Izrada programskog rješenja za automatiziranu proizvodnju izvještaja iz z/OS RACF sustava

Štefanec, Mia

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:236657

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-07-16**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

IZRADA PROGRAMSKOG RJEŠENJA ZA AUTOMATIZIRANU PROIZVODNJU IZVJEŠTAJA IZ z/OS RACF SUSTAVA

Diplomski rad

Mia Štefanec

Osijek, 2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

Obrazac D1: Obrazac za ocjenu diplomskog rada na sveučilišnom diplomskom studiju

Ocjena diplomskog rada na sveučilišnom diplomskom studiju

Ime i prezime pristupnika:	Mia Štefanec
Studij, smjer:	Sveučilišni diplomski studij Računarstvo
Mat. br. pristupnika, god.	D-1168R, 13.10.2020.
JMBAG:	0165071989
Mentor:	prof. dr. sc. Dominika Crnjac Milić
Sumentor:	prof. dr. sc. Krešimir Nenadić
Sumentor iz tvrtke:	Filip Đuričković
Predsjednik Povjerenstva:	izv. prof. dr. sc. Alfonzo Baumgartner
Član Povjerenstva 1:	prof. dr. sc. Dominika Crnjac Milić
Član Povjerenstva 2:	doc. dr. sc. Tomislav Galba
Naslov diplomskog rada:	Izrada programskog rješenja za automatiziranu proizvodnju izvještaja iz z/OS RACF sustava
Znanstvena grana diplomskog rada:	**Informacijski sustavi (zn. polje računarstvo)**
Zadatak diplomskog rada:	U teorijskom dijelu potrebno se osvrnuti na najčešće zahtjeve sigurnosnih revizora s kojima se korisnici IBM Z tehnologija susreću kao i mogućnosti koje sigurnosni menadžer RACF pruža za ostvarivanje tih zahtjeva pomoću ugrađenih revizorskih značajki. Osim navedenog, potrebno se osvrnuti na tehnologiju koju RACF, u kombinaciji sa z/OS operacijskim sustavom, koristi za bilježenje aktivnosti korisnika (SMF). Koristeći dostupne z/OS tehnologije, potrebno je izraditi programsko rješenje za automatizirano generiranje izvještaja o sigurnosnim iznimkama i stanju
Datum ocjene pismenog dijela diplomskog rada od strane mentora:	26.06.2024.
Ocjena pismenog dijela diplomskog rada od strane mentora:	Izvrstan (5)
Datum obrane diplomskog rada:	12.07.2024.
Ocjena usmenog dijela diplomskog rada (obrane):	Izvrstan (5)
Ukupna ocjena diplomskog rada:	Izvrstan (5)
Datum potvrde mentora o predaji konačne verzije diplomskog rada čime je pristupnik završio sveučilišni diplomski studij:	12.07.2024.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O IZVORNOSTI RADA

Osijek, 12.07.2024.

Ime i prezime Pristupnika:	Mia Štefanec
Studij:	Sveučilišni diplomski studij Računarstvo
Mat. br. Pristupnika, godina upisa:	D-1168R, 13.10.2020.
Turnitin podudaranje [%]:	5

Ovom izjavom izjavljujem da je rad pod nazivom**: Izrada programskog rješenja za automatiziranu proizvodnju izvještaja iz z/OS RACF sustava**

izrađen pod vodstvom mentora prof. dr. sc. Dominika Crnjac Milić

i sumentora prof. dr. sc. Krešimir Nenadić

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis pristupnika:

SADRŽAJ

1. UVOD 1

2. STANDARDI I POSTOJEĆA RJEŠENJA ZA IZRADU IZVJEŠTAJA 2

2.1. Standardi za informacijsku sigurnost 2

2.2. Postojeći alati za izradu izvještaja iz z/OS-a 4

3. TEHNOLOGIJE I ALATI KORIŠTENI ZA IZRADU IZVJEŠĆA 7

3.1. z/OS operacijski sustav 7

3.2. TSO/E i ISPF 7

3.3. RACF 9

3.4. ISPF uređivač 10

3.5. Programski jezik PL/I 10

3.6. SMF 12

3.7. Baza podataka Db2 13

3.8. IMS 13

4. PROGRAMSKO RJEŠENJE ZA ISPIS IZVJEŠĆA 15

4.1. Program za ispis izvješća o neuspješnim pokušajima pristupa sustavu (VJOBINIT) 17

4.1.1. Prevođenje i pokretanje programa 20

4.2. Program za ispis izvješća o neuspješnim zahtjevima za pristup podacima (VACCESS) 21

4.2.1. Prevođenje i pokretanje programa 23

5. PROGRAMSKO RJEŠENJE ZA IMS TRANSAKCIJU 24

5.1. PL/I programsko rješenje za 7Z5 IMS transakciju 26

5.2. PL/I programsko rješenje za 7Z6 IMS transakciju 31

5.3. Prikaz izlaznih ekrana transakcija 7Z5 i 7Z6 34

6. ZAKLJUČAK 36

LITERATURA 37

SAŽETAK 39

ABSTRACT 40

PRILOZI 41

ŽIVOTOPIS.....42

1. UVOD

U suvremenom informacijskom dobu, sigurnost podataka postala je jedna od najvažnijih tema unutar organizacija. One se sve više oslanjaju na tehnologije za pohranu i obradu važnih podataka te bi zbog toga trebale implementirati sigurnosne mjere za zaštitu svojih podataka. Jedan od ključnih elemenata sigurnosti je revizija. Revizija (engl. *audit*) je proces procjene i provjere učinkovitosti sigurnosnih kontrola [1] i koji pomaže organizacijama da identificiraju i riješe potencijalne ranjivosti prije nego što budu iskorištene u mogućem kibernetičkom napadu. Kako je većim organizacijama poput financijskih institucija, zdravstvenih ustanova i vladinih agencija ključna povjerljivost, integritet i dostupnost podataka, one u svojem poslovanju mogu koristiti središnje računalo (engl. *mainframe*) koje im, osim zaštite, može pružiti i efikasnu obradu podataka [2]. U ovom radu, naglasak će biti na središnjem računalu tvrtke IBM koje koristi operacijski sustav z/OS te njegovom sigurnosnom alatu RACF-u (engl. *Resource Access Control Facility*) koji će biti objašnjeni u nastavku rada.

U teorijskom dijelu ovog rada objasnit će se najčešći zahtjevi sigurnosnih revizora s kojima se korisnici IBM Z tehnologija susreću kao i mogućnosti koje sigurnosni menadžer RACF pruža za ostvarivanje tih zahtjeva uz pomoć ugrađenih revizorskih značajki. Osim navedenog, potrebno se osvrnuti na tehnologiju koju RACF, u kombinaciji s z/OS operacijskim sustavom, koristi za bilježenje aktivnosti korisnika (SMF, engl. *System Management Facilities*). U praktičnom dijelu ovog rada, koristeći dostupne z/OS tehnologije, bit će dano programsko rješenje za automatizirano generiranje izvještaja o sigurnosnim iznimkama i stanju sigurnosnog okruženja na temelju informacija dostupnih iz RACF baze podataka i z/OS sistemskih zapisa.

2. STANDARDI I POSTOJEĆA RJEŠENJA ZA IZRADU IZVJEŠTAJA

Kao što je spomenuto u uvodu, revizija je bitan element kojim se provjeravaju koje su, i kako su implementirane sigurnosne kontrole unutar organizacije. Prema [3] zaštita osjetljivih podataka neke organizacije je tehnički zahtjevna, ali i skupa te treba znati kako postaviti sigurnosne kontrole bez da pristup podacima bude ili previše slobodan ili previše rigorozan. Također, sistemski administratori koji imaju najvišu razinu pristupa podacima se trebaju kontrolirati i revidirati kako ne bi došlo do interne zloupotrebe podataka organizacije. Revizijom se mogu utvrditi mogući nedostatci u postojećim sigurnosnim kontrolama, otkriti neovlašten pristup ili pokušaj neovlaštenog pristupa resursu, provjeriti bilježi li se aktivnost korisnika te identificirati ranjivosti na vrijeme. Kako bi se provjerile ove sigurnosne kontrole, revizori zapravo trebaju utvrditi jesu li one usklađene sa sigurnosnim standardima. Svaka organizacija treba koristiti onaj sigurnosni standard koji najbolje opisuje njihove potrebe, a to može ovisiti o vrsti industrije, geografskoj lokaciji, vrsti podataka koju organizacija koristi i/ili zakonskim regulativama, a po potrebi se različiti standardi mogu kombinirati. U sljedećim potpoglavljima bit će ukratko opisana tri najčešća korištena standarda te postojeća rješenja koja pomažu organizacijama da ispune zahtjeve navedenih standarda s naglaskom na proizvodnju automatiziranih izvještaja kojima se prati aktivnost korisnika unutar organizacije.

2.1. Standardi za informacijsku sigurnost

ISO/IEC 27001 je standard za informacijsku sigurnost kojeg je kreirala Međunarodna organizacija za normizaciju (engl. *International Organization for Standardization*, u nastavku rada ISO). Prema [4] ovaj standard je najpoznatiji svjetski standard za sustave upravljanja informacijskom sigurnošću (u nastavku ISMS) koji nudi okvir za upravljanje rizicima te definira zahtjeve koje ISMS treba ispuniti. ISO 27001 okvir usmjerava organizacije kako „upravljati, kontrolirati, pregledavati, implementirati i održavati“ [4] informacijsku sigurnost, a kako bi organizacije dobile ili zadržale ISO 27001 certifikat, moraju provesti internu i vanjsku reviziju prilikom kojih se treba pripremiti sva dostupna dokumentacija u kojoj su zapisani svi procesi, procedure i upute za sigurnosne kontrole za ISMS neke organizacije, a na revizoru je da utvrdi jesu li opisane stvari implementirane u stvarnosti u ISMS-u [5].

PCI DSS (engl. *Payment Card Industry Data Security Standard*) sigurnosni je standard kojeg je razvilo PCI Vijeće za sigurnosne standarde (engl. *PCI Security Standards Council*). Prema [6] standard je namijenjen za one organizacije koje spremaju ili obrađuju informacije vlasnika platne

kartice u neke druge svrhe te je cilj ovog standarda zaštititi informacije vlasnika kartice i podatke za autentifikaciju. Prema ovom standardu, niti jedna strana koja je uključena u obradu kartičnog zahtjeva nikada ne smije spremati podatke za autentifikaciju nakon autorizacije. Neki od zahtjeva koje organizacije moraju ispuniti kako bi dobile certifikat za ovaj standard su [6]:

- Instalacija i održavanje konfiguracije vatrozida
- Šifriranje prijenosa podataka vlasnika kartice preko javnih mreža
- Zaštita sustava od virusa i razvoj sigurnosnih sustava
- Ograničiti pristup podacima kartice
- Pratiti i nadzirati pristup mrežnim resursima i podacima vlasnika kartice

GDPR (engl. *General Data Protection Regulation*) je sigurnosni zakon kojeg je donijela Europska Unija u svibnju 2018. godine kako bi se zaštitili podaci građana članica EU, a prema [7], ovaj zakon odnosi se i na države van Europske Unije, sve dok spremaju i koriste podatke građana EU, a ako neka organizacija prekrši pravila GDPR-a, trebat će platiti novčanu kaznu koja može doseći desetke milijuna eura. Prema GDPR-u, subjektu čiji se podaci obrađuju, ako je to dopustio, mora biti zajamčena zakonitost i transparentnost prilikom same obrade, trebaju se skupljati samo podaci nužni za zatraženu uslugu ili svrhu te se sakupljeni podaci moraju šifrirati. Također, revizor će utvrditi je li neka organizacija osigurala da pristup podacima imaju samo one osobe koje su dobile uputu pristupiti im, bilo zbog poslovnih razloga ili po zakonu EU ili njene članice [8].

Postoje još razni standardi, zakoni i sigurnosni okviri te svaka organizacija mora dobro proučiti koji će standard, ili više njih, najbolje zadovoljiti njihove potrebe i pouzdano zaštititi podatke koje koriste. Ono što je većini ovih standarda zajedničko jest nadzor i kontrola pristupa podacima kako bi u slučaju kibernetičkog napada postojali zapisi o tome tko je i kada pristupio podacima i kako bi se moglo reagirati što efikasnije, a ti podaci također mogu poslužiti revizorima za provjeru pristupaju li podacima samo oni zaposlenici koji zbog svoje uloge u poslovanju moraju pristupiti tim podacima te tko je sve neovlašteno pokušao pristupiti nekom resursu. Za kreiranje izvještaja o uspješnim i neuspješnim pokušajima pristupa mogu se koristiti postojeći alati, a kako se u praktičnom djelu ovog rada koristi z/OS koji je dio središnjeg računala, u nastavku će biti opisani alati koji se mogu koristiti za kreiranje izvještaja o pristupu prema zapisima i aktivnosti korisnika koji koriste z/OS u svome radu.

2.2. Postojeći alati za izradu izvještaja iz z/OS-a

Na središnjim računalima implementiraju se sigurnosni sustavi poput RACF, CA-ACF2 i CA-Top Secret koji se zajedničkim imenom nazivaju vanjski sigurnosni menadžeri (engl. *External Security Manager,* ESM). ESM-ovi omogućuju konfiguraciju sigurnosti prema potrebama korisnika kako bi spriječili neovlašteni pristup resursima kroz aplikacije ili transakcije [9]. Njihova implementacija i kasnije održavanje mogu postati kompleksni te zahtijevaju od administratora jedinstvene, ali i skupe, tehničke vještine kako bi konfiguracija bila postavljena bez grešaka. Prema [3] proces revizije može utjecati na dostupnost sistemskih inženjera i administratora koji rade na održavanju sigurnosnih sustava, a s obzirom na samu kompleksnost održavanja, za vrijeme revizije može doći i do stagnacije u poboljšanju sigurnosnih sustava. Ovo je jedan od razloga zašto se neke organizacije odlučuju za korištenje automatiziranih alata koji im olakšavaju provođenje sigurnosnih analiza, proizvodnji izvještaja i procjene rizika unutar sustava.

Jedan od takvih alata je *IBM zSecure* koji služi za procjenu rizika, pomaže pri planiranju i implementaciji sigurnosnih uputa i standarda te se koristi za praćenje sigurnosnih događaja, održavanje i poboljšavanje sigurnosnih kontrola središnjeg računala organizacije. Uz pomoć ovog alata se također mogu pisati i kreirati skripte za provjeru stanja sustava ili baza podataka te nudi praćenje prijetnji i izvještaje o pristupima u stvarnom vremenu, periodički ili na zahtjev [3].

Na slici 2.1. prikazano je sučelje zSecure alata i nekoliko opcija koje administrator ili sistemski inženjer može koristiti, u ovom slučaju to su *Setup* za opcije i ulazne podatkovne skupove, *RACF* za upravljanje navedenim sigurnosnim sustavom, *Audit* za revidiranje sigurnosti i sistemskih resursa, *Resource* za izvještaje o sigurnosti resursa, *Access* za praćenje pristupa i postavljanje kontrole pristupa, *Events* za bilježenje sigurnosnih događaja, *Command review* za pregled i korištenje naredbi, *CARLa* za rad s bibliotekama i upitima, *Information* za pretragu dokumentacije i opcija *Local* za pregled opcija koje su definirane lokalno. Također, IBM nudi i verziju programa namijenjenu korisnicima koji se osjećaju ugodnije raditi kroz grafičko korisničko sučelje (engl. *Graphical User Interface,* GUI) naziva IBM *Security zSecure Visual*.

Sl. 2.1. Sučelje početnog izbornika zSecure alata

Vanguard Advisor razvila je tvrtka *Vanguard Integrity Professionals* [10]. Prilagodljiv događajima i sigurnosnim podsustavima, alat šalje obavijesti i izvješća o sigurnosnim događajima u stvarnom vremenu sistemskim administratorima i ostalim osobama unutar organizacije kojima su potrebni podaci, a funkcionalnosti za praćenje mogu indicirati kada sigurnosne kontrole nisu više efikasne. Prednost ovog alata je mogućnost prilagođavanja izrade izvještaja prema potrebama organizacije bez potrebe za visokom vještinom programiranja.

_beta access za z/OS RACF proizvod je njemačke tvrtke _betasystems [11] koji pojednostavljuje nadzor i reviziju z/OS sigurnosnog sustava te pomaže organizacijama da što bolje implementiraju sigurnosne standarde i upute. Može se koristiti kroz web sučelje, kao *Windows* program ili kroz optimizirane ISPF panele, a uz pomoć aplikacije *_beta access easy* kojom se mogu delegirati zadaci lokalnim administratorima. *Windows* sučelje omogućuje revizorima korištenje sustava bez poznavanja RACF sigurnosne okoline što olakšava sam proces revidiranja. Zaposlenicima pruža jednostavniju izradu i analizu izvještaja o pristupima i nadzor kritičnih sigurnosnih događaja kao što je pristup osjetljivim podacima ili promjena korisničkih atributa.

Compliance Event Manager razvila je tvrtka *Broadcom* kako bi organizacijama pojednostavila usklađivanje sigurnosnih kontrola prema standardima i olakšala proces revizije uz pomoć naprednog upravljanja i otkrivanja prijetnji [12]. Prema [13], alat omogućuje korištenje više-faktorske autentifikacije za *RACF* (engl. *Resource Access Control Facility*), *Top Secret* i *ACF2* (engl. *Access Control Facility*) sigurnosne menadžere koja pokriva ključni problem u upravljanju pristupom središnjeg računala, a također šalje upozorenja i istražuje neovlaštene pristupe te pojednostavljuje izradu izvještaja.

Prilikom odabira rješenja za proizvodnju izvještaja, i upravljanje sigurnošću generalno, organizacije moraju utvrditi koje su im glavne potrebe koji alati se mogu najlakše implementirati u postojeće sustave, kakva im je mogućnost izrade izvještaja i koliko su kompliciranizahtjevi za održavanje spomenutih rješenja. Iako su ova rješenja provjerena i često korištena, organizacijama se savjetuje da isprobaju probne verzije proizvoda kako bi provjerili odgovara li njihovim potrebama i je li usklađen sa zahtjevima revizora koje organizacija mora ispuniti. Ovakva praksa se savjetuje zato što je licenciranje spomenutih proizvoda vrlo skupo te se ovisno o veličini organizacije cijene kreću od nekoliko tisuća do nekoliko desetaka tisuća eura godišnje. Kako neke organizacije ne mogu pronaći tolika financijska sredstva, one se odlučuju za kreiranje vlastitih rješenja uz pomoć funkcionalnosti koje im nude implementirane sigurnosne okoline i menadžeri poput RACF okruženja koji će se koristiti prilikom izrade rješenja za automatiziranu izradu izvješća u praktičnom djelu ovog rada.

3. TEHNOLOGIJE I ALATI KORIŠTENI ZA IZRADU IZVJEŠĆA

U izradi praktičnog dijela ovoga rada korištene su sljedeće tehnologije: z/OS okolina i njezine funkcionalnosti poput ISPF-a (engl. *Interactive System Productivity Facility*) i RACF-a koje se koriste kao dio testne okoline koju je za izradu ovog rada pružila tvrtka CROZ, PL/I programski jezik, ISPF uređivač za pisanje programskog rješenja, baza podataka Db2 te alati IMS i SMF.

3.1. z/OS operacijski sustav

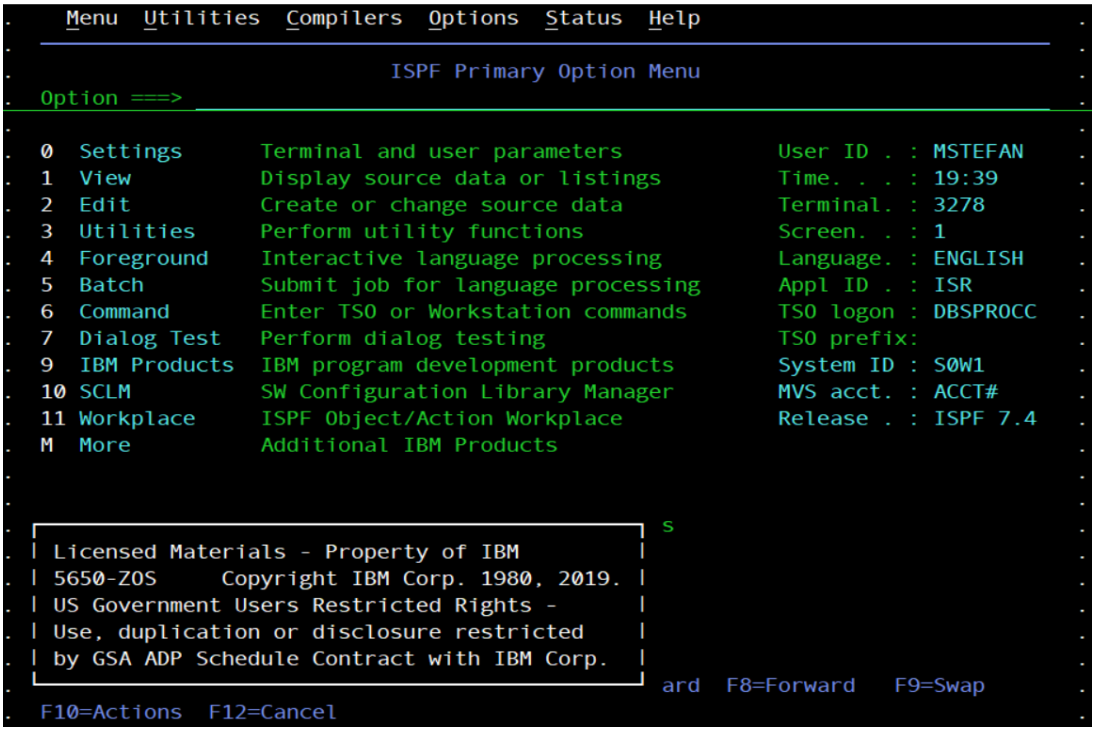
Prema [2], operacijski sustav z/OS je često korišten sustav kod središnjih računala jer pruža stabilnost, sigurnost, stalnu dostupnost i skalabilnu okolinu za aplikacije koje se koriste na središnjem računalu. Također podržava multi programiranje i multi procesiranje što ga sa mogućnošću pristupa i upravljanja velikom količinom memorije i ulazno-izlaznih operacija čini idealnim za procesiranje i izvođenje poslova i transakcija na središnjem računalu. Operacijski sustav z/OS dizajniran je kako bi u potpunosti iskoristio mogućnosti *hardware*-a IBM-ovog središnjeg računala i perifernih jedinica. Prema [14], sustav se sastoji od modula za učitavanje ili izvršnog koda, a prilikom instalacije moduli se kopiraju u biblioteke koje se nalaze na DASD (engl. *Direct Access Storage Devices*) jedinicama, *hardware*-ski dio se sastoji od procesora, diskova, magnetskih traka i korisničkih konzola, operacije z/OS-a se izvode u središnjoj procesorskoj memoriji, a korisnički programi također dijele spomenutu memoriju s z/OS-om. Važno je napomenuti kako je opis z/OS-a pojednostavljen kako bi se razumio njegov osnovni rad, a sami z/OS i središnje računalo su kompleksnija tema te se njihov detaljniji opis može pročitati u knjizi [2]. Za razliku od operacijskih sustava poput macOS-a, koji ima *user-friendly* korisničko sučelje i popularan je među kreativnim pojedincima, Windows OS-a, koji je popularan za osobno i poslovno korištenje zbog podrške velikom broju aplikacija i programa te Linuxa, koji ima visoku razinu sigurnosti te ga se često koristi i za osobne potrebe i poslovne servere, operacijski sustav z/OS se najčešće koristi u velikim organizacijama, poput banaka i zdravstva, koje zahtijevaju pouzdanost, sigurnost i skalabilnost za rukovanjem velikom količinom transakcija i zadataka.

3.2. TSO/E i ISPF

Korisnik se kroz z/OS može kretati uz pomoć naredbi ili korisničkog sučelja koje se bazira na stilu izbornika. *Time Sharing Option/Extensions* (TSO/E) omogućuje korisnicima prijavu u sustav i dijeljenje resursa na središnjem računalu, a nakon prijave, TSO vodi korisnika na ISPF panele koji nudi korisniku razne programe, uključujući uređivač u kojemu se mogu pisati programska

rješenja ili definirati zadatke (engl. *batch jobs*) koje središnje računalo treba izvesti, a mogu se pisati sadržaji datoteka, koji se u z/OS zovu *data sets*. Razina pristupa izbornicima koju ima svaki korisnik ovisi o implementiranim sigurnosnim kontrolama i atributima koje korisnikov profil ima [14]. Izgled TSO i ISPF panela prikazani su na slikama 3.1. i 3.2. .

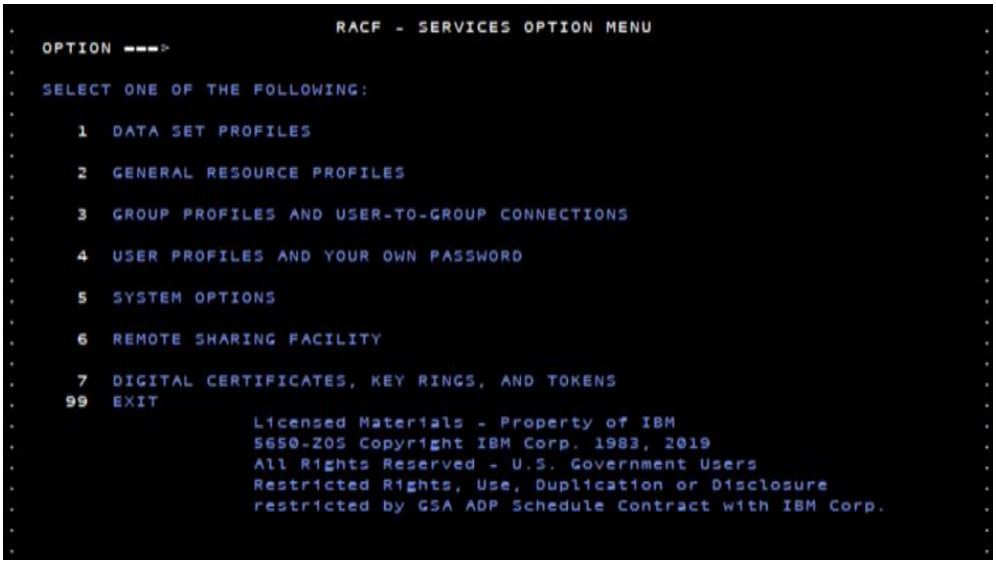
Sl. 3.1. Sučelje za prijavu u TSO/E



Sl. 3.2. ISPF panel

3.3. RACF

RACF [2] je vanjski sigurnosni menadžer koji pruža osnovne sigurnosne kontrole za sustav središnjeg računala i štiti njegove resurse dodjeljivanjem prava samo autoriziranim korisnicima. Informacije o korisnicima, resursima i pravima na pristup čuva u svojoj bazi podataka u strukturama koje se nazivaju profili te im pristupa nakon što korisnik pokuša pristupiti resursu kako bi se provjerilo ima li korisnik dovoljnu razinu prava na pristup za željeni zaštićeni resurs. Funkcionalnosti koje RACF nudi korisnicima i sistemskim administratorima su [2]: identificiranje i autentifikacija korisnika, davanje prava na pristup zaštićenim resursima, kontroliranje i bilježenje autoriziranih i uspješnih, ali i neautoriziranih i neuspješnih pokušaja pristupa te omogućavanje aplikacijama da koriste RACF naredbe. Na slici 3.3. prikazano je sučelje i opcije RACF programa koje korisnik može koristiti za postavljanje sigurnosnih kontrola za ostale korisnike ili resurse. U RACF okolini, svaki korisnik ima različitu sigurnosnu odgovornost ili potrebu za pristup resursima i podacima, jedan korisnik može imati izrazito visoku odgovornost prema sigurnosti, dok drugi nema skoro nikakvu te jedan korisnik može zahtijevati visoku potrebu za pristup resursima, drugi nisku, dok trećem pristup resursima može biti zabranjen u potpunosti. Kako bi se pojedinačnim korisnicima definirala odgovornost i razina pristupa, dodijeljeni su im RACF korisnički atributi, a tri najvažnija za revizore su *special*, kojeg koriste sigurnosni administratori, *roaudit* (engl. *read only audit*), koji omogućuje korisniku praćenje sustava, ali ne može postavljati nove revizijske postavke, i atribut *auditor* uz kojeg korisnik može postavljati i promatrati sigurnosne i revizijske postavke [15].



Sl. 3.3. RACF sučelje

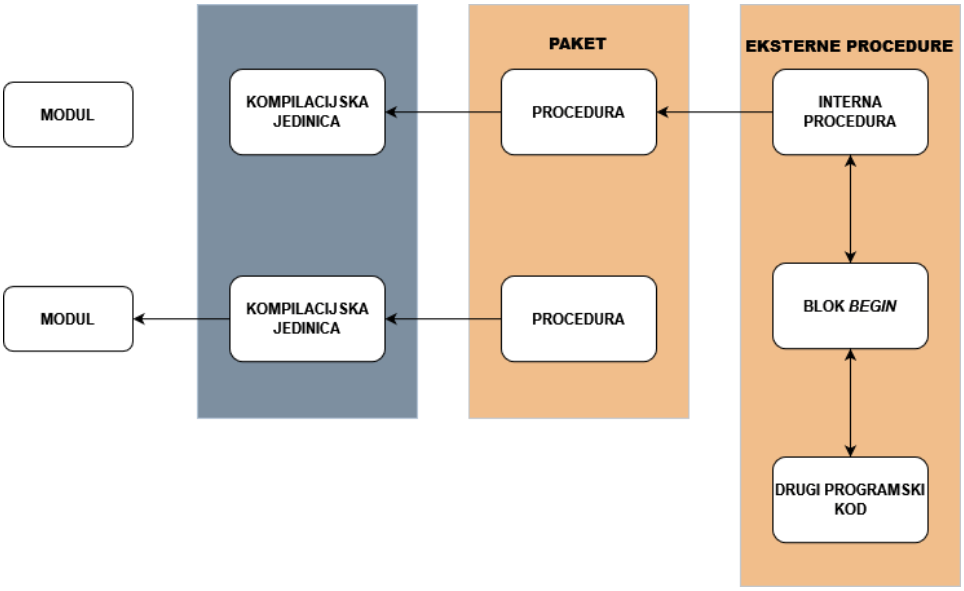
3.4. ISPF uređivač

ISPF editor interaktivni je tekstualni editor koji je dio operativnog sustava z/OS i koristi se za pregled i uređivanje skupova podataka, programskih rješenja, skripti, konfiguracijskih datoteka ili nekih drugih datoteka unutar ISPF biblioteka [16]. Neke od najčešćih naredbi koje se koriste u editoru su naredbe *FIND, CHANGE* i *EXCLUDE* koje se koriste za pretraživanje i zamjenu teksta ili izuzimanje određenih linija koda ili teksta. Ostale naredbe mogu se naći na [17]. Jedna od specifičnosti ovog editora su granice unutar kojih će biti vidljive promjene, a u ovom radu promjene će biti zabilježene ako one budu u stupcima između 1 i 72, a raspon se može promijeniti korištenjem naredbe *BOUNDS* [16].

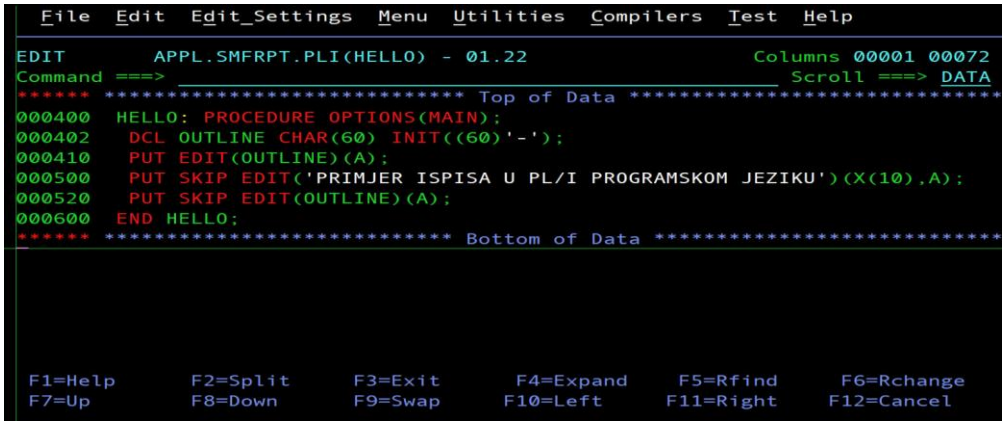
3.5. Programski jezik PL/I

Programming Language One ili PL/I, proceduralni je programski je jezik visoke razine kojeg je razvila tvrtka IBM 1960-ih godina za znanstvene i poslovne svrhe te za sistemsko programiranje [18]. Kombinira funkcionalnosti iz *Fortran* i *COBOL* programskih jezika koji su također namijenjeni za razvoj znanstvenih i poslovno orijentiranih aplikacija. Prema [2], programi napisani u ovom programskom jeziku sastoje se od blokova koje čine grupe izjava ili potprogrami, a upravo blokovi omogućuju pisanje modularnih aplikacija. Aplikacije koriste jedan ili više entiteta koji se nazivaju moduli za učitavanje (engl. *load modules*), a svaki ovakav modul čine jedna ili više odvojenih kompilacijskih jedinica (engl. *compilation units*) koji predstavljaju paket ili eksternu proceduru. Paketi mogu sadržavati više procedura, koje mogu biti eksterne ili interne, a procedure mogu sadržavati nijedan ili više blokova. Opisana struktura aplikacije pisane u PL/I programskom jeziku može se vidjeti na slici 3.4. Programski jezik PL/I ne smatra se karakterističnim objektno orijentiranim jezikom i nema klase poput C++ ili Jave, ali prethodno opisani koncept omogućuje strukturno programiranje za organizaciju programskog koda u zasebne logičke jedinice definiranjem struktura podataka i procedura za manipulaciju tih struktura. ISPF editor i programsko rješenje za jednostavni primjer prikazani su na slici 3.5., a ispis na 3.6.. Blok procedure započinje imenom procedure ili programa, u ovom slučaju *Hello*, ključnom riječju *PROCEDURE* te opcijom *MAIN* koja govori sustavu da je ova procedura prva u PL/I programu te se ona u jednoj proceduri ne smije pojaviti više od jednom. *PUT EDIT i PUT SKIP* se koriste za ispisivanje toka podataka, gdje *EDIT* omogućava formatiranje ispisa, a opcija *SKIP* prelazak u novi red. Dakle, u ovom kodu definirana je procedura *Hello* u kojoj se deklarira varijable *Outline* tipa *char* duljine 60 te je inicijalizirana na 60 znakova '-' i ova varijabla se prva ispisuje. Zatim se ispisuje prikazani tekst koji je formatiran korištenjem *PUT SKIP EDIT* naredbom i opcijom X koja

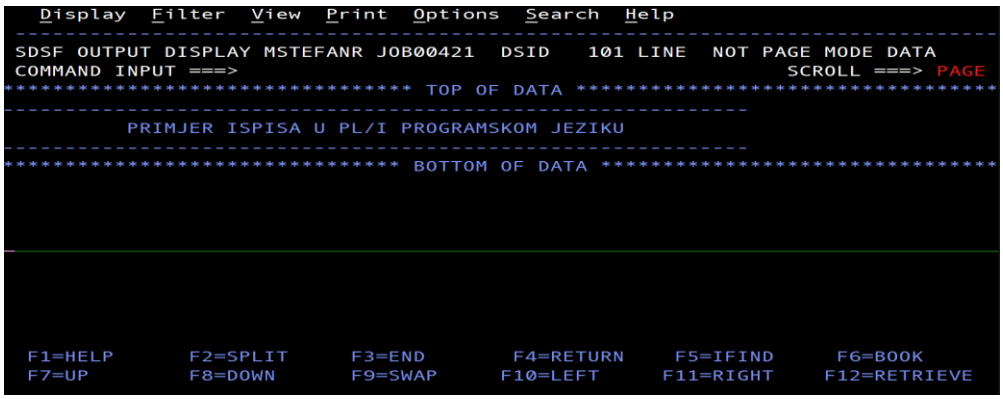
pomiče tekst u desno za 10 razmaka, varijabla *Outline* se ponovno ispisuje kako bi uokvirila ispisani tekst, a program završava naredbom *END* i nazivom procedure.



Sl. 3.4. Struktura PL/I aplikacije



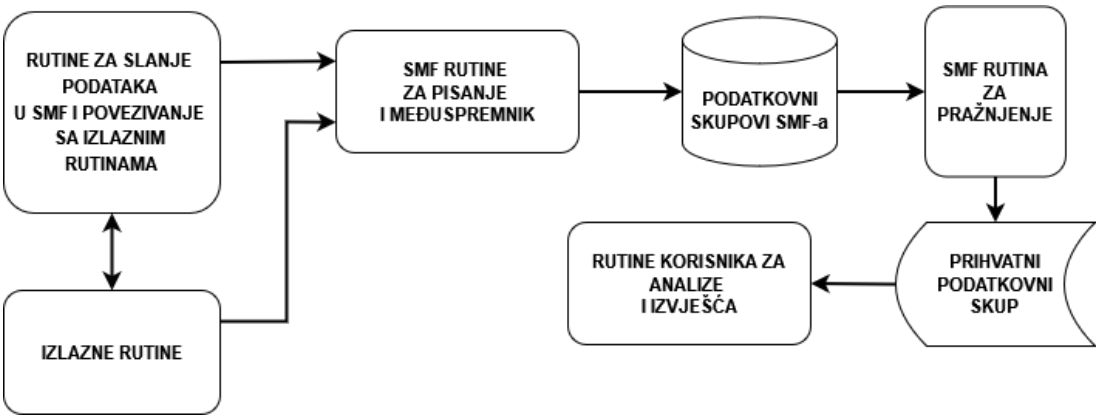
Sl. 3.5. Jednostavni PL/I program u ISPF editoru



Sl. 3.6. Ispis *Hello* primjera

3.6. SMF

System management facilities, u daljnjem tekstu SMF, prema [19], koristi se za prikupljanje i zapisivanje sistemskih poruka i podataka o pokrenutim zadacima, a koji se mogu koristiti u različite svrhe, poput: analize konfiguracije, planiranje zadataka, provjere pristupa podatkovnim skupovima, provjere korištenja resursa, održavanje sigurnosti sustava ili licenciranje korisnika. Prikupljene informacije organiziraju se u zapise o sustavu i zapise o pokrenutim poslovima, gdje zapisi o sustavu uključuju informacije o konfiguraciji, upravljanju memorijom i opterećenju sustava, dok zapisi o poslovima uključuju informacije procesorskom vremenu, aktivnosti podatkovnih skupova za svaki posao i pojedinačni korak unutar posla, transakcijskim programima i aktivnosti tijekom korištenja TSO/E sučelja, a zapisi se pohranjuju u SMF podatkovne skupove. SMF rutine za pohranjivanje zapisa i izlaz iz programa nezavisne su jedne o drugima te ih se može implementirati odvojeno ili po potrebi zajedno. Slikom 3.7. prikazan je kratki pregled rada SMF-a.



Sl. 3.7. Pregled rada SMF-a

Zapisi pohranjeni u SMF podatkovni skup se zatim mogu koristiti u raznim analizama i izvješćima. Kada se glavni SMF podatkovni skup popuni do maksimalnog kapaciteta, sustav šalje obavijest odgovornom korisniku kako bi se podatkovni skup morao isprazniti kako bi se stvorilo mjesta za nove zapise. Kako bi se navedeno ostvarilo, koristi se program *IFASMFDP* preko kojeg se zapisi iz jednog SMF podatkovnog skupa prebacuju u drugi kako bi se prvi ispraznio i mogao nastaviti s normalnim radom. Prilikom kreiranja posla za pozivanje *IFASMFDP* programa, također se definiraju tipovi zapisa koji će se kopirati u novi podatkovni skup, a za ovaj rad od važnosti su tipovi 30, 80, 81 i 83. Tip zapisa 30 sadrži informacije o početku i završetku rada radne jedinice poput TSO/E sjednice, pokretanju transakcijskog programa, pokrenutom zadatku, kao i informacijama o procesorskom vremenu svakog izvedenog koraka i broju procesiranih podataka.

Zapise tipa 80 kreiraju RACF procesi kada zabilježe sigurnosne događaje poput nedopuštenog pokušaja pristupa sustavu, dopuštenog ili nedopuštenog pristupa resursima koje izravno štiti RACF ili prilikom pokušaja izmijene sigurnosnih profila unutar RACF baze podataka, dok su zapisi tipa 81 kreirani nakon završetka inicijalizacije RACF-a. Na kraju, zapisi tipa 83 su zapisi kreirani korištenjem RACF naredbi za dodavanje, promjenu i brisanje podatkovnih skupova (*ADDSD, ALTDSD, DELDSD*), odnosno naredbi koji utječu na promjenu sigurnosnih oznaka nekog podatkovnog skupa.

3.7. Baza podataka Db2

Baza podataka koja će se koristiti u ovom radu je Db2 za z/OS operativni sustav, koju je razvio IBM. Baza kreirana za efikasno pohranjivanje, analiziranje i dohvaćanje podataka, koristi se u okolini središnjeg računala zbog svoje pouzdanosti i sigurnosnih značajki [20], a razumijevanje ove baze važno je kako bi se uspješno kreiralo rješenje za automatizaciju izvještaja. Jezik koji se koristi za rad s bazom je SQL (engl. *Structured Query Language*) kojeg čini niz naredbi: *CREATE, ALTER, DROP, SELECT, INSERT i UPDATE*, samo su neke od tih izjava, a koriste se za kreiranje, izmjenu, brisanje, odabir, umetanje i ažuriranje podataka u tablici.

Sigurnosne događaje koji će biti zapisani u SMF zapisima, potrebno je procesirati i spremiti u Db2 bazu podataka kako bi se dobivene informacije mogle koristiti za revizijske izvještaje. Kako bi se ovaj postupak uspješno proveo, u z/OS-u je prilikom instalacije potrebno kreirati bazu podataka, tablični prostor i odgovarajuće tablice [20] potrebne za uspješan rad između SMF i baze. Postoji nekoliko desetaka kreiranih tablica, a za potrebno programsko rješenje koristit će se samo dvije tablice koje sadrže ključne informacije za reviziju: JOBINIT tablica, u koju se zapisuju sve informacije povezane s inicijalizacijom poslova u z/OS sustavu, uključujući vrijeme pokretanja, ime posla i korisnika koji je posao i pokrenuo, te ACCESS tablica, u koje se zapisuju uspješni i neuspješni pokušaji pristupa resursima unutar sustava.

3.8. IMS

IMS (engl. *Information Management System*) transakcijski je upravitelj koji se temelji na porukama te ga čine dvije komponente: upravitelj baze podataka (IMS DB) i transakcijski upravitelj (IMS TM). IMS DB je sustav za upravljanje bazom podataka za definiranje strukture baze, organizaciju poslovnih podataka, izvođenje upite na podacima i transakcija. IMS TM je transakcijski upravitelj za procesiranje ulaznih i izlaznih poruka između korisnika i sustava te upravlja porukama u redu čekanja, sigurnošću, raspoređivanje, formatiranjem, zapisivanjem i

njihovim oporavkom. IMS se koristi u bankarstvu, financijama, zdravstvu i proizvodnji [21]. Svaka poruka koja se pošalje ili primi u IMS-u može sadržavati jedan ili više segmenata, segmenti ulazne poruke sastoje se od polja LLZZ duljine četiri bajta, gdje duljinu toka bitova čini prva dva bajta LL, a ZZ su dva bajta rezervirana za potrebe IMS-a, zatim transakcijskog koda (TRANCODE) koji identificira određenu transakciju koja će se izvesti, a ostatak ulaznog toka čine potrebne podatke, a izlazna poruka također sadrži polje duljine četiri bajta LLZZ i izlazne podatke, uključujući informacije o potencijalnim pogreškama prilikom izvođenja programa. IMS program prima i šalje u red čekanja jedan po jedan segment poruke te se prije primanja prvog segmenta iduće poruke briše kraj prethodne poruke.

4. PROGRAMSKO RJEŠENJE ZA ISPIS IZVJEŠĆA

Prvi dio programskog rješenja zahtjeva provjeru je li u sustavu omogućeno zapisivanje sigurnosnih događaja, u ovom slučaju zapisivanje događaja do kojih dolazi nakon neuspješnih pokušaja prijave ili zahtjeva za pristupom podacima. RACF ove informacije zapisuje u SMF koji služi za bilježenje i spremanje podataka o sustavu i informacija koje se prikupe tijekom izvođenja JCL (engl. *Job Control Language*) zadatka. Za provjeru i omogućavanje RACF opcija, potrebno je imati korisnički atribut AUDITOR. Nakon dobivanja ovog atributa, aktivne opcije provjeravaju se uz pomoć naredbe SETROPTS LIST (engl. *Set RACF Options*). Iako je prilikom instalacije RACF alata postavljeno da se bilježe sigurnosni događaji za većinu profila i klasa, može se dogoditi da je u postavkama isključena opcija praćenja klase kojoj pripada korišteni podatkovni skup te se u tom slučaju koristi naredba ALTDSD (engl. *Alter Data Set Profile*) kako bi se podatkovni skup dodao na listu za praćenje, na primjer, naredbom ALTDSD [ime podatkovnog skupa] GLOBALAUDIT(FAILURES(READ)) omogućuje se zapisivanje svih neautoriziranih pokušaja pristupa podatkovnom skupu u SMF zapise. Za ovo programsko rješenje, omogućeno je zapisivanje sigurnosnih događaja te je dalje bilo potrebno napisati JCL zadatak prema kojemu će se događaji iz SMF podatkovnog skupa zapisati direktno u odgovarajuće Db2 tablice prije nego se isprazne, točnije, SMF zapisi ispraznit će se u bazu podataka kako bi se oslobodio prostor za nove podatke. Ovaj JCL sastoji se od dva manja zadatka: prvi je LOAD i drugi je DELETE te će za svaki dio ovog JCL-a biti prikazan programski kod i opis. Dijagramom prikazan je tok razvoja ovog dijela programskog rješenja:

Korak LOAD (slika 4.1.) koji izvršava program DSNUPROC, a upravo taj program pokreće uslužne alate Db2 baze, a za parametar mu je predan naziv podsustava kojeg baza koristi, u ovom slučaju podsustav DBCG. Prvo se definiraju reference na podatkovne skupove koji će se kreirati i koristiti za uspješno izvršavanje programa. Referenca SYSREC obavezan je podatkovni skup jer poziva ulazni skup koji sadrži SMF zapise pomoću kojih će se ispuniti tablice, SYSERR i SYSDISC kreirat će nove radne skupove u koje će se spremati potencijalne informacije o pogreškama te kopije zapisa koji se nisu uspjeli spremiti u Db2 tablice, a skup SYSMAP je radni skup koji mapira identifikatore redova tablice u kojima je došlo do pogreške. Kako tablice sadrže indekse, potrebno je definirati ulazne i izlazne radne skupove SORTOUT i SYSUT1 za sortiranje ulaznih i izlaznih podataka. SYSIN ulazni je tok koji sadrži kontrolnu izjavu LOAD kojom se definira funkcija koju će posao izvesti, a korištene opcije su DATA (označava učitavanje podataka), INDDN predaje ulazni skup SYSREC, a opcija LOG YES omogućuje bilježenje

procesa punjenje tablica baze podataka. Na kraju, opcijom INTO TABLE daju se nazivi tablica i njihovi stupci koji se trebaju popuniti zapisima zabilježenim u SMF skupu.

Linija ***Kod***

```
1:      //LOAD     EXEC DSNUPROC,SYSTEM=DBCG
2:      //SYSREC   DD DSN=*.DUMP.OUTDD,DISP=SHR
3:      //SYSERR   DD DSN=SYS1.SMF.JCL.ERR,DISP=(NEW,CATLG),
4:      //            SPACE=(CYL,(50,10),RLSE)
5:      //SYSDISC  DD DSN=SYS1.SMF.JCL.DISC,DISP=(NEW,CATLG),
6:      //            SPACE=(CYL,(50,10),RLSE)
7:      //SORTOUT  DD UNIT=SYSDA,SPACE=(4000,(20,20))
8:      //SYSUT1   DD UNIT=SYSDA,SPACE=(4000,(20,20))
9:      //SYSMAP   DD UNIT=SYSDA,SPACE=(4000,(20,20))
10:     //SYSIN    DD *
11:     LOAD DATA
12:       INDDN SYSREC
13:       LOG YES
14:       INTO TABLE [ime tablice]
…         (…
10725     …)
```

Sl. 4.1. LOAD zadatak

Drugi korak ovog LOADUNLD JCL posla je korak DELETE (slika 4.2.) koji koristi sistemski program IEFBR14 koji vraća 0 kao povratnu vrijednost izvršenog koraka. Parametar COND = (0,NE) provjerava je li povratna vrijednost prethodnog JCL koraka različita od 0 (NE, engl. *Not equal to*), ako je ovaj uvjet točan, ovaj korak, odnosno brisanje radnih skupova, će se preskočiti što znači da je prilikom punjenja tablica došlo do pogreške koja se mora zapisati u pod. skupove namijenjeni za pisanje pogrešaka koji su se definirali u prethodnom koraku, a ako uvjet nije ispunjen, odnosno povratna vrijednost je nula, tada se brišu radni podatkovni skupovi.

Linija ***Kod***

```
10727:  /*
10728:  //DELETE EXEC PGM=IEFBR14,COND=(0,NE)
10729:  //SYSPRINT DD SYSOUT=*
10730:  //ERRDUM DD DSN=SYS1.SMF.JCL.ERR,DISP=(OLD,DELETE)
10731:  //DISCDUM DD DSN=SYS1.SMF.JCL.DISC,DISP=(OLD,DELETE)
```

Sl. 4.2. DELETE zadatak

4.1. Program za ispis izvješća o neuspješnim pokušajima pristupa sustavu (VJOBINIT)

Kod na slici 4.3. prikazuje definiranje pokazivača (engl. *cursor*) preko kojeg se dohvaćaju podaci iz tablice JOBINIT. Potrebni podaci za ispis izvještaja su oznaka događaja (EVENT_QUAL), oznaka korisnika koji je uzrokovao događaj(EVT_USER_ID), vrijeme i datum kada je došlo do događaja (TIME_WRITTEN, DATE_WRITTEN), puno ime korisnika (USER_NAME) te ime sustava i terminala (SYSTEM_SMFID, TERM) s kojih je zabilježen sigurnosni događaj, a navedeni podaci se dohvaćaju iz tablice JOBINIT ako ispunjavaju sljedeća dva uvjeta:

1. vrijeme događaja je unutar 24h od trenutka pokretanja posla
2. oznaka događaja je različita od oznaka koje označavaju uspješan pristup.

Također, 16. linija koda u program uključuje skup varijabli SQLCA (engl. *SQL Communication Area*) koje se ažuriraju nakon završetka izvođenja svake SQL naredbe i daju informacije o tome je li naredba uspješno izvršena ili ne, te gdje je došlo do pogreške tijekom izvođenja.

Linija ***Kod***

```
16:     EXEC SQL INCLUDE SQLCA;
17:     EXEC SQL INCLUDE VJOBINIT;
18:     EXEC SQL
19:
20:     DECLARE JOBINITC CURSOR FOR
21:     SELECT EVENT_QUAL,EVT_USER_ID, TIME_WRITTEN,DATE_WRITTEN, USER_NAME,
22:     SYSTEM_SMFID,TERM
23:     FROM VJOBINIT
24:     WHERE (TIMESTAMP_WRITTEN >= CURRENT TIMESTAMP - 1 DAY)
25:     AND (EVENT_QUAL NOT IN ('SUCCESS','TERM','SUCCESSI','SUCCESST',
26:         'RACINITI','RACINITD','SUCCESSP','SUCCESSM','MFAPSUCC'));
```

Sl. 4.3. Definiranje JOBINIT pokazivača za dohvaćanje podataka

Pokazivač se nakon definiranja otvara te se uz pomoć varijable SQLCODE iz skupa varijabli SQLCA provjerava je li došlo do pogreške tijekom otvaranja. Vrijednosti SQLCODE varijable mogu ispunjavati jedan od sljedeća četiri uvjeta:

1. ako vrijedi SQLCODE < 0, tada je izvedba SQL naredbe bila neuspješna
2. ako vrijedi SQLCODE = 100, tada unutar baze nisu pronađeni podaci
3. ako vrijedi SQLCODE = 0, tada je izvedba SQL naredbe bila uspješna
4. ako vrijedi SQLCODE > 0, tada je izvedba SQL naredbe bila uspješna, ali uz upozorenje.

Na slici 4.4. prikazano je što se događa u kodu kada SQLCODE vrati određenu vrijednost. U slučaju da je SQLCODE < 0, poziva se sistemska procedura DSNTIAR(*sqlca, message, lrecl*) koji pretvara povratni kod i podatke o pogrešci u tekstualnu poruku i rezultat se ispisuje uz pomoć PUT SKIP DATA izjave u izlaznu datoteku. Procedura za parametre prima varijablu SQLCA, *message*, odnosno varijablu ERROR_MESSAGE u koju se sprema poruka i *lrecl*, odnosno varijabla DATA_LEN koja predstavlja duljinu logičkog zapisa izlazne poruke. Nakon ispisivanja poruke o pogrešci, program se završava zbog izjave SIGNAL ERROR koja signalizira da je došlo do pogreške. Nadalje, ako je varijabla SQLCODE = 100, tada se u izlaznu datoteku ispisuje poruka o tome kako podaci nisu pronađeni. Oznaka X(16) omogućuje ispis 16 razmaka unutar datoteke prije nego li se ispiše poruka, a oznaka A je obavezna oznaka kada se koristi opcija EDIT za ispis poruke jer opisuje da se podatkovnom toku predaje znakovna vrijednost. Nakon ispisa poruke, pokazivač JOBINITC se zatvara. Ako je SQLCODE = 0, tada se dohvaćaju podaci iz pokazivača u prihvatne varijable koje se zatim koriste u izjavama za ispis izvještaja u izlaznu datoteku. Formatiranje ispisa izvedeno je uz pomoć oznake COL (engl. *column*) kojom se označava početna pozicija, odnosno početni stupac za ispisivanje teksta.

Ispod ispisanih podataka koji ispunjavaju uvjete SQL pokazivača, ispisuje se legenda oznaka i događaja koji korisniku, koji je pokrenuo posao, u ovom slučaju revizor ili sistemski inženjer, objašnjavaju značenje dobivenih podataka. Dio koda koji pokriva ispis legende prikazan je na slici 4.5..

Linija ***Kod***

```
29:     EXEC OPEN JOBINITC;
30:
31:     IF (SQLCODE < 0) THEN DO;
32:       CALL DSNTIAR (SQLCA, ERROR_MESSAGE, DATA_LEN);
33:       PUT SKIP DATA(ERROR_MESSAGE.ERROR_TEXT);
34:       SIGNAL ERROR;
35:     END;
36:
37:     IF (SQLCODE = 100) THEN DO;
38:       EXEC SQL CLOSE JOBINITC;
39:
40:     DO WHILE (SQLCODE=0);
41:      EXEC SQL
42:        FETCH JOBINITC INTO :VJOBINIT.EVENT_QUAL,
43:        :VJOBINIT.EVT_USER_ID,
44:        :VJOBINIT.TIME_WRITTEN,
45:        :VJOBINIT.DATE_WRITTEN,
46:        :VJOBINIT.USER_NAME,
47:        :VJOBINIT.SYSTEM_SMFID
48:        :VJOBINIT.TERM;
49:
50:      PUT SKIP EDIT('DO:',VJOBINIT.EVENT_QUAL,
51:      'ID:',VJOBINIT.EVT_USER_ID,
52:      'V:',VJOBINIT.TIME_WRITTEN,
53:      'DM:',VJOBINIT.DATE_WRITTEN,
54:      'S:',VJOBINIT.SYSTEM_SMFID)(A,COL(5),A,COL(15),A,COL(19),A,
55:      COL(29),A,COL(32),A,COL(42),A,COL(46),A,COL(58),A,COL(61),A);
56:
57:      PUT SKIP EDIT('T:',VJOBINIT.TERM,
58:      'IK:',VJOBINIT.USER_NAME)(A,COL(4),A,COL(15),A,
59:      COL(19),A);
60:
61:      PUT SKIP EDIT('#',(70)'-','#')(A);
62:     END;
```

Sl. 4.4. Rukovanje greškama, dohvaćanje i ispis podataka nakon otvaranja pokazivača

Linija ***Kod***

```
74:     PUT SKIP EDIT('LEGENDA DOGADAJA:')(A);
75:     PUT SKIP EDIT('INVPSWD','NETOCNA LOZINKA','PWDEXPR',
76:     'LOZINKA JE ISTEKLA','INVNPWD','NEVAZECA NOVA LOZINKA')
77:     (A,COL(11),A,SKIP);
…:      …
93:     END INITSQL;
```

Sl. 4.5. Primjer implementiranja legende događaja

4.1.1. Prevođenje i pokretanje programa

Kako bi korisnik mogao pokrenuti prikazani program, prvo ga je potrebno prevesti, a to se radi definiranjem JCL posla koji koristi sistemski implementiranu proceduru PLICMP#2, kojoj se predaje parametar MEM (engl. *member*) vrijednosti INITSQL, odnosno predaje se podatkovni skup koji sadrži programsko rješenje i projekt SMFRPT u kojem se ovaj član nalazi (slika 4.6.). Nakon uspješnog prevođenja programa (povratna vrijednost je 0 ili 4), program se može pokrenuti pomoću drugog JCL posla (slika 4.7.) koji učitava prevedeni program te će izvješće ispisati u izlaznu datoteku, SYSPRINT.

```
EDIT       APPL.SMFRPT.JCL(INITSQL) - 01.00              Columns 00001 00072
Command ===>     SUB                                      Scroll ===> DATA
****** ***************************** Top of Data ********************************
==MSG> -CAUTION- Profile changed to NUMBER ON STD (from NUMBER OFF).
==MSG>           Data has valid standard numbers.
000100 //INITSQL  EXEC PROC=PLICMP#2,
000200 // MEM=INITSQL,
000300 // APL=SMFRPT
****** ***************************** Bottom of Data *****************************
```

Sl. 4.6. JCL za prevođenje INITSQL-a

```
Command ===>       sub                                      Scroll ===> DATA
****** ***************************** Top of Data ********************************
000100 //RUNINIT  EXEC PGM=IKJEFT01,
000200 // REGION=0M,
000300 // DYNAMNBR=20
000400 //STEPLIB   DD  DSN=APPL.SMFRPT.LOAD,
000500 // DISP=SHR
000600 //          DD  DSN=DSNC10.SDSNLOAD,
000700 // DISP=SHR
000800 //SYSTSPRT DD SYSOUT=*
000900 //SYSPRINT DD SYSOUT=*
001000 //SYSTSIN  DD *
001100   PROFILE NOPREFIX
001200   DSN SYSTEM(DBCG)
001300   RUN PROGRAM(INITSQL)    LIB(APPL.SMFRPT.LOAD) PLAN(SMFRPT)
001400 /*
```

Sl. 4.7. JCL za pokretanje INITSQL programa

Slikom 4.8. prikazano je dobiveno izvješće nakon pokretanja programa te je vidljivo kako je dana 31.05.2024., isti korisnik prvo unio pogrešno korisničko ime koje ne postoji u sustavu (UNDFUSER), a zatim i pogrešnu lozinku (INVPSWD). Legenda ispod izvješća pojašnjava korisniku značenje oznake događaja.

Sl. 4.8. Dobiveno izvješće o neuspješnim pokušajima pristupa sustavu

4.2. Program za ispis izvješća o neuspješnim zahtjevima za pristup podacima (VACCESS)

Za podatke spremljene u tablicu VACCESS koristi se ista logika kao i kod prethodne tablice, ali je prilagođena prema potrebnim podacima, osim već prije spomenutih oznaka događaja, korisnika, vremena i datuma, korisnikova punog imena, imena sustava i terminala, ovdje su još bitne informacije o imenu resursa kojemu se pokušalo pristupiti (RES_NAME), klasi (CLASS) i zahtjevu za čitanjem ili pisanjem (REQUEST). Slikom 4.9. prikazano je definiranje pokazivača ACCESSC koji se koristi za dohvaćanje podataka iz tablice VACCESS. Kao i kod JOBINITC

Linija ***Kod***

```
17:     EXEC SQL INCLUDE SQLCA;
18:     EXEC SQL INCLUDE VACCESS;
19:     EXEC SQL
20:
21:     DECLARE ACCESSC CURSOR FOR
22:     SELECT EVENT_QUAL,EVT_USER_ID, TIME_WRITTEN,DATE_WRITTEN, USER_NAME,
23:     SYSTEM_SMFID, RES_NAME, TERM, CLASS, REQUEST
24:     FROM VACCESS
25:     WHERE (TIMESTAMP_WRITTEN >= CURRENT TIMESTAMP - 1 DAY)
26:     AND EVENT_QUAL <> 'SUCCESS'
```

Sl. 4.9. Definiranje ACCESSC pokazivača za dohvaćanje podataka

pokazivača, i ovdje se dohvaćaju podaci koji zapisani u zadnjih 24 sata od trenutka pokretanja posla te oznaka događaja mora biti različita od 'SUCCESS'.

Isto kao i kod otvaranja pokazivača JOBINITC, prvo se provjerava je li došlo do pogreške prilikom otvaranja ACCESSC pokazivača te postoje li podaci u tablici prema zadanim uvjetima, a na slici 4.10. prikazan je programski kod za dohvaćanje i ispis podataka nakon uspješnog izvršavanja SQL naredbe:

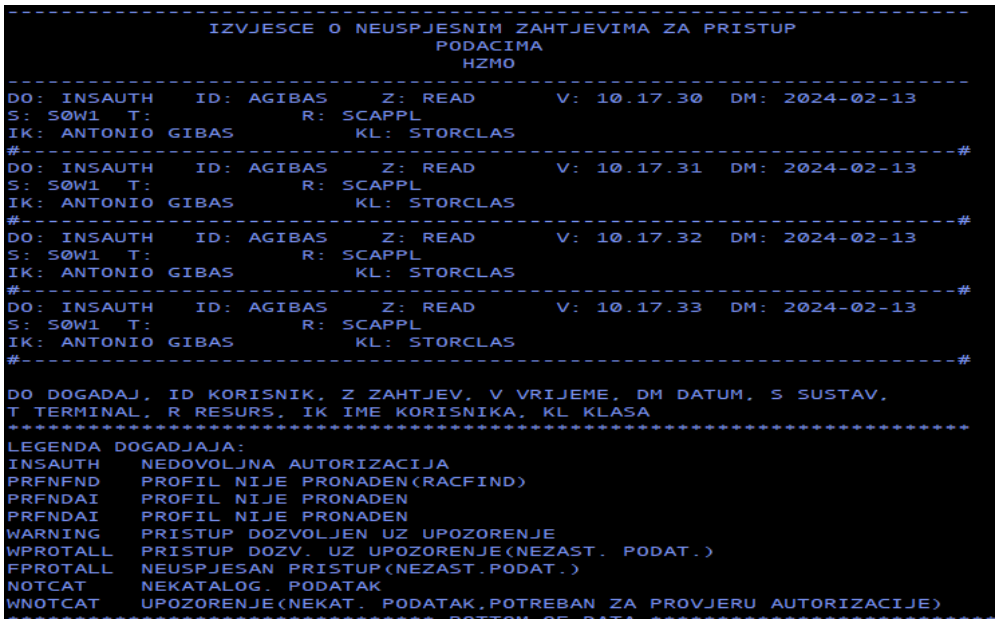
Linija Kod

```
40:     DO WHILE (SQLCODE=0);
41:      EXEC SQL
42:        FETCH ACCESSC INTO :VACCES.EVENT_QUAL,
43:        :VACCESS.EVT_USER_ID,
44:        :VACCESS.TIME_WRITTEN,
45:        :VACCESS.DATE_WRITTEN,
46:        :VACCESS.USER_NAME,
47:        :VACCESS.SYSTEM_SMFID,
48:        :VACCESS.RES_NAME,
49:        :VACCESS.TERM,
50:        :VACCESS.CLASS,
51:        :VACCESS.REQUEST;
52:      PUT SKIP EDIT('DO:',VACCESS.EVENT_QUAL,
53:      'ID:',VACCESS.EVT_USER_ID,
54:      'Z:',VACCESS.REQUEST,
55:      'V:',VACCESS.TIME_WRITTEN,
56:      'DM:',VACCESS.DATE_WRITTEN)(A,COL(5),A,COL(15),A,COL(19),A,
57:      COL(29),A,COL(32),A,COL(42),A,COL(45),A,COL(55),A,COL(59),A);
58:
59:      PUT SKIP EDIT('S:',VACCESS.SYSTEM_SMFID, ,
60       'T:',VACCESS.TERM,
61:      'R',VACCESS.RES_NAME)(A,COL(4),A,COL(10),A,COL(13),A,COL(23),
62:      A,COL(26),A(46));
63:
64:      PUT SKIP EDIT('IK:',VACCESS.USER_NAME,
65:      'KL:',VACCESS.CLASS) A,COL(5),A,COL(27),A,COL(31));
66:
67:      PUT SKIP EDIT('#',(70)'-','#')(A);
68:     END;
```

Sl. 4.10. Dohvaćanje i ispis podataka nakon otvaranja pokazivača ACCESSC

4.2.1. Prevođenje i pokretanje programa

Programsko rješenje koje se nalazi u pod. skupu ACCSSQL, prevodi se i pokreće na isti način kao i INITSQL na slikama 4.6. i 4.7., samo je parametar MEM prilagođen ACCSSQL programu. Izvješće dobiveno ovim programom prikazano je slikom 4.11., a prema njemu, korisnik je nekoliko puta pokušao pristupiti resursi iako za taj resurs nije imao autorizaciju. Kao i kod INITSQL programa, i ovdje postoji legenda događaja i oznaka koja korisniku omogućuje lakše razumijevanje dobivenog izvještaja.

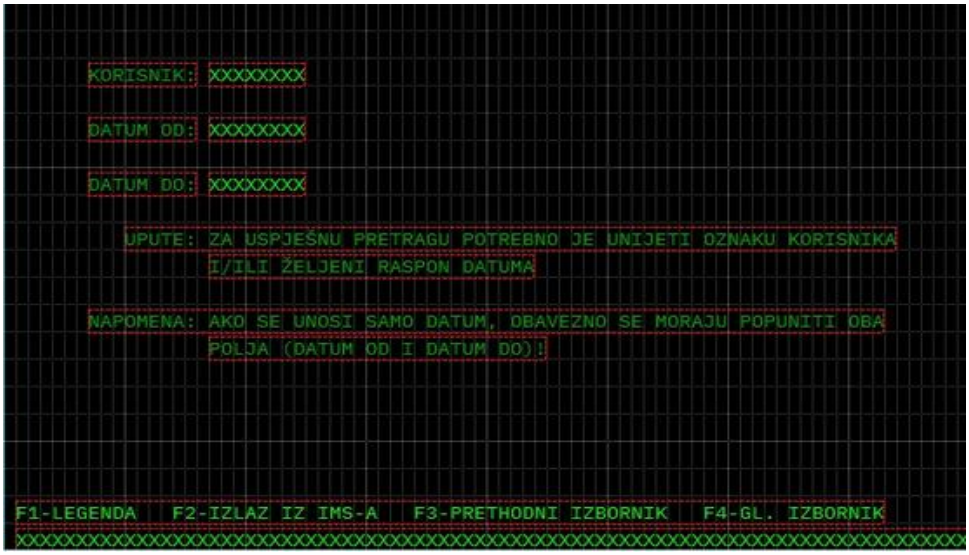


Sl. 4.11. Dobiveno izvješće o neuspješnim pokušajima pristupa podacima

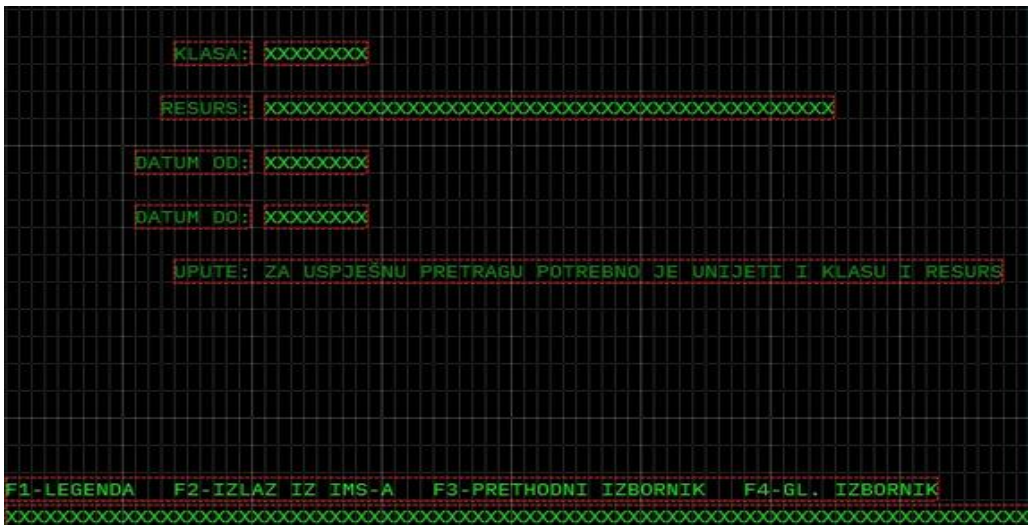
Ovime je završen prvi dio programskog rješenja koji će korisnicima, poput revizora i sistemskih administratora, omogućiti lakšu provjeru drugih korisnika koji pokušavaju s nedovoljnim autorizacijskim pravima pristupiti sustavu ili resursima, a program se u budućnosti može poboljšati dodatnim funkcijama kao što je slanje izvješća na email adresu korisnika koji zatraže izvješće o sigurnosnim događajima u protekla 24 sata.

5. PROGRAMSKO RJEŠENJE ZA IMS TRANSAKCIJU

Drugi dio programskog rješenja obuhvaća izradu aplikacije koja koristi IMS transakcije kako bi korisnik mogao dobiti ispis izvješća u stvarnom vremenu prema određenim uvjetima, za JOBINIT tablicu to su ime korisnika i raspon datuma, a kod ACCESS ime resursa, klase u kojoj se resurs nalazi te željeni raspon datuma. Prije nego što se započne pisanje programskog rješenja za korištenje IMS transakcije koristeći PL/I programski jezik, prvo je bilo potrebno dizajnirati izgled ekrana, odnosno sučelja koje će se koristiti za slanje ulaznih poruka kroz IMS transakciju, a izgledi ekrana prikazani su slikama 5.1. i 5.2., koji prikazuju izgled ulaznog ekrana za rad s podacima iz tablice JOBINIT i ACCESS.



Sl. 5.1. Ekran za unos ulaznih podataka za prikaz neuspješnih pokušaja pristupa sustavu



Sl. 5.2. Ekran za unos ulaznih podataka za prikaz neuspješnih zahtjeva prema resursima

Osim prostora za upis potrebnih uvjeta za pretragu, poput imena korisnika, klase, resursa i datuma, kreirani su i prečaci za lakše kretanje po sučelju, a donji dio ekrana koji je označen slovima X, rezerviran je za ispis poruke o mogućoj grešci prilikom pretrage uvjeta.

Dio programskog rješenja kojeg sadrže i rješenje za program za pretragu i ispis izvještaja o neuspješnim pokušajima pristupa sustavu i slanja zahtjeva za pristup resursima, je definiranje ulazne i izlazne strukture, prikazano slikom 5.3.:

Linija ***Kod***

```
1:      DCL
2:      1 ULAZ
3:        2 LEN       BIN FIXED(31,0),
4:        2 *         CHAR(2),
5:        2 TRN_CODE  CHAR(9),
6:        2 DATAU,
7:          [ulazni podaci u koje će biti spremljeni uneseni uvjeti];
8:
9:      DCL
10:     1 IZLAZ,
11:       2 LEN   BIN FIXED(31,0),
12:       2 *     CHAR(2),
13:       2 ATTR  CHAR(1),
14:       2 DATAI,
15:         [izlazni podaci u koje će biti spremljeni podaci iz baze];
```

Sl. 5.3. Primjer ulaznih i izlaznih struktura za rad s IMS-om

Varijabla LEN je duljina LL iz duljine polja LLZZ, dok ZZ predstavlja varijabla *, TRN_CODE je varijabla namijenjena za spremanje transakcijskog koda uz pomoću kojeg će se izvršiti odgovarajuća transakcija, a podstrukture DATAU i DATAI sadrže varijable u koje će biti spremljeni dohvaćeni podaci iz baze podataka prema unesenim uvjetima za pretragu, a ove se varijable razlikuju s obzirom na to koji program će korisniku biti potreban za pretragu, 7Z5 koji je namijenjen pretrazi za neuspješne prijave u sustav (JOBINIT tablica), ili 7Z6 koji je namijenjen za dohvaćanje neuspješnih zahtjeva za pristup resursima zbog nedovoljnog prava na autorizaciju (ACCESS tablica). Također, prije nego se uđe u petlju za provjeru unesenih podataka, u svakom programu se prvo poziva funkcija PLITDLI(*par_number, function, pcb area, io area*) gdje je par_number broj parametara koje će funkcija primiti, izuzev ovog parametra, function označava funkciju IMS-a koja će se koristiti, u ovom slučaju za ulaz je to funkcija GU (engl. *Get Unique*) koja dohvaća prvi segment nove poruke, pcb area (engl. *Program Specification Block*) je programski prostor DCPTR, a io area je prostor koji se koristi za primanje unesenih podataka, u

ovom slučaju je to struktura ULAZ. Poziv ove funkcije prikazan je sljedećom linijom koda: `CALL PLITDLI(PAR_NUM, 'GU  ', DCPTR, ULAZ);`

5.1. PL/I programsko rješenje za 7Z5 IMS transakciju

Kao i kod kreiranja izvještaja u prvom dijelu programskog rješenja, prvo je potrebno definirati pokazivač SEARCHDATA koji će olakšati dohvaćanje podataka iz baze, a za razliku od pokazivača kojim su se dohvaćali podaci samo za prethodna 24 sata, ovaj pokazivač dohvaća podatke iz stupaca EVENT_QUAL, EVT_USER_ID, TIME_WRITTEN, DATE_WRITTEN, USER_NAME i TERM ako se ime korisnika i raspon datuma podudaraju sa podacima poslanim kroz IMS transakciju i ako identifikator događaja ne sadrži uspješnu vrijednost. Ako korisnik u program ne upiše ime korisnika za kojeg želi ispisati aktivnost ili raspon datuma, koriste se *null* indikatori koji bazi govore da u tom slučaju koristi zadane vrijednosti, za nedostatak vrijednosti za EVT_USER_ID to je oznaka % kojom se bazi onda govori da dohvati sve vrijednosti imena korisnika za koje vrijedi da su u odabranom rasponu datuma izazvali sigurnosni događaj, a u slučaju neupisivanja oba datuma, zadana vrijednost prema kojoj se pretražuju datumi je od 01.01.0001. godine do trenutnog dana. Podaci se sortiraju od novijeg zapisa prema starijem, a slika 5.4. prikazuje kod za definiranje objašnjenog pokazivača.

Linija ***Kod***

```
72:     EXEC SQL
73:       DECLARE SEARCHDATA CURSOR FOR
74:       SELECT EVENT_QUAL,
75:       EVT_USER_ID,
76:       TIME_WRITTEN,
77:       DATE_WRITTEN,
78:       USER_NAME,
79:       TERM
80:       FROM VJOBINIT
81:     WHERE EVT_USER_ID LIKE CONCAT(NVL(:DATAU.KORISNIK:NULLIND1,'%'),'%')
82:       AND (DATE_WRITTEN BETWEEN NVL(:TPDATOD:NULLIND2,'0001-01-01')
83:       AND NVL(:TPDATDO:NULLIND3,CURRENT DATE))
84:       AND (EVENT_QUAL NOT IN ('SUCCESS','TERM','SUCCESSI','SUCCESST',
85:       'RACINITI','RACINITD','SUCCESSP','SUCCESSM','MFAPSUCC'))
86:       ORDER BY TIMESTAMP_WRITTEN DESC;
```

Sl. 5.4. Primjer ulaznih i izlaznih struktura za rad s IMS-om

Nakon definiranja pokazivača i pozivanja funkcije PLITDLI koja dohvaća prvi segment poruke, ulazi se u petlju koja provjerava ima li segmenata u redu čekanja, ako ima, prvo se provjerava je li primljeni status o porukama prazan, ako nije prazan poziva se funkcija RZ0AER koja služi za rukovanje pogreškama. Ako postoje segmenti u redu čekanja i status je prazan, provjeravaju se uneseni podaci, a taj dio koda vidljiv je na slici 5.5.:

Linija ***Kod***

```
100:    DO WHILE(CSTATUS <> 'QC');
101:      IF CSTATUS <> '  ' THEN
102:      CALL RZ0AER(DCMASK, 'CISRT-1', ULAZ, OPCIJA);
103:      IF(DATAU.KORISNIK <> '' & (DATAU.DATOD <> '' & DATAU.DATDO <> ''))
104:      THEN CALL FETCHDATA;
105:      ELSE IF(DATAU.KORISNIK <> '') THEN CALL FETCHDATA;
106:      ELSE
107:       IF(DATAU.DATOD <> '' & DATAU.DATDO <> '') THEN CALL FETCHDATA;
108:      ELSE
109:       DO;
110:        IZLAZ.PORUKA = RIGHT('NEISPRAVAN UNOS!',LENGTH(IZLAZ.PORUKA));
111:        CALL P_SEND;
112:       END;
113:      FIRSTREC = '1'B;
114:      COUNTREC = 0;
115:      PAR_NUM = 3;
116:      CALL PLITDLI(PAR_NUM, 'GU  ', DCPTR, ULAZ);
117:    END;
```

Sl. 5.5. Provjera unesenih podataka za program 7Z5

Funkcija FETCHDATA poziva se u slučaju kada je unesen korisnik i/ili oba datuma, ako nije unesen korisnik potrebno je unijeti oba datuma kako bi program ispravno radio, a ako se ne unese niti jedan podatak ili ako se unese korisnik i samo jedan datum, u izlaznu strukturu u zapisuje se poruka o neispravnom unosu te se poziva funkcija P_SEND koja šalje poruku na ekran. Nakon izlaska iz petlje, pomoćne varijable FIRSTREC i COUNTREC, koje se koriste u funkcijama FETCHDATA i P_SEND, postavljaju se na vrijednosti '1' bit i 0 te se ponovno poziva funkcija PLITDLI sa ulaznim parametrima kako bi se provjerilo ima li još poruka u redu za čekanje.

Nakon pozivanja funkcije FETCHDATA, prvo se provjeravaju jesu li uvjeti za korisnika, datum od ili datum do prema kojima će provesti pretraga i dohvaćanje podataka prazni. Ako je ijedan uvjet prazan, za njega se postavlja *null* indikator koji će SQL pokazivaču reći da umjesto prazne varijable, koristi zadanu vrijednost postavljenu u pokazivaču, a uz to se datumi preoblikuju iz

unesenog oblika (DDMMYYY) u oblik koji je zapisan u bazi podataka (YYYY-MM-DD), što je i prikazano slikom 5.6.:

Linija ***Kod***

```
123:    FETCHDATA: PROC;
124:     IF (DATAU.KORISNIK = '') THEN
125:       NULLIND1 = -1;
126:     IF (DATAU.DATOD = '') THEN
127:       NULLIND2 = -1;
128:     ELSE
129:       TPDATOD = REPATTERN(DATAU.DATOD,'YYYY-MM-DD','DDMMYYYY');
130:     IF (DATAU.DATDO = '') THEN
131:       NULLIND3 = -1;
132:     ELSE
133:       TPDATDO = REPATTERN(DATAU.DATDO,'YYYY-MM-DD','DDMMYYYY');
```

Sl. 5.6. Definiranje funkcije FETCHDATA i postavljanje null indikatora

Zatim se otvara pokazivač te provjerava vrijednost SQLCODE-a, kada je ona 0, tada se iz pokazivača dohvaćaju potrebne informacije te se poziva funkcija PREPARE_SCREEN kojom će se ekran pripremiti za ispisivanje dohvaćenih podataka, kada je vrijednost SQLCODE-a manja od 0, poziva se funkcija RZ0AER za rukovanjem pogreškama, a kada je vrijednost jednaka 100, ispisuje se poruka kako za tražene uvjete nema podataka ili kako više nema podataka za pretragu, ovisno o vrijednostima varijabli FIRSTREC i COUNTREC. Ako je vrijednost FIRSTREC varijable postavljena na '1' bit, a COUNTREC 0, to znači kako još niti jedan zapis nije dohvaćen te će se na izlaz ispisati poruka o nepostojanju podataka za tražene uvjete. Slika 5.7. prikazuje opisani kod.

Linija Kod

```
139:    EXEC SQL OPEN SEARCHDATA;
140:
141:    DO WHILE (SQLCODE=0);
142      REINIT VJOBINIT;
143:     EXEC SQL
144:       FETCH SEARCHDATA INTO :VJOBINIT.EVENT_QUAL,
145:       :VJOBINIT.EVT_USER_ID,
146:       :VJOBINIT.TIME_WRITTEN,
147:       :VJOBINIT.DATE_WRITTEN,
148:       :VJOBINIT.USER_NAME,
149:       :VJOBINIT.TERM;
150:     IF SQLCODE <> 0 THEN LEAVE;
151:     CALL PREPARE_SCREEN;
152:    END;
153:
154::   IF (SQLCODE < 0) THEN DO;
155:      CALL RZ0AER(SQLCA, 'S ERRFETCH, VJOBINIT, OPCIJA);
156:    END;
157:
158:    IF (SQLCODE = 100) THEN DO;
159:      IF (FIRSTREC & COUNTREC = 0) THEN
160:        IZLAZ.PORUKA = RIGHT('NEMA PODATAKA ZA TRAZENE UVJETE',
161:                        LENGTH(IZLAZ.PORUKA));
162:      ELSE
163:        DO;
164           IZLAZ.PORUKA = RIGHT('NEMA VISE PODATAKA',
165:                          LENGTH(IZLAZ.PORUKA));
166::       END;
167:      CALL P_SEND;
168:    END;
169:    EXEC SQL CLOSE SEARCHDATA;
170:    END FETCHDATA;
```

Sl. 5.7. Nastavak funkcije FETCHDATA, dohvaćanje podataka i rukovanje pogreškama

Kada se podaci uspješno dohvate i spreme u prihvatne varijable, poziva se funkcija PREPARE_SCREEN (slika 5.8.) kojom se prvo provjerava koliko je zapisa trenutno spremno za ispis na ekran, u slučaju da je COUNTREC = 13, to znači da je trenutni ekran za ispis prema dizajnu dosegao svoj maksimum te se ostali podaci pripremaju za ispis na sljedeći ekran te se poziva funkcija P_SEND koja će ispisati prikladnu poruku, a vrijednost varijable COUNTREC

vraća se na 0. Nakon te provjere, u varijable izlazne strukture zapisuju se podaci koji su spremljeni u prihvatne varijable te su spremni za slanje na ekran uz pomoć funkcije P_SEND (slika 5.9.)

Linija Kod

```
178:    PREPARE_SCREEN: PROC;
179:      IF COUNTREC = 13 THEN
180:        DO;
181:          IZLAZ.PORUKA = RIGHT('NASTAVAK',LENGTH(IZLAZ.PORUKA));
182:         CALL P_SEND;
183:         COUNTREC = 0;
184:        END;
185:      COUNTREC += 1;
186:      IZLAZ.UVJKORISNIK = DATAU.KORISNIK;
187:      IZLAZ.UVJDATOD = DATAU.DATOD;
188:      IZLAZ.UVJDATDO = DATAU.DATDO;
189:      IF (VJOBINIT.DATE_WRITTEN <> '') THEN
190:        IZLAZ.REZ_TABLICA(COUNTREC).DATUM =
191:        REPATTERN(VJOBINIT.DATE_WRITTEN, 'DDMMYYYY', 'YYYY-MM-DD');
192:      IZLAZ.REZ_TABLICA(COUNTREC).VRIJEME = VJOBINIT.TIME_WRITTEN;
193:      IZLAZ.REZ_TABLICA(COUNTREC).DOGADJAJ = VJOBINIT.EVENT_QUAL;
194:      IZLAZ.REZ_TABLICA(COUNTREC).KORISNIK = VJOBINIT.EVT_USER_ID;
195:      IZLAZ.REZ_TABLICA(COUNTREC).TERMINAL = VJOBINIT.TERM;
196:      IZLAZ.REZ_TABLICA(COUNTREC).IME_KOR = VJOBINIT.USER_NAME;
197:    END PREPARE_SCREEN;
```

Sl. 5.8. Definiranje funkcije PREPARE_SCREEN

Linija Kod

```
123:    P_SEND: PROC;
124:      IF FIRSTREC THEN
125:        DO;
126:          PAR_NUM = 4;
127:          FIRSTREC = '0'B;
128:        END;
129:      ELSE
130:        DO;
131:          PAR_NUM = 3;
132:        END;
133:      CALL PLITDLI(PAR_NUM,'ISRT', DCMASK, IZLAZ, DESIGN_NAME);
134:      IF (CSTATUS <> '  ') THEN
135:        CALL RZ0AER(DCMASK, 'C-ISRT', IZLAZ, OPCIJA);
136:      REINIT IZLAZ;
137:    END P_SEND;
```

Sl. 5.9. Definiranje funkcije P_SEND

Funkcija P_SEND provjerava šalje li se na ekran prvi zapis, ako je uvjet istinit, broj parametara potrebnih za poziv PLITDLI funkcije za ispis poprima vrijednost 4, s obzirom na to da će funkciji

trebati parametri ISRT, koja govori funkciji da pošalje trenutni segment na ekran, IZLAZ koji sadrži zapise za ispis i varijabla DESIGN_NAME koja predstavlja ime ekrana u IMS-u na kojem će se prikazati ispis. U slučaju bilo kakve pogreške, poziva se funkcija RZ0AER koja u sistemske podatkovne skupove ispisuje informacije o pogrešci. Ovime je završena implementacija programskog rješenja za program 7Z5, odnosno za prikaz korisnika koji su neuspješno pokušali pristupiti sustavu.

5.2. PL/I programsko rješenje za 7Z6 IMS transakciju

Slikom 5.10. prikazan je kod za definiranje SEARCHDATA pokazivača za dohvaćanje podataka iz tablice ACCESS, a dohvaćaju se resursi koji odgovaraju imenima klase i resursa koji su uneseni kao i rasponu datuma. Kao i kod pokazivača za transakciju 7Z5, u slučaju da neki od potrebnih podataka nije unesen, uz pomoć *null* indikatora se bazi govori da dohvati potrebne podatke sa zadanim vrijednostima.

Linija ***Kod***

```
81:     EXEC SQL
82:       DECLARE SEARCHDATA CURSOR FOR
83:       SELECT EVENT_QUAL,
84:       EVT_USER_ID,
85:       TIME_WRITTEN,
86:       DATE_WRITTEN,
87:       REQUEST,
88:       CLASS,
89:       RES_NAME
90:       FROM VACCESS
91:       WHERE CLASS LIKE CONCAT(NVL(:DATAU.KLASA:NULLIND1,'%'),'%')
92:       AND RES_NAME LIKE CONCAT(NVL(:DATAU.RESURS:NULLIND2,'%'),'%')
93:       AND (DATE_WRITTEN BETWEEN NVL(:TPDATOD:NULLIND3,'0001-01-01')
94:            AND NVL(:TPDATDO:NULLIND4,CURRENT DATE))
95:       AND EVENT_QUAL <> 'SUCCESS'
96:       ORDER BY TIMESTAMP_WRITTEN DESC;
```

Sl. 5.10. Definiranje pokazivača za rad s tablicom ACCESS

Pozivom funkcije PLITDLI provjerava se postoje li ulazni segmenti u redu čekanja, ako postoje, provjera se koji su podaci uneseni kako bi se nadalje mogla pozvati funkcija FETCHDATA ili kako bi se poruka o pogrešci ispisala na ekran uz pomoć funkcije P_SEND. Provjera unesenih podataka prikazana je slikom 5.11.. Kako bi se transakcija 7Z6 uspješno provela, potrebno je unijeti ili ime klase i resursa za koju se provjeravaju neuspješni pokušaji pristupa, ili raspon datuma. U slučaju da se unese samo ime resursa bez klase i samo jedan datum, krajnjem korisniku

će se prikazati poruka o pogrešnom unosu. Slikom 5.12. prikazana je implementacija funkcije FETCHDATA kojom se inicijaliziraju *null* indikatori za SQL upit i dohvaćaju podaci u prihvatne varijable, dok je slikom 5.13. prikazana funkcija PREPARE_SCREEN prilagođena za rad s transakcijom 7Z6. Funkcija P_SEND jednaka je kao i funkcija prikazana slikom 5.9.

Linija ***Kod***

```
100:    DO WHILE(CSTATUS <> 'QC');
101:      IF CSTATUS <> '  ' THEN
102:      CALL RZ0AER(DCMASK, 'CISRT-1', ULAZ, OPCIJA);
103:      IF((DATAU.KLASA <> '' & DATAU.RESURS <> '') &
104:      (DATAU.DATOD <> '' & DATAU.DATDO <> ''))
105:      ELSE
106:       IF(DATAU.KLASA <> '' & DATAU.RESURS <> '') THEN CALL FETCHDATA;
107:      ELSE
108:       IF(DATAU.DATOD <> '' & DATAU.DATDO <> '')) THEN CALL FETCHDATA;
109:      ELSE
110:       DO;
111:        IZLAZ.PORUKA = RIGHT('NEISPRAVAN UNOS!',LENGTH(IZLAZ.PORUKA));
112:        CALL P_SEND;
113:       END;
114:      FIRSTREC = '1'B;
115:      COUNTREC = 0;
116:      PAR_NUM = 3;
117:      CALL PLITDLI(PAR_NUM, 'GU  ', DCPTR, ULAZ);
118:    END;
```

Sl. 5.11. Provjera unesenih podataka za program 7Z6

Linija Kod

```
135:    FETCHDATA: PROC;
136:     IF (DATAU.KLASA = '') THEN
137:       NULLIND1 = -1;
138:     IF (DATAU.RESURS = '')
139:       NULLIND2 = -1;
140:     IF (DATAU.DATOD = '')
141:     THEN NULLIND3 = -1;
142:     ELSE TPDATOD = REPATTERN(DATAU.DATOD,'YYYY-MM-DD','DDMMYYYY');
143:     IF (DATAU.DATDO = '')
144:     THEN NULLIND4 = -1;
145:     ELSE  TPDATDO = REPATTERN(DATAU.DATDO,'YYYY-MM-DD','DDMMYYYY');
146:
147:    EXEC SQL OPEN SEARCHDATA;
148:
149:    DO WHILE (SQLCODE=0);
150:     REINIT VACCESS;
151:     EXEC SQL
152:       FETCH SEARCHDATA INTO :VACCESS.EVENT_QUAL,
153:       :VACCESS.EVT_USER_ID,
154:       :VACCESS.TIME_WRITTEN,
155:       :VACCESS.DATE_WRITTEN,
156:       :VACCESS.REQUEST,
157:       :VACCESS.CLASS,
158:       :VACCESS.RES_NAME;
159:     IF SQLCODE <> 0 THEN LEAVE;
160:     CALL PREPARE_SCREEN;
161:    END;
162:
163:    IF (SQLCODE < 0) THEN DO;
164:      CALL RZ0AER(SQLCA, 'S ERRFETCH, VJOB, OPCIJA);
165:    END;
166:
167:    IF (SQLCODE = 100) THEN DO;
168:      IF (FIRSTREC & COUNTREC = 0) THEN
169:        IZLAZ.PORUKA = RIGHT('NEMA PODATAKA ZA TRAZENE UVJETE',
170:                        LENGTH(IZLAZ.PORUKA));
171:      ELSE
172:        DO;
173:          IZLAZ.PORUKA = RIGHT('NEMA VISE PODATAKA',
174:                          LENGTH(IZLAZ.PORUKA));
175:        END;
176:      CALL P_SEND;
177:    END;
178:    EXEC SQL CLOSE SEARCHDATA;
179:    END FETCHDATA;
```

Sl. 5.12. Definiranje funkcije FETCHDATA i postavljanje null indikatora

Linija Kod

```
190:    PREPARE_SCREEN: PROC;
191:      IF COUNTREC = 5 THEN
192:        DO;
193:          IZLAZ.PORUKA = RIGHT('NASTAVAK',LENGTH(IZLAZ.PORUKA));
194:         CALL P_SEND;
195:         COUNTREC = 0;
196:        END;
197:      COUNTREC += 1;
198:      IZLAZ.UVJKLASA = DATAU.KLASA;
199:      IZLAZ.UVJRES = DATAU.RESURS;
200:      IZLAZ.UVJDATOD = DATAU.DATOD;
201:      IZLAZ.UVJDATDO = DATAU.DATDO;
202:      IF (VACCESS.DATE_WRITTEN <> '') THEN
203:        IZLAZ.REZ_TABLICA(COUNTREC).DATUM =
204:        REPATTERN(VACCESS.DATE_WRITTEN, 'DDMMYYYY', 'YYYY-MM-DD');
205:      IZLAZ.REZ_TABLICA(COUNTREC).VRIJEME = VACCESS.TIME_WRITTEN;
206:      IZLAZ.REZ_TABLICA(COUNTREC).DOGADJAJ = VACCESS.EVENT_QUAL;
207:      IZLAZ.REZ_TABLICA(COUNTREC).KORISNIK = VACCESS.EVT_USER_ID;
208:      IZLAZ.REZ_TABLICA(COUNTREC).ZAHTJEV = VACCESS.REQUEST;
209:      IZLAZ.REZ_TABLICA(COUNTREC).KLASA = VACCESS.CLASS;
210:     IZLAZ.REZ_TABLICA(COUNTREC).RESURS = SUBSTR(VACCESS.RES_NAME,1,44);
211:    END PREPARE_SCREEN;
```

Sl. 5.13. Definiranje funkcije PREPARE_SCREEN

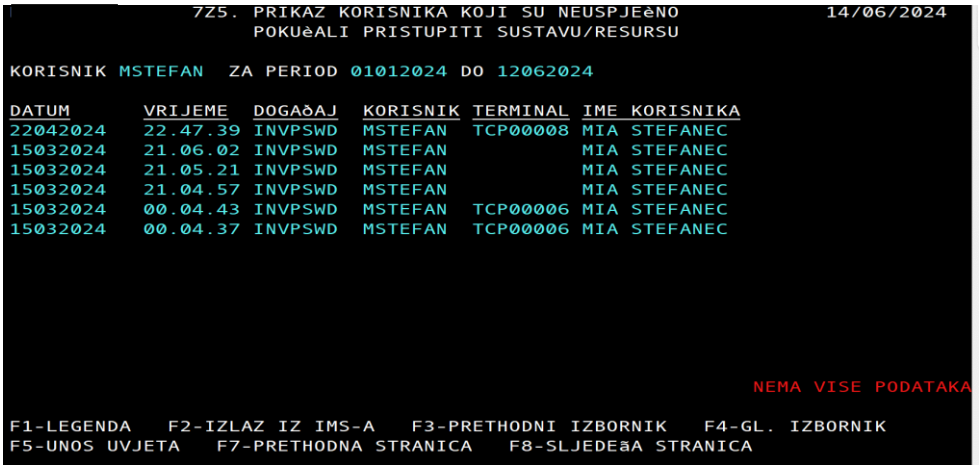
Ovime je završena implementacija programskog rješenja za transakciju 7Z6, a primjeri izlaza 7Z5 i 7Z6 transakcija bit će prikazani u sljedećem potpoglavlju.

5.3. Prikaz izlaznih ekrana transakcija 7Z5 i 7Z6

Nakon što se programi za izvršavanje IMS transakcije prevedu, korisnik poduzima sljedeće korake:

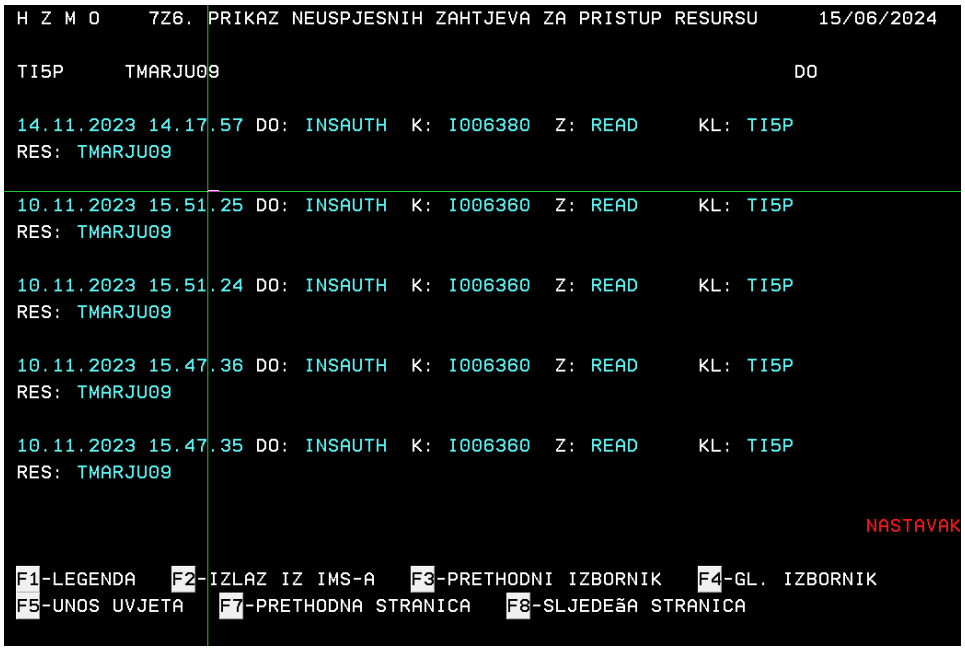
1. prijavljuje se u sustav za korištenje IMS transakcija
2. poziva naredbu IZBORNIK kroz koju bira koju transakciju će koristiti
3. unosi se traženi podaci za izvršavanje transakcije

Slikama 5.14., 5.15. i 5.16. prikazani su neki od mogućih ispisa rezultata za transakcije 7Z5 i 7Z6., na slikama je također vidljivo kako u donjem dijelu ekrana postoji opis prečica na tipkovnici koje korisniku pomažu da se lakše kreće po sučelju te što jednostavnije ponovi svoju pretragu ili izađe iz sustava.



Sl. 5.14. Izgled izlaznog ekrana transakcije 7Z5 kada su uneseni korisnik i raspon datuma

Sl. 5.15. Izgled izlaznog ekrana nakon pogrešnog unosa



Sl. 5.16. Izgled izlaznog ekrana transakcije 7Z6 kada su uneseni ime klase i resursa

6. ZAKLJUČAK

U ovom radu izrađeno je programsko rješenje za automatiziranu proizvodnju izvještaja iz z/OS RACF sustava koji štiti resurse na središnjem računalu dodjeljivanjem prava pristupa samo autoriziranim korisnicima te njegova pravilna upotreba značajno smanjuje rizik od neovlaštenih pristupa. Središnja računala su nezamjenjivi elementi u poslovanju velikih organizacija zbog svoje pouzdanosti, sigurnosti i sposobnosti obrade velikih količina podataka. Također, u teorijskom dijelu rada opisala se revizija koja kao proces procjene i provjere učinkovitosti sigurnosnih kontrola omogućava identificiranje potencijalnih sigurnosnih prijetnji, a opisani su i neki od poznatijih sigurnosnih standarda prema kojima bi organizacije trebale implementirati funkcionalnosti bitne za sigurnost. Korištenjem RACF-a i z/OS-a, moguće je bilježiti sve aktivnosti korisnika putem SMF zapisa, što omogućava detaljne analize sigurnosnih događaja, a implementacija automatiziranog rješenja za proizvodnju sigurnosnih izvještaja iz z/OS RACF sustava od velike je važnosti za učinkovito upravljanje sigurnosnim rizicima i osiguravanje usklađenosti s relevantnim sigurnosnim standardima. Automatizirana izrada sigurnosnih izvješća ključna je za učinkovitu reviziju i praćenje sigurnosnog stanja sustava. Praktični dio rada predstavio je programska rješenja za generiranje izvještaja o sigurnosnim iznimkama i stanju sigurnosnog okruženja, korištenjem PL/I i JCL jezika, Db2 baze podataka kao i IMS transakcija za dinamični pregled izvještaja. Ovo programsko rješenje povećava točnost i učinkovitost revizijskih postupaka te olakšava korisnicima da što lakše dođu do željenih informacija koje se mogu koristiti u internim ili vanjskim revizijskim procesima, a programska rješenja se nadalje mogu poboljšati implementiranjem funkcionalnosti za slanje izvještaja email-om korisniku koji je izvještaj zatražio ili slanjem preko pisača, a programska rješenja koja koriste IMS se mogu poboljšati implementacijom funkcije kojom će se provjeravati je li korisnik u ulazne segmente unio točan format datuma.

LITERATURA

[1] IBM, Auditing overview, IBM, 2021., dostupno na: https://www.ibm.com/docs/en/sva/9.0?topic=auditing-overview [10.01.2024.]

[2] M. Ebbers, J. Kettner, W. O'Brien, B. Ogden, Introduction to the New Mainframe: z/OS Basics, IBM Redbooks, SAD, 2011.

[3] A. Buecker, M. Cairns, M. Conway, M. S. Hahn, D. McLemore, J. Pease, L. Xie, IBM z/OS Mainframe Security and Audit Management Using the IBM Security zSecure Suite, IBM Redbooks, SAD, 2011.

[4] ISO, ISO/IEC 27001: Information security, cybersecurity and privacy protection - Information security management systems – Requirements, ISO/IEC, Švicarska, 2022.

[5] A. Magnusson, ISO 27001 Audit: Everything You Need to Know, StrongDM, 2023., dostupno na: https://www.strongdm.com/blog/iso-27001-audit [13.01.2024.]

[6] PCI Security Standard Council, PCI DSS Quick Reference Guide: Understanding the Payment Card Industry Data Security Standard, PCI Security Standards Council, Srpanj 2018.

[7] B. Wolford, What is GDPR, the EU's new data protection law?, Proton AG, dostupno na: https://gdpr.eu/what-is-gdpr/ [20.01.2024.]

[8] Intersoft Consulting, Security of processing, Intersoft Consulting, Njemačka, dostupno na: https://gdpr-info.eu/art-32-gdpr/ [20.01.2024.]

[9] IBM, Using an ESM with CICS, IBM, 2021., dostupno na: https://www.ibm.com/docs/en/cics-tx/10.1.0?topic=manager-using-external-security-cics [03.02.2024.]

[10] Vanguard Integrity Professionals, Vanguard Advisor for RACF, ACF2 & Top Secret, SAD, dostupno na: https://www.go2vanguard.com/offerings/audit-compliance/advisor/ [05.02.2024.]

[11] Beta Systems, Beta Access: Optimize Your Mainframe Access Control with Our z/OS Administration Tool, Beta Systems Software AG, Njemačka, dostupno na: https://www.betasystems.com/products/beta-access [05.02.2024.]

[12] Broadcom, Compliance Event Manager, Broadcom, dostupno na: https://www.broadcom.com/products/mainframe/compliance-data-protection/compliance-event-manager [05.02.2024.]

[13] Broadcom, Mainframe Security Solution, Broadcom, 2021., dostupno na: https://docs.broadcom.com/docs/mainframe-security-solution-brief [05.02.2024]

[14] IBM, z/OS Concepts, IBM Redbooks, SAD, 2010.

[15] IBM, Assigning optional user attributes, IBM, 2021., dostupno na: https://www.ibm.com/docs/en/zos/2.2.0?topic=groups-assigning-optional-user-attributes [25.03.2024.]

[16] IBM, ISPF Edit and Edit Macros, IBM Corp., SAD, 2023.

[17] IBM, Managing data, IBM, 2023., dostupno na: https://www.ibm.com/docs/en/zos/2.5.0?topic=editor-managing-data [06.04.2024.]

[18] DevX, Programming Language One, DevX, 2023., dostupno na: https://www.devx.com/terms/programming-language-one/ [23.04.2024.]

[19] IBM, MVS System Management Facilities (SMF), IBM Corp., SAD, 2024.

[20] IBM, Security Server RACF: Auditor's Guide, IBM Corp., SAD, 2023.

[21] IBM, IBM Information Management System (IMS), IBM, 2023., dostupno na: https://www.ibm.com/docs/en/integration-bus/10.1?topic=ims-information-management-system [17.06.2024.]

SAŽETAK

U ovom diplomskom radu opisani su sigurnosni standardi koji upućuju organizacije kako pravilno i efikasno implementirati razne sigurnosne kontrole, kao i postojeća rješenja koja prema ovim standardima mogu generirati revizorska izvješća. U radu je implementirano programsko rješenje za automatiziranu proizvodnju sigurnosnih izvještaja o neuspješnim pokušajima pristupa sustavu i neovlaštenim zahtjevima za pristup resursima sustava koristeći funkcionalnosti koje nudi središnje računalo i operacijski sustav z/OS . Najvažnije funkcionalnosti i alati ovog OS-a koji su se koristili kako bi se programsko rješenje uspješno izradilo su zapisi iz SMF-a, programski jezik PL/I, baza podataka Db2 i transakcijski upravitelj IMS. Prvo je opisana implementacija JCL zadatka kojim se potrebne tablice baze podataka pune potrebnim informacijama iz SMF podatkovnih skupova te se implementiralo rješenje PL/I programskim jezikom za ispis izvješća po potrebi korisnika. U drugom dijelu praktičnog rješenja, implementiran je PL/I program za rad s IMS transakcijom i sučeljem za ispis izvješća prema potrebnim uvjetima na zahtjev korisnika.

Ključne riječi: PL/I, programsko rješenje, RACF, sigurnost, središnje računalo, z/OS

ABSTRACT

Development of a Software Solution For the Automated Production of Reports From the z/OS RACF System

This thesis describes the security standards that instruct the organization how to correctly and effectively implement various security controls, as well as existing solutions that can generate audit reports according to these standards. The paper implemented a software solution for the automated production of security reports on unsuccessful attempts to access the system and unauthorized requests for access to system resources using the functionality offered by the central computer and the z/OS operating system. The most important functionalities and tools of this OS that were used to successfully create the software solution are records from SMF, programming language PL/I, database Db2 and transaction manager IMS. First, the implementation of the JCL task was described, which required the database tables filled with the necessary information from the SMF data sets, and the PL/I programming language solution was implemented for printing reports as needed by the user. In the second part of the practical solution, a PL/I program was implemented for working with IMS transactions and an interface for printing reports according to the necessary conditions at the request of the user.

Keywords: mainframe, PL/I, RACF, security, software solution, z/OS

PRILOZI

Prilog 1: Programsko rješenje nalazi se na GitHubu: https://github.com/mia-stefanec/RACFReport

ŽIVOTOPIS

Mia Štefanec rođena je 16. ožujka 1997. godine u Osijeku. Osnovnu školu Dobriša Cesarić u Osijeku završava 2012. godine te upisuje I. gimnaziju Osijek. Godine 2016. upisuje preddiplomski studij računarstva na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija Osijek. Nakon završenog preddiplomskog studija, upisuje diplomski studij Računarstva, smjer Informacijske i podatkovne znanosti.