STUDIA INFORMATICA 2000
Volume 21 Number 3(41)

Arkadiusz RZUCIDŁO
Politechnika Rzeszowska, Zakład Informatyki

ADAPTACJA SYSTEMU OCHRONY W BAZACH DANYCH MS ACCESS DO OBOWIĄZUJĄCYCH PRZEPISÓW PRAWNYCH

Streszczenie. Praca opisuje metody ochrony aplikacji obsługującej bazę danych budowanej z zastosowaniem MS Access. Treść zawiera dwa podstawowe rozwiązania, jakie należy przedsięwziąć, aby zbiór danych podlegał ochronie przewidzianej przez przepisy prawne. Są nimi odpowiednia administracja grupą roboczą oraz utworzenie dziennika zdarzeń.

THE ADAPTATION OF SYSTEM PROTECTION IN MS ACCESS DATA BASES TO THE FORCE OF LAW REGULATIONS

Summary. This paper presents the protection methods for data base applications build by MS Access. The contents includes two of basic solutions, must be undertaken for data source. The protections are provided by law regulations. There are work group administration and data base process recorder.

1. Ochrona danych w świetle obowiązujących przepisów

Zgodnie z Ustawą z dnia 29 sierpnia 1997 r. o ochronie danych osobowych (DzU z dnia 29 października 1997 r.) dane gromadzone w systemach informatycznych podlegają ochronie. Ich bezpieczeństwo uwarunkowane jest szeregiem paragrafów określających zasady gromadzenia, przetwarzania, przechowywania i udostępniania ich osobom do tego upoważnionym. Rozdział piąty tej Ustawy mówi, że :

Art. 36. Administrator danych jest obowiązany do zastosowania środków technicznych i organizacyjnych zapewniających ochronę przetwarzanych danych osobowych, a w szczególności powinien zabezpieczyć dane przed ich udostępnieniem osobom

nieupoważnionym, zabraniem przez osobę nieuprawnioną, uszkodzeniem lub zniszczeniem.

Art. 37. Do obsługi systemu informatycznego oraz urządzeń wchodzących w jego skład, służących do przetwarzania danych, mogą być dopuszczone wyłącznie osoby posiadające upoważnienie wydane przez administratora danych [1].

Z przepisów tych jasno wynika, że tworząc nawet najprostszy system informacyjny zawierający znamiona zbioru danych osobowych, należy przewidzieć sposób jego ochrony przed wglądem w jego zawartość osób nieupoważnionych. Zabezpieczony, skopiowany zbiór danych jest bezużyteczny, jeśli nie stanowi on źródła informacji.

Prócz zabezpieczeń poziomu użytkownika system informacyjny powinien posiadać również rejestrator zdarzeń mających miejsce w trakcie pracy bazy danych. Elementem takim jest dziennik zdarzeń usankcjonowany przepisami. W świetle § 16 pkt 3 i 4 Rozporządzenia Ministra Spraw Wewnętrznych i Administracji z dnia 3 czerwca 1998 r. w sprawie określenia podstawowych warunków technicznych i organizacyjnych, jakim powinny odpowiadać urządzenia i systemy informatyczne służące do przetwarzania danych osobowych (DzU Nr 133, poz. 883) dla każdej osoby, której dane przetwarzane są w systemie informatycznym, system ten powinien zawierać odnotowanie identyfikatora użytkownika wprowadzającego dane oraz informacje, komu, kiedy i w jakim zakresie dane zostały udostępnione, jeśli przewidziane jest udostępnianie danych innym podmiotom, chyba że dane te traktuje się jako dane powszechnie dostępne [2].

MS Access jest częścią składową pakietu Office. Stanowi on uzupełnienie bazy programów biurowych o narzędzie do tworzenia i administrowania bazami danych. Gromadzenie informacji (danych osobowych) powoduje konieczność zastosowania w aplikacji mechanizmów ochrony środowiska pracy oraz samych danych. Program ten posiada wewnętrzny system zabezpieczeń o strukturze wielowarstwowej. Podstawowym jego elementem jest plik grupy roboczej. W nim znajduje się szkielet zabezpieczeń poziomu użytkownika.

Dziennik zdarzeń buduje sam projektant. Aplikacja nie udostępnia rejestratora zdarzeń bazy danych. W treści opisu przedstawiono przykładowy sposób zamodelowania takiego rejestratora.

2. Unikatowe zabezpieczenie środowiska MS Access

System ochrony umieszczony jest w pliku grupy roboczej *system.mdw*, automatycznie tworzonej podczas instalacji programu MS Access w komputerze. Obsługiwany jest przez

program Wrkgadm.exe znajdujący się w folderze Windows\System. Zawartość zbioru ma strukturę hierarchiczną i składa się z konta: administratora oraz dwóch grup użytkowników: *Administratorzy* i *Użytkownicy*. Zarządzający jest z racji posiadanych uprawnień jednocześnie członkiem zarówno jednej, jak i drugiej grupy.

Podczas procesu projektowania bazy danych właścicielem obiektów systemu jak i samego pliku bazy jest użytkownik, który pełni rolę projektanta. Niezależnie od przydzielonych mu praw dostępu i przynależności grupowej zawsze może on otworzyć swoją bazę [3]. Najczęściej użytkownikiem tym jest *Administrator* będący jednocześnie właścicielem budowanego projektu. Systemowa grupa robocza jest grupą standardową. Oznacza to, że każda instalacja MS Access'a, jest w posiadaniu takiego pliku z identyczną hierarchią zabezpieczeń. Tak więc nadanie hasła administratorowi jest tylko zabezpieczeniem lokalnym, obowiązującym na stanowisku projektanta. W celu „ominięcia" tego typu zabezpieczenia wystarczy przenieść plik bazy danych na inną jednostkę wyposażoną w program MS Access, gdzie zabezpieczenia oparte są na grupie systemowej, a administrator nie ma określonego hasła. Z reguły aplikacje użytkowe (MS Access) nie udostępniają możliwości zabezpieczenia zbiorów przed kopiowaniem. Jest to domena systemów operacyjnych, tak więc w pracy pominięto problem ochrony pliku jako zasobu dyskowego.

Aby uniknąć tego typu problemu, należy utworzyć nową unikatową dla projektu grupę roboczą wykorzystując do tego wspomniany wyżej program Workgadm.exe. Wraz z uruchomieniem aplikacji MS Access następuje odwołanie do rejestru z zapytaniem o aktualną nazwę pliku zabezpieczeń. Modyfikacja hierarchii danych zbioru pozwala zbudować wzór sytemu ochrony obowiązującego na jednostce projektowej dla wszystkich projektowanych baz danych. Ilość wzorów zabezpieczeń może być dowolna, a struktura systemu zależy wyłącznie od przyjętego przez projektanta schematu.

Zabezpieczenia baz danych typu MS Access są identyczne w przypadku pracy na lokalnym stanowisku, jak i pracy w wersji sieciowej. Struktura plików grup roboczych obejmuje konta wszystkich osób mających udział w procesie gromadzenia i przetwarzania danych. Współużytkowanie danych w sieci opiera się na dwóch modelach:

- Współużytkowania bazy danych jako całości – gdzie baza danych wraz z elementami tworzącymi środowisko pracy jest w całości umieszczona na serwerze.
- Dzielonej bazy danych – gdzie elementami współużytkowanymi są tylko tabele, zaś środowisko jest dostosowywane przez samego użytkownika (formularze, kwerendy itp.).

Zarówno w pierwszym, jak i w drugim modelu użytkownik korzystający z danych musi być wyposażony w lokalną instalację Access'a. Każdy z tych programów posługuje się plikiem grupy roboczej. Aby ujednolicić system ochrony, należy wspólny zbiór zabezpieczeń umieścić

na serwerze, konfigurując jednocześnie lokalne wersje Access'a w ten sposób, aby ustalenia kont czerpały tylko z jednego pliku.

2.1. Unikatowy plik grupy roboczej

Tworzenie unikatowego pliku grupy roboczej jest elementem stanowiącym o bezpieczeństwie danych zawartych w tworzonej aplikacji. Jest to najbardziej newralgiczny punkt systemu ochrony, ponieważ osoba niepowołana przy próbie przejęcia danych spotyka go na samym początku. Jego zaprojektowanie winno być bardzo staranne i przemyślane. Nie istnieje algorytm, według którego należałoby budować schematy ochrony, jednak konto administratora z racji swoich uprawnień w każdym z projektów będzie miało taką samą postać. W tym opisie pominięto omówienie uprawnień kont zwykłych użytkowników, a skupiono się na ustawieniu odpowiedniego środowiska dla administratora.

Tworzenie indywidualnego pliku grupy roboczej dla projektu rozpoczyna się od uruchomienia programu Workgadm.exe. Aplikacja ta informuje o nazwie obecnie używanego pliku zabezpieczeń. Wybranie opcji *Utwórz* pozwoli na utworzenie nowej grupy. Po podaniu nazwy firmy oraz identyfikatora grupy (unikatowej wartości będącej symbolem grupy) program zapisuje na dysku pod podaną nazwą nowy schemat hierarchiczny, obowiązujący od tej chwili. Unikatowość grupy wynika tylko z jej odmiennego identyfikatora, natomiast wszelkie ustawienia schematu zabezpieczeń są jeszcze identyczne jak w przypadku grupy systemowej. Dalsze prace pozwolą na osiągnięcie określonego celu, czyli zunifikowania opisywanego systemu.

2.2. Reorganizacja uprawnień w nowym pliku grupy roboczej

Kolejnym krokiem w tworzeniu zabezpieczeń jest wykreowanie nowego konta użytkownika pełniącego funkcję administratora, któremu przydzielamy wszystkie dostępne prawa. Staje się on ponadto właścicielem projektów, przy jednoczesnym pozbawieniu powyższych uprawnień poprzedniego *Administratora*.

Ponieważ konta *Administratora* nie można usunąć oraz zmienić nazwy (jest to użytkownik stały dla pliku grupy roboczej), chcąc ograniczyć jego przywileje w środowisku, należy pozbawić go wszystkich uprawnień. Pozwoli to na separację jego konta od wszelkich prac projektowych.

Włączenie systemu zabezpieczeń następuje z chwilą nadania hasła jednemu z użytkowników np. *Administrator*. Po powtórnym uruchomieniu aplikacji tworzymy nowego administratora z nową nazwą. W niniejszym opisie przyjęto konto *nAdmin*. Przydzielając

nowego administratora do grupy *Administratorzy* nadajemy mu prawa równorzędne poprzedniemu właścicielowi - którego chcemy w efekcie ograniczyć.

Ponowne uruchomienie aplikacji pozwala na aktywację (logowanie) konta *nAdmin*.

Odbieranie uprawnień staremu *Administratorowi* jest procesem kilkuetapowym:

- odebranie praw *Administratora* na poziomie użytkownika - pozwala na ograniczenie jego możliwości do elementów bazy danych - nowych i już istniejących.
- prawa dziedziczności wynikające z przynależności do grup są niezależne od praw nadawanych indywidualnie. Ponieważ *Administrator* jest członkiem grup *Administratorzy*, należy go z niej wykluczyć.

  Istotną uwagą jest fakt, że Access pozwala na istnienie kilku administratorów projektów, ale posiada również wewnętrzne zabezpieczenie chroniące go przed zupełną utratą kontroli przez ich wszystkich. Należy zwrócić uwagę na to, by przypadkiem podczas usuwania użytkowników z grupy *Administratorzy* przez pomyłkę nie usunąć *nAdmin*. Cofnięcie tego polecenia nie jest możliwe. Jeżeli jednak taka sytuacja miałaby miejsce, należy powtórnie aktywować konto *Administratora* i rozpocząć procedurę przyznawania praw *nAdmina* od początku.

  Oprócz grupy *Administratorzy* istnieje również grupa *Użytkownicy*. Posiada ona również wszystkie prawa dostępu do obiektów bazy danych. Grupy tej nie można na stałe usunąć, ponieważ jest ona trwałym elementem systemu zabezpieczeń Access'a. Wykluczenie z niej *Administratora* nie jest również możliwe, ponieważ jest to grupa, której członkami są wszyscy użytkownicy. Zasada dziedziczności praw grupy nadal pozwala *Administratorowi* posiadać uprawnienia do bazy. W takim przypadku ograniczenie praw grupie *Użytkownicy* spowoduje, że *Administrator* nie będzie posiadał już żadnych uprawnień.
- utworzenie nowej – unikatowej – grupy np. *Pracownicy* pozwoli na umieszczenie tam innych kont użytkowników przewidzianych w systemie bez konta *Administratora*.

2.3. Zmiana właściciela bazy danych

Zmiana właściciela obiektów oraz pliku bazy danych jest bardzo ważna. Prawo własności pozwala właścicielowi, niezależnie od przydzielonych praw dostępu i przynależności grupowej, uruchomić budowany projekt. Ma on również przywilej przydzielania praw do utworzonych przez siebie części aplikacji.

Przeniesienie praw własności z *Administratora* na *nAdmin* nie rozwiązuje do końca problemu. Opis ten dotyczy przypadku, gdy baza danych została wykonana wcześniej niż (unikatowe) środowisko systemu zabezpieczeń. Mimo iż właścicielem elementów bazy danych (formularze, kwerendy, raporty itp.) jest nowy administrator, to jednak wyłączne prawo do otwierania bazy nadal posiada *Administrator*. Oznacza to, że *Administrator* nadal będzie miał

możliwość nadawania uprawnień otwierania projektu innym użytkownikom. Stanowi on nadal zagrożenie dla budowanego projektu.

Problem ten rozwiązać można tworząc nowy plik bazy danych i importując wszystkie elementy starego projektu do nowego środowiska, gdzie użytkownikiem nadrzędnym jest *nAdmin*. Właścicielem nowego projektu będzie od tej pory nowy administrator, zaś całość bazy danych będzie odzwierciedleniem jej poprzedniej wersji.

W przypadku nowych projektów ich zarządcą jest *nAdmin*. To czyni go automatycznie właścicielem obiektów i pliku bazy danych.

3. Dziennik zdarzeń

Access nie posiada mechanizmów dziennika zdarzeń, pozwala jednak na ich utworzenie. Istotą dziennika jest jego odrębność od całości aplikacji, a jednocześnie kontrola nad zdarzeniami mającymi miejsce w czasie pracy. Czynnik odrębności zdefiniować można tu jako niezależność od innych elementów biorących udział w pracy bazy danych. Innymi słowy, jest to tabela rejestrująca zdarzenia zachodzące w aplikacji. Wszystkie ich rodzaje traktowane są jednakowo. Zdarzenia tworzą oddzielne zapisy w dzienniku, zawierając istotne z ich punktu widzenia fakty.

3.1. Tabela dziennika

Struktura tabeli dziennika zawierać musi odpowiednie pola pozwalające na jednoznaczne określenie rodzaju operacji wraz z jej parametrami. Opisanie czynności jest bardzo istotne z punktu widzenia śledzenia wykonywanych zdarzeń. Przyjęto następującą strukturę:

Tabela 1

Struktura tabeli dziennika zdarzeń

Nazwa Pola	Znaczenie	Typ danych
ID	Identyfikator wiersza w tabeli dziennika	przyrostowy
Identyfikator modyfikacji danych	Identyfikator rekordu modyfikowanego	liczba
Tabela	Nazwa tabeli modyfikowanej	tekst
Pole	Nazwa pola poddanego modyfikacji	tekst
Użytkownik	Nazwa osoby „zalogowanej" do aplikacji	tekst
Data	Data modyfikacji danych	data
Czas	Czas modyfikacji danych	czas
WartPrzed	Wartość pola przed zamianą danych	tekst
WartPo	Wartość pola po zmianie danych	tekst

Tabela jest przeznaczona tylko do rejestracji zdarzeń zachodzących podczas pracy z bazą danych. Są nimi operacje na rekordach, uruchamianie i zamykanie tabel, aktywacja kont użytkowników oraz identyfikacja włamania do bazy. Pozyskiwanie wpisów jest niezależne od użytkownika. Dziennik pracuje całkowicie w tle aplikacji. To czyni go niewidocznym i sprawia, że użytkownik nie jest świadomy jego istnienia. W skład rejestratora zdarzeń, oprócz opisywanej tabeli, wchodzi również moduł z deklaracją stałych i zmiennych globalnych, funkcją wpisu do dziennika oraz funkcją logowania. W pracy tej wykorzystano do budowy mechanizmu rejestratora postać funkcji, ponieważ mogą być one używane w makropoleceniach. Procedury nie dają takiej możliwości.

Ważne jest odpowiednie zabezpieczenie tabeli dziennika przed bezpośrednim dostępem do niej osób nieupoważnionych. Zaleca się ustawienie atrybutu tabeli ukrytej, przez co staje się ona niewidoczna w panelu sterowania bazą, nawet w przypadku wykrycia możliwości przejścia w tryb projektowania. Ponieważ przyrost ilości wierszy w tabeli rejestratora jest bardzo duży, zadaniem administratora będzie periodyczne przeglądanie i archiwizowanie dziennika.

3.2. Mechanizmy dziennika zdarzeń

Moduł zawierający funkcje dziennika zdarzeń udostępnia je dla całości aplikacji bazy danych. Stanowi główną część całego mechanizmu rejestratora. Scenariusz pracy wymusza jednak istnienie procedur w opisie zdarzeń aplikacji w celu odpowiedniej modyfikacji zmiennych globalnych stanowiących o wpisach do dziennika. W efekcie procedury obsługi zdarzeń wraz z elementami modułu tworzą całość mechanizmu rejestrującego zdarzenia w bazie danych.

3.2.1. Moduł

Moduł jest trzonem mechanizmu współpracującego z tabelą dziennika. Zawiera deklarację stałych i zmiennych globalnych oraz funkcję wpisu do dziennika i funkcję logowania. Zadaniem funkcji wpisu do dziennika jest uruchomienie dynamicznego rekordu pełniącego ramkę transportową dla danych opisujących zdarzenie w bazie oraz ich zapis do tabeli dziennika. W gromadzeniu odpowiednich parametrów dla funkcji *WpisDziennik* pomocne są procedury obsługi zdarzeń. Funkcja logowania ma za zadanie notowanie autoryzacji użytkowników oraz przypisanie zmiennej *Koniec* wartości zabezpieczającej niezgodne ze schematem opuszczenie aplikacji. Opis modułu znajduje się w dalszej części pracy (rozdział 6).

Elementy modułu związane z obsługą dziennika zdarzeń to:

1. Deklaracja zmiennych i stałych globalnych – zmienne i stałe obowiązujące w aplikacji, wykorzystywane przez funkcję wpisu do dziennika, jak również przez procedury obsługi zdarzeń.

2. Definicja funkcji wpisu do dziennika *WpisDziennik* – deklaracja funkcji pobierającej dane i zapisującej zdarzenia do dziennika oraz informującej o braku możliwości zapisu danych w przypadku pojawienia się błędu w pracy rejestratora.
3. Definicja funkcji logowania – deklaracja funkcji sprawdzającej autoryzację uruchomienia bazy danych. Funkcja ta ma za zadanie rozpoznanie, czy użytkownik poprawnie zalogował się w aplikacji oraz przypisanie zmiennej *Koniec* wartości zabezpieczającej niezgodne ze schematem wyjście z aplikacji

3.2.2. Procedury obsługi zdarzeń

Procedury obsługi zdarzeń są elementami pełniącymi rolę uzupełniającą dla funkcji znajdujących się w module dziennika. Dotyczą one formularzy (jako obiektów) oraz samych pól danych.

Podczas uruchamiania formularza następuje jednoczesne uruchomienie tabeli lub tabel wywołanych w formularzu. Zdarzenie takie jest wpisywane do rejestratora zdarzeń, ponieważ powoduje wgląd użytkownika w dane. Nie jest to przypadek ingerencji w dane tylko w tabelę. Z uwagi na to wywołanie funkcji wpisu do dziennika musi nastąpić na poziomie formularza. Zdarzenia, które są interesujące z punktu widzenia tego przypadku, to: *Przy otwarciu*, *Przy bieżącym* oraz *Przy zamknięciu*. Pierwsza i ostatnia akcja powoduje wpis o otwarciu i zamknięciu tabeli danych. Akcja *Przy bieżącym* ma za zadanie odpowiednie sterowanie numerem aktualnie modyfikowanego rekordu. Identyfikator ten jest przypisywany do zmiennej *NrRek* jako składowej funkcji *WpisDziennik*.

Oprócz rejestracji zdarzeń związanych z tabelą (jako obiektem), podczas pracy aplikacji występuje również etap wprowadzania i modyfikacji danych. Zdarzenia tego typu będą zajmować najwięcej miejsca w rejestratorze.

Obsługa aplikacji bazy danych w MS Access polega na pracy z interfejsem użytkownika opartym na formularzach. Ingerencja użytkownika bezpośrednio w tabele nie jest wskazana z uwagi na możliwość naruszenia spójności relacji danych. Formularze schematyzują całość pracy. Ich pośrednictwo pomiędzy użytkownikiem a źródłem rekordów stanowi pewien próg bezpieczeństwa. Schemat pracy, jaki tworzą, znacznie upraszcza procesy gromadzenia i przeglądania danych, jak również uprzyjemnia pracę z bazą.

Zaletą graficznego interfejsu użytkownika jest również to, że dla poszczególnych pól danych reprezentujących edytowany rekord można określić procedury obsługi zdarzeń. Procedury te współdziałając z modułem będą pozyskiwać parametry potrzebne do rejestracji zdarzenia. Rodzaj gromadzonych w dzienniku informacji, a ściślej które pola będą brały udział w rejestracji, określa projektant systemu. Obsługą wpisu do dziennika zajmują się dwie akcje: *Przy wejściu* oraz *Przy wyjściu*. Pierwsza, ma za zadanie wprowadzenie (poprzez przypisanie

do zmiennej globalnej *WartPrzed*) wartości pola przed modyfikacją, bez względu na to, czy pole jest edytowane czy nie. Druga akcja pełni funkcję warunkującą. Jej praca polega na:

1. Pobraniu wartości pola po modyfikacji i przypisaniu jej do zmiennej *WartPo*.
2. Sprawdzeniu, czy wartość pola nie jest wartością pustą (nie wprowadzona wartość danej) oraz czy różni się od wartości przed wpisem?
3. Jeśli spełniony jest p.2, następuje wpis do dziennika zdarzeń (modyfikacja pola).
4. Jeśli nie spełniony jest p.2, kończy procedurę (nie zaistniała modyfikacja lub pole pozostaje puste).

Sekwencja taka jest powtarzana przy każdym polu wchodzącym w skład procesu rejestracji zdarzeń.

3.2.3. Procedury obsługi zdarzeń aktywacji, dezaktywacji użytkownika w aplikacji oraz wykrycie nieautoryzowanego wejścia

Mechanizmy pracy Access'a pozwalają na opuszczenie aplikacji w dowolnym momencie. W przypadku istnienia dziennika zdarzeń jest to efekt niedopuszczalny. Aplikacja posiada system kontrolowanego uruchamiania środowiska pracy za pomocą makropolecenia *Autoexec*. Autoryzowanie użytkownika, a ściślej odnotowanie tego zdarzenia w dzienniku następuje po wykonaniu wspomnianego skryptu. Jako pierwsze wykonane jest odwołanie do funkcji *Logowanie*. Funkcja ta powoduje:

1. Przypisanie zmiennej *Koniec* wartości *False*.
2. Przypisanie zmiennej *Wlamanie* wartości *1*.
3. Wywołanie funkcji *WpisDziennik* z wartością *Logowanie*.

Wykonanie funkcji powoduje przypisanie w dzienniku zdarzenia autoryzacji użytkownika.

Następnie uruchamiany jest formularz główny aplikacji. Obiekt ten pełni rolę panelu sterowania całością scenariusza pracy z bazą danych. Podczas uruchamiania formularza głównego następuje sprawdzenie autentyczności użytkownika. Autoryzacja wejścia do bazy opiera się na sprawdzeniu, czy każdy z etapów skryptu przygotowania aplikacji został wykonany do końca. W akcji *przy otwarciu* panelu sterowania następuje odczytanie wartości zmiennej *Wlamanie*. Pominięcie podczas otwarcia bazy danych makropolecenia *Autoexec* (nie uruchomienie funkcji *Logowanie*) powoduje, że wartość zmiennej *Wlamanie* pozostaje bez zmian i wynosi nadal „0". W tym wypadku system przypisze w dzienniku zdarzenie *Włamanie*. W przeciwnym wypadku będzie to poprawne *Logowanie*.

Ograniczenie bezpośredniego opuszczenia aplikacji ze środowiska Access'a można zrealizować za pomocą procedury wyjścia. W założeniu projektu zwykle przewiduje się opuszczenie aplikacji za pomocą jednego elementu sterującego (np. przycisk itp.), który kończy pracę z bazą danych. Zapisuje on wszystkie zmiany w rekordach, wprowadzając do dziennika notkę o poprawnym opuszczeniu programu jednocześnie zamykając środowisko.

Procedura kontrolowanego zamknięcia systemu musi być przypisana wszystkim formularzom pojawiającym się na ekranie podczas pracy. Dzięki temu nie jest możliwe zakończenie aplikacji w dowolnym momencie. Aby uschematyzować proces opuszczenia programu, wprowadzono zmienną globalną *Koniec*. Po wykonaniu makropolecenia *Autoexec* przyjmuje ona wartość *False*. Każdy z formularzy w obsłudze zdarzenia *Przy bieżącym* musi weryfikować na bieżąco wartość tej zmiennej. Dopóki posiada ona wartość *False*, na wszelkie próby nieodpowiedniego zamknięcia aplikacji system reagować będzie komunikatem o braku możliwości opuszczenia programu. Zmiana wartości zmiennej *Koniec* realizowana jest tylko w przypadku przycisku kończącego pracę aplikacji. Ten zmienia wspomnianą wartość na *True*, co powoduje możliwość poprawnego jej opuszczenie.

4. Szyfrowanie bazy danych

Ostatnim z etapów ochrony bazy jest jej szyfrowanie. Czynność ta jest niezależna w stosunku do opisywanego systemu zabezpieczenia i ma na celu ochronę źródła danych przed możliwością przeglądania go za pomocą innych aplikacji. Przeglądanie takie umożliwiają mechanizmy ODBC. Czynność szyfrowania i deszyfrowanie przeprowadzana jest automatycznie w momencie pracy aplikacji. Zmniejsza ona efektywność pracy bazy danych spowalniając ją o około 10%.

Szyfrowania bazy danych może dokonać tylko jej właściciel, ponieważ czynność ta wymaga posiadania prawa modyfikacji projektu dla każdej z tabel bazy danych [3]. Plik bazy nie może być w tym czasie używany. Sam proces szyfrowania uruchamiany jest z platformy MS Access.

5. Kod modułu

```
'deklaracja stałych globalnych
Global Const L_ = "Logowanie"
Global Const W_ = "Wylogowanie"
Global Const I_ = "Włamanie"
Global Const O_ = "Otwarcie"
Global Const Z_ = "Zamknięcie"
Global Const M_ = "Modyfikacja"
'deklaracja zmiennych globalnych dla dynamicznego rekordu
Global Wlamanie As Integer
```

```
Global Koniec As Integer
Global NTab As String
Global NrRek As Long
Global Wynik As Variant
Global WartPrzed, WartPo As String

Function WpisDziennik(Ident As Long, Tabela, Pole, Przed, Po, Akcja As Variant)
On Error GoTo Err_Dziennik
    'zmienna warunkująca możliwość wpisu do dziennika (true) lub jego barak (false)
WpisDziennik = True
    'przypisanie Baza jako bazy danych oraz Rek jako dynamiczny rekord
Dim Baza As Database, Rek As Recordset
    'uruchomienie „silnika bazy danych"
Set Baza = DBEngine.Workspaces(0).Databases(0)
    'przypisanie Rek jako transportera danych do tabeli „Dziennik" w bazie danych
Set Rek = Baza.OpenRecordset("Dziennik", DB_OPEN_TABLE)
    'instrukcja warunkująca dopisanie nowego rekordu do dziennika
If Rek.RecordCount <> 0 Then Rek.MoveLast
Rek.AddNew
    'przypisanie wartości dla pól dynamicznego rekordu
Rek.[Identyfikator] = Ident
Rek.[Akcja] = Akcja
Rek.[Tabela] = Tabela
Rek.[Pole] = Pole
Rek.[Użytkownik] = CurrentUser()
Rek.[Data] = Date
Rek.[Czas] = Time
Rek.[WartPrzed] = Przed
Rek.[WartPo] = Po
    'odświeżenie dziennika
Rek.Update
    'zamknięcie dynamicznego rekordu
Rek.Close
    'zamknięcie tabeli dziennika
Exit_Dziennik:
Exit Function
```

```
    'cześć obsługi błędów wpisu do dziennika
Err_Dziennik:
MsgBox "Błąd zapisu w dzienniku zdarzeń.", 48, "Błąd zapisu"
    'blokada wpisu do dziennika
WpisDziennik = False
Resume Exit_Dziennik
End Function

    'funkcja logowania
Public Function logowanie()
    'przypisanie zmiennej „koniec" wartości (false) blokującej nieschematyczne opuszczenie
    bazy danych
Koniec = False
    'wartość 1 zmiennej „włamanie" informuje o poprawnym logowaniu
Wlamanie = 1
    'wywołanie funkcji „WpisDziennik" z parametrami zapisu logowania do dziennika
Wynik = WpisDziennik(0, " ", " ", " ", " ", L_)
End Function
```

6. Procedury zdarzeń dla obiektów bazy danych

- Przykładowa procedura *Przy wejściu* dla pola [Imię]

```
Private Sub Imię_Enter()
    'przekazanie wartości pola [imie] do dynamicznego rekordu
WartPrzed = [Imię]
End Sub
```

- Przykładowa procedura dla akcji Przy wyjściu dla pola [Imię]. Zapisanie do dziennika operacji Modyfikacja

```
Private Sub Imię_Exit(Cancel As Integer)
WartPo = [Imię]
    'instrukcja rozpoznająca, czy wartość pola [Imię] została zmieniona
If Not IsNull([Imię]) And (WartPrzed <> WartPo) Then
    'wywołanie funkcji „WpisDziennik" z parametrami zapisu modyfikacji pola
Wynik = WpisDziennik(NrRek, NTab, "Imię", WartPrzed, WartPo, M_)
End If
```

```
End Sub
```

- Przykładowa procedura zapisu operacji zamknięcia tabeli

```
Private Sub Form_Close()
    'wywołanie funkcji „WpisDziennik" z parametrami zapisu zamknięcia tabeli
Wynik = WpisDziennik(0, NTab, " ", " ", " ", Z_)
End Sub
```

- Przykładowa procedura nadania zmiennej NrRek wartości identyfikatora modyfikowanego rekordu

```
Private Sub Form_Current()
    'przechwycenie identyfikatora modyfikowanego rekordu z bieżącego formularza
NrRek = Forms![Dane].[ID]
End Sub
```

- Przykładowa procedura zapisu operacji wylogowania do dziennika

```
Private Sub Polecenie30_Click()
    'wołanie operacji zamknięcia formularza „Panelu głównego"
DoCmd.Close A_FORM, "Panel Główny"
    'wywołanie funkcji „WpisDziennik" z parametrami zapisu wylogowania użytkownika
Wynik = WpisDziennik(0, " ", " ", " ", " ", W_)
DoCmd.Quit
End Sub
```

- Przykładowa procedura sprawdzająca poprawność wykonania skryptu logowania

```
Private Sub Form_Open(Cancel As Integer)
    'przypisanie do zmiennej „Ntab" nazwy otwieranej tabeli
NTab = Forms![Dane].RecordSource
    'wyświetlenie komunikatu o włamaniu
MsgBox Wlamanie
If Wlamanie = 0 Then Wlamanie = 2
    'wywołanie funkcji „WpisDziennik" z parametrami wykrycia „intruza w aplikacji"
Wynik = WpisDziennik(0, " ", " ", " ", " ", I_)
End If
    'wywołanie funkcji „WpisDziennik" z parametrami otwarcia tabeli w celu śledzenia
    czynności wykonywanych przez nieautoryzowanego użytkownika
Wynik = WpisDziennik(0, NTab, " ", " ", " ", O_)
End Sub
```

LITERATURA

1. Dziennik Ustaw z dnia 30 czerwca 1998 r. Na podstawie art. 45 pkt. 1 ustawy z dnia 29 sierpnia 1997 r. o ochronie danych osobowych (DzU Nr 133, poz. 883).
2. Rozporządzenie Ministra Spraw Wewnętrznych i Administracji z dnia 3 czerwca 1998 r. w sprawie określenia podstawowych warunków technicznych i organizacyjnych, jakim powinny odpowiadać urządzenia i systemy informatyczne służące do przetwarzania danych osobowych. (DzU z dnia 30 czerwca 1998 r. Nr 133, poz. 883).
3. Paul Cassel, Craig Eddy: Access 97 – Baza danych dla każdego, Helion, Gliwice 1998.
4. Alan Simpson, Elizabeth Olson: Access dla Windows 95, Helion, Gliwice 1997.
5. Dariusz Boratyn: MS Access 2.0, Croma, Wrocław 1995.

Recenzent: Dr inż. Marcin Skowronek

Wpłynęło do Redakcji 12 kwietnia 2000 r.

Abstract

This paper describes protection of database of MS Access application. Presented method is a basic minimum required by law "Protection of personal data". The administration of workgroup is presented in stage 2, gives trustworthiness security of data base file.

Copied by unauthorised person, doesn't allow access to data. Events registry (stage 3) gives the ability of following the process, taking place in application. There is described in paper (stage 5, 6) sample code of events registry of the application.

Solution described in this paper is enough universal to use it on every of developer place.