. Dit is echt dynamic SQL;
- embedded dynamic SQL, hierbij wordt de SQL in een variabele geplaatst en door een PREPARE en/of EXECUTE-statement uitgevoerd. DB2 lost dan de variabele op, waarbij dan een volledig SQL-statement moet ontstaan en voert het statement daarna uit. De programma's die embedded SQL gebruiken moeten vooraf worden geBIND;
- deferred embedded SQL, hierbij zijn de SQL-statements reeds in het programma verwerkt maar worden pas tijdens uitvoeringstijd gevalideerd. Deze techniek wordt gebruikt bij remote SQL-packages. Zie hiervoor de volgende paragraaf.

Op welke wijze autorisatie voor embedded en deferred embedded dynamic SQL-statements wordt gecontroleerd is afhankelijk van de BIND-parameter DYNAMICRULES. Deze parameter dient voor een package de volgende waarde te hebben:

```
DSN  BIND  PACKAGE   DYNAMICRULES(INVOKERUN)   OWNER(DB2PLAN)
```

De parameter voor een plan:

```
DSN  BIND  PLAN      DYNAMICRULES(RUN)   OWNER(DB2PLAN)
```

BIND voor remote DB2
Een applicatieplan is uitsluitend geldig op locale DB2-subsystemen. Binnen het DB2-subsysteem is in het applicatieplan een access-pad gedefinieerd. Een applicatieplan kan echter wel remote packages hebben geregistreerd. In dat geval worden in de package-list van het BIND PLAN-commando echter niet de lokale packages gespecificeerd maar de remote-package terwijl de remote-package op het remote systeem moeten worden geBIND. De eigenaar van het remote package, gedefinieerd met de BIND PACKAGE OWNER-parameter op het remote-systeem, moet wel alle rechten hebben om alle SQL-statements die in het package zijn gedefinieerd te kunnen uitvoeren. Een voorbeeld voor een remote-package BIND op het remote systeem *RDAM*:

```
DSN  BIND  PACKAGE(RDAM.COLL77)  MEMBER(PACK67)
                                 LIRARY(DB2DEV.ACCP.LOAD)
                                 OWNER(DB2PLAN)
```

Voor het BINDen van het applicatieplan op het lokale systeem kan het BIND PLAN worden uitgevoerd echter nu moet de locatienaam worden gedefinieerd. Een voorbeeld van het BINDen van een remote package in het applicatieplan op een lokaal systeem:

```
DSN  BIND  PLAN(PLAN95)  PKLIST(RDAM.COLL77.PACK67)
                         OWNER(DB2PLAN)
```

Een Database administrators mag uitsluitend een BIND kunnen uitvoeren indien deze is geautoriseerd voor een collection die in RACF is gedefinieerd. Dit moet op het installatiepanel DSNTIPP worden aangegeven. De implementatie:

```
BIND NEW PACKAGE  ===>  BIND
```

Overwegingen voor de keuze van een static of dynamic BIND zijn de:

- vereiste beveiliging;
- inspanningen die moeten worden verricht om de beveiligingsdefinities te onderhouden;
- technische mogelijkheden die voor de toepassing noodzakelijk zijn;

- performance die wordt vereist. Bij dynamic wordt de BIND herhaaldelijk, bij elke aanroep, uitgevoerd en heeft daarbij elke keer een autorisatie check tot gevolg.

22.12.2 BIND Plan

Een ‘plan’ of ‘applicatieplan’ is een controlstructuur dat tijdens het BIND-proces wordt gegenereerd. Het plan geeft DB2 het toegangspad en de toegangsautorisaties, welke nodig zijn om tijdens de uitvoering van een applicatie de betreffende SQL-statement uit te kunnen voeren. Het plan is in het kader van RACF van belang omdat hier de toegangscontrole wordt uitgevoerd van de vooraf (statische) gedefinieerde SQL-statements. Wanneer de autorisatie voor SQL-statements wordt geverifieerd hangt af van de gebruikte BIND-optie zoals hiervoor is beschreven. Uitsluitend een Production Binder, verantwoordelijk voor de verificatie van de applicatieprogrammatuur, mag een BIND uitvoeren. Een voorbeeld waarbij het DBRM direct in een plan wordt gebonden:

```
DSN  BIND  PLAN(PLAN84)  OWNER(DB2PLAN)  MEMBER(PACK68)
                                          LIRARY(DB2DEV.ACCP.LOAD)
```

De noodzakelijke autorisaties voor het BINDen van het applicatieplan:

```
SETROPTS  CLASSACT(MDSNPN)

RDEFINE   MDSNPN  DB2P.PLAN84.BIND
                  OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Production Binder groep *DB2APL3* autorisatie heeft om het plan, na bijvoorbeeld een aanpassing van het bij behorende programma, kan binden:

```
PERMIT    DB2P.PLAN84.BIND  CLASS(MDSNPN)
                  ID(DB2APL3)  ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd plan door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNPN een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE   MDSNPN  **  OWNER(SECADM)  UACC(NONE)
```

Een plan kan ook worden opgebouwd uit een vooraf geBIND packages die dan met een package-list in het applicatieplan kunnen worden geregistreerd. Voor het registreren van packages in een applicatieplan moet men BIND autorisatie op het applicatieplan hebben, zoals hiervoor beschreven en execute autorisatie op de betreffende packages. Uitsluitend een Production Binder, verantwoordelijk voor het beheer van de applicatieplannen mag EXECUTE autorisatie op de betreffende packages hebben. Een voorbeeld voor het registreren van packages in een applicatieplan:

```
DSN  BIND  PLAN(PLAN92)  OWNER(DB2PLAN)  PKLIST(COLL77.PACK69)

SETROPTS CLASSACT(MDSNPK)

RDEFINE   MDSNPK  DB2P.COLL77.PACK69.EXECUTE
                  OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Production Binder groep *DB2APL3* autorisatie heeft om het package in het applicatieplan te binden:

```
PERMIT    DB2P.COLL77.PACK69.EXECUTE  CLASS(MDSNPK)
                  ID(DB2APL3)  ACCESS(READ)
```

Remote Package
Indien een programma een remote-package voor het applicatieplan gebruikt moet op het lokale systeem de BIND worden geautoriseerd. Uitsluitend een Production Binder, verantwoordelijk voor het beheer van de applicatieplannen mag autorisatie voor het betreffende applicatieplan hebben. Autorisaties voor de betreffende package wordt op uitvoeringstijd gecontroleerd. Een voorbeeld van een remote-package BINDing:

```
DSN  BIND  PLAN(PLAN95)  OWNER(DB2POWN)
                          PKLIST(RDAM.COLL77.PACK67)

RDEFINE   MDSNPN  DB2P.PLAN95.BIND
                          OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Production Binder groep *DB2APL3* autorisatie heeft om het plan kan binden:

```
PERMIT    DB2P.PLAN95.BIND  CLASS(MDSNPN)
                          ID(DB2APL3)  ACCESS(READ)
```

22.12.3 BIND Package

Een ‘application package’ of ‘package’ is een set uitvoerbare SQL-statements. Een package wordt samengesteld door middel van het BIND (commando). Voor het uitvoeren van een package moet tevens een plan worden gegenereerd waarin het toegangspad en de toegangsautorisaties voor de betreffende SQL-statements worden gedefinieerd. Een plan wordt door het BIND PLAN-commando zoals hiervoor beschreven uitgevoerd. Voor het apart genereren van packages uit DBRMs wordt het BIND PACKAGE commando gebruikt. Als het package reeds in een package-list van een applicatieplan was opgenomen hoeft het applicatieplan niet te worden reBIND. Bij het veranderen van een package zonder autorisatie op het applicatieplan kan de programmalogica anders functioneren. Het BINDen van de packages moet daarom worden geautoriseerd. Uitsluitend een Production Binder, verantwoordelijk voor de verificatie van packages, mag een BIND PACKAGE uitvoeren. Een voorbeeld voor het BINDen (inclusief BIND, REBIND, FREE en DROP) van een package:

```
DSN  BIND  PACKAGE(COLL77)  OWNER(DB2POWN)  MEMBER(PACK69)
                            LIRARY(DB2DEV.ACCP.LOAD)
```

De noodzakelijke autorisaties voor het BINDen van een package:

```
RDEFINE   MDSNPK  DB2P.COLL77.PACK69.BIND
                          OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Database administrator groep *DB2APL4* autorisatie heeft om een package te gebruiken voor de BIND PACKAGE:

```
PERMIT    DB2P.COLL77.PACK69.BIND  CLASS(MDSNPK)
                          ID(DB2APL4)  ACCESS(READ)
```

Een voorbeeld autorisatie voor het BINDen van meerdere packages uit een collection:

```
RDEFINE   MDSNPK  DB2P.COLL77.PACK*.BIND
                          OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Database administrators groep *DB2APL3* autorisatie heeft om een package te binden:

```
PERMIT    DB2P.COLL77.PACK*.BIND  CLASS(MDSNPK)
                          ID(DB2APL3)  ACCESS(READ)
```

Een voorbeeld autorisatie voor het BINDen van packages uit alle collections die met dezelfde 4 karakters (*PACK*) beginnen:

```
RDEFINE    MDSNPK   DB2P.*.PACK*.BIND
                        OWNER(SECADM)   UACC(NONE)

    Een voorbeeld hoe de Production Binder groep DB2APL4 autorisatie heeft
    om een package te binden:

PERMIT     DB2P.*.PACK*.BIND   CLASS(MDSNPK)
                        ID(DB2APL4)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd package door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNPK een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNPK   **    OWNER(SECADM)   UACC(NONE)
```

22.12.4 Collection

Een collection is een groep packages die dezelfde qualifier hebben. Door het groeperen van packages in een collectie kan specifieke collection-autorisatie worden gegeven voor het gebruik en beheer van packages. Voor het kunnen gebruiken van een collection-naam bij het BIND PACKAGE commando moet Package Administration voor deze collectie worden geautoriseerd. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNCL)

RDEFINE    MDSNCL   DB2P.COLL77.CREATEIN
                        OWNER(SECADM)   UACC(NONE)

Een voorbeeld hoe de Package Administration groep DB2APL4 autorisatie heeft
om packages uit een collectie te binden:

PERMIT          DB2P.COLL77.CREATEIN   CLASS(MDSNCL)
                        ID(DB2APL4)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd collection door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNCL een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNCL   **    OWNER(SECADM)   UACC(NONE)
```

Uitsluitend een Package Administration, verantwoordelijk voor het beheer van collection(s), mag voor het beheer voor één of meerdere collections geautoriseerd zijn. Een voorbeeld:

```
RDEFINE    DSNADM   DB2P.COLL88.PACKADM
                        OWNER(SECADM)   UACC(NONE)

PERMIT     DB2P.COLL88.PACKADM   CLASS(DSNADM)
                        ID(DB2APL55)   ACCESS(READ)
```

22.12.5 Schema

Een schema is een logische groep user-defined functies, distinct types, triggers and stored procedures. Om deze objecten te kunnen gebruiken moeten deze in een schema zijn opgenomen (gedefinieerd). Uitsluitend een Package Administration, verantwoordelijk voor schema's en de

gerelateerde objecten, mag voor het toevoegen van deze objecten in schema worden geautoriseerd. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNSC)

RDEFINE    MDSNSC   DB2P.SCHEMA12.CREATEIN
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de Package Administration groep *DB2APL77* autorisatie heeft om in het schema functies te kunnen opnemen:

```
PERMIT          DB2P.SCHEMA12.CREATEIN   CLASS(MDSNSC)
                        ID(DB2APL77)   ACCESS(READ)
```

Andere autorisatie mogelijkheden voor schemabeheer zijn:

```
RDEFINE    MDSNSC   DB2P.SCHEMA12.STORP14.DROPIN
                      OWNER(SECADM)   UACC(NONE)
                                        Drop a Stored Procedures

RDEFINE    MDSNSC   DB2P.SCHEMA12.STORP14.ALTERIN
                       OWNER(SECADM)   UACC(NONE)
                                        Alter or Comment ON a Stored
                                        Procedures
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd schema door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNSC een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNSC   **    OWNER(SECADM)   UACC(NONE)
```

22.12.6 Stored Procedures

Stored procedures (routine package) zijn over het algemeen installatie eigen programma's. Het geeft de mogelijkheid om een set SQL-statements door middel van één aanroep (CALL) uit te voeren. Door het gebruik van stored procedures wordt vermeden dat er meerdere SQL-aanroepen achtereenvolgens over een netwerk moeten waardoor gecommuniceerd. Door het gebruikt van stored procedures wordt daarom veel communicatie overhead vermeden. Voor het beheer van stored procedures is het niet nodig om andere autorisaties te hebben dan aangegeven in het hiervoor beschreven beheer van schema's. Wel dient autorisatie te worden gegeven voor het activeren en deactiveren van de stored procedures. Een voorbeeld om alle procedures te activeren:

```
-START      PROCEDURE
```

Een voorbeeld om 2 specifieke stored procedures te activeren:

```
-START      PROCEDURE(SCHEMA5.SPROC35,SCHEMA5.SPROC36)
```

Het gebruik van het operator commando START, STOP PROCEDURE mag uitsluitend worden geautoriseerd voor Systems Operations. Een voorbeeld:

```
RDEFINE    DSNADM   DB2P.SYSOPR
 OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de System Operation groep *DB2OPR9* voor het activeren/deactiveren (START/STOP) van alle stored procedures wordt geautoriseerd:

```
PERMIT          DB2P.SYSOPR   CLASS(DSNADM)
```

ID(***DB2OPR9***) ACCESS(**READ**)

Een voorbeelden hoe de System Operation groep *DB2OPR9* voor het displayen (DISPLAY) van alle stored procedures van schema5 wordt geautoriseerd:

SETROPTS CLASSACT(**MDSNSP**)

RDEFINE MDSNSP *DB2P.SCHEMA5.*.***DISPLAY**
OWNER(*SECADM*) UACC(NONE)

PERMIT *DB2P.SCHEMA5.*.***DISPLAY** CLASS(MDSNSP)
ID(***DB2OPR9***) ACCESS(**READ**)

Om te voorkomen dat een niet specifiek in RACF gedefinieerd stored procedures door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNSP een sluitprofiel worden gedefinieerd. De implementatie:

RDEFINE MDSNSP ****** OWNER(*SECADM*) UACC(NONE)

DB2-established address space
Voor het uitvoeren van stored procedures moet de (DB2-established) address space waarin deze worden uitgevoerd van de Call Attachment Facility (CAF) gebruik kunnen maken om DB2 te kunnen benaderen. Zie voor verdere informatie paragraaf ‘DB2 Attachment Facilities’. De implementatie:

PERMIT *DB2P.***BATCH** CLASS(DSNR) ID(*SDB2PSPR*) ACCESS(**READ**)

Om te voorkomen dat een niet specifiek in RACF gedefinieerde caller door DB2 wordt geautoriseerd moet in de RACF general resource class DSNR een sluitprofiel worden gedefinieerd. De implementatie:

RDEFINE DSNR ****** OWNER(*SECADM*) UACC(NONE)

WLM address space
Voor het optimaal gebruik maken van de z/OS-resources bij de uitvoering van stored procedures kan de Workload Manager (WLM) worden toegepast. Hiervoor moet de address space die door WLM beheerd moet worden als een server worden gedefinieerd. De implementatie:

RDEFINE **SERVER** DB2.*DB2P.TELEFOON*
OWNER(*SECADM*) UACC(NONE)

PERMIT DB2.*DB2P.TELEFOON* CLASS(SERVER)
ID(***SDB2PWLM***) ACCESS(**READ**)

Voor het uitvoeren van stored procedures moet de WLM address space, waarin deze worden uitgevoerd, van de RRS Attachment Facility (RRSAF) gebruik kunnen maken om DB2 te kunnen benaderen. Zie voor verdere informatie paragraaf ‘DB2 Attachment Facilities’. De implementatie:

PERMIT *DB2P.***RRSAF** CLASS(DSNR) ID(*SDB2PWLM*)
ACCESS(**READ**)

Voor het definiëren van een stored procedures of installatie gedefinieerde functies (UDF) in een WLM-omgeving moet de Database administrator, verantwoordelijk voor deze procedures en functies worden geautoriseerd. WLM kan zo de verschillende soorten applicatieomgevingen gescheiden houden zodat beveiligingsgevoelige applicaties in een geïsoleerde omgeving worden uitgevoerd. Een voorbeeld:

RDEFINE DSNR *DB2PSYSP.***WLMENV.*SALESFI***
OWNER(*SECADM*) UACC(NONE)

Een voorbeeld hoe de Package Administration groep *DB2APL7* voor het beheer van stored procedures en installatie gedefinieerde functies in de WLM-omgeving *SALESFI* wordt geautoriseerd:

```
PERMIT          DB2PSYSP.WLMENV.SALESFI  CLASS(DSNR)
                                         ID(DB2APL7)  ACCESS(READ)
```

Toegang tot non-DB2 resources
Indien de stored procedure toegang moet hebben tot IMS, CICS of MVS/APPC transacties of VSAM-files moeten deze worden geautoriseerd. Bij het gebruik van een stored procedures (DB2-established) address space moet het userid van de address space, in het voorbeeld *SDB2PSPR* worden geautoriseerd. Het gebruik van een stored procedures (DB2-established) address space voor het benaderen van niet-DB2-middelen is daarom niet toegestaan.

Het gebruik van een stored procedures in een WLM address space voor het benaderen van niet-DB2-middelen is echter wel toegestaan mits de autorisatie met het userid van de aanroeper wordt uitgevoerd. De stored procedure moet dan als volgt worden gedefinieerd:

```
CREATE    PROCEDURE  H    WLM ENVIRONMENT SALESFI    SECURITY USER
```

22.12.7 User defined functions

Een installatie gedefinieerde functie (user defined function (UDF) of routine package) is, in tegenstelling tot de standaard DB2 built-in functies, een installatie gedefinieerde functie die in elk SQL-statement kan worden toegepast. Deze functies kunnen worden gecategoriseerd als user-defined scalar functions (functie response is één waarde) of als een user-defined table functions (functie response is een tabel). Een installatie gedefinieerde functie kan zijn een:

- external function, en kan een scalar of table function zijn die gemaakt is met een voor het platform gebruikelijke programmeertaal;
- sourced function, en kan uitsluitend een scalar function zijn gebaseerd op bestaande built-in functies of andere installatie gedefinieerde functies (UDFs);
- SQL-function, en kan uitsluitend een scalar function zijn bestaande uit source-code die opgenomen is in de UDF-definitie.

Installatie gedefinieerde functies moeten worden geactiveerd om in SQL-statements te kunnen worden gebruikt. DB2 activeert een nieuwe functie automatisch als deze door een SQL-statement wordt gerefereerd. Als een functie door een operator is gedeactiveerd (STOPed) kan deze door het DB2 START FUNCTION operator commando worden gereactiveerd. Een voorbeeld om alle functies te activeren:

```
-START      FUNCTION   SPECIFIC
```

Een voorbeeld om 2 specifieke functies te activeren:

```
-START      FUNCTION   SPECIFIC(SCHEMA5.UFUNC55,SCHEMA5.UFUNC56)
```

Het gebruik van het operator commando START, STOP en DISPLAY FUNCTION mag uitsluitend worden geautoriseerd voor Systems Operations. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNUF)

RDEFINE    MDSNUF   DB2P.SCHEMA5.*.START
                        OWNER(SECADM)  UACC(NONE)
```

Een voorbeeld hoe de Systems Operation groep *DB2OPR9* voor het activeren van alle user-functies uit schema *SCHEMA5* wordt geautoriseerd:

```
PERMIT   DB2P.SCHEMA5.*.START   CLASS(MDSNUF)
                                ID(DB2OPR9)   ACCESS(READ)
```

De andere voorbeelden van de operator commando's voor functies zijn:

```
RDEFINE   MDSNUF   DB2P.SCHEMA5.*.STOP
                               OWNER(SECADM)   UACC(NONE)

RDEFINE   MDSNUF   DB2P.SCHEMA5.*.DISPLAY
                               OWNER(SECADM)   UACC(NONE)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd user defined functions door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNUF een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE   MDSNUF   **   OWNER(SECADM)   UACC(NONE)
```

22.12.8 User defined distinct type

Een installatie gedefinieerde data type (user-defined distinct type (UDT)) is een standaard gedefinieerd data type op een, voor de gebruiker, andere manier gerepresenteerd formaat. Deze vertaling, bijvoorbeeld een andere vorm van dag/maand/jaar representatie, kan door de installatie worden gedefinieerd. Voor het beheer van installatie gedefinieerde data typen is het niet nodig om andere autorisaties te hebben dan aangegeven in het hiervoor beschreven beheer van schema's.

22.12.9 Trigger

Een trigger is een set SQL-statements in een DB2-database die wordt uitgevoerd als een vooraf bepaalde gebeurtenis plaatsvindt in een DB2-tabel. Voor het creëren van een trigger is TRIGGER-autorisatie op de tabel noodzakelijk. Uitsluitend Package Administration mag een trigger definiëren. Een voorbeeld:

Voor het definiëren van een trigger op een tabel (*TBP29*):

```
RDEFINE   MDSNTB   DB2P.DB2TAB.TBP29.TRIGGER
                               OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de Package Administration groep *DB2DBA44* een trigger op tabel (*TBP29*) kan definiëren:

```
PERMIT    DB2P.DB2TAB.TBP29.TRIGGER   CLASS(MDSNTB)
                                ID(DB2DBA44)   ACCESS(READ)
```

22.13 Expliciete autorisaties voor gegevensbenadering

22.13.1 Plan

Voor het uitvoeren van een applicatieprogramma (CICS, IMS of batch) waarbij statische SQL-statements worden gebruikt moet de gebruiker uitsluitend voor het betreffende applicatieplan worden geautoriseerd. De eigenaar (BIND OWNER-parameter) van het applicatieplan moet geautoriseerd zijn om al de SQL-statements uit te voeren, dus niet de gebruiker van het

programma/applicatieplan. Uitsluitend in RACF geautoriseerde eindgebruikers (Program End Users) mogen een plan kunnen uitvoeren. Een voorbeeld:

```
RDEFINE    MDSNPN   DB2P.PLAN94.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *SALES* autorisatie heeft om het plan en daardoor de bijbehorende applicatie te kunnen uitvoeren:

```
PERMIT     DB2P.PLAN94.EXECUTE   CLASS(MDSNPN)
                        ID(SALES)   ACCESS(READ)
```

22.13.2 Package

Indien in een applicatieplan een remote-package geregistreerd is moet de remote-package voor uitvoering van de SQL-statements op het remote-systeem zijn geautoriseerd. Als tussen het lokale- en remote-systeem het DB2-private communicatieprotocol wordt toegepast moet de eigenaar van het applicatieplan voor de betreffende remote-package worden geautoriseerd. Bij het toepassen van het DRDA- communicatieprotocol moet de uitvoerder (Program End Users) van het plan/package voor de betreffende remote-package worden geautoriseerd.

Een voorbeeld voor het DB2-private communicatieprotocol:

```
DSN  BIND  PLAN(PLAN95)  PKLIST(RDAM.COLL77.PACK67)
                        OWNER(DB2AMS)
```

Op het lokale systeem *AMS* met de eindgebruikers:

```
RDEFINE    MDSNPN   DB2P.PLAN95.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *CRUISE64* autorisatie heeft om de package en daardoor de bijbehorende SQL-statements te kunnen uitvoeren:

```
PERMIT     DB2P.PLAN95.EXECUTE   CLASS(MDSNPN)
                        ID(CRUISE64)   ACCESS(READ)
```

Op het remote systeem *RDAM* waar het package zich bevindt:

```
RDEFINE    MDSNPK   DB2P.COLL77.PACK67.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eigenaar (*DB2AMS*) van *PLAN95* voor de package op het remote systeem moet worden geautoriseerd:

```
PERMIT     DB2P.COLL77.PACK67.EXECUTE   CLASS(MDSNPK)
                        ID(DB2AMS)   ACCESS(READ)
```

Een voorbeeld voor het generieke DRDA-communicatieprotocol:

```
DSN        BIND  PLAN(PLAN95)   PKLIST(RDAM.COLL77.PACK67)
           OWNER(DB2AMS)
```

Op het lokale systeem (*AMS*) met de eindgebruikers:

```
RDEFINE    MDSNPN   DB2P.PLAN95.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *CRUISE64* autorisatie heeft om de package en daardoor de bijbehorende SQL-statements te kunnen uitvoeren:

PERMIT *DB2 DB2P.PLAN95.***EXECUTE** CLASS(MDSNPN)
ID(*CRUISE64*) ACCESS(**READ**)

Op het remote systeem (*RDAM*) waar het package zich bevindt:

RDEFINE MDSNPK *DB2P.COLL77.PACK67.***EXECUTE**
OWNER(*SECADM*) UACC(NONE)

Een voorbeeld hoe de eigenaar *CRUISE64* voor het de package *COLL77.PACK67* op het remote systeem *RDAM* moet worden geautoriseerd:

PERMIT *DB2P.COLL77.PACK67.***EXECUTE** CLASS(MDSNPK)
ID(***CRUISE64***) ACCESS(**READ**)

22.13.3 Intermediate systemen

Als via een ander systeem, een zogenaamde HOP, een remote DB2-database wordt benaderd, dan kan de autorisatie op deze HOP op twee manieren worden geverifieerd. Indien er sprake is van ‘cliënt (untrusted)’-‘Trusted HOP’-‘Trusted Target systeem’ relatie moet op de HOP-side de authenticatie van de aanvrager worden gecontroleerd. De gebruiker moet worden geautoriseerd voor het uitvoeren van statische en dynamische SQL-statements. De implementatie voor het geven van deze instructie van het Trusted Target systeem naar de HOP:

AUTH AT HOP SITE ===> ***RUNNER***

22.13.4 Dynamische SQL-statements

Voor het uitvoeren van dynamische SQL-statements, waarbij geen vooraf gedefinieerd applicatieplan of package aanwezig is, moet de gebruiker specifiek voor het benaderen van de tabelgegevens worden geautoriseerd. In de paragraaf ‘Table, index en view’ zijn hiervan voorbeelden opgenomen.

Voor autorisatiecontrole bij dynamische SQL wordt het SQL-ID gebruikt die:

- gelijk is aan het primaire autorisatie ID (RACF userid) als het SQL-statement SET CURRENT SQLID(USER) wordt gegeven (is de default);
- gelijk is aan een secondary autorisatie ID (RACF groepid) als bijvoorbeeld het SQL-statement SET CURRENT SQLID(*SALES*) wordt gegeven. De gebruiker kan dit alleen uitvoeren als dit een voor de gebruiker valide connect-groep is;
- door de connectie exit of sign-on exit wordt gezet, maar dit is niet in deze standaard toegestaan;
- in het SQL-statement SET CURRENT SQLID(een host variabele of een constante wordt gedefinieerd). De vertaling van host variabelen en constanten kan in de BIND parameter NOREOPT(VARS) worden voorkomen of toegestaan;
- door een gebruiker met SYSADM met het SQL-statement SET CURRENT SQLID(*SALES*) wordt gegeven.

22.13.5 Stored Procedures

Voor het kunnen uitvoeren van een stored procedure moet de aanroeper EXECUTE-autorisatie op de procedure hebben. Uitsluitend geautoriseerde gebruikers (Program End Users) mogen een stored procedure uitvoeren. Een voorbeeld:

```
RDEFINE    MDSNSP   DB2P.DB2SCH43.PROC49.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *SALES* autorisatie heeft om het plan en daardoor de bijbehorende applicatie te kunnen uitvoeren:

```
PERMIT     DB2P.DB2SCH43.PROC49.EXECUTE   CLASS(MDSNSP)
                        ID(SALES)   ACCESS(READ)
```

22.13.6 User defined functions

Het gebruik van een installatie gedefinieerde functie (user defined function (UDF) of routine package) moet worden geautoriseerd. Uitsluitend geautoriseerde gebruikers (Program End Users) mogen deze uitvoeren. Een voorbeeld:

```
RDEFINE    MDSNUF   DB2P.SCHEMA5.UFUNC55.EXECUTE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *CRUISE64* voor de user-functie *UFUNC55* uit schema *SCHEMA5* wordt geautoriseerd:

```
PERMIT     DB2P.SCHEMA5.UFUNC55.EXECUTE   CLASS(MDSNUF)
                        ID(CRUISE64)   ACCESS(READ)
```

22.13.7 User distinct type

Het gebruik van een installatie gedefinieerde data type (user-defined distinct type (UDT)) moet worden geautoriseerd. Uitsluitend geautoriseerde gebruikers (Program End Users) mogen deze uitvoeren. Een voorbeeld:

```
SETROPTS CLASSACT(MDSNUT)

RDEFINE    MDSNUT   DB2P.SCHEMA5.UTYPE35.USAGE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de groep *CRUISE64* voor de user-functie *UTYPE35* uit schema *SCHEMA5* wordt geautoriseerd:

```
PERMIT     DB2P.SCHEMA5.UTYPE35.USAGE   CLASS(MDSNUT)
                        ID(CRUISE64)   ACCESS(READ)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerd user distinct type door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNUT een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNUT   **    OWNER(SECADM)   UACC(NONE)
```

22.13.8 Table, index en view

Uitsluitend in RACF geautoriseerde eindgebruikers (Program End Users) mogen de gegevens in een tabel kunnen benaderen. De eindgebruiker mogen niet meer autorisaties op de tabel (en kolommen) hebben dan die noodzakelijk zijn voor de uitvoering van zijn/haar rol/functie. De eindgebruikers worden altijd voor een tabel en kolommen geautoriseerd, voor indexen en views is dit niet anders alleen zullen de tabellen en kolommen via een toegangspad worden benaderd.

Zoals aangegeven in de paragraaf ‘BIND’ moet de eigenaar (OWNER) van een plan/package geautoriseerd zijn voor alle objecten die de SQL-statements in het plan/package refereren. Voor interactieve en ODBC dynamische SQL-statements moet juist de individuele gebruiker (process

runner, requester) voor de betreffende tabellen, kolommen, views, etc. worden geautoriseerd. Een aantal voorbeelden:

Voor het benaderen van een tabel (TBP29):

```
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.SELECT
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep VERKOOP een tabel (TBP29) kan benaderen:

```
PERMIT     DB2P.DB2TAB.TBP29.SELECT   CLASS(MDSNTB)
                        ID(VERKOOP)   ACCESS(READ)
```

Voor het aanpassen van een tabel (TBP29):

```
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.UPDATE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de Plan-eigenaar *DB2PLE34* de gegevens in tabel TBP29 kan aanpassen:

```
PERMIT     DB2P.DB2TAB.TBP29.UPDATE   CLASS(MDSNTB)
                        ID(DB2PLE34)   ACCESS(READ)
```

Voor het aanpassen van een kolom in een tabel:

```
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.COLNAW.UPDATE
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep VERKOOP de gegevens uitsluitend van één kolom (COLNAW) in een tabel (TBP29) kan aanpassen:

```
PERMIT     DB2P.DB2TAB.TBP29.COLNAW.UPDATE   CLASS(MDSNTB)
                        ID(VERKOOP)   ACCESS(READ)
```

Andere eindgebruikersautorisatie mogelijkheden zijn:

```
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.INSERT
                        OWNER(SECADM)   UACC(NONE)
RDEFINE    MDSNTB   DB2P.DB2TAB.TBP29.DELETE
                        OWNER(SECADM)   UACC(NONE)
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde tabel door DB2 wordt geautoriseerd moet in de RACF general resource class MDSNTB een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    MDSNTB   **    OWNER(SECADM)   UACC(NONE)
```

Het autoriseren van tabelgegevens via een index of een view wordt op dezelfde manier gedefinieerd als bij het benaderen van een tabel. Uitsluitend geautoriseerde eindgebruikers (Program End Users) mogen een index of view kunnen benaderen. Een voorbeeld:

```
RDEFINE    MDSNTB   DB2P. DB2TAB.INDX86.SELECT
                        OWNER(SECADM)   UACC(NONE)
```

Een voorbeeld hoe de eindgebruikersgroep *DB2ABC5* een index (*INDX86*) kan benaderen:

```
PERMIT     DB2P. DB2TAB.INDX86.SELECT   CLASS(MDSNTB)
```

```
                                    ID(DB2ABC5)  ACCESS(READ)
```

22.13.9 Platform autorisatie

Indien bepaalde informatie over meerdere DB2-subsystemen dezelfde autorisaties moet hebben kan een DB2-subsysteem overstijgend RACF profiel worden gedefinieerd. Een voorbeeld van een generiek RACF-profiel dat toegang geeft tot een tabel ongeacht in welk DB2-subsysteem deze zich bevindt:

```
RDEFINE    MDSNTB   *.TELEPHON.SELECT
                        OWNER(SECADM)  UACC(NONE)
```

Opmerking: het gebruik van een RACF-variabele, te definiëren in de general resource class RACFVARS, voor de qualifiers is een andere mogelijkheid om een meer generiek profiel te definiëren.

22.14 DB2 commando's

DB2-commando’s gegeven vanaf het MVS-console hebben de bevoegdheid van SYSOPR. Deze paragraaf moet nog worden ontwikkeld.

22.15 DB2 Attachment Facilities

DB2 is een relationele database manager waarin gebruikersgegevens kunnen worden opgeslagen. Het aanroepen van DB2 moet als eerste worden gecontroleerd zodat niet elke willekeurige applicatie of subsysteem DB2 kan aanroepen en eventueel de werking zou kunnen beïnvloeden. Het aanroepen van DB2 wordt gecontroleerd door de RACF general resource class DSNR. Een voorbeeld hoe achtereenvolgend een selectie van DB2-aanroepers IMS (inclusief MPP, BMP, Fast Path en DL/I batch), CICS, DDF, RRSAF en BATCH (inclusief TSO, CAF, batch en DB2 utility jobs) worden geautoriseerd:

```
SETROPTS CLASSACT(DSNR)

RDEFINE    DSNR   (DB2P.MASS)   OWNER(SECADM)  UACC(NONE)
RDEFINE    DSNR   (DB2P.SASS)   OWNER(SECADM)  UACC(NONE)
RDEFINE    DSNR   (DB2P.DIST)   OWNER(SECADM)  UACC(NONE)
RDEFINE    DSNR   (DB2P.RRSAF)  OWNER(SECADM)  UACC(NONE)
RDEFINE    DSNR   (DB2P.BATCH)  OWNER(SECADM)  UACC(NONE)
```

Het gebruik van de ENABLE-parameter in het BIND-commando, waarmee het gebruik door andere subsystemen van applicatieplannen en packages wordt geautoriseerd, is niet toegestaan. Uitsluitend de RACF general resource class DSNR mag hiervoor worden gebruikt. De implementatie:

```
DSN  BIND  DISABLE
```

Om te voorkomen dat een niet specifiek in RACF gedefinieerde privilege door DB2 wordt geautoriseerd moet in de RACF general resource class DSNR een sluitprofiel worden gedefinieerd. De implementatie:

```
RDEFINE    DSNR   **   OWNER(SECADM)  UACC(NONE)
```

Opmerking: de DB2 attachment stelt vast welk applicatieplan bij een bepaald programma hoort. Voor het uitvoeren van een applicatie voert DB2 autorisatie controle uit bij het creëren van een

DB2-thread. Indien één thread voor meerdere transacties wordt gebruikt geeft het DB2-attachment voor autorisatie controle voor elke transactie een SIGNON-command naar DB2.

22.15.1 IMS

Voor IMS is een DB2 attachment facility aanwezig die via het MVS subsystem interface (SSI) wordt aangeroepen en voor het uitvoeren van database calls met DB2 via Cross Memory (XM) communiceert. Een IMS-operator kan met het IMS-commando IDENTIFY een DB2-connectie aanvragen. Het gebruik van een DB2-connectie door IMS moet worden geautoriseerd. Een voorbeeld:

```
PERMIT    DB2P.MASS    CLASS(DSNR)  ID(SIMSCNTL)  ACCESS(READ)
```

Opmerking: voor autorisatie van DB2-objecten wordt door DB2 het IMS-eindgebruikers userid welke door IMS samen met de transactieaanvraag aan DB2 wordt meegestuurd.

22.15.2 CICS

Voor CICS is een DB2 attachment facility aanwezig die via het MVS subsystem interface (SSI) wordt aangeroepen en voor het uitvoeren van database calls met DB2 via Cross Memory (XM) communiceert. Een CICS-operator kan met het CICS-commando IDENTIFY een DB2-connectie aanvragen. Het gebruik van een DB2-connectie door CICS moet worden geautoriseerd. Een voorbeeld:

```
PERMIT    DB2P.SASS    CLASS(DSNR)  ID(SCICS16)    ACCESS(READ)
```

Autorisatie wordt uitgevoerd met het CICS eindgebruikers userid of het CICS started task userid of het userid is de CICS-transactienaam. Dit wordt aangegeven in Resource Control Table (RCT) van de betreffende CICS met de AUTH-parameter. De implementatie waarbij het userid van de CICS-eindgebruikers voor DB2-autorisatie wordt gebruikt:

```
AUTH=USERID
```

Indien de betreffende CICS een trusted status heeft kan worden overwogen om niet het userid van de eindgebruikers voor DB2-autorisatie controle te gebruiken maar het CICS started task userid. De implementatie:

```
AUTH=SIGNID
```

Voor DB2-autorisatiecontrole is het gebruik niet toegestaan van een:

- vast gedefinieerd userid, AUTH=*XKRENT01;*
- CICS-transactiecode als userid AUTH=TXID;
- groepid, AUTH=GROUP;
- operator-id, AUTH=USER;
- terminal-id, AUTH=TERM.

22.15.3 TSO/DB2I/SPUFI/Batch

Onder TSO in de foreground of background kan DB2 met de TSO attachment facility worden aangeroepen. De TSO met de TSO attachment facility kan verder voor de volgende functies worden toegepast:

- voor het uitvoeren van het DB2/TSO-command DSN;

- als vehikel voor het DB2I-panel interface, hoofdzakelijk voor het beheer van het DB2-subsysteem en DB2-objecten;
- als vehikel voor het SPUFI-interface, voor het interactief uitvoeren van SQL-statements, de echte dynamische SQL;
- voor het uitvoeren van Batch-programma's.

Voor het uitvoeren van SQL-statements, zonder dat daarvoor een applicatieprogramma voor moet worden geschreven, kan de DB2I-gebruiker de SPUFI facility gebruiken. SQL-statements kunnen dan direct en interactief worden uitgevoerd. Het gebruik van de TSO attachment facility moet worden geautoriseerd. Uitsluitend geautoriseerde eindgebruikers (Query Users) mogen de TSO attachment facility gebruiken. Een voorbeeld:

PERMIT *DB2P*.**BATCH** CLASS(DSNR) ID(*VERKOOP*) ACCESS(**READ**)

22.15.4 CAF (Call Attachment Facility)

De Call Attachment Facility (CAF) is een DB2 attachment facility voor TSO (foreground en background) Batch programma's en voor de stored procedure address space. De CAF wordt via het MVS subsystem interface (SSI) aangeroepen en voor het uitvoeren van database calls wordt met DB2 via Cross Memory (XM) communiceert. De CAF is een alternatief voor het DB2 TSO-commando DSN en heeft voor de uitvoering van SQL-statements meer controle mogelijkheden. Het gebruik van de Call Attachment Facility moet worden geautoriseerd. Uitsluitend geautoriseerde eindgebruikers (Query Users) mogen de TSO attachment facility gebruiken. Een voorbeeld:

PERMIT *DB2P*.**BATCH** CLASS(DSNR) ID(*VERKOOP*) ACCESS(**READ**)

22.15.5 RRSAF (Recoverable Resource Manager Services Attachment Facility)

De DB2 Recoverable Resource Manager Services Attachment Facility (RRSAF) maakt gebruik van z/OS Transaction Management en Recoverable Resource Manager Services (RRS). RRS is ontwikkeld voor systeembrede coordinatie van two-phase commit operaties. De RRSAF kan worden toegepast voor het aanroepen van DB2 door alle z/OS Resource Managers die gebruikt maken van RRS. De met RRSAF uitgevoerde database calls zijn dynamische SQL-statements. Binnen DB2 is voor RRSAF geen applicatieplan aanwezig, de autorisaties worden daarom op runtime gecontroleerd. Het gebruik van RRSAF moet worden geautoriseerd. Uitsluitend geautoriseerde eindgebruikers (Query Users) mogen de TSO attachment facility gebruiken. Een voorbeeld:

PERMIT *DB2P*.**RRSAF** CLASS(DSNR) ID(*PRODUCT*) ACCESS(**READ**)

22.15.6 DDF (Distributed Data Facility)

De Distributed Data Facility (DDF) is een DB2-faciliteit op z/OS waarmee vanaf een remote site met DB2 kan worden gecommuniceerd. De remote site kan elk willekeurig database systeem zijn mits deze het Distributed Relational Database Architecture (DRDA) protocol ondersteund. Uitsluitend geautoriseerde eindgebruikers (Query Users) mogen DDF gebruiken. Een voorbeeld hoe een remote-gebruiker DDF kan gebruiken:

PERMIT *DB2P*.DIST CLASS(DSNR) ID(*VERKOOP*) ACCESS(**READ**)

Voorbeeld voor koppeling aan het SNA-netwerk *RDAM*:

PERMIT *DB2P*.DIST CLASS(DSNR) ID(*VERKOOP*) ACCESS(**READ**)
WHEN(APPCPORT(*RDAM*))

Voorbeeld voor koppeling aan het TCP/IP-netwerk:

PERMIT *DB2P*.DIST CLASS(DSNR) ID(*VERKOOP*) ACCESS(**READ**)
WHEN(APPCPORT(**TCPIP**))

22.16 DB2 Connect

DB2 Connect is een cliënt implementatie voor het opzetten van een verbinding met bijvoorbeeld DB2 op z/OS waarna de eindgebruiker database queries kan uitvoeren. Voor autorisaties op database-objecten gelden dezelfde maatregelen als beschreven voor dynamic SQL. DB2 Connect wordt onder andere gebruikt voor het kunnen veranderen van het wachtwoord. De mogelijkheid voor het veranderen van het gebruikers wachtwoord moet tijdens DB2-installatie op het installatiepanel DSNTIPR worden aangegeven. De implementatie:

EXTENDED SECURITY ===> **YES**

22.17 Netwerk control

DB2 heeft verschillende mogelijkheden voor identificatie en authenticatie vanuit het netwerk. Deze zijn:

- alleen het userid. Dit is alleen toegestaan als de aanvrager op een vertrouwd (trusted) systeem is aangelogd en deze identificatie en authenticatie van de gebruiker heeft uitgevoerd en er geen mogelijkheid voor andere (sub)systemen zijn, die geen verificatie hebben uitgevoerd, het systeem te benaderen;
- userid en wachtwoord in clear tekst, is niet toegestaan;
- userid en Passticket. Zie hiervoor de beschrijving in hoofdstuk ‘Passtickets’, wordt aanbevolen tussen DB2s op z/OS systems;
- Kerberos tickets. Zie hiervoor de beschrijving in hoofdstuk ‘Kerberos’, wordt aanbevolen met Windows 2000;
- DRDA password encryption support en Database Connection Services (DCS), ondersteund door DB2 Connect, voor:
  - Unencrypted userid en encrypted wachtwoord;
  - Encrypted userid en encrypted wachtwoord;
  - Userid, oude wachtwoord en nieuwe wachtwoord eventueel encrypted.

22.17.1 SNA

Zie voor volledige beschrijving van SNA-beveiliging het hoofdstuk ‘Virtual Telecommunication Access Method (VTAM)’.

Aangezien DB2 een netwerkapplicatie is moet de DB2-applicatie met de RACF general resource class APPL worden beschermd. Een voorbeeld waarbij DB2 door VTAM/RACF wordt beschermd:

De naam die DB2 gebruikt om zich aan VTAM bekend te maken wordt in installatiepanel DSNTIPR opgegeven. Een voorbeeld:

DB2 NETWORK LUNAME ===> ***ADAMDB2P***

De RACF profielen:

```
RDEFINE   APPL   ADAMDB2PI   OWNER(SECADM)
                             UACC(NONE)   NOTIFY(SECADM)
```

Een voorbeeld waarmee de groep *INKOOP* netwerktoegang tot DB2 krijgt:

```
PERMIT    ADAMDB2P   CLASS(APPL)   ID(INKOOP)   ACCESS(READ)
```

De SNA-netwerkcommunicatie met de DB2 Distributed Data Facility (DDF) address space moet worden geautoriseerd. Een voorbeeld waarbij de groep *VERKOOP* DB2 via het SNA-netwerk, waarbij de remote-locatie moet worden opgegeven, mag benaderen:

```
PERMIT    DB2P.DIST    CLASS(DSNR)   ID(VERKOOP)   ACCESS(READ)
                                     WHEN(APPCPORT(RDAM))
```

Het userid, oude wachtwoord en nieuwe wachtwoord moeten encrypted over het netwerk worden verzonden. Het gebruik van encrypted userids en wachtwoorden kan voor SNA worden afgedwongen in de DB2-subsysteemtabel SYSIBM.LUNAMES. De kolom SECURITY_IN moet voor elke remote user staan op ‘V’ (verify) wat inhoud dat DB2-VTAM-RACF alle aanvragen van remote-gebruikers verifieert en de kolom ENCRYPTPSWDS moet op ‘Y’ staan wat aangeeft dat alle ontvangen wachtwoorden encrypt moeten zijn.

Voor aanvragen naar een andere DB2 subsysteem (sending requests) kan het beste worden volstaan met RACF Passtickets waarbij het password niet over het netwerk wordt verzonden. Dit wordt in de DB2-subsysteemtabel SYSIBM.LUNAMES in de kolom SECURITY_OUT met een ‘R’ aangegeven.

22.17.2 TCP/IP

Zie voor volledige beschrijving van TCP/IP-beveiliging het hoofdstuk ‘TCP/IP’.

Voor het gebruik van TCP/IP door DB2 moet de Distributed database manager address space de Unix System Services (USS) kunnen gebruiken. Uitsluitend de Distributed database manager address space mag binnen DB2 privileges hebben om de USS voor TCP/IP te kunnen gebruiken. Zie voor volledige beschrijving van USS-beveiliging het hoofdstuk ‘USS’. Een voorbeeld:

```
ALTUSER   SDB2PDST OMVS(UID(0))
```

DB2 moet identificatie en authenticatie van een gebruiker uitvoeren. Het is niet toegestaan om een DB2-aanvraag die over een extern (buiten de box) TCP/IP-netwerk te vertrouwen. Verificatie moet daarom op het DB2-subsysteem worden uitgevoerd. De implementatie op het installatiepanel DSNTIP5:

```
TCP/IP ALREADY VERIFIED   ===>   NO
```

Aangezien DB2 een netwerkapplicatie is kan worden overwogen om de DB2 applicatie met de RACF general resource class APPL te beschermen. Een voorbeeld waarbij DB2 door TCP/IP-RACF wordt beschermd:

De TCP/IP-poortnummers die DB2 gebruikt wordt in installatiepanel DSNTIP5 opgegeven. Een voorbeeld:

```
DRDA PORT        ===>   446
RESYNC PORT      ===>   5020
```

De RACF-profielen:

```
RDEFINE   APPL   ADAMDB2P
          OWNER(SECADM)  UACC(NONE)  NOTIFY(SECADM)
```

Een voorbeeld waarmee de groep *INKOOP* netwerktoegang tot DB2 krijgt:

```
PERMIT    ADAMDB2P  CLASS(APPL)  ID(INKOOP)  ACCESS(READ)
```

De TCP/IP-netwerkcommunicatie met de DB2 Distributed Data Facility (DDF) address space moet worden geautoriseerd. Een voorbeeld waarbij de groep *VERKOOP* DB2 via het TCP/IP-netwerk, waarbij het keyword TCPIP moet worden gebruikt, mag benaderen:

```
PERMIT    DB2P.DIST  CLASS(DSNR)  ID(VERKOOP)  ACCESS(READ)
                                   WHEN(APPCPORT(TCPIP))
```

Aangezien de PI-standaard voor het efficiënte beheer er vanuit gaat dat groepsid(s) worden toegepast in de autorisatieprofielen zal de connectie-exit DSN3@ATH voor TCP/IP-netwerkcommunicatie moeten worden aangepast. De exit zal zo moeten worden gemodificeerd dat deze de secondary IDs (groepids) uit RACF haalt.

Opmerking: de Sign-on exit wordt niet bij TCP/IP-netwerkcommunicatie gebruikt.

Voor aanvragen naar een andere DB2-subsysteem (sending requests) kan het beste worden volstaan met RACF Passtickets waarbij het password niet over het netwerk wordt verzonden. Dit wordt in de DB2-subsysteemtabel SYSIBM.IPNAMES in de kolom SECURITY_OUT met een ‘R’ aangegeven.

23 UNIX System Services (USS)

23.1 Introductie

23.2 Status

Dit hoofdstuk is op OS/390 release 2.8 en moet nog naar z/OS 1.2 worden gebracht. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

23.3 Gebruikersprofielen

Alle gebruikers die de Unix System Services (USS) gebruiken moeten binnen RACF zijn gedefinieerd.
Als eerste moet een normaal RACF gebruikersprofiel met daarbij een OMVS-segment worden gedefinieerd om de USS te kunnen gebruiken. De naam OMVS is een erfenis van de oude naam van de USS, OpenEdition/MVS. De implementatie om een userid voor de USS te definiëren:

ADDUSER *XKRET01* NAME(*'T. Krens'*) OWNER(*HRMSTAF*)
OMVS(UID(*451230*) **HOME**(*'/u/xkret01'*) **PGM**(*'/bin/sh'*))

Als een OMVS-segment wordt gedefinieerd moet de Security administrator minimaal de volgende richtlijnen in acht nemen:

- een algoritme voor toekenning van het UNIX UID (User ID), bijvoorbeeld:
  - het UID 0 (nul) mag nooit aan een gebruiker worden toegekend, behalve wanneer dit voor technische redenen noodzakelijk is en er geen andere implementatie mogelijkheid aanwezig is;
  - de eerste 1000 UIDs mogen uitsluitend voor USS-processen worden gebruikt;
  - voor interne gebruikers een relatieve plaats in de numerieke schaal van 1.000 tot 1.000.000. Bijvoorbeeld: T Krens, waar de letter K ongeveer in het midden van het alfabet ligt krijgt het UID van 451230. Wanneer medewerkers een employeenummer hebben kan overwogen worden deze te gebruiken;
  - voor externe gebruikers, inhuur, kort lopende opdrachten, etc., een relatieve plaats in de numerieke schaal van 1.000.000 tot 2.000.000.
  - Een alternatief voor toekennen van UIDs kan zijn dat het personeelsnummer wordt gebruikt met een prefix van:
    - 10 voor productie userids;
    - 20 voor intern personeel;
    - 30 voor extern personeel.
- een beleid voor het toekennen van het mount point van de gebruikers home directory. Het beleid moet minimaal de volgende punten bevatten:
  - normale eindgebruikers mag nooit toegestaan worden het mount point van de 'root' directory ('/') te gebruiken;
  - het gebruik van de 'root' directory is uitsluitend toegestaan door de USS-systeembeheerders voor het onderhouden van de USS-systeemfiles;

  - het gebruik van de ‘root’ directory is uitsluitend toegestaan voor USS daemons en processen wanneer daar een technische reden voor is.
- een beleid voor het toekennen van het programma dat de USS initieel voor de gebruiker opstart. Het programma hangt af van het gebruik van de USS door de gebruiker maar zal veelal een UNIX shell zijn.

23.4 Groepsprofielen

Zoals bij RACF-gebruikers van toepassing is moeten USS-gebruikers met een OMVS-segment aan één of meer groepen zijn gekoppeld. Een gebruiker wordt met een normale RACF CONNECT aan de betreffende groep gekoppeld. De groepen echter moeten een UNIX GID (Groep ID) krijgen die door de USS voor autorisatie controle wordt gebruikt. Het GID wordt door middel van een OMVS-segment bij het groepsprofiel gedefinieerd. Een voorbeeld:

```
ADDGROUP  PIGRP  OMVS(GID(551670))
```

Als een groeps-OMVS-segment wordt gedefinieerd moet de Security administrator minimaal de volgende richtlijnen in acht nemen:

- een algoritme voor toekenning van het GID, bijvoorbeeld:
  - het GID 0 (nul) mag nooit aan een groep worden toegekend, behalve wanneer dit voor technische redenen noodzakelijk is en er geen andere implementatie mogelijkheid aanwezig is;
  - de eerste 1000 GIDs mogen uitsluitend voor USS-processen worden gebruikt;
  - voor interne gebruikers een relatieve plaats in de numerieke schaal van 1.000 tot 1.000.000. Bijvoorbeeld: PI-groep (PIGRP), waar de letter P ongeveer in het midden van het alfabet ligt krijgt het GID van 551670. Wanneer afdelingen of groepen al een nummer hebben kan overwogen worden dit te toe te passen;
  - voor externe gebruikers, inhuur, kort lopende opdrachten, etc., een relatieve plaats in de numerieke schaal van 1.000.000 tot 2.000.000.
  - Een alternatief voor toekennen van GIDs kan zijn dat het afdelingsnummer wordt gebruikt.

Opmerking: Een UID en GID kan een numerieke waarde hebben van 0 tot 2147483647.

23.5 superuser

Om processen met ‘superuser’ autoriteit te laten uitvoeren, bijvoorbeeld processen door de ‘cron’ daemon gestart, moet een userid met een UID van 0 en een groepid met een GID van 0 worden gedefinieerd. Dit userid is standaard BPXROOT maar kan met het SUPERUSER(USSROOT) statement in het SYS1.PARMLIB(BPXPRMxx) member worden veranderd naar een installatie specifiek userid. In dit implementatie voorbeeld is het USSROOT:

```
ADDUSER  USSROOT  NAME('USS root userid')  OWNER(SECADM)
         NOPASSWORD  DFLTGRP(USSGRP)
         OMVS(UID(0)  HOME('/')  PROGRAM())
```

Voor het ‘talk’, ‘write’ en ‘mesg’ commando, vereist USS de groepsnaam ‘tty’ met een installatie unieke GID. Het is niet toegestaan gebruikers aan deze groep te koppelen. De naam van de groep kan naar een installatie specifieke naam worden veranderd met het

TTYGROUP(USSTTYG) statement in het SYS1.PARMLIB(BPXPRMxx) member. In dit implementatie voorbeeld is het veranderd naar USSTTYG:

```
ADDGROUP  USSTTYG   SUPGROUP(USSMAIN)   OMVS(GID(2))
```

23.6 Default userid en groepid

Als gebruikers zich niet hoeven te identificeren en autoriseren, bijvoorbeeld wanneer de USS wordt gebruikt als Internet server, dient een default userid en groepid te worden aangemaakt die uitsluitend die rechten heeft die een onbekende Internet gebruiker maximaal binnen de USS en OS/390 mag hebben. Zowel de UID als GID moeten eenvoudig te onderkennen zijn. Bij gebruik van een default userid en groepid, die voor elke willekeurige gebruiker toegankelijk is, dient de installatie er voor te zorgen dat de fileprotectiebits en de mogelijkheid voor het uitvoeren van commando's adequaat zijn afgeschermd. Het default userid en groepid kunnen door middel van het BPX.DEFAULT.USER profiel in de FACILITY class aan de USS worden bekendgemaakt. Bij gebruik van de Webserver is het niet toegestaan het default userid (BPX.DEFAULT.USER) te gebruiken. Hiervoor in de plaats dient het surrogate user mechanisme te worden gebruikt zoals in het hoofdstuk 'Webserver' is beschreven. De implementatie:

```
ADDGROUP  USSDFLTG  SUPGROUP(USSMAIN)  OWNER(USSMAIN)
                                OMVS(GID(1002))

ADDUSER  USSDFLTU  NAME('USS default user')  OWNER(SECADM)
                   NOPASSWORD  DFLTGRP(USSDFLTG)
                   OMVS(UID(1002)  HOME('/tmp/ussdflt')  PROGRAM('/bin/sh'))

RDEFINE    FACILITY   BPX.DEFAULT.USER
                   APPLDATA('USSDFLTU/ USSDFLTG')

SETROPTS RACLIST(FACILITY)  REFRESH
```

23.7 Cross reference UIDs en GIDs

Door de USS kan een cross reference van de UIDs en GIDs in de RACF class UNIXMAP worden bijgehouden. Omdat de norm is dat een UID en GID niet bij meerdere userids of groepids mogen worden gedefinieerd en dus uniek moeten zijn tenzij het technisch noodzakelijk is om nummers dubbel toe te kennen, kan door de Security administrator de UNIXMAP class worden geactiveerd. De UIDs en GIDs worden dan automatisch door RACF in de UNIXMAP class gedefinieerd wanneer deze bij een RACF-gebruikersprofiel of -groepsprofiel worden gedefinieerd.
Voor een UID wordt een profiel Uxxx aangemaakt waarbij de xxx het UID is. De userids die het betreffende UID hebben toegewezen gekregen worden op de access list van dit betreffende profiel geplaatst.
Voor het GID wordt een profiel Gxxx profiel aangemaakt waarbij xxx het GID is. De groepids die het betreffende GID hebben toegewezen gekregen worden op de access list van dit betreffende profiel geplaatst. De Security administrator kan de UNIXMAP profielen listen en dubbele uitgiften van UIDs en GIDs controleren. Om de UNIXMAP class te activeren wordt het volgende commando gebruikt:

```
SETROPTS CLASSACT(UNIXMAP)
```

Uitsluitend de Security administrator (SPECIAL user) mag de informatie in het OMVS-segment lezen en wijzigen. Gewone gebruikers mogen niet de mogelijkheid hebben deze informatie te lezen of te veranderen door middel van RACF of UNIX commando's. Dit kan door middel van de volgende definities worden bereikt. De implementatie:

```
RDEFINE   FIELD  USER.OMVS.UID          UACC(NONE)
RDEFINE   FIELD  USER.OMVS.HOME         UACC(NONE)
RDEFINE   FIELD  USER.OMVS.PROGRAM      UACC(NONE)
RDEFINE   FIELD  USER.OMVS.CPUTIME      UACC(NONE)
RDEFINE   FIELD  USER.OMVS.ASSIZE       UACC(NONE)
RDEFINE   FIELD  USER.OMVS.FILEPROC     UACC(NONE)
RDEFINE   FIELD  USER.OMVS PROCUSER     UACC(NONE)
RDEFINE   FIELD  USER.OMVS THREADS      UACC(NONE)
RDEFINE   FIELD  USER.OMVS MMAPAREA     UACC(NONE)

SETROPTS CLASSACT(FIELD)
SETROPTS RACLIST(FIELD)  REFRESH
```

23.8 USS kernel

De USS-kernel is een OS/390 started task. Deze started task moet worden opgestart onder een RACF userid met een UID en een groepid met een GID. Deze userid en groepid mogen verder geen toegangsrechten op OS/390 resources hebben. Voor onderhoud en beveiligingsredenen, zoals beveiligingsdelegatie, wordt het aanbevolen om een nieuwe RACF-groep (high level qualifier) voor alle USS gerelateerde resources te definiëren. De implementatie:

```
ADDGROUP  USSMAIN  NAME('USS primaire resource groep')
                   OWNER(SECADM)

ADDGROUP  USSGRP   SUPGROUP(USSMAIN)  OWNER(USSMAIN)
                   OMVS(GID(0))

ADDUSER   USSU     NAME('USS kernel userid')  OWNER(SECADM)
                   NOPASSWORD  DFLTGRP(USSGRP)
                   OMVS(UID(0)  HOME('/')  PGM())
```

De USS started task moet in de RACF class STARTED worden gedefinieerd. De implementatie:

```
SETROPTS GENERIC(STARTED)

RDEFINE   STARTED  OMVS.*  OWNER(SECADM)  STDATA(USER(USSU)
          GROUP(USSGRP)  PRIVILEGED(NO)  TRUSTED(NO)
          TRACE(YES))

SETROPTS CLASSACT(STARTED)
SETROPTS RACLIST(STARTED)
```

De USS heeft het initialisatie proces BPXOINIT nodig die als started task wordt uitgevoerd en dient in de STARTED class te worden gedefinieerd. De implementatie:

```
RDEFINE   STARTED  BPXOINIT.*  OWNER(SECADM)
                   STDATA(USER(USSU) GROUP(USSGRP)
                   PRIVILEGED(NO)  TRUSTED(NO)  TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

Omdat de norm is dat de USS niet als een 'trusted' task mag wordt gedefinieerd moet deze expliciet toegang tot de HFS datasets worden gegeven. Een voorbeeld:

```
ADDSD     'OMVS.**'  OWNER(SECADM)  UACC(NONE)
PERMIT    'OMVS.**'  ID(USSGRP)     ACCESS(UPDATE)
```

Indien andere HFS-datasets moeten worden gebruikt moeten ook deze worden geautoriseerd. Een voorbeeld:

```
ADDSD    'PROD.HFS.**'  OWNER(SECADM)  UACC(NONE)
PERMIT   'PROD.HFS.**'  ID(USSGRP)     ACCESS(UPDATE)
```

Om hacking attempts te reduceren en het laden van installatie vreemde programma's te voorkomen moeten alle programma's door RACF worden gecontroleerd (program controlled). Programma's die niet bij RACF bekend zijn kunnen dan niet worden geladen en daardoor niet worden uitgevoerd. De implementatie voor programma control:

```
SETROPTS WHEN(PROGRAM)

ADDSD    'SYS1.LINKLIB'    OWNER(SECADM)  UACC(READ)
ADDSD    'PROD.SCEERUN'    OWNER(SECADM)  UACC(READ)
ADDSD    'SYS1.SEZALINK'   OWNER(SECADM)  UACC(READ)

RDEFINE  PROGRAM  **  ADDMEM('SYS1.LINKLIB'/ /NOPADCHK)
                        OWNER(SECADM)  UACC(READ)
RALTER   PROGRAM  **  ADDMEM('PROD.SCEERUN'/ /NOPADCHK)
                        UACC(READ)
RALTER   PROGRAM  **  ADDMEM('SYS1.SEZALINK'/ /NOPADCHK)
                        UACC(READ)

SETROPTS WHEN(PROGRAM)  REFRESH
```

23.9 Daemons

Om de gebruikersidentiteit te kunnen veranderen met de system calls setuid, seteuid, setreuid en spawn met een userid, kan door middel van het RACF-profiel BPX.DAEMON in de FACILITY class worden geautoriseerd. Als dit profiel gedefinieerd is met UACC(NONE) dan moeten alle UID(0) gebruikers, die deze system calls moeten kunnen uitvoeren, worden geautoriseerd. Over het algemeen hebben daemons die onder UID(0) worden gestart zoals inetd, rlogind, cron, Im, uucpd, syslog, etc., deze system calls nodig.

Met het BPX.DAEMON profiel wordt ook bereikt dat alle programma's die door een daemon worden geladen 'program controlled' moeten zijn. Hierdoor kunnen geen installatie vreemde programma's worden geladen en uitgevoerd. De implementatie hoe de USS daemon eigenschappen krijgt:

```
RDEFINE  FACILITY  BPX.DAEMON  OWNER(SECADM)  UACC(NONE)
PERMIT   BPX.DAEMON  CLASS(FACILITY)  ID(USSU)  ACCESS(READ)

SETROPTS CLASSACT(FACILITY)
SETROPTS RACLIST(FACILITY)  REFRESH
```

Alle daemons moeten onder hun eigen userid/groepid worden geactiveerd. De implementatie:

```
ADDGROUP  UUCPGRP  SUPGROUP(USSMAIN)  OWNER(USSMAIN)
                     OMVS(GID(0))

ADDUSER   UUCPU   NAME('uucp daemon user ID')  OWNER(SECADM)
                  NOPASSWORD  DFLTGRP(UUCPGRP)
                  OMVS(UID(0)  HOME('/usr/spool/uucppublic')  PGM())

PERMIT    BPX.DAEMON  CLASS(FACILITY)  ID(UUCPU)  ACCESS(READ)

ADDGROUP  RLOGGRP  SUPGROUP(USSMAIN)  OMVS(GID(0))
```

```
                                OWNER(USSMAIN)
ADDUSER  RLOGU  NAME('rlogin daemon user ID')  OWNER(SECADM)
                NOPASSWORD  DFLTGRP(RLOGGRP)
                OMVS(UID(0)  HOME('/')  PGM())

PERMIT   BPX.DAEMON  CLASS(FACILITY)  ID(RLOGU)  ACCESS(READ)

SETROPTS RACLIST(FACILITY)  REFRESH
```

23.10 Autorisatie USS faciliteiten

De USS heeft een aantal faciliteiten die door middel van RACF kunnen worden geautoriseerd. Deze faciliteiten moeten worden gelimiteerd tot uitsluitend die gebruikers die vertrouwd zijn en waarvoor het noodzakelijk is dat zij deze autorisatie krijgen. Deze faciliteiten mogen uitsluitend via het Change Management proces worden toegekend. De implementatie:

```
RDEFINE  FACILITY  BPX.DEBUG              UACC(NONE)
RDEFINE  FACILITY  BPX.DAEMON             UACC(NONE)
RDEFINE  FACILITY  BPX.DEFAULT.USER        UACC(NONE)
RDEFINE  FACILITY  BPX.FILEATTR.APF        UACC(NONE)
RDEFINE  FACILITY  BPX.FILEATTR.PROGCTL    UACC(NONE)
RDEFINE  FACILITY  BPX.JOBNAME             UACC(NONE)
~~RDEFINE  FACILITY  BPX.SAFFASTPATH      UACC(NONE)~~
```

Het profiel BPX.SAFFASTPATH moet niet worden gedefinieerd zodat SAF fastpath niet wordt geactiveerd. Door het niet te activeren van SAF fastpath kan alle toegang en toegangspogingen tot HFS files worden gelogged.

```
RDEFINE  FACILITY  BPX.SERVER             UACC(NONE)
RDEFINE  FACILITY  BPX.SMF                UACC(NONE)
RDEFINE  FACILITY  BPX.STOR.SWAP          UACC(NONE)
RDEFINE  FACILITY  BPX.SUPERUSER          UACC(NONE)
RDEFINE  FACILITY  BPX.WLMSERVER          UACC(NONE)

PERMIT   BPX.FILEATTR.APF  CLASS(FACILITY)
         ID(SECADM)  ACCESS(UPDATE)
PERMIT   BPX.FILEATTR.PROGCTL  CLASS(FACILITY)
         ID(SECADM)  ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY)  REFRESH
```

Geen enkele eindgebruiker mag een UID van 0 en een GID van 0 toegekend krijgen. Als een eindgebruiker toch taken moet uitvoeren waarvoor 'superuser' autorisatie nodig is, zoals voor aanpassing van systeem files, dient de eindgebruiker voor het BPX.SUPERUSER profiel in de FACILITY class te worden geautoriseerd. Een gebruiker blijft hiermee zijn eigen UID houden bijvoorbeeld 451230, maar kan dan met het shell commando 'su' zonder wachtwoord de superuser eigenschappen krijgen. De implementatie:

```
RDEFINE  FACILITY  BPX.SUPERUSER          UACC(NONE)
PERMIT   BPX.SUPERUSER  CLASS(FACILITY)
         ID(SECADM)  ACCESS(READ)

SETROPTS RACLIST(FACILITY)  REFRESH
```

23.11 Autorisatie USS commando's

Een gebruiker dient uitsluitend die commando's te kunnen uitvoeren die strikt noodzakelijk zijn voor de uitvoering van zijn/haar taak. UNIX-commando's die de integriteit van het gehele systeem kunnen beïnvloeden moeten meestal onder de 'superuser' autoriteit worden uitgevoerd.

Indien niet nodig is dat een gebruiker de volledige ‘superuser’ autoriteit behoeft te krijgen maar voldoende is om de mogelijkheid te hebben op een specifiek commando uit te voeren dienen profielen in de RACF class UNIXPRIV te worden gedefinieerd waarop de gebruiker wordt geautoriseerd. Deze autorisatie dient via Change Management te worden verkregen. De implementatie:

```
RDEFINE   UNIXPRIV  CHOWN.UNRESTRICTED            UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS             UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS.CHOWN       UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS.MOUNT       UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS.QUIESCE     UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS.PFSCTL      UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.FILESYS.VREGISTER   UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.IPC.RMID            UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.PROCESS.GETPSENT    UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.PROCESS.KILL        UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.PROCESS.PTRACE      UACC(NONE)
RDEFINE   UNIXPRIV  SUPERUSER.SETPRIORITY         UACC(NONE)

PERMIT    SUPERUSER.FILESYS.CHOWN  CLASS(UNIXPRIV)
          ID(SECADM)  ACCESS(READ)

SETROPTS CLASSACT(UNIXPRIV)
SETROPTS RACLIST(UNIXPRIV)  REFRESH
```

23.12 Logging

Voor logging doeleinden moet de daemon ‘syslogd’ als een started task worden geactiveerd. De implementatie:

```
ADDGROUP  SYSLOGRP  SUPGROUP(USSMAIN)  OMVS(GID(0))
ADDUSER   SYSLOGU   NAME('Syslogd daemon user ID')  OWNER(SECADM)
                    NOPASSWORD  DFLTGRP(SYSLOGRP)
                    OMVS(UID(0)  HOME('/')  PROGRAM())

RDEFINE   STARTED   SYSLOGRP.*  STDATA(USER(SYSLOGU)
                    GROUP(SYSLOGRP)  PRIVILEGED(NO)
                    TRUSTED(NO)  TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

Voor het loggen van RACF- en USS-informatie dienen de volgende classes te worden geactiveerd. In deze classes dienen geen profielen te worden gedefinieerd. De implementatie:

```
SETROPTS CLASSACT(BPXAS  BPXOINIT  OMVS)
SETROPTS CLASSACT(DIRAUTH  DIRSRCH  DIRACC)
SETROPTS CLASSACT(FSOBJ  FSSEC)
SETROPTS CLASSACT(IPCOBJ  PROCACT  PROCESS)
```

Door middel van RACF opties kan de mate van de UNIX-logging worden bepaald. Een implementatie voorbeeld:

```
SETROPTS LOGOPTIONS(FAILURES(DIRSRCH))
```

24 TCP/IP

24.1 Introductie

24.2 Status

Dit hoofdstuk is op OS/390 release 2.8 en moet nog naar z/OS 1.2 worden gebracht. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

24.3 IP adres control

Toegang tot FTP kan door middel van FTP-exits worden gecontroleerd. Het gebruik van exits wordt echter sterk afgeraden. Toegang dient zoveel mogelijk met standaard RACF faciliteiten te worden geregeld. Een voorbeeld:

Alle werkstations die niet specifiek zijn gedefinieerd kunnen door alle gebruikers worden gebruikt:

```
RDEFINE   TERMINAL  **  UACC(READ)
```

Het werkstation dat door een specifieke gebruiker mag worden gebruikt voor bijvoorbeeld een FTP-sessie moet worden gedefinieerd. Een voorbeeld:

```
RDEFINE   TERMINAL  94568393  UACC(NONE)
PERMIT    CLASS(TERMINAL)  94568393  ID(XMIEP01)  ACCESS(READ)
```

AANVULLENDE MAATREGEL. Het gebruik van het werkstation moet als conditie worden opgenomen voor het gebruik van het FTP-programma. De implementatie:

```
RDEFINE   APPL  FTP*  OWNER(SECADM)     UACC(NONE)
PERMIT    CLASS(APPL)  FTP*    ID(XMIEP01)  ACCESS(READ)
                                WHEN(TERMINAL(94568393))
```

De TERMINAL en APPL class moeten geactiveerd worden. De implementatie:

```
SETROPTS CLASSACT(TERMINAL  APPL)
```

24.4 Userid en groepid

TCP/IP en FTP zijn OS/390 started tasks waarvan de betreffende userids en groepids met een UID en GID moeten worden gedefinieerd. De implementatie:

```
ADDGROUP  TCPIPGRP  SUPGROUP(SUSSMAIN)  OWNER(SUSSMAIN)
                    OMVS(GID(0))
ADDUSER   TCPIPU    NAME('TCP/IP daemon user ID')  OWNER(SECADM)
                    NOPASSWORD  DFLTGRP(STCPGRP)
                    OMVS(UID(0)  HOME('/')  PGM())

ADDGROUP  FTPGRP    SUPGROUP(USSMAIN)  OWNER(USSMAIN)
                    OMVS(GID(0))
ADDUSER   FTPU      NAME('FTP daemon user ID')  OWNER(SECADM)
                    NOPASSWORD  DFLTGRP(SFTPGRP)
                    OMVS(UID(0)  HOME('/')  PGM())
```

```
RDEFINE   STARTED  TCPIP.*  OWNER(SECADM)
                    STDATA(USER(STCPU)  GROUP(TCPIPGRP)
                    PRIVILEGED(NO)  TRUSTED(NO)  TRACE(YES))
RDEFINE   STARTED  FTPD.*   OWNER(SECADM)
                    STDATA(USER(SFTPU)  GROUP(FTPGRP)
                    PRIVILEGED(NO)  TRUSTED(NO)  TRACE(YES))
```

Er mag geen mogelijkheid zijn voor het gebruik van het TCP/IP en FTP daemon userid voor de uitvoering van file transfers door eindgebruikers (userid propagation) en dient daarom altijd te worden afgeschermd voor propagatie. De implementatie:

```
RDEFINE   PROPCNTL  STCPU  SFTPU
```

24.5 TCP/IP en FTP daemon

Zowel TCP/IP als FTP moeten als een daemon in de BPX.DAEMON profiel in de FACILITY class worden gedefinieerd. De implementatie:

```
PERMIT    BPX.DAEMON  CLASS(FACILITY)  ID(STCPU)  ACCESS(READ)
PERMIT    BPX.DAEMON  CLASS(FACILITY)  ID(SFTPU)  ACCESS(READ)

SETROPTS RACLIST(FACILITY)  REFRESH
```

24.6 Autorisatie z/OS commando's

Alle TCP/IP and FTP gerelateerde OS/390 commando's dienen door RACF te worden gecontroleerd. De implementatie:

```
SETROPTS  GENERIC(OPERCMDS)
SETROPTS  GENCMD(OPERCMDS)

RDEFINE   OPERCMDS  (MVS.VARY.TCPIP.DROP)  OWNER(SECADM)
                    UACC(NONE)
RDEFINE   OPERCMDS  (MVS.VARY.TCPIP.OBEYFILE)  OWNER(SECADM)
                    UACC(NONE)
PERMIT    'MVS.VARY.TCPIP.DROP'  CLASS(OPERCMDS)
                    ID(OPERGRP2)  ACCESS(CONTROL)
PERMIT    'MVS.VARY.TCPIP.OBEYFILE'  CLASS(OPERCMDS)
                    ID(OPERGRP2)  ACCESS(CONTROL)

RDEFINE   OPERCMDS  (MVS.ROUTEMGR.OMPROUTE)  OWNER(SECADM)
                    UACC(NONE)
PERMIT    'MVS.ROUTEMGR.OMPROUTE'  CLASS(OPERCMDS)
                    ID(OPERGRP2)  ACCESS(CONTROL)

RDEFINE   OPERCMDS  (MVS.ROUTEMGR.OROUTED)  OWNER(SECADM)
                    UACC(NONE)
PERMIT    'MVS.ROUTEMGR.OROUTED'  CLASS(OPERCMDS)
                    ID(OPERGRP2)  ACCESS(CONTROL)

SETROPTS  CLASSACT(OPERCMDS)
SETROPTS  RACLIST(OPERCMDS)  REFRESH
```

25 Firewall

25.1 Introductie

De z/OS firewall is een beveiligingsfaciliteit voor een IP-netwerk die naar behoefte kan worden geactiveerd. Met de z/OS firewall kan z/OS op een veiligere manier aan een TCP/IP-netwerk zoals een Intranet of het Internet worden gekoppeld. De basis van de z/OS firewall beveiligingsfaciliteit is IP-packet filtering. De filtering wordt op basis van filterregels door de Firewall uitgevoerd, zo kan het opzetten van een netwerkverbinding worden toegestaan of worden afgewezen.

Naast het definiëren van de Firewall filterregels met behulp van lijncommando's is er ook een meer gebruikersvriendelijke interface, gebaseerd op een graphical user interface (GUI), aanwezig. Voor het gebruik van deze interface, die op een personal computer kan worden geïnstalleerd, moet de Firewall configuration server worden geïmplementeerd.

Naast de basis z/OS Firewall service biedt de leverancier andere netwerkbeveiligingsfaciliteiten die zijn ondergebracht in het Firewall technologies product. Implementatie van de z/OS Firewall is echter een voorwaarde om deze additionele netwerkbeveiligingsfaciliteiten te kunnen inrichten. Deze extra services zijn:

- Network Address Translation (NAT);
- Domain Name Services (DNS);
- Proxy servers (bijvoorbeeld de FTP (Application Gateway) proxy);
- SOCKS server;
- RealAudio Support.

25.1.1 Internet Protocol

De Firewall services die in deze standaard zijn beschreven zijn gebaseerd op IP versie 4 (IPV4). De standaard voor het IPsec (IP versie 6) protocol, de basis voor een Virtual Private Network (VPN) wordt in een ander hoofdstuk beschreven.

25.1.2 Virtual Private Network

De onderstaande netwerkbeveiligingsfaciliteiten die de leverancier tot de Firewall Technologies rekent wordt in het hoofdstuk Virtual Private Network (VPN) en Internet Key Exchange (IKE) behandeld:

- Virtual Private Network (VPN);
- ISAKMP server, die het Internet Security Association and Key Management Protocol implementeerd voor Internet Key Exchange (IKE).

25.1.3 End-node Firewall

De opzet van deze z/OS Firewall standaard gaat uit van het 'end-node of Private Firewall' concept. Alhoewel dit mogelijk is het niet de bedoeling de Firewall netwerkverkeer te laten routeren naar andere netwerken. De Firewall-opzet in deze standaard zal uitsluitend netwerkveer wel of niet tot de achterliggende z/OS applicaties toelaten.

25.1.4 Leeswijzer

Voor extra verduidelijking zijn andere dan uitsluitend beveiligingsgerelateerde teksten opgenomen. Voor een volledig operationeel overzicht dienen de officiële publicaties van de leverancier te worden genomen.

25.2 Installatie

25.2.1 Firewall Kernel

De Firewall-kernel is een z/OS started task. Deze started task moet worden opgestart onder een RACF userid met een UID en een groepid met een GID. Deze userid en groepid mogen verder geen toegangsrechten op z/OS-resources hebben dan strik noodzakelijk voor de werking. Voor onderhoud en beveiligingsredenen, zoals beveiligingsdelegatie, wordt het aanbevolen om een nieuwe RACF-groep (high level qualifier) voor alle Firewall gerelateerde resources te definiëren. De implementatie:

```
ADDGROUP FWGRP SUPGROUP(SYS1) OWNER(USSMAIN)
                    OMVS(GID(10))
```

Voor de Firewall moet een apart userid worden gedefinieerd waaraan specifieke toegangsrechten kunnen worden verleend. Het userid mag niet voor logon kunnen worden gebruikt en moet, gezien het technische ontwerp, de root (superuser) privilege hebben. De implementatie:

```
ADDUSER FWU NAME('Firewall kernel userid') OWNER(SECADM)
            DFLTGRP(FWGRP) NOPASSWORD
            OMVS(UID(0) HOME('/u/fwkern/') PGM())
            AUTHORITY(CREATE) UACC(ALTER)
```

De Firewall started task moet in de general resource class STARTED als non-privileged en non-trusted worden gedefinieerd zodat de toegangsrechten worden beperkt tot specifieke toegangsrechten. De implementatie:

```
SETROPTS GENERIC(STARTED)
SETROPTS CLASSACT(STARTED)

RDEFINE  STARTED FWKERN OWNER(SECADM) STDATA(USER(FWU)
         GROUP(FWGRP) PRIVILEGED(NO) TRUSTED(NO)
         TRACE(YES))

SETROPTS RACLIST(STARTED)
```

25.2.2 Dataset protectie

De Firewall gebruikt TCP/IP-programmatuur. Omdat de norm is dat het Firewall started task userid als een 'non-trusted' task moet worden gedefinieerd moet expliciet toegang tot de TCP/IP-datasets worden gegeven. Een voorbeeld:

```
ADDSD    'SYS1.TCPIP.*' OWNER(SECADM) UACC(NONE)
PERMIT   'SYS1.TCPIP.*' ID(FWU) ACCESS(READ)
```

De Firewall started task wordt met JCL opgestart waardoor het started task userid geautoriseerd moet zijn om de JCL te kunnen lezen. Indien de JCL voor de gebruikte deamons in een aparte JCL-library wordt gedefinieerd zal ook deze voor het Firewall started task userid moeten worden geautoriseerd. Een voorbeeld van autorisatie van een JCL-library:

```
ADDSD    'SYS1.FIREWALL.PROCLIB' GENERIC
                OWNER(SECADM) UACC(NONE)
```

```
PERMIT     'SYS1.FIREWALL.PROCLIB'
                    ID(FWU)  ACCESS(READ)
```

25.2.3 Programma protectie

Om zich te kunnen gedragen als een daemon moet het userid waaronder de daemon wordt gestart met het general resource profiel BPX.DAEMON in de FACILITY class worden geautoriseerd. Uitsluitend een userid met UID(0), wat het Firewall started task userid is, kan hiervoor worden geautoriseerd. Met het BPX.DAEMON profiel wordt ook bereikt dat alle programma's die door een daemon worden geladen 'program controlled' moeten zijn. Hierdoor kunnen geen installatie vreemde programma's worden geladen en uitgevoerd. De implementatie waarbij het Firewall started task userid, waaronder alle daemons worden uitgevoerd, de daemon-eigenschappen krijgt:

```
RDEFINE    FACILITY  BPX.DAEMON  OWNER(SECADM)  UACC(NONE)
PERMIT     BPX.DAEMON  CLASS(FACILITY)  ID(FWU)  ACCESS(READ)

SETROPTS RACLIST(FACILITY)  REFRESH
```

Een eigenschap van daemon programmatuur is dat deze 'program controlled' moet zijn. De programmadatasets moeten daarom in de general resource class PROGRAM worden openomen. Een voorbeeld:

```
RDEFINE    PROGRAM  *  ADDMEM('SYS1.ICA.SICALMOD'//NOPADCHK)
                         UACC(READ)

SETROPTS WHEN(PROGRAM)  REFRESH
```

Voor de Firewall Configuratie GUI moet gebruik worden gemaakt van Secure Socket Layer (SSL) programmatuur. De library met SSL-programmatuur moet 'program controlled' worden gemaakt. Een voorbeeld:

```
RDEFINE    PROGRAM  *  ADDMEM('SYS1.SSL.SGSKLOAD'//NOPADCHK)
                         UACC(READ)

SETROPTS WHEN(PROGRAM)  REFRESH
```

25.2.4 TCP/IP stack

De Firewall gebruikt de TCP/IP started task-naam als de TCP/IP stack-naam. In het voorbeeld is de stack-naam *TCPIP* voor productie en *TCPIPT* voor een andere TCP/IP stack, bijvoorbeeld voor testdoeleinden. Voor beschrijving van de beveiligingsaspecten van TCP/IP verwijzen we naar het betreffende hoofdstuk. De voorbeelden:

Voor een productie TCP/IP(-stack):

```
RDEFINE    STARTED  TCPIP.*  OWNER(SECADM)
                      STDATA(USER(STCPU)  GROUP(TCPIPGRP)
                      PRIVILEGED(NO)  TRUSTED(NO)  TRACE(YES))
```

Voor de test TCP/IP-stack genaamd *TCPIPT*:

```
RDEFINE    STARTED  TCPIPT.*  OWNER(SECADM)
                      STDATA(USER(STCPU)  GROUP(TCPIPGRP)
                      PRIVILEGED(NO)  TRUSTED(NO)  TRACE(YES))
```

25.2.5 TCP/IP Configuratie datasets

Voor het configureren van TCP/IP zijn verschillende parametersets noodzakelijk. Voor het configureren van applicatie gerelateerde componenten zijn dat de zogenaamde TCPIP.DATA configuratiestatements. Deze statements kunnen in verschillende datasets worden opgenomen. Zo kan er bijvoorbeeld de *SYS1.TCPIP.**DATA***, *SYS1.TCPPARMS(TCPIPT)*, of andere door de installatie te kiezen naamgeving worden gebruikt.

Voor hardware gerelateerde componenten zijn de zogenaamde PROFILE.TCPIP configuratiestatements aanwezig. Gelijk de TCPIP.DATA kunnen ook deze configuratiestatements in veschillende, door de installatie te kiezen, datasets worden opgenomen. Voorbeelden hiervan zijn *SYS1.TCPIP.**PROFILE**.TCPIP*, *SYS1.TCPIP.PROFILE(TCPIPT)*, etc.

Afhankelijk van de wens van de installatie kan een passende dataset worden gekozen. Bij het maken van een keuze dient altijd de meest recente installatie documentatie te worden geraadpleegd. De zogenaamde resolver zal met een voorgedefinieerde zoekvolgorde een datasetlibrary en daarna een parameterset lokaliseren en toepassen.

25.2.6 TCP/IP Configuratie parameters

Het lokaliseren van een parameterset voor TCP/IP-stack initialisatie wordt gedaan met behulp van de TCP/IP started task naam. De met de TCP/IP started task naam moet overeenkomen met de TCPIPJOBNAME-parameter in de parameterset in een TCPIP.DATA-parameterdataset. De parameter HOSTNAME in de gelokaliseerde parameterset geeft de host-naam aan voor de betreffende TCP/IP-stack. Een voorbeeld uit een TCP/IP-parameterset waarbij de overige parameters zijn weggelaten:

```
SYST:          TCPIPJOBNAME   TCPIPT
HOSTNAME       PRODA
```

Het gebruik van stacks door een gebruiker (server) kan met een EZB.STACKACCESS.*.* profiel in de general resource class SERVAUTH worden geautoriseerd. Dit is in het TCP/IP-hoofdstuk verder beschreven.

Net als TCP/IP is de Firewall een proces onder de z/OS Unix System Services (USS). Na het activeren van de Firewall software zal het netwerkverkeer door de Firewall moeten worden geleid. Binnen TCP/IP zal het gebruik van de z/OS Firewall met de FIREWALL-parameters in de PROFILE.TCPIP-dataset moeten worden gedefinieerd. Het gebruik van de DATAGRAMFWD-parameter, waardoor er routing van IP-netverkeer door de Firewall plaatsvindt, mag niet worden toegepast. De implementatie voor activering van de Firewall:

```
IPCONFIG   FIREWALL   DATAGRAMFWD
```

Voor het gebruik van het netwerk zal de Firewall een TCP/IP-stack moeten kunnen benaderd. TCP/IP onder z/OS heeft de mogelijkheid om verschillende TCP/IP-stacks te definiëren. Welke TCP/IP-stack de Firewall moet gebruiken zal moeten worden gedefinieerd. Een voorbeeld van het fwstack lijncommando voor het koppelen van de Firewall aan een TCP/IP-stack:

```
fwstack      cmd=add  \
             stack="TCPIPT"  \
             force=yes

fwstack      cmd=query

Systeemoutput fwstack commando (voorbeeld):
             stack        = TCPIPT
```

```
status          = Up
type            = Firewall
active filters  = No
filter logging  = Yes
active NAT      = No
Nat logging     = No
```

25.2.7 Firewall STACK daemon

Voor het benaderen van de TCP/IP-stack moet binnen de Firewall de stack-daemon worden geïmplementeerd en geactiveerd. De Firewall stack daemon is een z/OS started task en moet voor de Firewall worden geautoriseerd. Een voorbeeld:

```
RDEFINE   STARTED  ICAPSTAK.**  OWNER(SECADM)
          STDATA(USER(FWU)  GROUP(FWGRP)  PRIVILEGED(NO)
          TRUSTED(NO)  TRACE(YES))
```

25.2.8 Firewall activering

Het starten om daarna gebruik te maken van de Firewall resources moet worden geautoriseerd met behulp van het FWKERN.START.REQUEST profiel in de general resource class FACILITY. De implementatie:

```
SETROPTS CLASSACT(FACILITY)

RDEFINE   FACILITY  FWKERN.START.REQUEST
                              OWNER(SECADM)  UACC(NONE)
PERMIT    FWKERN.START.REQUEST  CLASS(FACILITY)
                              ID(FWU)  ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY)  REFRESH
```

Om er zeker van te zijn dat de Firewall kernel wordt gestart moet deze in de automatische logon faciliteit van TCP/IP worden opgenomen. Met de definitie van FWKERN in het AUTOLOG / ENDAUTOLOG statementblok in de PROFILE.TCPIP-dataset wordt de Firewall kernel automatisch na TCP/IP-initialisatie gestart. De implementatie waarbij de Firewall kernel wordt gedefinieerd:

```
AUTOLOG
FWKERN
LPSERVE
NAMESRV
SMTP
ENDAUTOLOG
```

Opmerking: als een z/OS-image is opgezet voor uitsluitend een routing Firewall, welke hier niet wordt beschreven en aangeraden, mag alleen de FWKERN in het AUTOLOG-statementblok worden worden gedefinieerd aangezien op een Firewall-z/OS-image geen verwerking plaats zal vinden.

De Firewall (FWKERN) zal na initialisatie de Firewall gerelateerde daemons starten indien deze met het ‘fwdaemon cmd=change’ lijncommando in de Firewall configuratie zijn toegevoegd. Deze daemons kunnen zijn:

- FWSTACK voor de Firewall stack;
- CFGSRV voor Firewall configuratie;
- SOCKS Proxy;

- FTP Proxy;
- ISAKMPD voor VPN en IKE implementatie.

Voor het benaderen van de TCP/IP-stack moet de Firewall stack daemon worden geactiveerd. De implementatie:

```
fwdaemon   cmd=change  \
           daemon=fwstackd  \
           started=yes
```

De Firewall kan naast de automatische start door TCP/IP ook met het 'fwkern' z/OS operatorcommando worden gestart echter dit is alleen aan te raden bij eventuele operationele problemen. Gelijk met de automatische start door TCP/IP zal na initialisatie van de Firewall kernel de gedefinieerde daemons, zoals de stack daemon, automatisch worden opgestart. Het voorbeeld van het z/OS-operatorcommando:

```
START      fwkern
```

25.3 Netwerkcomponenten

Om netwerkresources door de Firewall te laten controleren zullen deze binnen de Firewall bekend moeten worden gemaakt. Als eerste worden de netwerkresources voor TCP/IP gebruik gedefinieerd daarna wordt de resource informatie in de Firewall-configuratie gedefinieerd. De netwerkresources worden aan TCP/IP met de DEVICE en LINK parameterstatements bekend gemaakt. De netwerkresourceparameters worden in een PROFILE.TCPIP-dataset gedefinieerd. Enige voorbeelden voor verduidelijking van de parameters:

Voor een Channel-to-Channel verbinding:

```
DEVICE     CTC3       CTC  D00
LINK       CTCD00     CTC  0      CTC3
```

Voor een HYPERchannel verbinding:

```
DEVICE     HCH2       HCH  E00
LINK       HCHE00     HCH  2      HCH2
```

Voor een Gigabit Ethernet verbinding:

```
DEVICE     GIGAETH1   MPCIPA
LINK       GIGALINK1  IPAQENET   GIGAETH1
```

Met de PRIMARYINTERFACE-parameter wordt de host-naam gedefinieerd. Een voorbeeld:

```
PRIMARYINTERFACE   PRODA
```

Met de HOME-parameter krijgen de netwerkverbindingen, dus de fysieke verbindingen, een IP-adres. De IP-adressen worden binnen de Firewall gebruik om controle op het netwerkverkeer uit te voeren. De voorbeelden:

```
HOME
172.16.234.1       PRODA
172.16.200.0       CTCD00
172.16.210.0       HCHE00
172.16.240.0       GIGALINK1
```

25.4 Firewall Configuratie

De basis van de Firewall is het conditioneel toelaten van een netwerkverbinding gebaseerd op IP-adressen en op het soort netwerkverkeer. Hiervoor dienen door de Firewall administrator filterregels (filter rules) te worden aangemaakt. Om de Firewall-regels te definiëren moeten in het HFS (Hierarchical File System) de voor de firewall service specifieke comfiguratie files worden opgenomen. De filterregels kunnen in de files met lijncommando's of met de configuratie GUI worden gedefinieerd. De keuze van de filterregels is altijd installatie specifiek en dient op de behoefte van de omgeving te worden aangepast. Echter voor de duidelijkheid geven wij hierna enkele basisbegrippen voor het definiëren van filterregels met lijncommando's. Natuurlijk kunnen de gegeven voorbeelden ook met de configuratie GUI worden geïmplementeerd.

De Firewall filterregels moeten eerst worden gedefinieerd om daarna te kunnen worden geactiveerd. Als er geen filterregels worden gedefinieerd zal er via de z/OS Firewall geen TCP/IP netwerkverkeer met z/OS applicaties of subsystemen mogelijk zijn. Verder zijn er een aantal standaard filterregels aanwezig die naar behoefte kunnen worden geactiveerd. De definities worden uiteindelijk in configuratiefiles opgeslagen ongeacht of deze met behulp van lijncommando's of met de GUI worden gedefinieerd. Het definiëren van filterregels in de betreffende configuratiefiles met een editor is niet toegestaan omdat syntax controle op de filterregels daarmee niet wordt uitgevoerd.

Hieronder zijn voorbeelden van lijncommando's opgenomen waarmee filterregels kunnen worden gedefinieerd. Voor definities gemaakt met de configuratie GUI zullen dezelfde gegevens moeten worden gebruikt echter dit is gebruikersvriendelijker voor de Firewall security administrator.

25.4.1 Firewall Configuratie files

Voor het veilig gebruik van de Firewall heeft de leverancier standaard (default) configuratie files meegeleverd. Deze configuratiefiles dienen in het active filesysteem te worden gekopieerd zoals in de installatie voorschriften is aangegeven. De Firewall configuratiefiles zijn:

/etc/security/**fwobjects**.cfg
/etc/security/**fwrules**.cfg
/etc/security/**fwservices**.cfg
/etc/security/**fwsocks**.cfg
/etc/security/**fwaudio**.cfg
/etc/security/**fwdaemon**.cfg
/etc/security/**fwahtran**.cfg
/etc/security/**fwesptran**.cfg

25.4.2 Identifier Filterregels

Bij de definitie van een Firewall filterregel krijgt deze een uniek identifier (id) van het systeem toebedeeld waarvan de eerste 500 identifiers gereserveerd en gebruikt worden door voorgedefinieerde filterregels. Voor de eenvoud bij verdere bewerking kan deze identifier in andere lijncommando's worden opgenomen. De toebedeelde identifier kan met de list-parameter worden opgevraagd. Een voorbeeld:

fwnwobj cmd=**list** name="*PROD AMS*"

Met het voorbeeld geeft het systeem de volgende output met daarin de toebedeelde identifier (id = 834):

id = 834

```
type = Host
name = PROD AMS
desc = Productiesysteem lokatie Amsterdam
addr = 172.16.234.1
mask = 255.255.255.255
```

Voor aanpassing van de definitie kan in opvolgende lijncommando's dit voorgaande id=1 worden toegepast. Een voorbeeld van een aanpassing, die bijvoorbeeld nodig zijn als de TCP/IP-HOME-parameter wordt veranderd, en een verwijdering van een netwerkobjectdefinitie:

```
fwnwobj     cmd=change   id=834   addr=172.16.234.5

fwnwobj     cmd=delete   id=834
```

25.4.3 Voorgedefinieerde filterregels

Het is van belang om de Firewall documentatie te raadplegen op reeds aanwezige rule-definities. De leverancier heeft voor veel voorkomende netwerkdiensten definities opgenomen. Indien de voorgedefinieerde filterregel-definities niet het gewenste resultaat zal geven kunnen deze als model voor het definiëren van organisatie specifieke filterregel-definities worden toegepast. Voorgedefinieerde filterregels hebben gereserveerde identificatienummers (id=) van 1 t/m 500. Deze filterregels kunnen niet worden verwijderd en de filterregels 1 t/m 450 kunnen tevens niet worden aangepast.

25.5 Role Based Firewall Control

Voor het beheerbaar en controleerbaar opzetten van Firewall filterregels dient Role Based Firewall Control (RBFC) te worden gehanteerd. Hiermee worden gebruikers/gebruikersgroepen systematisch netwerkresources zoals applicaties toebedeeld. RBFC bestaat uit 3 matrixes waarvan de eerste informatie over rollen in de organisatie bevat, de derde matrix bevat het IP-nummerplan/schema en de tweede, tussenliggende, matrix koppelt de rol-informatie aan de IP-nummers.

25.6 Configuratie Filterregels

Na het opstellen van de RBFC-matrixes kunnen de Firewall filterregels worden gedefinieerd. De logische volgorde hierbij is de definities voor:

1. netwerkobjecten
2. netwerkobjectgroepen
3. filterregels
4. service
5. connecties
6. activering

25.6.1 Netwerkobjecten

Allereerst zullen de te controleren netwerkresources aan de Firewall bekend moeten worden gemaakt door deze te definiëren met het 'fwnwobj' lijncommando. De typen netwerkresources zijn een host, network, firewall, router, interface of VPN. Enige voorbeelden van netwerkresource definities:

```
Definitie van een z/OS-systeem in Amsterdam:
fwnwobj     cmd=add   \
            name="PROD AMS"   \
```

```
                desc="Productiesysteem lokatie Amsterdam"  \
                type=Host  \
                addr=172.16.234.1  \
                mask=255.255.255.255

Definitie van een z/OS-systeem in Rotterdam:
fwnwobj         cmd=add  \
                name="PROD ROT"  \
                desc="Productiesysteem lokatie Rotterdam"  \
                type=Host  \
                addr=172.16.235.2  \
                mask=255.255.255.255

Definitie van een z/OS-systeem in Utrecht:
fwnwobj         cmd=add  \
                name="PROD UTR"  \
                desc="Productiesysteem lokatie Utrecht"  \
                type=Host  \
                addr=172.16.236.3  \
                mask=255.255.255.255

Definitie van een Local Area Network in Amsterdam:
wnwobj          cmd=add  \
                name="LAN AMS1"  \
                desc="Local Area Network Amsterdam, 4e etage"  \
                type=Network  \
                addr=172.16.240.0  \
                mask=255.255.255.0

Definitie van een tweede Local Area Network in Amsterdam:
fwnwobj         cmd=add  \
                name="LAN AMS2"  \
                desc="Local Area Network Amsterdam, 5e etage"  \
                type= Network  \
                addr=172.16.245.0  \
                mask=255.255.255.0

Definitie van een derde, beperkt, Local Area Network in Amsterdam:
fwnwobj         cmd=add  \
                name="LAN AMST"  \
                desc="Local Area Network Amsterdam, Test"  \
                type= Network  \
                startaddr=172.16.246.1  \
                endaddr=172.16.246.12

Definitie van een derde, beperkt, Local Area Network in Rotterdam:
fwnwobj         cmd=add  \
                name="LAN ROTT"  \
                desc="Local Area Network Rotterdam, Test"  \
                type= Network  \
                startaddr=172.16.246.17  \
                endaddr=172.16.246.31

Definitie van de Firewall in Rotterdam:
fwnwobj         cmd=add  \
                name="Firewall Rotterdam"  \
                desc="Firewall op productie systeem Rotterdam"  \
                type=Firewall  \
                addr=172.16.240.17  \
```

mask=*255.255.255.255*

Opmerking: Op de website www.iana.org kan men informatie vinden over het gebruik van IP-nummers. De IANA organisatie stelt dat een aantal reeksen IP-nummers niet voor extern gebruik zijn toegestaan. Deze zijn:

- 10.x.x.x voor een gereserveerd classe A-netwerk;
- 172.16-31.x.x voor 16 verschillende classe B-netwerken;
- 192.168.x.x voor 256 verschillende classe C-netwerken.

In de praktijk kunnen en worden deze nummerreeksen veelal wel voor intern gebruik toegepast. Zie hiervoor het Network Address Translation (NAT) gedeelte.

De Firewall heeft een voorgedefinieerd netwerkobjectdefinitie, beschrijvend de totale netwerkomgeving genaamd 'The World'. De definitie kan met de list-parameter zichtbaar worden gemaakt:

```
fwnwobj     cmd=list   name="The World"
```

Het systeem geeft de volgende output:

```
id = 1
type = Network
name = The World
desc =
addr = 0.0.0.0
mask = 0.0.0.0
startaddr =
endaddr =
```

25.6.2 Netwerkobjectgroepen

Netwerkobjecten kunnen voor vereenvoudiging van administratie tot groepen worden samengevoegd. Een voorbeeld waarbij de productiesystemen *PROD AMS* en *PROD ROT* in één netwerkobjectgroep worden gedefinieerd en een voorbeeld waarbij *LAN AMS1* en *LAN AMS2* in één netwerkobjectgroep worden gedefinieerd:

```
fwnwgrp     cmd=create  \
            name="Productiesystemen"  \
            desc="Productiesystemen Amsterdam en Rotterdam"  \
            namelist="PROD AMS|PROD ROT"

fwnwgrp     cmd=create  \
            name="Verkoop afdeling"  \
            desc="Verkoop afdeling in Amsterdam"  \
            namelist="LAN AMS1|LAN AMS2"
```

Elk gedefinieerd netwerkobjectgroep krijgt een uniek identifier (id) toebedeeld. Voor de eenvoud bij verdere bewerking van de netwerkobjectgroepdefinitie kan deze identifier in andere lijncommando's worden opgenomen. De toebedeelde identifier kan met de list-parameter worden opgevraagd. Een voorbeeld:

```
fwnwgrp     cmd=list  \
            name="Productiesystemen"
```

Met het voorbeeld geeft het systeem de volgende output met daarin de toebedeelde identifier (id = 929):

```
id = 929
```

```
name = Productiesystemen
desc = Productiesystemen Amsterdam en Rotterdam
idlist = 834, 925
```

Naar behoefte kunnen netwerkobjecten aan de netwerkobjectgroep worden toegevoegd. Een voorbeeld:

```
fwnwgrp     cmd=add   \
            id=929   \
            namelist="LAN AMST"
```

Verder is het mogelijk om de netwerkobjecten uit de netwerkobjectgroep te verwijderen met de cmd=remove parameter of in het uiterste geval de gehele netwerkobjectgroep te verwijderen met de cmd=delete parameter. De voorbeelden:

Uit de lijst verwijderen van één of meerdere netwerkobjecten:

```
fwnwgrp     cmd=remove   \
            id=929   \
            namelist="LAN AMST"
```

De lijst in zijn geheel verwijderen:

```
fwnwgrp     cmd=delete   \
            id=929
```

25.6.3 Filterregels

Voor het toelaten of blokkeren van netwerkverkeer moeten filterregels worden gedefinieerd. Door implementatie van de standaard door de leverancier meegeleverde filterregels wordt geen enkel netwekverkeer toegestaan en organisatie specifieke filterregels moeten dan worden gedefinieerd om netwerkverkeer mogelijk te maken. De volgende standaard filterregels moeten worden geïmplementeerd:

Naast de standaard filterregels, die geen enkel netwerkverkeer toelaat, zal de Firewall Security administrator de noodzakelijke filterregels moeten definieren om het gewenste netwerkverkeer toe te laten. De filterregels geven aan welke poortnummers voor welk protocol mogen worden gebruikt. Voor informatie over poortnummers verwijzen wij naar het Internet-adres www.iana.org/assignments/port-numbers. Hieronder een voorbeeld van specifieke filterregels voor het toestaan van HTTP-netwerkverkeer:

Een filterregel voor het toestaan van een aanvraag voor een HTTP-sessie (voor het kunnen opvragen van webpages) vanuit bijvoorbeeld Internet:

```
fwfrule     cmd=add   \
            name="Web requests"   \
            desc="Web request vanuit Internet "   \
            type=permit   \
            protocol=tcp   \
            srcopcode=gt   srcport=1023   \
            destopcode=eq   destport=80   \
            interface=nonsecure   \
            routing=local   \
            direction=inbound   \
            log=yes
```

Een filterregel voor het kunnen accepteren van een HTTP-sessie naar bijvoorbeeld een Internet-gebruiker:

```
fwfrule     cmd=add   \
            name="Web response"   \
```

```
desc="Web response naar Internet "   \
type=permit   \
protocol=tcp/ack   \
srcopcode=eq   srcport=80   \
destopcode=gt   destport=1023   \
interface=nonsecure   \
routing=local   \
direction=outbound   \
log=yes
```

Voor het opvragen van de toebedeelde identificaties (id=) kan de list-parameter worden toegepast. De voorbeelden:

Voorbeeld 1:

```
fwfrule      cmd=list   \
             name="Web requests"
```

Met het voorbeeld geeft het systeem de volgende output met daarin de toebedeelde identifier (id = 948):

```
id = 948
name = Web requests
desc = Web request vanuit Internet
```

Voorbeeld 2:

```
fwfrule      cmd=list   \
             name="Web response"
```

Met het voorbeeld geeft het systeem de volgende output met daarin de toebedeelde identifier (id = 949):

```
id = 949
name = Web response
desc = Web response naar Internet
```

25.6.4 Filterregels voor gekoppelde netwerken

In de praktijk zullen over het algemeen meerdere netwerkadapters voor één enkele z/OS-image in gebruik zijn. Gesteld kan worden dat het netwerkverkeer van buiten een z/OS onveilig is tenzij het een netwerkverbinding naar een ander, vertrouwde z/OS-omgeving is. Voor de filtering van netwerkverkeer kan de Firewall dan aparte filterregels toepassen. Als een netwerkadapter een netwerkverbinding geeft met een onveilige omgeving moet deze als onveilig worden gedefinieerd. Een voorbeeld van een Gigabit Ethernet adapter:

```
fwadapter    cmd=change
             addr=172.16.240.0
             state=nonsecure
```

Voor netwerkverkeer vanuit een andere z/OS-omgeving met een gelijk beveiligingsniveau kan de netwerkadapter als veilig worden beschouwd. Een netwerkverbinding met een systeem op een lager beveiligingsniveau, zoals een testsysteem, zal als onveilig moeten worden gedefinieerd. Een voorbeeld definitie van een CTC-netwerkverbinding op een productiesysteem dat een koppeling verzorgt naar een z/OS-testsysteem:

```
fwadapter    cmd=change
             addr=172.16.200.0
             state=nonsecure
```

Een voorbeeld op een z/OS-systeem van een HyperChannel dat een netwekverbinding verzorgt met een gelijkwaardig beveiligingsniveau zoals een ander z/OS-systeem in een andere LPAR met eenzelfde beveiligingsniveau:

```
fwadapter   cmd=change
            addr=172.16.210.0
            state=secure
```

Door de ‘fwadapter’ definitie, die aangeeft of een adapter veilig of onveilig netwerkverkeer met zich meebrengt, kan de Firewall beoordelen of filterregels wel of niet moeten worden toegepast. De Firewall zal filterregels toepassen voor een veilige of onveilige adapter afhankelijk van de ‘fwfrule interface=’ parameter. Een voorbeeld voor HTTP-verkeer met Internet-gebruikers:

```
fwfrule     cmd=add   \
            name="Web requests"   \
            desc="Web request vanuit Internet "   \
            type=permit   \
            protocol=tcp   \
            srcopcode=gt   srcport=1023   \
            destopcode=eq   destport=80   \
            interface=nonsecure   \
            routing=local   \
            direction=inbound   \
            log=yes

fwfrule     cmd=add   \
            name="Web response"   \
            desc="Web response naar Internet "   \
            type=permit   \
            protocol=tcp/ack   \
            srcopcode=eq   srcport=80   \
            destopcode=gt   destport=1023   \
            interface=nonsecure   \
            routing=local   \
            direction=outbound   \
            log=yes
```

25.6.5 Services

Met filterregels kan worden aangegeven welke typen sessie kunnen worden opgezet. Met de netwerkobject- en netwerkobjectgroepdefinities kan worden aangegeven over welke netwerkgebruikers- en systemen een netwerksessie kan worden toegestaan. Deze veschillende definities kunnen worden gecombineerd tot één definities waarbij dan zowel de IP-adressen waarbij een sessie mag plaatsvinden en het sessie type hier tussen wordt gedefinieerd. Het geheel is daarbij als services gedefinieerd. Een voorbeeld waarbij elke willekeurige Internet/Intranet gebruiker een HTTP-sessie met de productiesystemen kan opzetten:

```
fwservice   cmd=create   \
            name="Web toegang"   \
            desc="Toegang productie Web Servers vanuit externe netwerken"   \
            rulelist=948/f,949/b
```

Opmerking: bij de lijst van filterregels zijn de extra parameters opgenomen /f en /b opgenomen. De parameter /f betekend foreward wat inhoud dat de IP adressen voor de source (aanvrager) en destination (beantwoorder) worden gecontroleerd op de volorde zoals gespecificeerd in een filterregel, echter in het voorbeeld is geen IP-adres in de definitie opgenomen. Een source en destination IP-adres is echter wel aanwezig in het IP-packet dat van de aanvrager komt en zal door de firewall worden geregistreerd. Voor het uitgaande verkeer, als antwoord (acknowlegment) op de aanvraag, zal het origineel geregistreerde source IP-adres en het

destination IP-adres omgekeerd moeten worden gecontroleerd. Het omgekeerd controleren van de source als destination en destination als source IP-adres kan worden aangegeven door de /b (backward) parameter voor het betreffende filterregel in de lijst van filterregels. Voor het definieren van /f en /b parameter zal de flow van het betreffende netwerkprotocol bekend moeten zijn. De leverancier en algemene netwerkdocumentatie kan hier uitsluitsel over geven. Een voorbeeld van de /f en /b parameter:

```
fwservice    cmd=create  \
             name="Web Server"  \
             desc="Toegang productie Web Servers vanuit externe netwerken"  \
             rulelist=948/f,949/b
```

Met behulp van tijdslots kan de service worden beperkt. Een voorbeeld waarbij de Web-service uitsluitend op Vrijdag 16 juli 2004 van 11.00 – 12.00 uur gebruikt kan worden:

```
fwservice    cmd=change  \
             name="Web Server"  \
             rulelist=948/f,949/b  \
             time=1100-1200  \
             month=Jul  \
             day=16  \
             timefilter=activate
```

Voor het opvragen van de toebedeelde identificaties (id=) kan de list-parameter worden toegepast. De voorbeelden:

```
fwservice    cmd=list  \
             name="Web Server"
```

Met het voorbeeld geeft het systeem de volgende systeemoutput met daarin de toebedeelde identifier (id = 993):

```
id = 993
name = Web Server
desc = Toegang productie Web Servers vanuit externe netwerken
rulelist=948/f,949/b
```

Opmerking: voor het aanpassen of verwijderen van filterrules zijn de parameters cmd=add, cmd=move, cmd=remove en cmd=delete aanwezig welke hier verder worden beschreven.

Zoals eerder aangegeven zijn er voor veel voorkomende netwerkdiensten voorgedefinieerde definities aanwezig. Hieronder een voorbeeld van de aanwezige service-definitie voor het ping-commando:

```
fwservice    cmd=list  id=18
```

Met de tandaard ping-service-definitie (id=18) geeft het systeem de volgende output:

```
id = 18
name = Ping
desc = Permit Ping outbound secure network to anywhere
rulelist = 13/f,12/b
log =
fragment =
tunnel =
time =
month =
day =
weekday =
```

```
timefilter =
```

De ping-service-definitie is opgebouwd uit de 2 rule-definities id=13 en id=12 waarvan hieronder de voorbeelden.

```
fwfrule     cmd=list   id=13
```

Met de standaard ping-rule-definitie (id=13) geeft het systeem de volgende output:

```
id = 13
type = permit
name = Ping
desc = ICMP port 8
protocol = icmp
srcopcode = eq
srcport = 8
destopcode = eq
destport = 0
interface = both
routing = both
direction = both
log = no
tunnel =
fragment = yes
```

```
fwfrule     cmd=list   id=12
```

Met de standaard ping-rule-definitie (id=12) geeft het systeem de volgende output:

```
id = 12
type = permit
name = Ping Response
desc = ICMP port 0
protocol = icmp
srcopcode = eq
srcport = 0
destopcode = eq
destport = 0
interface = both
routing = both
direction = both
log = no
tunnel =
fragment = yes
```

25.6.6 Connecties

De doelstelling van een netwerk is om een verbinding (connectie) tot stand te brengen terwijl de Firewall het gebruik van deze verbinding controleerd. Met een Firewall connection-definitie kan een service-definitie worden gekoppeld aan een specifieke netwerkverbinding. In het onderstaande voorbeeld kunnen Internet-gebruikers van de z/OS HTTP Server (Web Server) gebruik maken:

```
fwconns     cmd=create   \
            name="Internet verbinding"    \
            desc="Internet verbinding voor publiek gebruik"    \
            source="The World"   \
            destination="PROD ROT"   \
            servicelist=993
```

Verder zijn er verschillende ‘fwconns’ subcommando’s om services in de connectiedefinitie aan te passen of in zijn geheel te verwijderen. Dit zijn de cmd=change voor het aanpassen van de connectiedefinitie, de cmd=delete voor het verwijderen van connectiedefinities en de cmd=addservice, moveservice en removeservice voor het aanpassen van de service in de connectdefinitie.

Verder is er het ‘fwconns cmd=move’ subcommando voor het aanpassen van de zoekvolgorde van de verschillende connectdefinities. De relatieve plaats van de connectiedefinitie bepaald welke het eerst wordt gevonden en is daarmee bepalend voor het wel of niet toestaan van een verbinding. Een meer generieke connectdefinitie moet daarom ook als laatste worden gevonden terwijl een meer specifieke connectdefinitie als eerste moet worden gevonden. Verder is het van belang de meest gebruikte connectdefinities, rekeninghoudend met de noodzaak voor het plaatsen van specifieke connectdefinities aan het begin van de zoekvolgorde, vooraan in de gehele set te plaasten om de efficiëntie van het zoeken te verhogen en daarmee de performance van de Firewall te optimaliseren.

25.6.7 Activeren filterregels

Na het aanbrengen van Firewall-definities moeten deze voor gebruik worden geactiveerd. Het dient aanbeveling de Firewall-definities als eerste op syntax te toetsen met het fwfilter cmd=verify lijncommando en bij fouten deze eerst te corrigeren. Het voorbeeld:

```
fwfilter  cmd=verify
```

Het activeren van de Firewall-definities wordt met het fwfilter lijncommando uitgevoerd. Het voorbeeld:

```
fwfilter  cmd=update
```

Indien het noodzakelijk is om de Firewall filterregels te deactiveren kan het fwfilter cmd=shutdown lijncommando worden gebruikt. Echter dit is niet aan te raden aangezien de standaard filterregels die dan worden geactiveerd alle netwerkverkeer toelaten wat in de meeste gevallen nooit mag plaatsvinden. Een betere oplossing is om in noodgevallen een filterregel te definiëren en te activeren die al het netverkeer beperkt. Een voorbeeld:

```
fwconns     cmd=create   \
            name="Stop alle netwerkverkeer naar Rotterdam"    \
            desc="Uitsluitend voor noodgevallen"    \
            source="The World"   \
            destination="PROD ROT"   \
            servicelist=

fwfilter    cmd=update
```

25.6.8 Globale Filterregels

Voor situaties waarbij netwerkverkeer direct moet worden gestopt zijn voorgedefinieerde filterregels aanwezig. Hieronder zijn deze globale filterregels opgenomen die in het systeem aanwezig zijn echter voordat deze in de praktijk worden toegepast moet altijd de laatste informatie vanuit het systeem worden geraadpleegd. Dit kan met het ‘fwsecpolicy cmd=list’ lijncommando. Een voorbeeld van de aanwezige globale filterregels:

```
fwsecpolicy  cmd=list
```

Het systeem geeft de volgende output waarbij ‘No’ aangeeft dat deze filterregel inactief is:

```
id=54  No Permit configuration client to secure interface
id=55  No Permit configuration client to non-secure interface
```

```
id=20 No Permit DNSqueries
id=21 No Permit DNSzone transfers
id=34 No Deny Socks to non-secure interface
id=23 No Deny UDP traffic to non-secure interface
id=1  No Deny traffic to non-secure interface
id=26 No Deny traffic to secure interface
id=37 No Deny all traffic
id=25 No Permit all traffic
```

Bij een urgente situaties waarbij netwerkverkeer zo snel mogelijk moet worden gestopt en weinig tijd is om verschillende filterregels aan te passen, kunnen deze globale filterregels worden geactiveerd. Voor het activeren van de globale filterregels wordt het 'fwsecpolicy cmd=change' lijncommando gebuikt. Een voorbeeld waarbij al het netwerkverkeer vanuit onveilige netwerken wordt gestopt:

```
fwsecpolicy  cmd=change   servicelist=54,1,23

fwsecpolicy  cmd=list

Het systeem geeft de volgende output waarbij 'Yes' aangeeft dat deze filterregel
actief zijn:
id=54 Yes Permit configuration client to secure interface
id=1  Yes Deny traffic to non-secure interface
id=23 Yes Deny UDP traffic to non-secure interface
id=55 No Permit configuration client to non-secure interface
id=20 No Permit DNSqueries
id=21 No Permit DNSzone transfers
id=34 No Deny Socks to non-secure interface
id=26 No Deny traffic to secure interface
id=37 No Deny all traffic
id=25 No Permit all traffic
```

Opmerking: de volgorde waarin de globale filterregels worden geactiveerd is van belang voor de uiteindelijke filtering. Indien de eerste filterregel opgaat zal de volgende niet worden gebruikt. Verder zal na het definiëren van de globale filterregels deze met de 'fwfilter cmd=update'nog moeten worden geactiveerd.

```
fwfilter        cmd=update
```

Voor het deactiveren van de globale filterregels zal deze moeten worden verwijderd. De implementatie:

```
fwsecpolicy  cmd=change   servicelist=
fwfilter        cmd=update
```

25.7 Firewall Configuration GUI

Het configureren van Firewall regels kan door het definieren van deze regels in de betreffende files met behulp van een file editor. Echter voor gebruiksgemak is een Graphical User Interface (GUI)-interface meegeleverd die op een PC moet worden geïnstalleerd.

25.7.1 Configuration Server

Voor communicatie tussen de GUI op de PC en de Firewall wordt het TCP/IP-protocol gebruikt. Om de Firewall regels te configureren moet de gebruiker daarbij aan de Firewall Configuration Server aanloggen. Hiervoor moet de Firewall Configuration Server als daemon (z/OS started task) worden geïmplementeerd. De implementatie:

```
RDEFINE   STARTED  ICAPCFGS.*  OWNER(SECADM)
          STDATA(USER(FWU)  GROUP(FWGRP)  PRIVILEGED(NO)
          TRUSTED(NO)  TRACE(YES))

SETROPTS RACLIST(STARTED)
```

Voor het kunnen configureren van Firewall regels is het noodzakelijk dat de gebruiker aan de Firewall-groep, in dit geval *FWGRP*, is geconnect. In geval dat de gebruiker superuser (root) privileges heeft is dit niet nodig echter de norm is dat gebruikers geen superuser prvileges mogen hebben tenzij dit technisch noodzakelijk is. Een voorbeeld:

```
CONNECT  XRIJL01     GROUP(FWGRP)  OWNER(HRMCENTR)
```

Voor het gebruik van de Firewall Configuration Server zal de gebruiker (z/OS Firewall administrator), ook al heeft deze superuser privileges, moeten worden geautoriseerd met het ICA.CFGSRV-profiel in de general resource class FACILITY. Een voorbeeld:

```
RDEFINE   FACILITY  ICA.CFGSRV  OWNER(SECADM)  UACC(NONE)
PERMIT    ICA.CFGSRV  CLASS(FACILITY)  ID(XRIJL01)  ACCESS(READ)

SETROPTS RACLIST(FACILITY)  REFRESH
```

De Configuratie Server daemon zal door de Firewall moeten worden gestart. De implementatie:

```
fwdaemon   cmd=change  \
           daemon=cfgsrv  \
           started=yes
```

25.7.2 Configuration GUI Filterregels

Hieronder is een voorbeeld beschreven voor het gebruik van de Firewall Confguration GUI vanuit een veilig netwerk. Het gebruik van de GUI is voor het gebruiksgemak aan te bevelen echter het gebruik van de lijncommando's voor configuratie van de Firewall filterregels moet voor noodgevallen, waarbij de GUI niet kan worden gebruikt, te allen tijde mogelijk blijven. Een voorbeeld van filterregels voor het autoriseren van de GUI:

```
Definitie van een Configuratie GUI-werkstation:
fwnwobj    cmd=add  \
           name="Firewall Admin"  \
           desc="Firewall administratie werkstation"  \
           type=Host  \
           addr=172.16.236.9  \
           mask=255.255.255.255

fwnwobj    cmd=list  name="Firewall Admin"

Systeemoutput:
           id = 944
           type = Host
           name = Firewall Admin
           desc = Firewall administratie werkstation
           addr = 172.16.236.9
           mask = 255.255.255.255
           startaddr =
           endaddr =

Een filterregel voor het toestaan van een aanvraag voor een HTTP-sessie:
fwfrule    cmd=add  \
           name="Firewall Admin requests"  \
           desc="Firewall administratie requests van werkstation"  \
```

```
                type=permit   \
                protocol=tcp   \
                srcopcode=gt   srcport=1023   \
                destopcode=eq   destport=1014   \
                interface=secure   \
                routing=local   \
                direction=inbound   \
                log=yes
```

Een filterregel voor het kunnen accepteren van een HTTP-sessie naar bijvoorbeeld een Internet-gebruiker:

```
fwfrule         cmd=add   \
                name="Firewall Admin response"   \
                desc="Firewall administratie response naar werkstation"   \
                type=permit   \
                protocol=tcp/ack   \
                srcopcode=eq   srcport=1014   \
                destopcode=gt   destport=1023   \
                interface=secure   \
                routing=local   \
                direction=outbound   \
                log=yes

fwfrule         cmd=list   name="Firewall Admin requests"
```

Systeemoutput fwfrule commando:

```
                id = 956
                name = Firewall Admin requests
                desc = Firewall administratie requests van werkstation


fwfrule         cmd=list   name=" Firewall Admin response"
```

Systeemoutput fwfrule commando:

```
                id = 957
                name = Firewall Admin response
                desc = Firewall administratie response naar werkstation
```

Een definitie van de GUI-service voor het administreren van de Firewalls regels:

```
fwservice       cmd=create   \
                name="Firewall Admin GUI"   \
                desc=" Firewall administratie GUI toegang"   \
                rulelist=956/f,957/b

fwservice       cmd=list   name="Firewall Admin GUI"
```

Systeemoutput fwservice commando:

```
                id = 958
                name = Firewall Admin GUI"
                desc = Firewall administratie GUI toegang"
                rulelist = 956/f,957/b

fwconns         cmd=create   \
                name="Firewall Admin verbinding"   \
                desc="Firewall administrator GUI verbinding"   \
                source="Firewall Admin"   \
                destination="PROD ROT"   \
                servicelist=958
```

25.8 Network Address Translation (NAT) services

Over het algemeen is het niet gewenst om de intern gebruikte IP-adressen aan de buitenwereld (Internet) bekend te maken. Het beste kunnen de gebruikte externe naar interne IP-adressen worden vertaald. Hiervoor kan de Network Address Translation (NAT) service worden toegepast. Met NAT kan bijvoorbeeld een extern IP-adres van bijvoorbeeld 194.109.145.116 vertaald worden naar een intern IP-adres van 10.85.46.44. Met deze adresvertaling wordt er geen informatie aan het Internet bekend gemaakt die enig inzicht geeft in de opbouw van het interne netwerk. Alhoewel het geheel vrij is om alle IP-adressen intern te gebruiken kan het beste de Internet Assigned Numbers Authority (IANA) www.iana.org geraadplegen worden om na te gaan welke IP-adressen het beste voor intern gebruik kunnen worden toegepast. Voor informatie over de toegewezen externe IP-adressen kan het beste de Internetprovider waarbij men is aangesloten worden benaderd.

De netwerkadresvertaling door NAT gebeurt, bij inkomend netwerkverkeer, voordat er TCP/IP routing of IP-filtering plaatsvindt. Bij uitgaand netwerkverkeer vindt netwerkadresvertaling plaats nadat TCP/IP routing of filtering heeft plaatsgevonden. Bij gebruik van NAT dient daarom rekening te worden gehouden met de juiste, vertaalde netwerkadressen in de firewall filterregels.

NAT wordt niet gebruikt voor IP-packets die door een Proxy server worden aangemaakt. Een voorbeeld hiervan is FTP dat zich in de firewall bevindt. FTP gebruikt in dit geval als source-adres het IP-netwerkadres van de firewall. Verder wordt FTP ondersteund door toepassing van IP-adresvertaling door middel van het port-commando.

Opmerking: NAT kan uitsluitend IP-netwerkadressen van TCP en UDP packets vertalen. IP-netwerkadressen van ICMP packets worden dus niet vertaald.

25.9 Vertalingsregels

De NAT heeft 3 mogelijkheden om een IP-adres te vertalen. Deze mogelijkheden zijn:

1. vertaling (map) via een vaste NAT-tabel waarbij altijd dezelfde één op één vertaling van een specifiek extern naar specifiek intern IP-adres plaatsvindt.
2. vertaling (translate) met een dynamische NAT-tabel waarbij een extern IP-adres een willekeurig intern IP-adres binnen een aangegeven range krijgt toegekend.
3. geen vertaling (exclude) van extern naar intern IP-adres en wordt daarmee uitgesloten van vertaling.

Voor het definiëren van de NAT-vertaalregels wordt het ‘fwnat cmd=add type=’ gebruikt.

25.9.1 IP-adres vaste vertaling

Een voorbeeld waarbij een specifiek intern (secure) IP-adres naar een geregistreerd remote (non-secure) IP-adres wordt vertaald:

```
fwnat       cmd=add   \
            type=map   \
            secaddr=10.85.46.44   \
            remmask=194.109.145.116
```

25.9.2 IP-adres dynamische vertaling

Een voorbeeld waarbij aangegeven wordt dat in een A-classe netwerk alle secure IP-adressen (intern netwerk) moet worden vertaald als deze een connectie zoeken naar een non-secure netwerk zoals het Internet of andere niet betrouwbare netwerken:

```
fwnat       cmd=add   \
            type=translate   \
            addr=10.0.0.0   \
            mask=255.0.0.0
```

Een voorbeeld waarbij aangegeven wordt dat in een C-classe netwerk alle secure IP-adressen (intern netwerk) moet worden vertaald als deze een connectie zoeken naar een non-secure netwerk zoals het Internet of andere niet betrouwbare netwerken:

```
fwnat       cmd=add   \
            type=translate   \
            addr=10.0.0.0   \
            mask=255.255.255.0
```

25.9.3 IP-adres zonder vertaling

Een voorbeeld van een NAT-definitie waarbij voor een range IP-adressen geen vertaling plaatsvindt maar het gebruik van de originele IP-adressen (secure is non-secure) wordt toegestaan:

```
fwnat       cmd=add   \
            type=exclude   \
            addr=10.0.1.0   \
            mask=255.255.255.0
```

25.9.4 Activering / Deactivering NAT vertaalregels

Na het aanbrengen of aanpassen van de NAT-vertaalregels dienen deze te worden geactiveerd. De implementatie:

```
fwnat       cmd=update
```

Deactivering van NAT-vertaalregels mag in de meeste gevallen niet plaatsvinden. Zeker niet als door het deactiveren van de NAT-vertaalregels er IANA-geregistreerde IP-adressen voor het Internet bekend worden die niet extern mogen worden toegepast. Dit zou een overtreding van de Internet gebruiksregels tot gevolg hebben. Voor het deactiveren is het volgende lijncommando aanwezig:

```
fwnat       cmd=shutdown
```

Opmerking: gebruik van ‘cmd=shutdown’ dient met grote zorgvuldigheid te worden toegepast.

25.10 SOCKS services

De SOCKS Server (proxy) is een netwerk gerelateerd applicatieve interface die tussen een client- en serverprogramma gepositioneerd is en daarmee een intermediair is voor al het netwerkverkeer tussen client en server. Bij gebruik van de SOCKS-proxy zoekt een clientprogramma verbinding met de proxy waarbij deze met voorgedefinieerde SOCKS-filterregels (autorisatieregels) controleerd of het gevraagde IP-adres en portnummer mag worden benaderd.

Additioneel kan identificatie en authenticatie van de gebruiker plaatsvinden. De gebruiker moet dan een identity daemon (identd) op zijn systeem (bijvoorbeeld PC) hebben geïmplementeerd. De SOCKS-proxy gebruikt dan de 'identd' als server om met de eindgebruiker te communiceren en identificatie en authenticatie informatie uit te wisselen. De SOCKS-proxy kan het verkregen userid voor verdere autorisatie van de netwerksessie gebruiken en dit aan de server communiceren als dit vereist is.

De netwerksessie kan afhankelijk van de geïmplementeerde encryptie mogelijkheden tussen de client en SOCKS-proxy worden encrypt zodat de integriteit en vertrouwelijkheid van het netwerkverkeer over onveilige netwerken wordt gewaarborgd zonder dat hiervoor extra maatregelen nodig zijn.

Een voorwaarde voor het gebruik van de SOCKS-proxy, en daarmee het SOCKS-autorisatie mechanisme, is dat de client-applicatie zogenaamd 'socksified' moet zijn, oftewel de SOCKS-application interface gebruikt.

25.10.1 SOCKS Proxy toepassing

Bij gebruik van de SOCKS-proxy op een end-node host kan het cliëntprogramma, op bijvoorbeeld een PC, door de SOCKS-proxy worden geïdentificeerd en geauthenticeerd en afhankelijk van de geïmplementeerde encryptie mogelijkheden de sessie encrypten. De eindgebruiker kan daarom via of vanuit een onveilig netwerk toegang zoeken met behoud van de vertrouwelijkheid en integriteit van zijn/haar gegevens.

Is de gewenste applicatieserver niet op dezelfde host geïmplementeerd dan zal de sessie tussen de SOCKS-proxy en de applicatieserver alsnog moeten worden geëncrypt om de vertrouwelijkheid en integriteit van het netwerkverkeer te waarborgen. In het geval van z/OS zal in de meeste gevallen zowel de SOCKS-proxy als de gegevens op dezelfde host aanwezig zijn. Van netwerkverkeer van de SOCKS-proxy via een onveilig netwerk naar de geraadpleegde gegevens zal dan geen sprake zijn.

De SOCKS-proxy kan tevens voor uitgaand netwerkverkeer worden toegepast. Met uitgaande netwerkverkeer zal deze van de eigen vertrouwde netwerkomgeving naar een onveilig netwerk zoals het Internet of een Intranet gaan. De SOCKS-proxy kan dan zorgdragen voor het verborgen houden van het originele netwerkadres voor de buitenwereld.

25.10.2 SOCKS Started Task

De SOCKS-proxy of SOCKS-server is een daemon die als z/OS started task wordt geactiveerd. Een voorbeeld:

```
RDEFINE   STARTED  ICAPSOCK.*  OWNER(SECADM)
          STDATA(USER(FWU)  GROUP(FWGRP)  PRIVILEGED(NO)
          TRUSTED(NO)  TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

De SOCKS-server daemon zal als onderdeel van de Firewall moeten worden gestart en in de configuratie moeten worden opgenomen. De implementatie:

```
fwdaemon   cmd=change  \
           daemon=sockd  \
           started=yes
```

25.10.3 Socks filterregels

Voor het definiëren van de SOCKS-filterregels (filter rules) kan het lijncommando “fwsrule” of de configuratie GUI worden toegepast. Een voorbeeld waarbij altijd de SOCKS-proxy wordt gebruikt voor identicatie en authenticatie van de gebruiker voor het gebruik van FTP (File Transport Protocol).

```
fwsrule     cmd=add   \
            name=”FTP“    \
            desc=”Socksified FTP“    \
            type=permit    \
            identd=Always    \
            userid=XKRET01,XRIJL01    \
            operator=eq    \
            port=21    \
            rulecommand=”FTP“
```

Voor het opvragen van de toebedeelde identificaties (id=) kan de list-parameter worden toegepast. De voorbeelden:

```
fwsrule     cmd=list   \
            name=”FTP”
```

Met het voorbeeld geeft het systeem de volgende output met daarin de toebedeelde identifier (id = 998):

```
id      = 998
name = FTP
desc   = Socksified FTP
```

25.10.4 Connections en services definities

De doelstelling van een netwerk is om een verbinding (connectie) tot stand te brengen terwijl de Firewall het gebruik van deze verbinding controleert. Met een SOCKS-connection-definitie kan een SOCKS-definitie aan een specifieke netwerkverbinding worden gekoppeld. In het onderstaande voorbeeld kunnen Internet-gebruikers van de z/OS SOCKS-proxy gebruik maken om daarmee andere services zoals FTP, geïdentificeerd en geauthenticeerd te benaderen:

```
fwconns     cmd=create   \
            name=”Internet SOCKS-verbinding”    \
            desc=”Internet SOCKS-verbinding voor publiek gebruik”    \
            source=”The World”   \
            destination=”PROD ROT”   \
            sockslist=998
```

25.10.5 Emergency Socks shutdown

Bij een urgente situaties waarbij het SOCKS-netwerkverkeer zo snel mogelijk moet worden gestopt en weinig tijd is om verschillende filterregels aan te passen, kunnen de reeds beschreven globale filterregels worden geactiveerd. Voor het activeren van de globale filterregels wordt het ‘fwsecpolicy cmd=change’ lijncommando gebuikt. Een voorbeeld waarbij al het SOCKS-netwerkverkeer vanuit onveilige netwerken wordt gestopt:

```
fwsecpolicy  cmd=change   servicelist=34

fwsecpolicy  cmd=list
```

Het systeem geeft de volgende output waarbij ‘Yes’ aangeeft dat de SOCKS-filterregel actief is:

```
    id=34 Yes Deny Socks to non-secure interface
```

```
id=54 No Permit configuration client to secure interface
id=55 No Permit configuration client to non-secure interface
id=20 No Permit DNSqueries
id=21 No Permit DNSzone transfers
id=23 No Deny UDP traffic to non-secure interface
id=1  No Deny traffic to non-secure interface
id=26 No Deny traffic to secure interface
id=37 No Deny all traffic
id=25 No Permit all traffic
```

Opmerking: bij gebruik van meerdere globale filterregels is de volgorde waarin deze worden geactiveerd van belang voor de uiteindelijke filtering. Indien de eerste filterregel opgaat zal de volgende niet worden gebruikt. Verder zal na het definiëren van de globale filterregels deze met de ‘fwfilter cmd=update’nog moeten worden geactiveerd.

```
fwfilter        cmd=update
```

Voor het deactiveren van de globale filterregels zal deze moeten worden verwijderd. De implementatie:

```
fwsecpolicy  cmd=change   servicelist=
fwfilter        cmd=update
```

26 Intrusion Detection System (IDS)

26.1 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

27 Webserver

27.1 Introductie

27.2 Status

Dit hoofdstuk is op OS/390 release 2.8 en moet nog naar z/OS 1.2 worden gebracht. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

27.3 Webserver proces

De OS/390 Webserver is een started task waarvan het betreffende userid en groepid met een UID en GID moeten worden gedefinieerd. De implementatie:

```
ADDGROUP  WEBGRP  SUPGROUP(USSMAIN) OWNER(USSMAIN)
          OMVS(GID(0))
ADDUSER   WEBU    NAME('Webserver daemon user ID')  OWNER(SECADM)
          NOPASSWORD  DFLTGRP(WEBGRP)
          OMVS(UID(0)  HOME('/')  PGM())

RDEFINE   STARTED  IMWEBSRV.*  OWNER(SECADM)
          STDATA(USER(WEBU) GROUP(WEBGRP)  PRIVILEGED(NO)
          TRUSTED(NO)  TRACE(YES))

SETROPTS RACLIST(STARTED)  REFRESH
```

Om hacking attempts te reduceren en het laden van installatie vreemde programma's te voorkomen moeten alle programma's door RACF worden gecontroleerd (program controlled). Programma's die niet bij RACF bekend zijn kunnen dan niet worden geladen en daardoor niet worden uitgevoerd. De implementatie voor programma control voor de Webserver programma's:

```
ADDSD     'SYS1.SIMWMOD1'  GENERIC  OWNER(SECADM)  UACC(READ)
RALTER    PROGRAM  **  ADDMEM('SYS1.SIMWMOD1'/ /NOPADCHK)
          UACC(READ)

SETROPTS WHEN(PROGRAM)  REFRESH
```

27.4 Daemon

Om de gebruikersidentiteit te kunnen veranderen met de system calls setuid, seteuid, setreuid en spawn met een userid, kan door middel van het RACF-profiel BPX.DAEMON in de FACILITY class worden geautoriseerd. Als dit profiel gedefinieerd is met UACC(NONE) dan moeten alle UID(0) gebruikers, die deze system calls moeten kunnen uitvoeren, worden geautoriseerd. Over het algemeen hebben daemons die onder UID(0) worden gestart zoals inetd, rlogind, cron, Im, uucpd, syslog, etc., deze system calls nodig.

Met het BPX.DAEMON profiel gedefinieerd wordt ook bereikt dat alle programma's die door een daemon worden geladen en deze wil uitvoeren 'program controlled' moeten zijn. Hierdoor kunnen geen installatie vreemde programma's worden geladen en uitgevoerd. De implementatie hoe de Webserver de daemon eigenschappen krijgt:

```
RDEFINE   FACILITY  BPX.DAEMON  OWNER(SECADM)  UACC(NONE)
PERMIT    BPX.DAEMON  CLASS(FACILITY)  ID(WEBU)  ACCESS(READ)
```

```
SETROPTS RACLIST(FACILITY)  REFRESH
```

De Webserver is gedefinieerd als een server en dient in de BPX.SERVER profiel in de FACILITY class te worden gedefinieerd. Door het definiëren in de BPX.SERVER profiel geeft de Webserver rechten om een processen onder het userid van de Client (werkstation) uit te voeren en de toegangsrechten van de Client op OS/390 resources te controleren. De implementatie:

```
RDEFINE   FACILITY  BPX.SERVER  OWNER(SECADM)  UACC(NONE)
PERMIT    BPX.SERVER  CLASS(FACILITY)  ID(WEBU)  ACCESS(UPDATE)

SETROPTS RACLIST(FACILITY)  REFRESH
```

27.5 Ongedefinieerde gebruikers

Voor het geven van service aan niet gedefinieerde gebruikers, zoals onbekende Internet gebruikers, maakt de Webserver gebruik van een specifiek userid (executing userid) voor het opstarten van processen voor deze gebruikers. Het executing userid mag uitsluitend die rechten hebben die nodig zijn om gebruikers de noodzakelijke service te geven. Voor het opstarten van processen onder het executing userid, dient dit userid aan de Webserver bekend te worden gemaakt. De Webserver leest hiervoor het RACF-profiel BPX.SRV.userid in de SURROGATE class. Tevens dient het Webserver userid (surrogating userid) te worden geautoriseerd om een proces onder het executing userid te mogen opstarten. De implementatie:

```
ADDGROUP  WEBPUBG  SUPGROUP(USSMAIN)  OWNER(USSMAIN)
            OMVS(GID(1003))
ADDUSER   WEBPUBU  NAME('Webserver userid voor surrogating')
          OWNER(SECADM)
          NOPASSWORD  DFLTGRP(WEBPUBG)
          OMVS(UID(1003)  HOME('/')  PGM('/bin/sh'))

RDEFINE   SURROGAT  BPX.SRV.WEBPUBU  OWNER(SECADM)
          UACC(NONE)

PERMIT    BPX.SRV.WEBPUBU  CLASS(SURROGAT)  ID(WEBGRP)
          ACCESS(READ)

SETROPTS CLASSACT(SURROGAT)
SETROPTS RACLIST(SURROGAT)
```

27.6 Certificaten

De Webserver kan gebruik maken van certificaten voor onder andere de Secury Socket Layer (SSL) technologie. Dit gedeelte is nog niet uitgewerkt.

28 WebSphere

28.1 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

29 Public Key Infrastructure (PKI) en Digitale Certificaten

29.1 Introductie

In een Public Key Infrastructuur-omgeving (PKI) kan een gebruiker, die met een andere gebruiker wil communiceren, een publieke (openbare) sleutel bij een PKI-server opvragen. Met de publieke sleutel kan de gebruiker alle informatie die hij/zij over een netwerk naar de eigenaar van de publieke sleutel wil verzenden gebruiken voor het versleutelen van het bericht. Alle informatie versleuteld met de publieke sleutel kan uitsluitend met persoonlijke (geheime) sleutel van de eigenaar van de publieke sleutel worden ontsleuteld. De Security Server RACF levert een PKI voor het genereren, registreren, opslaan, verstrekken en terugtrekken van publieke en persoonlijke sleutels en digitale certificaten.

29.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

30 Virtual Private Network (VPN) en Internet Key Exchange (IKE)

30.1 Introductie

In een Virtual Private Networks (VPNs) wordt alle informatie, die over een netwerkverbinding wordt verzonden, versleuteld. Omdat het versleutelen op het netwerk niveau plaatsvindt, genaamd IPSec (IP security), is het niet nodig om applicaties aan te passen voor een veilige informatie overdracht. Voor het versleutelen wordt een symmetrische algoritme toegepast welke in netwerkcomponenten aan weerszijde van de netwerkverbinding aanwezig moet zijn. De Security Server RACF kan het sleutelbeheer, Internet Key Exchange (IKE) bestaande uit aanmaak, opslag en verstrekken van sleutels, voor een VPN-omgeving uitvoeren.

30.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

31 Secure Socket Layer (SSL)

31.1 Introductie

In een Secure Socket Layer (SSL)-omgeving, onderdeel van het TCP/IP-netwerkprotocol, wordt identificatie en authenticatie van een applicatieserver naar een gebruiker toe met een asymmetrisch versleutelalgoritme (public key encryption) uitgevoerd. De openbare sleutel (public key) wordt door de applicatie server met een certificaat aan de gebruiker gedistribueerd. Na verificatie, door de gebruiker (werkplek), van de echtheid van het certificaat kunnen de gebruiker (werkplek) en de applicatie (server) een symmetrische sleutel uitwisselen die voor verdere communicatie wordt gebruikt.

Eventueel kan in de SSL-omgeving de gebruiker ook worden gevraagd zich met een openbare sleutel (public key van de gebruiker) zich te identificeren.

31.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

32 Light-weight Directory Access Protocol (LDAP)

32.1 Introductie

Elektronische certificaten kunnen worden gebruikt om de identiteit van een persoon in het electronische communicatieverkeer vast te kunnen vastleggen. Deze certificaten (identiteit van de persoon) en persoonlijke gegevens zoals functie, telefoonnummer, kamernummer etc. alsmede de plaats van een persoon in een organisatie of instelling kunnen in een directory worden ondergebracht. Bij elektronische gegevensverwerking is te verwachten dat dit type directories intensief zullen worden geraadpleegd. Voor deze directories is een apart protocol ontwikkeld, het Light-weight Directory Access Protocol (LDAP). De Security Server RACF kan samen met de z/OS LDAP-server de RACF-gegevens via het LDAP ontsluiten. Als meer informatie, dan in een RACF repositorie kan worden ondergebracht moet worden opgeslagen kan dit in een DB2-database worden ondergebracht. Voor toepassingen die gebruik maken van de z/OS LDAP-server is het transparant of deze fysiek zijn opgeslagen in de RACF repositorie of DB2-database.

32.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

33 Enterprise Identity Management (EIM)

33.1 Introductie

Met Enterprise Identity Management (EIM) kunnen relaties tussen verschillende userids en/of accounts (identity mapping) worden gelegd. De gebruikersregistraties (registries of directories) waarin de userids en accounts zijn gedefinieerd kunnen daarbij op elk willekeurig type besturingssysteem zijn geplaatst. De relatie tussen de verschillende userids en/of accounts worden in een LDAP-server vastgelegd. Elke applicatie of besturingssysteem dat hiervoor is ingericht kan daarbij de LDAP-server voor identificatie en authenticatie gebruiken. EIM geeft daarbij een central-point-of-user-relations.

33.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

34 Distributed Computing Environment (DCE)

34.1 Introductie

De Distributed Computing Environment (DCE) is een programma interface (middleware) om applicaties platform onafhankelijk te kunnen ontwikkelen en te laten functioneren. De op DCE ontwikkelde applicaties kunnen op praktisch elk modern platform worden uitgevoerd. Het biedt een directory service waarmee een applicatie andere actieve applicatie componenten in het netwerk gevonden kunnen worden. Deze functie wordt de remote procedure call (RPC) genoemd.

Voor het geautoriseerd kunnen aanroepen van andere applicatie componenten is een security server aanwezig die gebaseerd is op het Kerberos protocol. Aangezien de door DCE gebruikte sleutels een beperkte geldigheidsduur kunnen hebben moeten de security server, een geïntegreerd componenten van DCE, de gebruiker en applicatie server dezelfde tijd hebben. Voor het synchroniseren van de tijd over alle aangesloten platformen is een geïntegreerde time server binnen DCE aanwezig.

Verder heeft de DCE-architectuur een eigen gedistribueerd file system. De DCE Security Server gebruikersinformatie (principal information) is opgeslagen in de Security Server RACF registry. Hierdoor is het niet nodig om gebruikersinformatie op twee plaatsen te moeten beheren en kan worden volstaan met normale RACF commando’s voor het gebruikersbeheer.

34.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

35 Kerberos

35.1 Introductie

In een kerberos-omgeving wordt de communicatie tussen een gebruiker (werkplek) en een applicatie (server), met een symmetrische algoritme versleuteld. De sleutel wordt door de kerberos-server gegenereerd en naar de gebruiker en server versleuteld over het netwerk verstuurd. Om een sleutel te krijgen moet de gebruiker en de server zich bij de kerberos-server identificeren en authenticeren. De Kerberos server geeft na een positieve identificatie en authenticatie de gebruiker en applicatieserver een token met daarin versleuteld een symmetrische sleutel. De RACF Security Server kan als Kerberos-server zorgdragen voor het identificeren en authenticeren van de gebruiker en server en daaropvolgend, aan gebruiker en server voor onderlinge communicatie, een symmetrische sleutel verstrekken.

35.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

36 Passtickets

36.1 Introductie

Een PassTicket is een alternatief voor een wachtwoord. Het PassTicket kan door de Security Server RACF of door een geautoriseerd programma voor een gebruiker worden aangemaakt. De gebruiker die het PassTicket heeft aangevraagd kan deze gedurende 10 minuten gebruiken voor identificatie en authenticatie naar een applicatie. Tijdens dit proces wordt geen wachtwoord in leesbare vorm over het netwerk wordt verstuurd. Applicaties kunnen van dit standaard mechanisme gebruik van maken. Voorbeelden van applicaties die hiervoor zijn ingericht zijn:

- Host On-Demand
- Tivoly Global Sign-On
- WebSphere
- APPC, CICS, IMS, TSO en Batch

36.2 Status, nader uit te werken

Dit hoofdstuk moet nog worden uitgewerkt. De lezer die hiervoor zijn bijdrage wil aanleveren is van harte welkom.

37 Bijlage: RACF Classes

RACF class	Description from RACF Command Language Reference SC28 1919-06	Hoofdstuk
ALC Class		
ALCSAUTH	Supports the Airline Control System/MVS (ALCS/MVS) product.	
Terminal Classes		
TERMINAL	Terminals (TSO or VM). See also GTERMINL class.	TCP/IP en FTP
GTERMINL	Resource group class for TERMINAL class.	
APPC Classes		
APPCLU	Verifying the identity of partner logical units during VTAM session establishment.	VTAM
APPCPORT	Controlling which user IDs can access the system from a given LU(APPC port of entry). Also, conditional access to resources for users entering the system from a given LU.	VTAM
APPCSERV	Controlling whether a program being run by a user can act as a server for a specific APPC transaction program (TP).	VTAM
APPCSI	Controlling access to APPC side information files.	VTAM
APPCTP	Controlling the use of APPC transaction programs.	VTAM
Program Classes		
APPL	Controlling access to applications.	TCP/IP
VTAMAPPL	Controlling who can open ACBs from non-APF authorized programs.	
PROGRAM	Controlled programs (load modules).	Protecting dataset, DASD en tapes

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
PMBR	Member class for PROGRAM class (not for use on RACF commands).	N/A
Operations Classes		
CONSOLE	Controlling access to MCS consoles. Also, conditional access to other resources for commands originating from an MCS console.	z/OS Facilities
OPERCMDS	Controlling who can issue operator commands (for example, JES and MVS, and operator commands).	z/OS Facilities
ICSF Classes		
CSFKEYS	Controlling use of Integrated Cryptographics Service Facility (ICSF) cryptographic keys. See also the GCSFKEYS class.	ICSF
GCSFKEYS	Resource group class for CSFKEYS class.	ICSF
CSFSERV	Controlling use of Integrated Cryptographics Service Facility (ICSF) cryptographic services.	ICSF
Device Classes		
DASDVOL	DASD volumes. See also the GDASDVOL class.	
GDASDVOL	Resource group class for DASDVOL class.	
DEVICES	Used by MVS allocation to control who can allocate devices such as: Unit record devices (printers and punches) (allocated only by PSF, JES2, or JES3). Graphics devices (allocated only by VTAM). Teleprocessing (TP) or communications devices (allocated only by VTAM)	JES2
TAPEVOL	Tape volumes.	Protecting dataset, DASD en tapes
Digital Certificates Classes		

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
DIGTCERT	Contains digital certificates and information related to them.	Digitale Certificaten
DIGTRING	Contains a profile for each key ring and provides information about the digital certificates that are part of each key ring.	Digitale Certificaten
PTKTDATA	PassTicket key class enables the security administrator to associate a RACF secured signon secret key with a particular mainframe application that uses RACF for user authentication. Examples of such applications are IMS, CICS, TSO, VM, APPC, and MVS batch.	Digitale Certificaten
z/OS Facility Classes		
FACILITY	Miscellaneous uses. Profiles are defined in this class so that resource managers (typically program products or components of MVS or VM) can check a user's access to the profiles when the users take some action. Examples are catalog operations (DFP) and use of the vector facility (an MVS component). RACF does not document all of the resources used in the FACILITY class by other products. For information on the FACILITY class resources used by a specific product (other than RACF itself), see the product's documentation. Also used for Unix System Service (USS).	xxxxx
DLFCLASS	The data lookaside facility.	z/OS Facilities
TEMPDSN	Controlling who can access residual temporary data sets. You cannot create profiles in this resource class.	xxxxxx
RACF Facility Classes		
FIELD	Fields in RACF profiles (field-level access checking).	Unix System Services (USS)

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
STARTED	Used in preference to the started procedures table to assign an identityduring the processing of an MVS START command.	Groepen en gebruikers, CICS, USS, TCP/IP en FTP, Webserver
SURROGAT	If surrogate submission is allowed, and if allowed, which user IDs can act as surrogates.	Groepen en gebruikers, USS, TCP/IP en FTP, Webserver
PROPCNTL	Controlling if user ID propagation can occur, and if so, for which user IDs (such as the CICS or IMS main task user ID), user ID propagation is not to occur.	CICS
GLOBAL	Global access checking table entry.	Protecting datasets, DASD en tapes
GMBR	Member class for GLOBAL class (not for use on RACF commands).	
RACGLIST	Class of profiles that hold the results of RACROUTE REQUEST=LIST,GLOBAL=YES or a SETROPTS RACLIST operation.	
RACFVARS	RACF variables. In this class, profile names, which start with & (ampersand), act as RACF variables that can be specified in profile names in other RACF general resource classes.	Protecting datasets, DASD en tapes
RVARSMBR	Member class for RACFVARS (not for use on RACF commands).	
RRSFDATA	Used to control RACF remote sharing facility functions.	RRSF
SDSF Classes		
SDSF	Controls the use of authorized commands in the System Display and Search Facility (SDSF). See also GSDSF class.	SDSF
GSDSF	Resource group class for SDSF class.	SDSF
OPC/ESA Class		

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
IBMOPC	Controlling access to OPC/ESA subsystems.	OPC
JES2 Classes		
JESINPUT	Conditional access support for commands or jobs entered into the system through a JES input device.	JES2
JESJOBS	Controlling the submission and cancellation of jobs by job name.	JES2
JESSPOOL	Controlling access to job data sets on the JES spool (that is, SYSIN and SYSOUT data sets).	JES2
WRITER	Controlling the use of JES writers.	JES2
NODES	Controlling the following on MVS systems: Whether jobs are allowed to enter the system from other nodes. Whether jobs that enter the system from other nodes have to pass user identification and password verification checks	JES2
NODMBR	Member class for NODES class (not for use on RACF commands).	JES2
PSFMPL	Used by PSF to perform security functions for printing, such as sepa-rator page labeling, data page labeling, and enforcement of the user printable area.	JES2
B1 Security Classes		
SECDATA	Security classification of users and data (security levels and security categories).	Gebruikers- en data-categorisering
SCDMBR	Member class for SECDATA class (not for use on RACF commands).	
SECLABEL	If security labels are used, and, if so, their definitions.	Gebruikers- en data-categorisering
System Object Manager Class		
SOMDOBJS	Controlling the client's ability to invoke the	

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
	method in the class.	
System View Class		
SYSMVIEW	Controlling access by the SystemView for MVS Launch Window to SystemView for MVS applications.	
Novell Class		
NDSLINK	Mapping class for Novell Directory Services for z/OS user identities.	
Lotus Notes Class		
NOTELINK	Mapping class for Lotus Notes for z/OS user identities.	
CICS Classes		
ACICSPCT	CICS program control table.	CICS
BCICSPCT	Resource group class for ACICSPCT class.	CICS
CCICSCMD	Used by CICS/ESA 3.1, or later, to verify that a user is permitted to use CICS system programmer commands such as INQUIRE, SET, PERFORM, and COLLECT.1	CICS
DCICSDCT	CICS destination control table.	CICS
ECICSDCT	Resource group class for DCICSDCT class.	CICS
FCICSFCT	CICS file control table.	CICS
GCICSTRN	Resource group class for TCICSTRN class.	CICS
HCICSFCT	Resource group class for FCICSFCT class.	CICS
JCICSJCT	CICS journal control table.	CICS
KCICSJCT	Resource group class for JCICSJCT class.	CICS
MCICSPPT	CICS processing program table.	CICS
NCICSPPT	Resource group class for MCICSPPT class.	CICS

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
PCICSPSB	CICS program specification blocks or PSBs	CICS
QCICSPSB	Resource group class for PCICSPSB class.	CICS
SCICSTST	CICS temporary storage table.	CICS
TCICSTRN	CICS transactions.	CICS
UCICSTST	Resource group class for SCICSTST class.	CICS
VCICSCMD	Resource group class for the CCICSCMD class.	CICS
CICS System Manager Classes		
CPSMOBJ	Used by CICSPlex System Manager, which provides a central point of control when running multiple CICS systems, to determine operational controls within a CICS complex.	CICS System Manager
GCPSMOBJ	Resource grouping class for CPSMOBJ.	CICS System Manager
CPSMXMP	Used by CICSPlex System Manager to identify exemptions from security controls within a CICS complex.	CICS System Manager
DB2 Classes		
DSNR	Controlling access to DB2 subsystems.	DB2
DSNADM	DB2 administrative authority class	DB2
GDSNBP	Grouping class for DB2 buffer pool privileges	DB2
GDSNCL	Grouping class for DB2 collection privileges	DB2
GDSNDB	Grouping class for DB2 database privileges	DB2
GDSNPK	Grouping class for DB2 package privileges	DB2
GDSNPN	Grouping class for DB2 plan privileges	DB2
GDSNSC	Grouping class for DB2 schemas	DB2
GDSNSG	Grouping class for DB2 storage group	DB2

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
	privileges	
GDSNSM	Grouping class for DB2 system privileges	DB2
GDSNSP	Grouping class for DB2 stored procedures	DB2
GDSNTB	Grouping class for DB2 table, index, or view privileges	DB2
GDSNTS	Grouping class for DB2 tablespace privileges	DB2
GDSNUF	Grouping class for DB2 user-defined functions	DB2
GDSNUT	Grouping class for DB2 user-defined distinct types	DB2
MDSNBP	Member class for DB2 buffer pool privileges	DB2
MDSNCL	Member class for DB2 collection privileges	DB2
MDSNDB	Member class for DB2 database privileges	DB2
MDSNPK	Member class for DB2 package privileges	DB2
MDSNPN	Member class for DB2 plan privileges	DB2
MDSNSC	Member class for DB2 schemas	DB2
MDSNSG	Member class for DB2 storage group privileges	DB2
MDSNSM	Member class for DB2 system privileges	DB2
MDSNSP	Member class for DB2 stored procedures	DB2
MDSNTB	Member class for DB2 table, index, or view privileges	DB2
MDSNTS	Member class for DB2 tablespace privileges	DB2
MDSNUF	Member class for DB2 user-defined functions	DB2
MDSNUT	Member class for DB2 user-defined distinct types	DB2
MVS/DFP and DFSMS/MVS		

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
Classes		
MGMTCLAS	SMS management classes.	SMS
STORCLAS	SMS storage classes.	SMS
SUBSYSNM	Authorizes a subsystem (such as a particular instance of CICS) to open a VSAM ACB and use VSAM Record Level Sharing (RLS) functions.	SMS
IMS Classes		
TIMS	Transaction (trancode).	IMS
GIMS	Grouping class for transaction.	IMS
AIMS	Application group names (AGN).	IMS
CIMS	Command.	IMS
DIMS	Grouping class for command.	IMS
PIMS	Database.	IMS
QIMS	Grouping class for database.	IMS
SIMS	Segment (in database).	IMS
UIMS	Grouping class for segment.	IMS
FIMS	Field (in data segment).	IMS
HIMS	Grouping class for field.	IMS
OIMS	Other.	IMS
WIMS	Grouping class for other.	IMS
Information Management Classes		
GINFOMAN	Resource group class for Information Management Version 5.	

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
INFOMAN	Member class for Information Management Version 5.	
LFS/ESA Classes		
LFSCLASS	Controls access to file services provided by LFS/ESA.	
MQSeries Classes		
GMQADMIN	Grouping class for MQM administrative options.	MQSeries
GMQCHAN	Reserved for MQM/ESA.	MQSeries
GMQNLIST	Grouping class for MQM namelists.	MQSeries
GMQPROC	Grouping class for MQM processes.	MQSeries
GMQQUEUE	Grouping class for MQM queues.	MQSeries
MQADMIN	Protects MQM administrative options.	MQSeries
MQCHAN	Reserved for MQM/ESA.	MQSeries
MQCMDS	Protects MQM commands.	MQSeries
MQCONN	Protects MQM connections.	MQSeries
MQNLIST	Protects MQM namelists.	MQSeries
MQPROC	Protects MQM processes.	MQSeries
MQQUEUE	Protects MQM queues.	MQSeries
NetView Classes		
NETCMDS	Controlling which NetView commands the NetView operator can issue.	
NETSPAN	Controlling which NetView commands the NetView operator can issue against the resources in this span.	
NVASAPDT	NetView/Access Services.	

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
PTKTVAL	Used by NetView/Access Services Secured Single Signon to store information needed when generating a PassTicket.	
RMTOPS	NetView Remote Operations.	
RODMMGR	NetView Resource Object Data Manager (RODM).	
z/OS UNIX System Services Classes		
DIRAUTH	Setting logging options for RACROUTE REQUEST=DIRAUTH requests. Also, if the DIRAUTH class is active, security label authorization checking is done when a user receives a message sent through the TPUT macro or the TSO SEND, or LISTBC commands. Profiles are not allowed in this class.	Unix System Services
DIRACC	Controls auditing (using SETROPTS LOGOPTIONS) for access checks for read/write access to HFS directories. Profiles are not allowed in this class.	Unix System Services
DIRSRCH	Controls auditing (using SETROPTS LOGOPTIONS) of HFS directory searches. Profiles are not allowed in this class.	Unix System Services
FSOBJ	Controls auditing (using SETROPTS LOGOPTIONS) for all access checks for HFS objects except directory searches. Controls auditing (using SETROPTS AUDIT) of creation and deletion of HFS objects. Pro-files are not allowed in this class.	Unix System Services
FSSEC	Controls auditing (using SETROPTS LOGOPTIONS) for changes to the security data (FSP) for HFS objects. Profiles are not allowed in this class.	Unix System Services
IPCOBJ	Controlling auditing and logging of IPC security checks.	Unix System Services
PROCACT	Controls auditing (using SETROPTS LOGOPTIONS) of functions that look at data	Unix System Services

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
	from, or affect the processing of, z/OS UNIX processes. Profiles are not allowed in this class.	
PROCESS	Controls auditing (using SETROPTS LOGOPTIONS) of changes to UIDs and GIDs of z/OS UNIX processes. Controls auditing (using SETROPTS AUDIT) of dubbing and undubbing of z/OS UNIX processes. Profiles are not allowed in this class.	Unix System Services
UNIXMAP	Contains profiles that are used to map z/OS UNIX UIDs to RACF user IDs and z/OS UNIX GIDs to RACF group names.	Unix System Services
UNIXPRIV	Contains profiles that are used to grant z/OS UNIX privileges.	Unix System Services
Webserver Class		
CBIND	Controlling the client's ability to bind to the server.	Webserver
SERVER	Controlling the server's ability to register with the daemon.	Webserver
SERVAUTH	Contains profiles that are used by servers running on z/OS to check a client's authorization to use the server or to use resources managed by the server.	Webserver
JAVA	Contains profiles that are used by Java for z/OS applications to perform authorization checking for Java for z/OS resources.	Webserver
z/OS DCE Classes		
DCEUUIDS	Used to define the mapping between a user's RACF user ID and the corresponding DCE principal UUID.	DCE
KEYSMSTR	Holds a key to encrypt the DCE password.	DCE
Tivoli Classes		
ROLE	Specifies the complete list of resources and associated access levels that are required to	

RACF class	**Description from RACF Command Language Reference SC28 1919-06**	**Hoofdstuk**
	perform the particular job function this role represents and defines which RACF groups are associated with this role.	
TMEADMIN	Maps the user IDs of Tivoli administrators to RACF user IDs.	
TSO Classes		
ACCTNUM	TSO account numbers.	TSO
PERFGRP	TSO performance groups.	TSO
TSOAUTH	TSO user authorities such as OPER and MOUNT.	TSO
TSOPROC	TSO logon procedures.	TSO
SMESSAGE	Controlling to which users a user can send messages (TSO only).	TSO
IBM Classes		
LOGSTRM	Reserved for MVS/ESA.	
DBNFORM	Reserved for future IBM use.	

38 Bijlage: SETROPTS parameters

Hieronder is een overzicht opgenomen van de SETROPTS parameters volgens de in deze standaard aangegeven instellingen. Verder is een output listing gegeven van het SETROPTS LIST commando, uitgevoerd door een gebruiker met het system-AUDITOR attribuut.

SETROPTS
NOADDCREATOR
NOADSP
[APPLAUDIT | NOAPPLAUDIT]
  [AT([node] .userid ...) | ONLYAT([node] .userid ...)]
AUDIT (*)
[CATDSNS (FAILURES | WARNING) | NOCATDSNS]
CLASSACT (FACILITY RACFVARS TAPEVOL CONSOLE OPERCMDS DLFCLASS STARTED
JESSPOOL JESINPUT JESJOBS WRITER SURROGAT PROPCNTL APPL VTAMAPPL FIELD)
CLASSACT (UNIXMAP UNIXPRIV BPXAS BPXOINIT OMVS DIRAUTH DIRSRCH DIRACC FSOBJ
FSSEC IPCOBJ PROCACT PROCESS) *met USS actief*
CLASSACT (SUBSYSNM TCICSTRN GCICSTRN ACICSPCT BCICSPCT DCICSDCT ECICSDCT
FCICSFCT HCICSFCT JCICSJCT KCICSJCT MCICSPPT NCICSPPT SCICSTST UCICSTST
PCICSPSB QCICSPSB CCICSCMD VCICSCMD) *met CICS actief*
CLASSACT (TIMS GIMS AIMS IMS DIMS PIMS QIMS SIMS UIMS FIMS
HIMS OIMS WIMS) *met IMS acief*
CLASSACT (DSNR DSNADM GDSNBP GDSNCL GDSNDB GDSNPK GDSNPN GDSNSC
GDSNSG GDSNSM GDSNSP GDSNTB GDSNTS GDSNUF GDSNUT
MDSNBP MDSNCL MDSNDB MDSNPK MDSNPN MDSNSC MDSNSG
MDSNSM MDSNSP MDSNTB MDSNTS MDSNUF MDSNUT) *met DB2 actief*
CLASSACT (TERMINAL) *aanvullende maatregel*
CLASSACT (SECDATA SECLABEL) *met B1 faciliteiten actief*
CLASSACT (CSFSERV CSFKEYS) *met ICSF actief*
CMDVIOL
NOCOMPATMODE
EGN
ERASE
GENCMD (*)
GENERIC (*)
GENERICOWNER
[{GENLIST | NOGENLIST} (class-name ...)]
GLOBAL (DATASET JESSPOOL)
GRPLIST
JES (
BATCHALLRACF
[EARLYVERIFY | NOEARLYVERIFY]
XBMALLRACF
NJEUSERID(*UNJEUND*)
UNDEFINEDUSER(*UJESUND*)
)
INACTIVE(90)
INITSTATS
[LANGUAGE(
  [PRIMARY(language)]
[SECONDARY(language)]
)]
LOGOPTIONS(
ALWAYS(OPERCMDS TAPEVOL)
| NEVER(class-name, ...), ...
| SUCCESSES(class-name, ...), ...
FAILURES(DATASET) a*anvullende maatregel*
FAILURES(DIRSRCH) *met USS actief*
DEFAULT(DATASET)
)
MLACTIVE (FAILURES)
[MLQUIET | NOMLQUIET]
MLS (FAILURES)
MLSTABLE

OPERAUDIT
PASSWORD(
HISTORY(13)
INTERVAL(60)
REVOKE(3)
RULE 1 (LENGTH(6:8) ALPHANUM))
WARNING(10))
[PREFIX(prefix) | NOPREFIX]
PROTECTALL (FAILURES)
RACLIST (APPCSERV DEVICES PROPCNTL SECLABEL UNIXPRIV APPCTP FIELD PSFMPL SYSMVIEW VTAMAPPL CSFKEYS NODES PTKTDATA SERVAUTH STARTED CSFSERV OPERCMDS RACFVARS ACCTNUM DBNFORM JESINPUT PERFGRP TERMINAL ALCSAUTH DCEUUIDS JESJOBS PTKTVAL TMEADMIN APPCPORT DIGTCERT JESSPOOL RRSFDATA TSOAUTH APPCSI DIGTRING KEYSMSTR SDSF TSOPROC APPL DLFCLASS LFSCLASS SERVER CBIND DSNR LOGSTRM SMESSAGE VMCMD CONSOLE FACILITY MGMTCLAS SOMDOBJS CPSMOBJ FCICSFCT MQCMDS STORCLAS CPSMXMP INFOMAN MQCONN SUBSYSNM WRITER DASDVOL JAVA NETCMDS SURROGAT)
RACLIST deze classes indien deze zijn geactiveerd met CLASSACT
REALDSN
[RETPD(nnnnn)]
RVARYPW (SWITCH(switch-pw) STATUS(status-pw))
SAUDIT
SECLABELAUDIT *aanvullende maatregel B1 Security*
SECLABELCONTROL *aanvullende maatregel B1 Security*
SECLEVELAUDIT (*CONFIDENTIAL*) *aanvullende maatregel B1 Security*
[SESSIONINTERVAL(n) | NOSESSIONINTERVAL]
STATISTICS (DATASET)
TAPEDSN
TERMINAL(READ)
WHEN(PROGRAM)

SETROPTS LIST *commando dient nog nader te worden ingevuld.*

39 Appendix: Referentie documenten

39.1 Algemene literatuur

39.2 Internet sites

www.s390.ibm.com

39.3 Presentaties

39.4 Documentatie

39.4.1 Security Server (RACF)

Command Language Reference
Command Syntax Summary
Data Areas
Diagnosis Guide
General User's Guide
Introduction
Macros and Interfaces
Messages and Codes
Migration
RACROUTE Macro Reference
Security Administrator's Guide
Security Auditor's Guide
System Programmer's Guide
Security Server (RACF) Planning: Installation and Migration

39.4.2 z/OS

z/Architecture Principles of Operation
Network Job Entry Formats and Protocols
Parallel Sysplex Recovery
Parallel Sysplex Test Report
MVS JCL User's Guide
MVS Auth Assembler Services Guide
MVS Auth Assm Services Reference
MVS Auth Assm Services Reference ENF-IXG
MVS Auth Assm Services Reference LLA-SDU
MVS Auth Assm Services Reference SET-WTO
MVS Callable Services for High Level Languages
MVS Conversion Notebook
MVS Data Areas, Vol 1 (ABEP-DALT)
MVS Data Areas, Vol 2 (DCCD-ITTCTE)
MVS Data Areas, Vol 3 (IVT-RCWK)
MVS Data Areas, Vol 4 (RD-SRRA)
MVS Data Areas, Vol 5 (SSAG-XTLST)
MVS Diagnosis: Procedures
MVS Diagnosis: Reference
MVS Diagnosis: Tools and Service Aids
MVS Dump Output Messages
MVS Extended Addressability Guide

MVS Initialization and Tuning Guide
MVS Initialization and Tuning Reference
MVS Installation Exits
MVS IPCS Commands
MVS IPCS Customization
MVS IPCS User's Guide
MVS JCL Reference
MVS Planning: Global Resource Serialization
MVS Planning: Workload Management
MVS Programming: Assembler Services Guide
MVS Programming: Assembler Services Reference
MVS Programming: Product Registration
MVS Programming: Resource Recovery
MVS Programming: Sysplex Services Guide
MVS Programming: Sysplex Services Reference
MVS Routing and Descriptor Codes
MVS Setting Up a Sysplex
MVS System Codes
MVS System Commands
MVS System Data Set Definition
MVS System Management Facilities
MVS System Messages, Vol 1 (ABA-ASA)
MVS System Messages, Vol 2 (ASB-ERB)
MVS System Messages, Vol 3 (EWX-IEB)
MVS System Messages, Vol 4 (IEC-IFD)
MVS System Messages, Vol 5 (IGD-IZP)
MVS Using the Subsystem Interface
MVS Workload Management Services
MVS Device Validation Support
MVS Writing Servers for APPC/MVS
MVS Writing Transaction Sched for APPC/MVS
MVS JES Common Coupling Services
MVS Planning: Operations
MVS Recovery and Reconfiguration Guide
MVS Using the Functional Subsystem Interface
MVS Planning: APPC/MVS Management
MVS Product Management
MVS Programming: Writing Transaction Programs for APPC/MVS
Parallel Sysplex Overview
Parallel Sysplex Application Migration

39.4.3 JES2

Introduction
Initialization and Tuning Guide
Initialization and Tuning Reference
Commands
Installation Exits
Data Areas, Vol 1($ALINDEX-$EVT)
Data Areas, Vol 2 ($FCLWORK-$OUTWORK)
Data Areas, Vol 3 ($PADDR-$XRQ)
Diagnosis
Macros
Messages
Migration

39.4.4 SMS

Introduction
Implementing System Managed Storage
Using Data Sets
Using Magnetic Tapes
Using the Interactive Storage Management Facility
Using the Volume Mount Analyzer
Program Management

Managing Catalogs
Access Method Services for Catalogs
Character Data Representation Architecture: Reference & Registry
Installation Exits
Macro Instructions for Data Sets
Migration
DFSMSdfp Advanced Services
DFSMSdfp Checkpoint/Restart
DFSMSdfp Diagnosis Guide
DFSMSdfp Diagnosis Reference
DFSMSdfp Storage Administration Reference
DFSMSdfp Utilities
DFSMSdss Storage Administration Guide
DFSMSdss Storage Administration Reference
DFSMShsm Data Recovery Scenarios
DFSMShsm Implementation and Customization Guide
DFSMShsm Managing Your Own Data
DFSMShsm Storage Administration Guide
DFSMShsm Storage Administration Reference
DFSMShsm Storage Administration Reference Summary
DFSMSrmm Application Programming Interface
DFSMSrmm Diagnosis Guide
DFSMSrmm Guide and Reference
DFSMSrmm Implementation and Customization Guide
DFSMSrmm Reporting
Object Access Method Plan, Install and Storage Admin. Gd. Object Support
Object Access Method Plan, Install Guide, and Admin. Gd for Tape Library
Object Access Method Application Programmer's Reference
DFSMS/MVS Advanced Copy Services

39.4.5 TSO

General Information
User's Guide
Primer
Administration
Command Reference
System Programming Command Reference
Customization
CLISTs
Messages
Guide to the SRPI
Programming Guide
Programming Services
REXX User's Guide
REXX Reference
System Diagnosis: Data Areas

39.4.6 ISPF

39.4.7 SDSF

Guide and Reference
Customization and Security

39.4.8 VTAM

Communications Server: AnyNet SNA over TCP/IP
Communications Server: AnyNet Sockets over SNA
Communications Server: APPC Application Suite Administration
Communications Server: APPC Applications Suite Programming
Communications Server: APPC Applications Suite User's Guide
Communications Server: CSM Guide
Communications Server: Resource Definition Samples
Communications Server: IP Configuration Guide

Communications Server: IP Configuration Reference
Communications Server: IP User's Guide and Commands
Communications Server: IPv6 Network and Application Design Guide
Communications Server: SNA Migration and Exploitation
Communications Server: SNA Network Implementation
Communications Server: SNA Operations
Communications Server: SNA Resource Definition Reference
Communications Server: SNA Programmer's LU 6.2 Guide
Communications Server: SNA Programmer's LU 6.2 Reference

39.4.9 OPC

39.4.10 ICSF

Overview
Administrator's Guide
Application Programmer's Guide
System Programmer's Guide
Messages
TKE Workstation User's Guide 2000
TKE Workstation User's Guide
System SSL Programming Gde and Ref
Open Cryptographic Services Facility Appl. Dev.'s Gde. & Ref.
Open Cryptographic Service Facility Svc.Prov.Mod.Dev.Gde.&Ref.

39.4.11 CICS

Transaction Server Planning for Installation
Transaction Server Release Guide
Transaction Server Migration Guide
Transaction Server Installation Guide
Transaction Server Program Directory
Transaction Server Licensed Program Specification
General
Master Index
User's Handbook
Transaction Server Glossary (softcopy only)
Administration
System Definition Guide
Customization Guide
Resource Definition Guide
Operations and Utilities Guide
Supplied Transactions
Programming
Application Programming Guide
Application Programming Reference
System Programming Reference
Front End Programming Interface User's Guide
C++ OO Class Libraries
Distributed Transaction Programming Guide
Business Transaction Services
Diagnosis
Problem Determination Guide
Messages and Codes
Diagnosis Reference
Data Areas
Trace Entries
Supplementary Data Areas
Communication
Intercommunication Guide
CICS Family: Interproduct Communication
CICS Family: Communicating from CICS on System/390
External Interfaces Guide
Internet Guide

Special topics
Recovery and Restart Guide
Performance Guide
IMS Database Control Guide
RACF Security Guide
Shared Data Tables Guide
Transaction Affinities Utility Guide

CICS DB2 Guide

Application Programming Primer (VS COBOL II)
Application Migration Aid Guide
CICS Family: API Structure
CICS Family: Client/Server Programming
CICS Family: General Information
Applications Guide

CICS/ES.1 Sample A XRF Guide

39.4.12 CICSPlex documentatie

General
SM Master Index
SM Concepts and Planning
SM User Interface Guide
SM View Commands Reference Summary
Administration and Management
SM Administration
SM Operations Views Reference
SM Monitor Views Reference
SM Managing Workloads
SM Managing Resource Usage
SM Managing Business Applications
Programming
SM Application Programming Guide
SM Application Programming Reference
Diagnosis
SM Resource Tables Reference
SM Messages and Codes
SM Problem Determination

39.4.13 MQSeries

Brochure,
An Introduction to Messaging and Queuing
Planning Guide
Intercommunication book
Queue Manager Clusters
Clients book
System Administration book
Command Reference
Programmable System Management book
Administration Interface Programming Guide and Reference
Messages book
Application Programming Guide
Application Programming Reference
Application Programming Reference Summary
Using C++
Using Java
MQSeries for z/OS
Release 1Licensed Program Specifications
Release 1Program Directory
System Management Guide
Messages and Codes

Problem Determination Guide

39.4.14 IMS

Glossary
Administration Guide: Database Manager
Administration Guide: System
Administration Guide: Transaction Manager
Release Planning Guide
Customization Guide
Operations Guide
Operator's Reference
Summary of Operator Commands
Sample Oper Procs
Utilities Ref: DB
Utilities Ref: System
Utilities Ref: TM
Application Programming: DB Manager
Application Programming: Design Guide
Application Programming: EXEC DLI
Application Programming: Transaction Manager
Database Recovery Control Guide & Reference
Installation Vol 1: Installation & Verification
Installation Vol 2: System Definition & Tailoring
Messages and Codes
Common Queue Server Guide & Reference
Open Transaction Manager Access
IMS/ESA: "Why Migrate"

39.4.15 DB2

Master Index
Installation Guide
Administration Guide
Administration Tool
Administration Tool User's Guide
Command Reference
Utility Guide and Reference
Application Programming and SQL Guide
Application Programming Guide and Reference for Java
Data Sharing: Planning and Administration
Data Sharing Quick Reference Card
Diagnosis Guide and Reference
Diagnostic Quick Reference Card
Image, Audio, and ideo Extenders Administration and Programming
Licensed Program Specifications
Messages and Codes
ODBC Guide and Reference
Reference for Remote DRDA Requesters and Serers
Reference Summary
Release Planning Guide
SQL Reference
Text Extender Administration and Programming
What's New?
XML Extender for OS/390 and z/OS Administration and Programming
Program Directory

DB2 Connect User's Guide
DB2 Connect Personal Edition Quick Beginnings
DB2 Connect Enterprise Edition for OS/2 and Windows: Quick Beginnings
DB2 Connect Enterprise Edition for UNIX: Quick Beginnings

Net.Data Library: Reference
Net.Data Library: Administration and Programming Guide for OS/390 and z/OS
Net.Data Library: Language Enironment Interface Reference

Net.Data Library: Messages and Codes

39.4.16 UNIX System Services

UNIX System Services User's Guide
UNIX System Services Planning
UNIX System Services Command Reference
UNIX System Services File System Interface Reference
UNIX System Services Connection Scaling Reference
UNIX System Services Messages and Codes
UNIX System Services Programming Tools
UNIX System Services Assembler Callable Services
UNIX System Services Using REXX

39.4.17 TCP/IP

Communications Server: IP User's Guide
Communications Server: IP Migration Guide
Communications Server: IP Configuration Guide
Communications Server: IP Configuration Reference
Communications Server: Resource Definition Samples
Communications Server: IP Network Print Facility
Communications Server: IP IMS Sockets Guide
Communications Server: IP CICS Sockets Guide
Communications Server: IP Programmer's Reference
Communications Server: IP Applications Programming Interface Guide
Communications Server: CMIP Services & Topology Agent Guide
Communications Server: IP and SNA Codes
Communications Server: IP Diagnosis Guide
Communications Server: IP Messages Volume 1 (EZA)
Communications Server: IP Messages Volume 2 (EZB)
Communications Server: IP Messages Volume 3 (EZY-EZZ-SNM)

39.4.18 Webserver

HTTP Server Planning, Installing, and Using
IBM Web Traffic Express for Multiplatforms User's Guide

39.4.19 Firewall

Security Server Firewall Technologies

39.4.20 Private Key Infrastructure

Security Server Cryptographic Services PKI Services Guide and Reference

39.4.21 VPN

Security Server Network Authentication Server Administration
Security Server Network Authentication Server Programming

39.4.22 LDAP

Security Server LDAP Server Administration and Use

39.4.23 DCE

SecureWay Security Server DCE Overview

40 Bijlage: Index