VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA PODNIKATELSKÁ
FACULTY OF BUSINESS AND MANAGEMENT

ÚSTAV INFORMATIKY
INSTITUTE OF INFORMATICS

TECHNIKA SQL INJECTION - JEJÍ METODY A ZPŮSOBY OCHRANY
SQL INJECTION TECHNIQUE - ITS METHODS AND METHODS OF PROTECTION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR
Bc. Beáta Bahureková

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Jiří Kříž, Ph.D.

BRNO 2020

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA PODNIKATELSKÁ

Zadání diplomové práce

Ústav:	Ústav informatiky
Studentka:	**Bc. Beáta Bahureková**
Studijní program:	Systémové inženýrství a informatika
Studijní obor:	Informační management
Vedoucí práce:	**Ing. Jiří Kříž, Ph.D.**
Akademický rok:	2019/20

Ředitel ústavu Vám v souladu se zákonem č. 111/1998 Sb., o vysokých školách ve znění pozdějších předpisů a se Studijním a zkušebním řádem VUT v Brně zadává diplomovou práci s názvem:

Technika SQL injection – její metody a způsoby ochrany

Charakteristika problematiky úkolu:

Úvod
Cíle práce, metody a postupy zpracování
Teoretická východiska práce
Analýza současného stavu
Vlastní návrhy řešení
Závěr
Seznam použité literatury
Přílohy

Cíle, kterých má být dosaženo:

Cílem práce jsou metody SQL injection z teoretického a praktického hlediska a způsoby ochrany proti napadení databázové vrstvy programu.

Základní literární prameny:

ABLON, Lillian, Martin C. LIBICKI a Andrea A. GOLAY. Markets for cybercrime tools and stolen data: hackers' bazaar. Santa Monica, CA: RAND Corporation, 2014. ISBN 9780833087119.

CLARKE, Justin. SQL injection attacks and defense. Waltham, MA: Elsevier, 2012. ISBN 9781597499637.

CLARKE, Justin. SQL Injection Attacks and Defense [online]. Elsevier, 2012 [cit. 2020-02-29]. ISBN 9781597499637.

ONDRÁK, Viktor, Petr SEDLÁK a Vladimír MAZÁLEK. Problematika ISMS v manažerské informatice. Brno: Akademické nakladatelství CERM, 2013. ISBN 978-80-7204-872-4.

WHITMAN, Michael E. a Herbert J. MATTORD. Principles of information security. Australia: Cengage Learning, 2018. ISBN 978-1-337-10206-3.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2019/20

V Brně dne 29.2.2020

L. S.

doc. RNDr. Bedřich Půža, CSc. ředitel	doc. Ing. et Ing. Stanislav Škapa, Ph.D. děkan

ABSTRAKT

SQL injection je technika namierená proti webovým aplikáciám využívajúcim SQL databázu, ktorá môže predstavovať obrovské bezpečnostné riziko. Ide v nej o vloženie kódu do SQL databáze, pričom tento útok využíva zraniteľnosti v databázovej alebo aplikačnej vrstve. Hlavným cieľom mojej diplomovej práce je oboznámenie sa s podstatou SQL injection, pochopenie jednotlivých metód tejto útočnej techniky a ukázanie si spôsobov ako sa proti nej možno brániť. Prácu možno rozdeliť na tieto hlavné časti, ktoré rozoberiem nasledovne. V úvodnej časti práce zmieňujem teoretické východiska týkajúce sa SQL injection problematiky. Nasledujúca kapitola je zameraná na jednotlivé metódy tejto techniky. Analytická časť je venovaná zmapovaniu súčasného stavu testovacích subjektov, skenovacích nástrojov, ktoré tvoria základ pre optimálne skúmanie a testovanie jednotlivých SQLi metód, ktoré sú v tejto časti rozobraté z praktického hľadiska spolu s analýzou príkazov. V poslednej časti budem implementovať SQLi metódy na vybrané subjekty a na základe výstupov vytvorím univerzálne návrhové riešenie ako sa proti takýmto útokom brániť.

KĽÚČOVÉ SLOVÁ

SQL, Cyber Security, SQL injection, detection, Code Injection, cyber-attack, SQLi

ABSTRACT

SQL injection is a technique directed against web applications that use an SQL database that can pose a huge security risk. It involves inserting code into a SQL database, and this attack exploits vulnerabilities in the database or application layer. The main aim of my diploma thesis is to get acquainted with the essence of SQL injection, to understand the individual methods of this technique and to show ways how to defend against this technique. The work can be divided into these main parts, which I will discuss as follows. In the introductory part of the thesis I mention the theoretical basis of SQL injection. The following chapter is focused on individual methods of this technique. The analysis of the current state is analysed on the basis of outputs of various injection scanners, which will be compared with each other through their efficiency. This section will also focus on how to practically find a vulnerable site, how to attack. In the last part I will suggest solutions how to defend.

KEY WORDS:

SQL, Cyber Security, SQL injection, detection, Code Injection, cyber-attack, SQLi

BIBLIOGRAFICKÁ CITÁCIA

BAHUREKOVÁ, Beáta. Technika SQL injection - jej metódy a spôsoby ochrany [online]. Brno, 2020 [cit. 2020-05-17]. Dostupné z: https://www.vutbr.cz/studenti/zav-prace/detail/127646. Diplomová práce. Vysoké učení technické v Brne, Fakulta podnikateľská, Ústav informatiky. Vedúci práce Jiří Kříž.

ČESTNÉ PREHLÁSENIE

Prehlasujem, že predložená diplomová práca je pôvodná a spracovala som ju samostatne. Prehlasujem, že citácie použitých prameňov je úplná, že som vo svojej práci neporušila autorské práva (v zmyslu Zákona č. 121/2000 Zb., o práve autorskom a o právach súvisiacich s právom autorským).

V Brne dňa 9. mája 2020

.....

Bc.Beáta Bahureková

POĎAKOVANIE

Na tomto mieste by som sa chcela poďakovať svojmu vedúcemu pánovi Ing. Jiřímu Křížovi, Ph.D., za odborné a cenné rady pri vypracovávaní tejto práce. ktorý mi bol oporou pri písaní tejto práce a udeľoval mi cenné rady. Ďalej by som sa chcela poďakovať spoločnostiam a správcom ich webových aplikácii za sprostredkovanie možnosti testovania, slovenčinárke Mgr. Soni Slamkovej za usmerňovanie pri korektúre textu. V neposlednom rade by som sa chcela poďakovať pánovi Ing. Petrovi Sedlákovi za cenné rady ohľadom bezpečnosti počas štúdia, ktoré mi boli veľmi nápomocné pri vypracovaní tejto práce.

OBSAH

ÚVOD

V súčasnej dobe informačných technológií, kedy používanie webových aplikácií (bankovníctvo), emailov, sociálnych sietí a iných služieb sa stali novým bežným spôsobom komunikácie. Lenže táto doba prinášajúca vymoženosti informačných technológií nám ruku v ruke s výhodami, ktoré nám prináša, nás súčasne aj s našimi osobnými dátami vystavuje nebezpečenstvu. Je kľúčové v dnešnej dobe si nielen zvyšovať povedomie o bezpečnosti, ale súčasne, či už programujeme/spravujeme alebo sprostredkovávame stránky pre druhých, tak mať na pamäti, že ak podceníme bezpečnosť, tak hazardujeme s osobnými údajmi/ dátami iných ľudí a podľa toho by sme k tomu mali pristupovať. Vyššie spomínané webové aplikácie a ich dáta, ku ktorým majú prístup sú väčšinou cieľom rôznych útokov. Jedným z najčastejších útokov sú práve SQLi útoky, ktoré sú realizovateľné vďaka bezpečnostných nedostatkoch v aplikáciách. Pokiaľ má stránka základ v SQL databáze, tak môže byť napadnutá prostredníctvom SQL injection útoku. Takýto typ útoku je schopný spôsobiť obrovské škody, či už z pohľadu krádeže osobných dát alebo škôd, ktoré môžu byť na databáze napáchané.

V prvej časti tejto práce sa venujem teoretickým poznatkom z oblasti SQLi a SQL v spojitosti s bezpečnosťou. Nasledujúca časť práce je venovaná analýze jednotlivých SQLi metód s praktickými príkladmi, krokov pri hackingu webovej aplikácie, bezpečnosti dát, analýze rizík a zhrnutie najčastejšie sa vyskytovaných metód SQLi. Návrhová časť práce je venovaná testovaniu vybraných SQLi metód na dvoch testovacích webových stránkach a dvoch, ktoré patria organizáciám, návrhu bezpečnostných odporučení na obranu proti SQLi metódam, ktoré obsahujú aj návrh opatrení mierených priamo na správcu. Práca je zakončená hodnotením výstupov testovania, návrhu bezpečnostných opatrení a celkového prínosu práce.

CIELE PRÁCE, METÓDY A POSTUPY SPRACOVANIA

Cieľom práce sú metódy SQL injection z teoretického a praktického hľadiska a spôsoby ochrany proti napadnutiu databázovej vrstvy programu.

Pri tvorbe tejto diplomovej práce bolo využitých niekoľko metód získavania informácii potrebných k implementácii SQLi metód. Pri získavaní potrebných informácií o testovacom Subjekte A boli čerpané z testovacích webových stránok na kt. bola väčšina potrebných informácií rozpísaná. Počas získavania informácií o Subjekte B som priamo kládla otázky správcovi webovej aplikácie, pri Subjekte C tieto informácie poskytla detailná dokumentácia, kt. je uložená na stránkach github. O Subjekte D boli všetky potrebné informácie v dokumentácii aplikácie na stránkach, kde bola stiahnutá.

Ďalšou z metód na získavanie informácií bolo priame pozorovanie jednotlivých aplikácií a procesu prihlasovania sa do nich, a reakcie serveru skrz jednotlivé spôsoby prihlasovania sa.

Jednou z metód na získavanie potrebných informácií bol Google Hacking, čo je technika používaná na zbieranie informácií, ktorú používa útočník pričom využíva pokročilé techniky vyhľadávania Google. Vyhľadávacie dotazy spoločnosti Google pri hackovaní sa dajú použiť na identifikáciu slabých miest v zabezpečení webových aplikáciách. Táto metóda bola v práci konkrétne používaná na zhromažďovanie týchto typov údajov: chybových správ odhaľujúcich citlivé informácie, informácie o individuálnych cieľoch a ďalšie citlivé údaje.

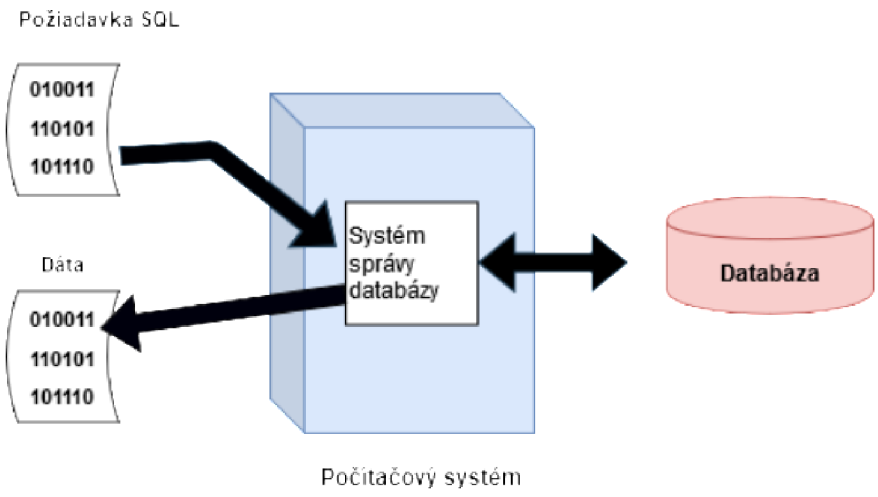
V neposlednom rade pri zbieraní informácií zohrala obrovskú úlohu nástroj na skenovanie webovej aplikácie.

1 TEORETICKÁ VÝCHODISKA PRÁCE

Prvá časť diplomovej práce je venovaná teoretickej časti zameranej na oboznámenie sa s SQL jazykom, architektúrou webu a kybernetickou bezpečnosťou. V tejto práci je teoretická časť kľúčová pre pochopenie komplexnosti problematiky, ktorá je spracovaná v praktickej časti.

1.1 SQL a základné pojmy týkajúce sa tohoto jazyka

Relačné databázové systémy a jazyk SQL, na ktorého základe sú postavené, predstavujú jednu zo základných technológií počítačového priemyslu. Jazyk SQL je nástroj organizovania, správy a získavania dát uložených v počítačovej databázy. Skratka SQL v preklade znamená štruktúrovaný dotazovací jazyk. SQL pracuje s jedným špecifickým typom databáze, ktorý je nazývaný relačná databáza (8).



Obr. 1: Použitie jazyka SQL pre prístup k databáze (vlastné).

Popis obrázka: Na obrázku vidíme spôsob akým pracujeme s jazykom SQL. Počítačový systém obsahuje databázu v kt. sú uložené dôležité informácie. Počítačový program, kt. riadi databázu sa nazýva databázový riadiaci systém (DBRS). Keď potrebujeme z databázy získať dáta, tak napíšeme požiadavku v jazyku SQL. Následne túto

požiadavku databázový systém spracuje, načíta dáta a vráti odpovedajúci informáciu. Celý tento proces, kedy požadujeme dáta z databázy a systém nám vráti výsledok, sa nazýva dotaz (query) (8).

Definícia dát – SQL dáva možnosť užívateľovi definovať štruktúru a organizáciu uložených dát spolu s definíciou ich vzájomných vzťahov (8).

Získavanie dát - ďalej SQL umožňuje užívateľovi či aplikačnému programu z databáze uložené dáta získavať a používať (8).

Riadenie prístupu – SQL je možné použiť obmedzeniu schopností užívateľa dáta čítať, pridávať a modifikovať, čím ich je možné chrániť pred neautorizovaným prístupom (8).

Manipulácia s dátami – SQL umožňuje užívateľovi alebo aplikačnému programu databázu aktualizovať pridávaním nových dát, odstraňovaním starých alebo zmenou už predtým uložených dát(8).

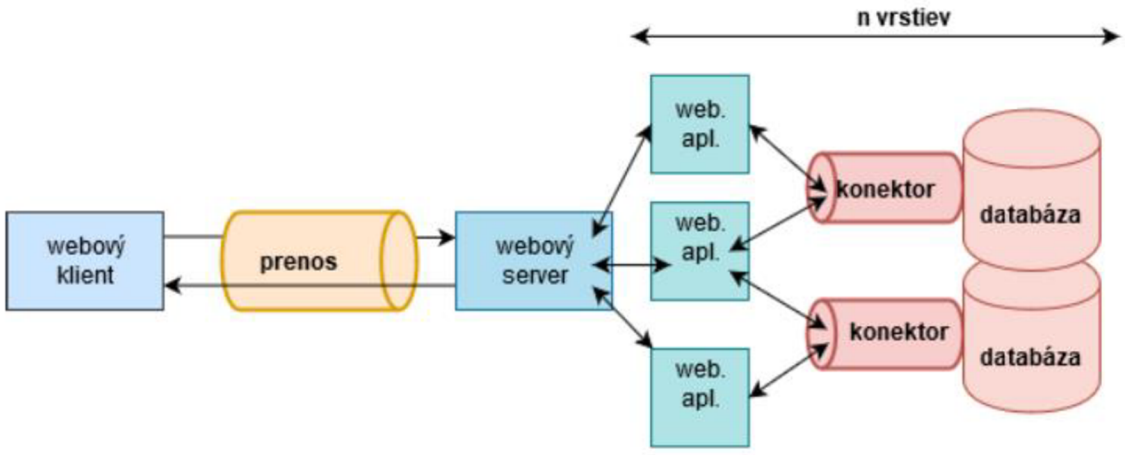
Zdieľanie dát – SQL sa používa k zladení zdieľania dát medzi viac užívateľov a k zaistením toho, že sa užívatelia medzi sebou navzájom nerušia(8).

Integrita dát – SQL definuje v databázy obmedzenie integrity, čím ju chráni pred porušením, kt. môže vzniknúť vďaka neúplným aktualizáciám alebo systémovému zlyhaniu (8).

SQL je z vyššie zmienených dôvodov braný ako komplexný jazyk pre riadenie a ovplyvňovanie databáze. Samostatný jazyk SQL nieje riadiaci systém databázy a ani sa tu nejedná o samostatný produkt. SQL je skôr integrovanou súčasťou databázového systému, ako jazyk a nástroj pre komunikáciu s ním (8).

1.2 Architektúra webových aplikácií

Architektúra webových aplikácií je najviac podobná centralizovanému výpočtovému modelu s veľkým množstvom distribuovaných klientov. Spomínaní klienti majú za úlohu vykonávať viac než len prezentáciu dát a pripojujú sa k centrálnemu serveru, na ktorom dochádza k podstatnej časti spracovania týchto dát. Hlavným rozdielom medzi webovou architektúrou a centralizovaným výpočtovým modelom je to, že webová architektúra sa môže skladať z niekoľkých serverov a niekoľkých databáz, oproti centralizovanému výpočtovému modelu, kde je len jeden master server (7).



Obr. 2: Architektúra webových aplikácií (vlastné)

U typických webových aplikácií beží časť kódu na strane užívateľa vo webovom klientovi (CSS, HTML, ...), na strane servera beží užívateľovi neprístupný kód reagujúce na HTTP požiadavky . Webový server ďalej komunikuje s webovými aplikáciami pomocou ďalších protokolov, nielen HTTP. Webové servery zobrazujú obsah stránok, oproti tomu webové aplikácie majú na starosti logiku a interakciu medzi užívateľom a stránkami. Webové aplikácie potom priamo komunikujú s ovládačmi databázy a vznáša požiadavky na dáta, ktoré spätne spracujú a odošlú webovému serveru, aby ukázal koncovému užívateľovi (29, 30).

1.3 Bezpečnosť týkajúca sa webových stránok a aplikácií

V dnešnej dobe internetu sú medzi sebou prakticky prepojené všetky počítače, či už sú to stolné počítače, servery, mobilné telefóny a rôzne iné zariadenia rozmanitých prevedení. Pre spoločnosti, podnikateľov a vývojárov softwaru tým vzniká obrovské spektrum príležitostí, ktoré vzhľadom na prepojenie systémov so sebou prináša aj hrozbu útoku.

Hlavným problémom je to, že niekedy vývojári aplikácie nepočítajú s tým, že by ich aplikácia mohla byť v prevádzke v silne prepojenom prostredí, kde by mohla byť dostupná aj pre útočníkov. Keďže je internet značne nepriateľské prostredie, tak by sme mali akýkoľvek kód navrhovať takým spôsobom, aby bol schopný odolať útokom.

Bezpečný produkt- *„predstavuje produkt, ktorý chráni dôvernosť, integritu a dostupnosť informácií zákazníka a súčasne dostupnosť a integritu zdrojov pre spracovanie, ktoré sú pod kontrolou vlastníka alebo administrátora webu“* (4).

Zraniteľné miesto v bezpečnosti- je typ chyby v produkte, kvôli ktorej užívateľ nevie ani pri správnom užívaní produktu zabrániť útočníkovi v získavaní vyšších oprávneniach v systéme užívateľa, napadnutiu jeho dát, v zásahu do jeho činností (4).

V tejto tematike platí pravidlo : bezpečný systém = kvalitný systém. Platí, že pokiaľ kód od samotného základu navrhujeme a zostavujeme s ohľadom na bezpečnosť, ktorá je súčasťou jeho základných vlastností, je mohutnejší, než kód do kt. implementujeme bezpečnosť až nakoniec vo forme doplnkových funkcií (4).

Pri programovaní aplikácie je dôležité nikdy nepočítať s tým, že daná aplikácia bude v prevádzke vždy iba v jednom z niekoľkých predom známych prostredí. Z tohto dôvodu je dôležité pri písaní kódu konať tak akoby aplikácia mala bežať v maximálne nepriateľskom prostredí a na základe toho ju navrhnúť, naprogramovať a adekvátne otestovať.

1.3.1 Zákon o kybernetickej bezpečnosti

V priebehu roku 2014 bol prijatý zákon o kybernetickej bezpečnosti č. 181/2014 Sb., kt. nadobudol účinnosť 1.1.2015 a kt. upravuje práva a povinnosti osôb, pôsobenie a právomoci orgánov verejnej moci v oblasti kybernetickej bezpečnosti. Cieľom zákona

je stanoviť podmienky spolupráce medzi súkromím sektorom a verejnou správou za účelom riešenia kybernetických bezpečnostných incidentov (5,6).

Zákon o kybernetickej bezpečnosti - je špecificky zameraný na ochranu funkčnosti sieťového prostredia umožňujúceho vznik, spracovanie, uchovávanie a komunikáciu informácií, kt. je tvorené informačnými systémami, službami a sieťami elektronických komunikácií. (5,6).

Kybernetický priestor- podľa §2 písm. a)"*sa kybernetickým priestorom rozumie digitálne prostredie umožňujúce vznik, spracovanie a výmenu informácií, tvorené informačnými systémami, a službami a sieťami elektronických komunikácií.(5)*" Podľa DZ bol kybernetický priestor definovaný ako informačné prostredie k realizácii informačných transakcií, kt. je vytvorené technológiami, ktorých definície a podmienky užívania upravujú zvláštne zákony, patria sem: Informačné systémy, službami a sieťami elektronických informácií.

Kyberpriestor je chápaný ako nehmotný svet informácií, ktorý vzniká vzájomným prepojením informačných a komunikačných systémov. Toto prostredie nám umožňuje vytvárať, uchovávať, využívať a vzájomne si vymieňať informácie a je realizované prostredníctvom počítačov prepojených komunikačnými sieťami v celosvetovom meradle (5).

Poverenec pre ochranu osobných údajov – (data protection officer) monitoruje súlad spracovanie osobných údajov s povinnosťami vyplývajúcimi z nariadenia, auditov, školenia a celkové riadenie agendy internej ochrany dát.

Pseudonymizácia - proces skrytie identity. Účelom je zbieranie ďalších údajov, ktoré sa týkajú rovnakého jednotlivca, aby bolo nutné poznať jeho totožnosť (napr. Kódovanie pomocou kľúča).

Súhlas so spracovaním osobných údajov - v prípade, že je spracovanie založené na súhlase, je potrebné, aby bol správca schopný doložiť, že fyzická osoba udelila súhlas so spracovaním údajov o svojej osobe slobodne a súhlas bol konkrétny, informovaný, jednoznačný a ničím nepodmienečný.

Správca osobných údajov - určuje účel a prostriedky pre spracovanie osobných údajov. Primárne zodpovedá za spracovanie OÚ.

Subjekt údajov - fyzická osoba, ku ktorej sa vzťahujú osobné údaje. Nejedná sa o

právnickú osobu.

Spracovanie - ľubovoľný úkon, ktorý správca (alebo spracovateľ) systematický vykonáva s osobnými údajmi.

1.3.2 Hacker

V dnešnej dobe sú pojmy „hacker“ či „hacking“ alebo „hacker“ brané ako negatívne výrazy pod kt. si verejnosť predstavuje nezákonnú činnosť páchanú v kyberpriestore. Pojem hacker sa v jednotlivých oblastiach vysvetľuje rozdielne avšak v rámci práce je podstatný tento spôsob vysvetlenia. Hacker je výraz pre počítačového programátora manipulujúceho s technológiami (24). Aktuálne sa hackeri delia na tieto štyri skupiny:

- Black hat (čierny klobúk) - sem patria všetci hackeri, kt. sa podieľajú na nezákonných činnostiach s cieľom obohatiť sa, uškodiť.
- White hat (biely klobúk) - patria sem etický hackeri, kt. využívajú svoje schopnosti tak aby neboli v rozpore s etickými zásadami. Väčšinou sa zabývajú testovaním bezpečnostných systémov (penetračné testovanie).
- Grey hat (sivý klobúk) - je to typ hackera, kt. je aj z etického hľadiska niekde medzi čiernym a bielym hackerom. Patria sem napríklad ľudia, kt. nabúrajú systém bez povolenia, ale dáta nijak nezneužijú, namiesto toho o tom informujú administrátora, aby vedel o bezpečnostných nedostatkoch systému (24).
- Blue hat (modrý klobúk) - je niekto mimo poradenských firiem v oblasti počítačovej bezpečnosti, ktorý pred spustením testuje systém a hľadá zneužitia, aby ich bolo možné uzavrieť. Blue hat hacker sa tiež odvoláva na profesionálov v oblasti bezpečnosti pozvaných spoločnosťou Microsoft, aby našli zraniteľné miesta v systéme Windows. Tento výraz je tiež spájaný s výročnou konferenciou o bezpečnosti spoločnosti Microsoft, ktorej neoficiálne meno pochádza z modrej farby spojenej s odznakmi zamestnancov spoločnosti Microsoft (23).

1.3.3 ePrivacy

Ide o nariadenie o súkromí a elektronických komunikáciách. Tento návrh súvisí s GDPR a rozširuje okruh subjektov a služieb, na ktoré sa vzťahuje ochrana osobných údajov. Pri elektronickej komunikácii musí byť zabezpečené, že ako obsah súkromných správ medzi dvoma subjektmi, tak údaje o mieste a čase odoslania a subjektoch nesmú zverejniť nikomu, okrem zúčastnených strán. Dôvodom podania návrhu je fakt, že doteraz služby ako VoIP, instant messaging a e-mailové služby nepodliehali úniovému rámci pre elektronické komunikácie. Po schválení tohto návrhu spadajú prevádzkovatelia OTT (Over-the-Top) pod právo na rešpektovanie súkromia a komunikácie (28).

Interpersonálne komunikačné služba - ide o službu, ktorá "umožňuje priamu interpersonálne a interaktívnu výmenu informácií medzi konečným počtom osôb" (29).

"Klasické" interpersonálne komunikačné služby - ide hlavne o služby určené na telekomunikáciu, ako Skype, WhatsApp, Messenger, ... (28).

"Nové" interpersonálne komunikačné služby - návrh rozširuje pôsobnosť ePrivacy na ďalšie služby, ako napr .: sociálne siete (obsahujú chat), online hry (obsahujúci chat), aplikácie a programy umožňujúce komunikáciu (TeamViewer) a časti IoT umožňujúci komunikáciu s koncovým užívateľom (Google Glass, Siri, ...) (28).

Spracovanie dát - podľa čl. 6 ePrivacy poskytovatelia služieb môžu "spracovávať dáta el. Komunikácií v prípade, že to je nevyhnutné pre prenos komunikácie po dobu nutnú na tento účel, alebo je to nevyhnutné na zachovanie a obnovu bezpečnostných služieb a sietí " (28).

Spracovanie metadát - poskytovatelia môžu spracovávať metadáta, ak *"je to nevyhnutné pre splnenie povinných požiadaviek na kvalitu služby [...], po dobu nutnú na tento účel, alebo je to nevyhnutné na vyúčtovanie, výpočet platieb za prepojenie, odhalenie podvodného užívania alebo zneužívania služieb elektronických komunikácií, zamedzenie takému podvodné užívanie alebo zneužívanie alebo prihlásiť sa k užívaniu týchto služieb, alebo príslušný koncový používateľ udelil svoj súhlas so spracovaním metadát svojich komunikácií pre jeden alebo viac konkrétnych účelov, vrátane poskytovania konkrétnych služieb takýmto koncovým užívateľom, za predpokladu, že tento účel alebo účely nemožno splniť spracovaním anonymizovaných informácií "*(28).

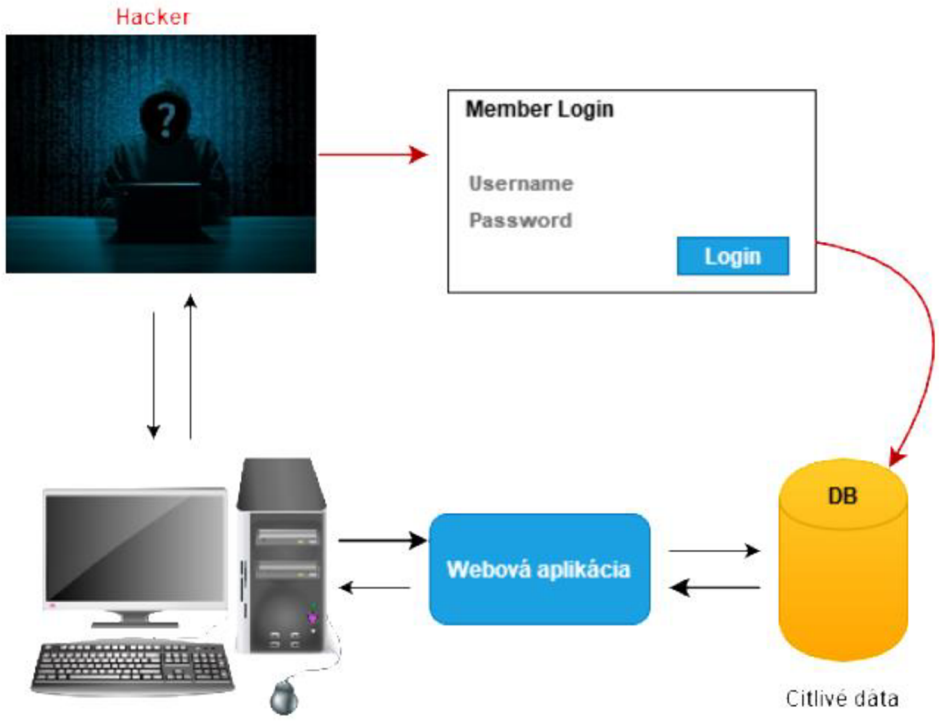
Spracovanie obsahu - poskytovateľ môže spracovávať obsah komunikácie *"za účelom poskytovania konkrétnej služby koncovému užívateľovi, ak príslušný koncový používateľ alebo koncoví užívatelia udelili svoj súhlas so spracovaním svojho obsahu elektronických komunikácií a danú službu nemožno bez spracovanie tohto obsahu poskytnúť, alebo ak všetci dotknutí koncoví užívatelia udelili svoj súhlas so spracovaním svojho obsahu elektronických komunikácií pre jeden alebo viac konkrétnych účelov, ktoré nemožno splniť spracovaním anonymizovaných informácií, a poskytovateľ konzultoval dozorný úrad. Pre konzultácii dozorného úradu sa uplatňuje článok. 36 ods. 2 a 3 nariadenia (EÚ) 2016/679 "(28)*

1.4 SQL Injection

Tento druh útoku je pokladaný za jednu z najzničujúcejších slabín, ktorá môže mať obrovský dopad na biznis, pretože môže viesť k odhaleniu všetkých citlivých informácií uložených v databáze. Informácií ako sú napríklad: prihlasovacie údaje, heslá, mená, adresy, telefónne čísla, podrobné informácie o kreditných kartách (1).

SQL Injection je chyba, ktorá spôsobí to, že ak útočníkovi poskytnete schopnosť ovplyvňovať dotazy na jazyk SQL, tak aplikácia prejde do databázy typu back-end. Keďže sa útočníkovi ponúka možnosť ovplyvniť to, čo sa odovzdáva do databázy, tak vie využiť syntax a schopnosti samotného SQL. Okrem toho môže využiť funkcie OS, ktoré má databáza k dispozícii (1).

SQL Injection je útok, ktorý sa pokúša získať neoprávnený prístup k databáze vložením kódu a využitím dotazu SQL. Pre lepšie pochopenie použijeme nasledujúci príklad. Povedzme, že existuje webová stránka banky, ktorá umožňuje používateľom prihlásiť sa pomocou používateľského mena a hesla. Keď používateľ zadá platné používateľské meno a heslo, overenie prebehne a bude sa môcť prihlásiť(3).



Obr. 3: Ukážka SQL injection (vlastné)

Nasledujúci dotaz je vytvorený na ukážku neoprávneného pokusu o prihlásenie za pomoci klauzule where:

Username = bba

Password = bba987

SQL dotaz:

```
SELECT*FROM users WHERE name = 'bba' and password = 'bba987'
```

Obr. 4: SQL dotaz (vlastné)

Užívateľ so zlým úmyslom by mohol zadať nasledujúci vstup do polí užívateľského mena a hesla na akomkoľvek webe.

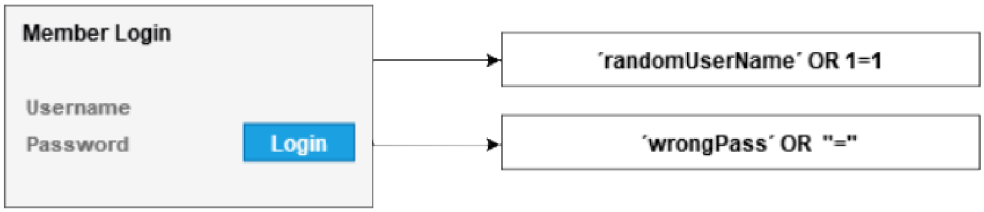
Username = bba

Password = ’or ‘1’ = ‘1‘

SQL dotaz by v tomto prípade vypadal nasledovne:

```
SELECT* FROM users WHERE name = ‘bba’ and password = ‘’or ‘1’ = ‘1’
```

Obr. 5: SQL dotaz where (vlastné)



Obr. 6: Prihlasovanie užívateľa (vlastné)

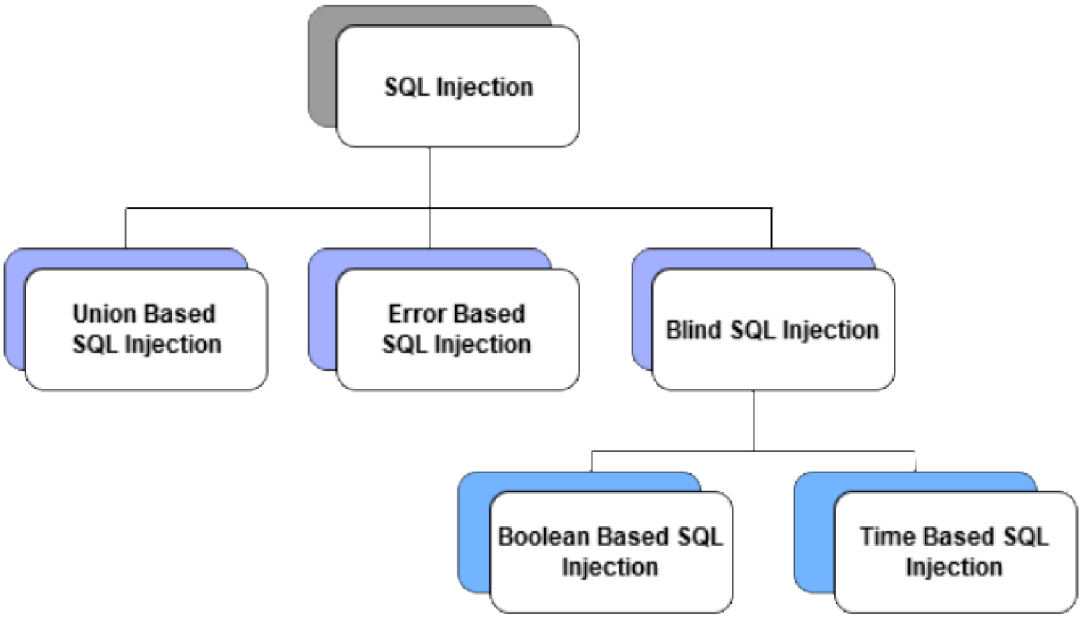
Kým 1=1 bude pravdivé, tak tomuto užívateľovi bude vždy umožnené prihlásiť sa na webovú stránku. Za predpokladu vyššie zmieneného získa používateľ(útočník) neoprávnený prístup k podrobnostiam účtu iného používateľa, čo by mohla smerovať k vážnym následkom za krádež informácií o účte, ktorá spadá pod porušenie ochrany osobných údajov. Vyššie zmienený SQL injection útok, slúžil len ako príklad na lepšie pochopenie, väčšina webových stránok by v dnešnej dobe by takémuto útoku ľahko zabránila.

1.5 Metódy SQL injection

SQLi je možno realizovať vďaka tomu, že užívateľský vstup je vložený do existujúceho dotazu SQL bez správnej validácie.

SQL Injection útoky je možno rozdeliť do týchto hlavných troch kategórií:

1. Union Based SQL Injection
2. Error Based SQL Injection
3. Blind SQLi útok - SQLi založená na boolean
   - SQLi založená na čase



Obr. 7: Rozdelenie metód SQL injection (vlastné)

1.5.1 Union SQLi

Útoky založené na UNION umožňujú testerovi ľahko extrahovať informácie z databázy. Pretože operátor UNION je možné použiť iba v prípade, že obidva dotazy majú presne rovnakú štruktúru, musí útočník vytvoriť príkaz SELECT podobný pôvodnému dotazu. Na tento účel musí byť známy platný názov tabuľky, ale je tiež potrebné určiť počet stĺpcov v prvom dotaze a ich typ údajov (20).

Kľúčové slovo UNION umožňuje vykonať jeden alebo viac ďalších SELECT dotazov a pripojiť výsledky k pôvodnému dotazu. Napríklad:

```
SELECT a, b FROM table1 UNION SELECT c, d FROM table2
```

Obr. 8: Ukážka na Union SQLi (20).

Tento dotaz SQL vráti jediný súbor výsledkov s dvoma stĺpcami, ktorý obsahuje hodnoty zo stĺpcov „a“ a „b“ in table1 a „stĺpec c“ a „d„ v table2. Aby dotaz UNION fungoval, musia byť splnené dve kľúčové požiadavky:

- Jednotlivé dotazy musia vrátiť rovnaký počet stĺpcov. Typy údajov v každom stĺpci musia byť kompatibilné medzi jednotlivými dotazmi.
- Ak chcete vykonať útok SQL UNION, musíte sa uistiť, že váš útok spĺňa tieto dve požiadavky. To zvyčajne zahŕňa zisťovanie:
    - Koľko stĺpcov sa vracia z pôvodného dotazu?
    - Ktoré stĺpce vrátené z pôvodného dotazu sú vhodným typom údajov na uchovávanie výsledkov z injektovaného dotazu?

Pri vykonávaní útoku SQLi UNION existujú dve účinné metódy na určenie toho, koľko stĺpcov sa vracia z pôvodného dotazu (20).

Prvá metóda zahŕňa vstrekovanie série klauzúl ORDER BY a zvýšenie indexu zadaného stĺpca, kým nedôjde k chybe. Napríklad za predpokladu, že bod vstrekovania je citovaný reťazec v rámci klauzule WHERE pôvodného dotazu, odošlite:

```
' ORDER BY 1--
' ORDER BY 2--
' ORDER BY 3--
```

Obr. 9: Order by (20).

Táto séria užitočných dát modifikuje pôvodný dotaz tak, aby výsledky usporiadal do rôznych stĺpcov v sade výsledkov. Stĺpec v klauzule ORDER BY môže byť určený indexom, takže nemusíte poznať názvy žiadnych stĺpcov. Keď zadaný index stĺpcov prekročí počet skutočných stĺpcov v množine výsledkov, databáza vráti chybu, napríklad: ORDER BY pozícia čísla 3 je mimo rozsahu počtu položiek vo výberovom zozname (20).

Druhá metóda spočíva v predložení série užitočných zaťažení UNION SELECT špecifikujúcich iný počet nulových hodnôt.

```
' UNION SELECT NULL--
' UNION SELECT NULL,NULL--
' UNION SELECT NULL,NULL,NULL--
```

Obr. 10: UNION SELECT (20).

Pokiaľ počet null nezodpovedá počtu stĺpcov, databáza vráti chybu, kt. môže vypadať nasledovne: „*Všetky dotazy kombinované pomocou operátora UNION, INTERSECT alebo EXCEPT musia mať vo svojich cieľových zoznamoch rovnaký počet výrazov*“(20).

Aplikácia môže opäť vrátiť túto chybovú správu alebo iba vrátiť všeobecnú chybu alebo žiadne výsledky. Keď sa počet nulových hodnôt zhoduje s počtom stĺpcov, databáza vráti v riadku výsledkov ďalší riadok obsahujúci hodnoty null v každom stĺpci. Účinok na výslednú odozvu HTTP závisí od kódu aplikácie. Ak budete mať šťastie, v odpovedi uvidíte nejaký ďalší obsah, napríklad ďalší riadok v tabuľke HTML. V najhoršom prípade by odpoveď mohla byť nerozoznateľná od reakcie, ktorá je spôsobená nesprávnym počtom null, čo spôsobí, že táto metóda určenia počtu stĺpcov bude neúčinná (20).

1.5.2 SQLi založené na chybe

Technika založená na chybe (Error Based SQLi) je užitočná, keď tester nemôže zneužiť zraniteľnosť SQL injekcie pomocou inej techniky, napríklad UNION. Technika založená na chybe spočíva v nútení databázy vykonať nejakú operáciu, pri ktorej výsledkom bude chyba. Potom sa pokúste extrahovať niektoré údaje z databázy a zobraziť ich v chybovom hlásení. Napríklad ak by bola možnosť nájsť názov tabuľky a zároveň by to bolo súčasťou cieľu útoku, tak najlepším spôsobom ako nájsť tieto informácie by bolo prostredníctvom systémových tabuliek. Je pravdou, že DBMS majú odlišnú formu pomenovávania, avšak tých populárnejších je len pár, vďaka čomu by teoreticky nemalo byť tak zložité nájsť systémové tabuľky. Je tu možné použiť minimálny príkaz prostredníctvom SELECT, skrz čo nie je potrebné špecifikovať názvy stĺpcov (19).

Užívateľský vstup (MySQL system table).

1 UNION SELECT 1 FROM information_schema.tables

Vygenerovaný dotaz

SELECT name, description, price FROM products WHERE category=1 UNION
SELECT 1 FROM information_schema.tables

Vrátená chyba

ORA-00942: tabuľka alebo pohľad neexistuje.

Obr. 11: Ukážka Error Based SQLi (19).

1.5.3 Slepá SQLi

Blind SQLi (SQLi na slepo) je typ útoku SQL Injection, ktorý kladie otázky a určuje odpoveď na základe odpovedí aplikácií (pravda/nepravda). Tento útok sa často používa, keď je webová aplikácia nakonfigurovaná tak, aby zobrazovala všeobecné chybové správy, ale nezmiernila kód, ktorý je náchylný na injekciu SQL. Blind SQL injection sa rozdeľuje na Boolean based SQLi (založenú na boolean) a Time based SQLi (založená na časovej odozve) (19).

1. Boolean SQLi - Útočník môže overiť, či sa odoslaná odpoveď vrátila pravdivá alebo nepravdivá niekoľkými spôsobmi, čo bude ukázané na nasledujúcich príkladoch. Pomocou jednoduchej stránky, ktorá zobrazuje ako parameter článok s daným ID, môže útočník vykonať niekoľko jednoduchých testov, aby zistil, či je stránka zraniteľná voči útokom SQL Injection.
    - Máme napríklad túto URL: http://bahurekova.com/items.php?id=2.
    - Databáze bude odoslaný takýto dotaz :

      SELECT title, description, body FROM items WHERE ID = 2

    - Útočník skúsi vstreknúť tento dotaz, čo vráti odpoveď „false“, bude to vypadať takto : http://bahurekova.com/items.php?id=2 and 1=2
    - Vďaka tomu by dotaz vypadal nasledovne

      SELECT title, description, body FROM items WHERE ID = 2 and 1=2

    - Ak je webová aplikácia zraniteľná voči SQL Injection, pravdepodobne už nič nevráti. Útočník vloží dotaz, ktorý vráti hodnotu „true“:
    - http://bahurekova.com/items.php?id=2 and 1=1
    - Po jeho overení sú jediným obmedzením oprávnenia nastavené správcom databázy, iná syntax SQL a predstavivosť útočníka (18).
2. SQLi založená na časovej odozve- tento typ slepej SQLi sa spolieha na pozastavenie databázy na zadanú dobu, potom vrátenie výsledkov, čo naznačuje úspešné vykonanie dotazu SQL. Pomocou tejto metódy útočník vymenuje každé písmeno požadovanej časti údajov pomocou nasledujúcej logiky, čo by mohlo vypadať nasledovne:
   Ak je prvé písmeno názvu prvej databázy „A“, počkajte 10 sekúnd (18).

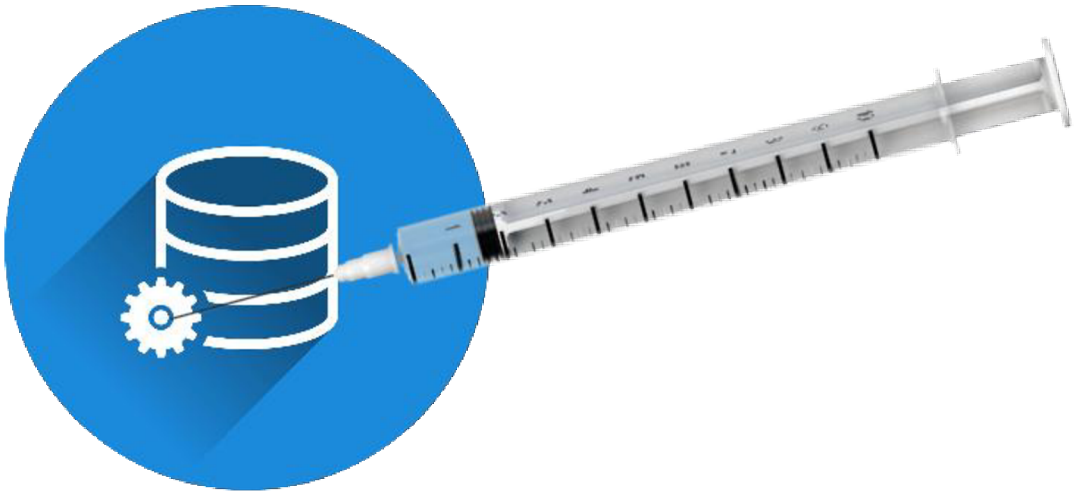
1.6 SQL bezpečnostné riziká

Ľahká dostupnosť aplikácie pre užívateľa so sebou prináša rôzne bezpečnostné riziká. SQL injection útoky, kt. využívajú zraniteľnosti sú síce komplikovanejšie, ale riziko napadnutia je napriek tomu vysoké.

Medzi hlavné SQL injection riziká patria:

- Získanie prístupu k dátam, ku ktorým nie sme autorizovaný mať prístup,
- Zmena dát, ktorá bola mienená len pre účel čítania,
- Možnosť vstúpiť do administračných častí internetovej aplikácie,
- Vyvolanie príkazov SQL serveru, ktoré nám umožnia ovládnuť zariadenie, na ktorom server beží,
- Zmazanie tabuliek, prípadne celej databázy,
- Poškodiť alebo zničiť integritu/dôvernosť/dostupnosť.

(31,15)



Obr. 12: Ilustrácia využitia zraniteľnosti databázy a jej injektovanie (vlastné).

1.7 Etický hacking webových aplikácií – postup pri príprave útoku

V tejto časti stručne rozpíšem návod akým budem neskôr postupovať pri realizácii etického hackingu, konkrétne SQL injection útoku.

Postup SQL injection útoku, kt. realizujem, sa dá zhrnúť v nasledujúcich bodoch:

- Príprava útoku(testovania)
    - (Voľba nástroja na skenovanie a testovanie webu)
- Nastavenie prostredia
- Skenovanie
- SQL injection útok

1.7.1 Príprava útoku

Nachádzame sa v prípravnej fáze, kedy je nutné si ujasniť nasledovné:

- Čo budeme testovať?
- Aké nástroje použijeme na skenovanie a testovanie webu ?
- Čas, kedy plánujeme testovať.

Než si zvolíme svoj nástroj na skenovanie, tak je potrebné sa zamyslieť a zodpovedať si nasledujúce otázky:

- Čo budeme testovať?
- Koľko sme ochotný do nástroja na skenovanie webovej aplikácie investovať ?

Tento krok nám výrazne uľahčí výber nástroja.

1.8 Obrana proti SQL injection

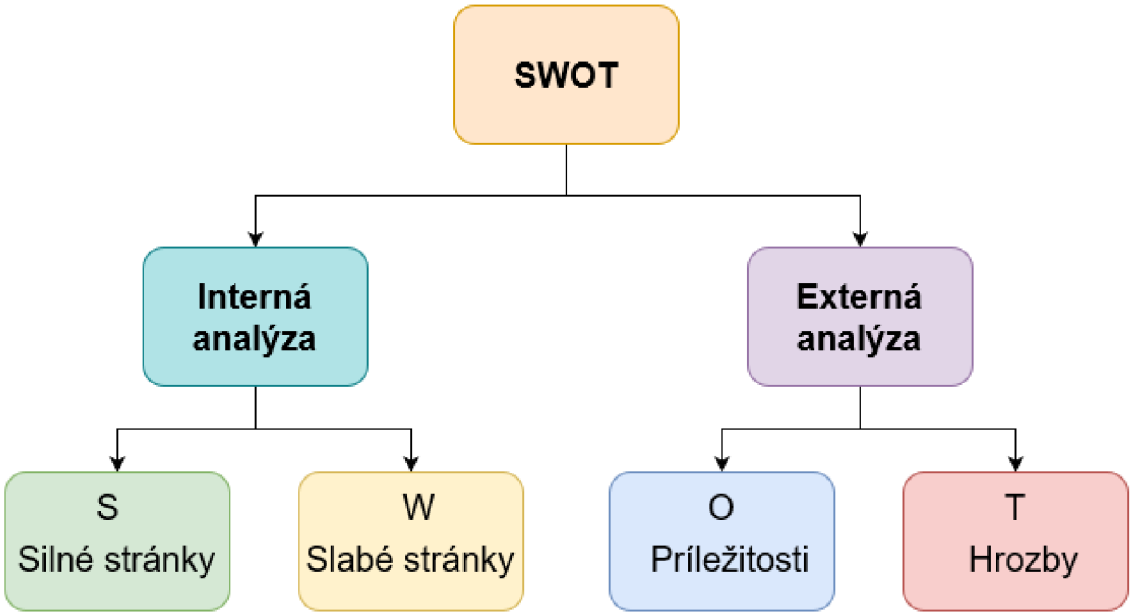
Každý vstup od užívateľa predstavuje pre aplikáciu potencionálne nebezpečie. Preto by sme mali brať na vedomie fakt, že ak vkladáme premenné do SQL dotazu, tak sa vystavujeme bezpečnostnému riziku. Aby sme mali dobre zabezpečenú obranu, tak je dôležité mať na pamäti, že SQLi je riziková všade tam, kde tvorca webovej aplikácie zabudol na bezpečnostné pravidlo, kt. znie nasledovne : „všetky vstupy do aplikácie je nutné kontrolovať na povolené hodnoty a typy.“ Pre bezpečnosť sa musia odstrániť potenciálne prvky škodlivého kódu, ako sú jednoduché úvodzovky. Ako k potencionálne rizikovým častiam aplikácie by sa malo pristupovať ku všetkým, kde je akýkoľvek vstup z vonka. Keď sa chceme brániť proti SQL injection útoku, tak je potrebné aby sme si vedeli overiť bezpečnosť webovej aplikácie, čo urobíme nasledovne. Začneme sa hrať so vstupmi do aplikácie u ktorej si chceme overiť bezpečnosť. Pre ručný test na SQLi slúži ako dôležitá pomôcka nasledujúci reťazec: ' or 1=1 --. Za predpokladu, že nám takýto reťazec od serveru vráti chybu, tak si môžeme byť istý, že sú v aplikácii nejaké neošetrené vstupy (15).

Je tiež dobré vypnúť viditeľnosť databázových chýb na vašich výrobných serveroch. Databázové chyby sa dajú použiť s SQLi na získanie informácií o vašej databáze. Ak zistíte zraniteľnosť aplikácie napríklad pomocou skenovacieho nástroja, pravdepodobne ju nebudete schopní okamžite opraviť. Táto chyba môže byť napríklad v otvorenom zdrojovom kóde. V takýchto prípadoch môžete dočasne dezinfikovať vstup pomocou brány firewall webovej aplikácie. Prevencia zraniteľností SQL Injection nie je jednoduchá. Konkrétne techniky prevencie závisia od podtypu zraniteľnosti SQLi, od databázového nástroja SQL a od programovacieho jazyka. Existujú však určité všeobecné strategické zásady, ktoré by ste mali dodržiavať, aby ste zaistili bezpečnosť svojej webovej aplikácie:

- Trénujte a udržiavajte informovanosť
- Neverte žiadnemu vstupu používateľa
- Používajte zoznamy povolených položiek, nie zoznamy zakázaných položiek
- Prijať najnovšie technológie / aktualizovať
- Zamestnávať overené mechanizmy
- Pravidelne skenovať webovú aplikáciu prostredníctvom nástroja (21)

1.9 SWOT analýza

Táto metóda nám prostredníctvom analýzy umožňuje nájsť v procese silné a slabé stránky a rovnako aj príležitosti a hrozby. V rámci analýzy sú mapované interné procesy, medzi ktoré zaraďujeme silné a slabé stránky procesu, a externé procesy, pod ktoré spadajú príležitosti a hrozby. SWOT analýza je mnoho funkcionálny nástroj, ktorý sa dá využívať napr. pre prípravu dielčieho inovačného projektu, pre prípravu taktiky víťazstva vo výberovom riadení. Analýzu má zmysel realizovať hlavne v prípadoch, kedy chceme niečo pozitívne zmeniť, keď hľadáme nové spôsoby, čo a akým spôsobom robiť lepšie. Je kľúčové nepodceňovať určovanie jednotlivých prvkov do kolóniek S,W,O a T. Pri tomto určovaní môžu byť veľmi nápomocné rôzne doplňujúce otázky, ktoré sú do istej miery veľmi unikátne pre jednotlivé situácie (13, 14).



Obr. 13: SWOT analýza a jej rozdelenie na jednotlivé prvky (13).

2 ANALYTICKÁ ČASŤ

Táto kapitola diplomovej je venovaná charakteristike súčasného stavu prostredia, na ktorom sú realizované testy v praktickej časti. podstaty SQL injection útoku a jeho priebehu, konkrétnym metódam s praktickými príkladmi, SQL skenerom a nástrojom, ktoré budú využité ako pomocný inštrument pri analýze vybraných webou.

2.1 Predstavenie testovacieho prostredia

Subjektami diplomovej práce budú 2 normálne fungujúce anonymizované webové aplikácie, 1 webová stránka vytvorená vo WordPresse pre účely testovania a 1 aplikácia, kt. vytvorili etický hackeri za edukačnými účelmi, ktorá bude hostovaná na mojom druhom notebooku.

2.1.1 Webové aplikácie určené pre učenie sa a testovanie etického hackingu

Pokiaľ si chcem vyskúšať nadobudnuté vedomosti ohľadom SQLi metód a nástrojov s ktorými sme sa oboznámili vyššie v súlade so zákonom, tak máme k dispozícii hneď niekoľko možností. Môžeme svoje služby ponúknuť spoločnosti, tak ako som to urobila ja alebo využijeme jednu z webových aplikácii, ktorá bola vytvorená presne pre tieto účely. V tabuľke číslo 1 porovnávam jednotlivé webové aplikácie, ktoré boli vytvorené pre účely testovania etickým hackermi, nadšencami pre bezpečnosť a celkovo ľudí, kt. sa chcú v tejto oblasti zdokonaľovať. Konkrétne porovnávam možnosti, ktoré jednotlivé aplikácie ponúkajú skrz testovanie, to pre akú úroveň schopností sú odporúčané a v neposlednom rade to, či je v aplikácii možné testovať SQLi útok, čo je pre moje testovanie primárnou podmienkou. Na základe týchto všetkých podmienok si jednu z týchto aplikácii vyberiem ako testovací subjekt D.

Tab. 1: Porovnávanie webových aplikácií pre účely etického hackingu (vlastné).

Webové aplikácie	Popis webovej aplikácie	Pre koho je aplikácia určená	Možnosť využiť aplikácie na precvičovanie SQLi
bWAPP	Táto aplikácia nám ponúka na testovanie nasledujúce :XSS, XST,CSRF,Man-in-the-middle attacks,SSRF,DoS útok, SQLi, HTML, iFrame, SSI, OS Command, PHP, XML, XPath, LDAP, Host Header and SMTP injections.	študenta, nadšenca pre bezpečnosť, developera, profesionála	✓
HackThisSite	Aplikácia nám v rámci testovania ponúka :Realistické misie, Aplikačné misie,Telefónne misie, Forenzné misie, Programovacie misie.	pre kohokoľvek od začiatočníka až po niekoho s rozšírenejšími schopnosťami hackingu	x
Google Gruyere	Táto aplikácia nám ponúka na testovanie nasledujúce : XSS, XRF,Vzdialené spustenie kódu, DoS útok, Zverejňovanie informácií.	pre začiatočníkov, ktorý chcú zlepšiť svoje zručnosti	x
DVIA	Mobilná aplikácia nám v rámci testovania ponúka : Phishing, Detekcia úteku z väzenia, Debugging, Dotykové / Face ID bypass, Únik údajov z postranného kanála, Zlomená kryptografia, Zabezpečenie sieťovej vrstvy, Oprava aplikácie.	nadšenca pre bezpečnosť v mobile, developera, profesionála	x
Hellbound Hackers	Aplikácia nám v rámci testovania ponúka : Hackovanie aplikácií, Základné webové hackovanie, Hacking v skripte Javascript, Problémy so zakorenením, Problémy s testovaním perom	pre kohokoľvek od začiatočníka až po niekoho s rozšírenejšími schopnosťami hackingu	x
OWASP Mutilidae II	Aplikácia bsahuje viac ako 40 zraniteľností a obsahuje k tomu aj veľké množstvo zraniteľností OWASP Top 10.	pre študenta, nadšenca pre bezpečnosť, developera, profesionála	x
Defend the Web	Obsahuje vyše 60 úrovní hackerstva a články, ktoré pokrývajú všetky oblasti bezpečnosti vrátane tých, ktoré sú na úrovni konkrétne obsiahnuté.	pre kohokoľvek od začiatočníka až po niekoho s rozšírenejšími schopnosťami hackingu	x
WebGoat	Táto aplikácia je vysoko nezabezpečená a plná bežných chýb zo strany servera, čo ju robí ideálnou na učenie sa o bezpečnosti aplikácií a precvičovaní zručností. Hlavné zo zraniteľných miest a útokov kt. nám táto aplikácie umožnuje skúmať sú: Otrava cache, SQLi, útoky trójskych koní, spyware, kódovanie Unicode	pre kohokoľvek od začiatočníka až po niekoho s rozšírenejšími schopnosťami hackingu	✓
Root Me	Je to skvelá platforma na bezpečnostné testovanie a rozvoj hackerských schopností. Rôyne subjektz testovania, kt. pokrýva Root Me: Digitálne vzšetrovanie, Automatizácia, Rozlomenie šifrovania, Cracking, Sieťové výzvy a SQLi.	Nieje určená pre začiatočníkov, ale pre všetkých na vyššej úrovni hackingu	✓
OverTheWire	Je ideálnym miestom pre zábavu a učenie sa hackingu na rôynzch úrovniach schopností. Každá úroveň obsahuje špecifické scenár.	pre kohokoľvek od začiatočníka až po niekoho s rozšírenejšími schopnosťami hackingu	x

Podľa analýzy možností, ktoré jednotlivé aplikácie pre testovanie útokov ponúkajú mi ako vhodné pre účely realizácie SQLi útokov vyšli tieto 3: bWAPP, WebGoat, Root Me. Vzhľadom k tomu, že Root Me je určené iba pre vyššie úrovne hackingu, tak ju nebudem vo svojom testovaní používať. Vzhľadom k tomu, že bWAPP používa MySQL databázu ,môže byť hostovaná na Linuxe aj Windowse tak z možností, ktoré mi ostali vychádza

ako najideálnejšia varianta pre moje testovanie. Z vyššie zmienených dôvodov určujem bWAPP ako testovací Subjekt D.

Tab. 2: Subjekty diplomovej práce (vlastné).

Anonymizované stránky	Charakteristika	Typ webovej aplikácie	Certifikácie
Subjekt A	Acunetix testový web	naprogramovaná	nie
Subjekt B	Je webová stránka základnej školy.	prenajatá od Wordpress	áno
Subjekt C	Sociálna sieť	naprogramovaná	áno
Subjekt D	Webová aplikácia, ktorú vytvorili etický hackeri za účelom edukácie a etického testovania úrovne hackingu	bWAPP	nie

Hlavným poslaním subjektov práce je sprostredkovanie rozmanitého prostredia na účely etického testovania SQLi útoku, vďaka čomu posúdim efektívnosť rôznych SQLi metód v spojitosti s rôznym prostredím a zabezpečením webových aplikácií.

2.1.2 Nastavenie prostredia pred penetračným testovaním

V rámci penetračného testovania budú nakonfigurované dve rôzne prostredia, prvé s Windows 10 a druhé s Kali Linuxom. Zmyslom tejto časti je si vytvoriť taký základový systém o kt. sa viem, že bude v testoch konzistentný.

Predtým ako sa začne prevádzkovať virtuálny stroj a inštalovať naň nástroje potrebné k penetračným testom, tak je dôležité sa presvedčiť, že je k dispozícii taký počítač, kt. zvládne všetko potrebné.

2.1.2.1 Základné požiadavky na hardware

Než sa započne nakonfigurovanie vybraného prostredia, respektíve prostredí, tak je potrebné overiť či zariadenie, kt. je plánované používať na tieto účely spĺňa minimálne odporúčané požiadavky hardwaru a softwaru. Môže sa zdať, že tieto požiadavky sú trochu prehnané, ale malo by sa brať na vedomie, že sa bude spúšťať viac virtuálnych strojov, kt. jednoducho vyžadujú lepší hardware. V nasledujúcej časti prebehne inventúra HW, ktorý je na toto testovanie k dispozícii.

Minimálne odporúčané požiadavky na hardware notebooku z ktorého bude realizované testovanie:

- Notebook s aspoň 8 GB RAM

- 500 GB priestoru na pevnej jednotke, ideálny je pre túto prácu SSD
- Procesor i7 Intel Quad Core
- WMware Workstations/Fusion/Player/Virtual Box (9)

HW ktorým mám k dispozícii:

Tab. 3: Špecifikácie serveru pre subjekt D (vlastné).

HW serveru pre Subjekt D	
Model	Lenovo Thinkpad E520
Procesor	Core i5
RAM	16,00 GB
Grafická karta interná	Intel HD Graphics 2000
Grafická karta externá	AMD Radeon HD 6630 M 2GB vlastnej pamäte
Disk	SSD 500 GB
Operačný systém	Windows 10 PRO

Tab. 4: Špecifikácie testovacej stanice (vlastné).

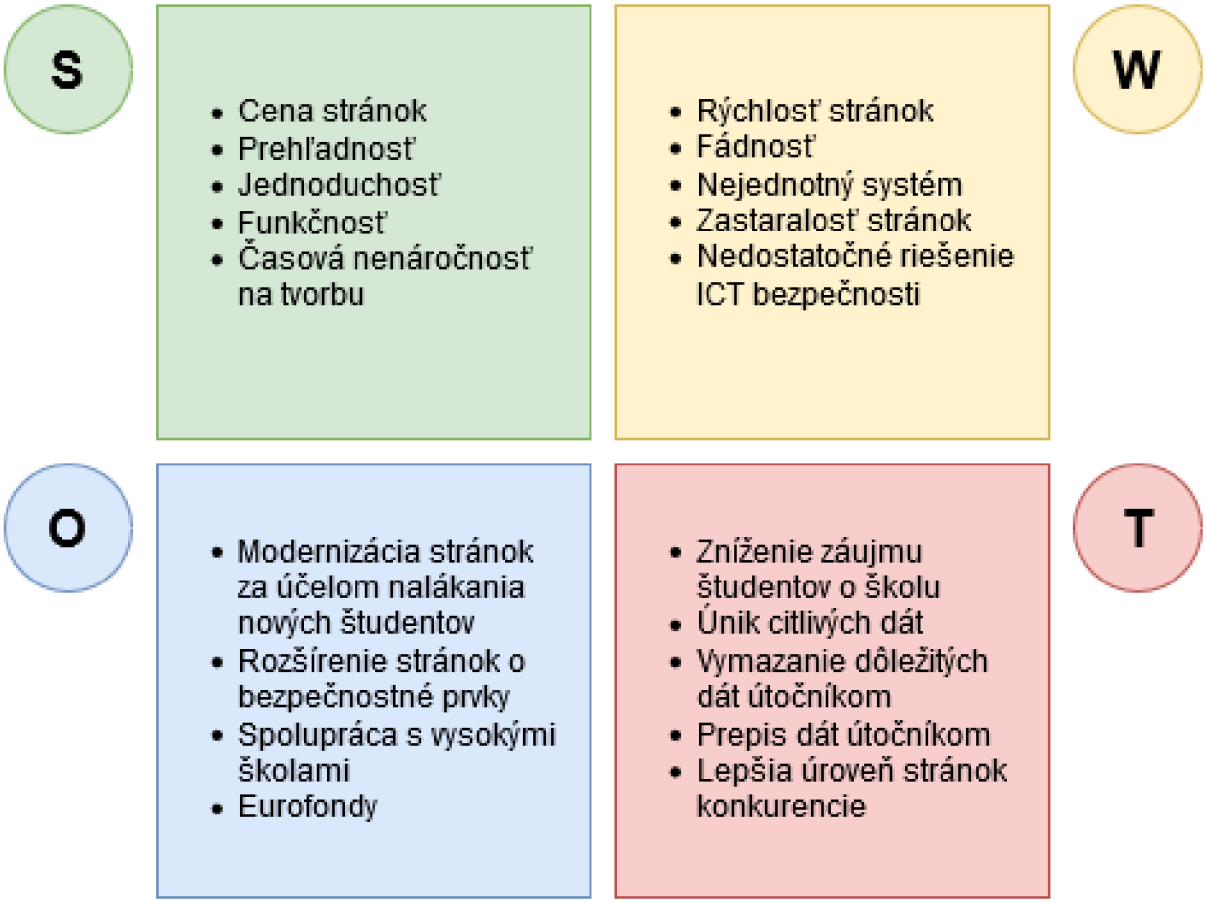
HW notebooku z ktorého bude prebiehať testovanie	
Model	ASUS Zenbook PRO UX550VD
Procesor	Intel Core i7-770HQ
RAM	8,00 GB
Grafická karta interná	Intel HD Graphics 630
Grafická karta externá	NVIDIA GeForce GTX 1050 Ti
Disk	SSD 500 GB
Operačný systém	Windows 10 PRO
Vistual Box OS	Kali Linux

2.2 SWOT analýza testovacích Subjektov

Nasledujúca časť diplomovej práce je venovaná SWOT analýze testovacím subjektom, ktorá pozostáva z ich silných, slabých stránok, príležitostí a hrozieb z pohľadu webových aplikácií.

2.2.1 SWOT analýza subjektu B

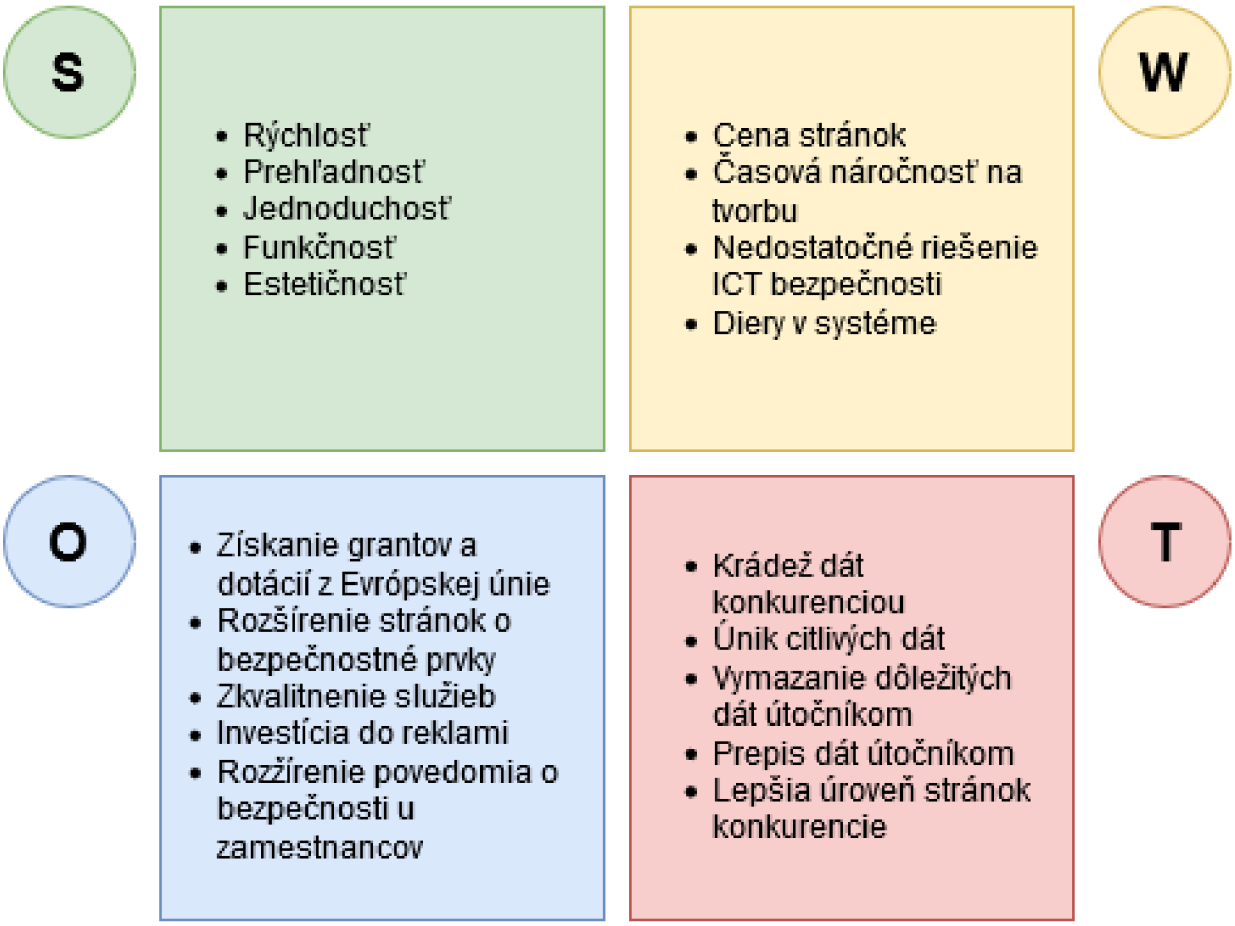
V tejto podkapitole je rozobraná SWOT analýza webových stránok základnej školy, kt. bola vytvorená vo WordPresse. Tieto webové stránky sú v práci z dôvodu pseudonymizácie označované ako testovací Subjekt B.



Obr. 14: SWOT analýza Subjektu A (vlastné).

2.2.2 SWOT analýza subjektu C

V tejto podkapitole je rozobraná SWOT analýza webových stránok spoločnosti , ktorá sprostredkúva sociálnu sieť. Tieto webové stránky sú v práci z dôvodu pseudonymizácie označované ako testovací Subjekt C.



Obr. 15: SWOT analýza testovacieho Subjektu B (vlastné).

2.3 SQL injection

2.3.1 Praktická ukážka SQLi metód

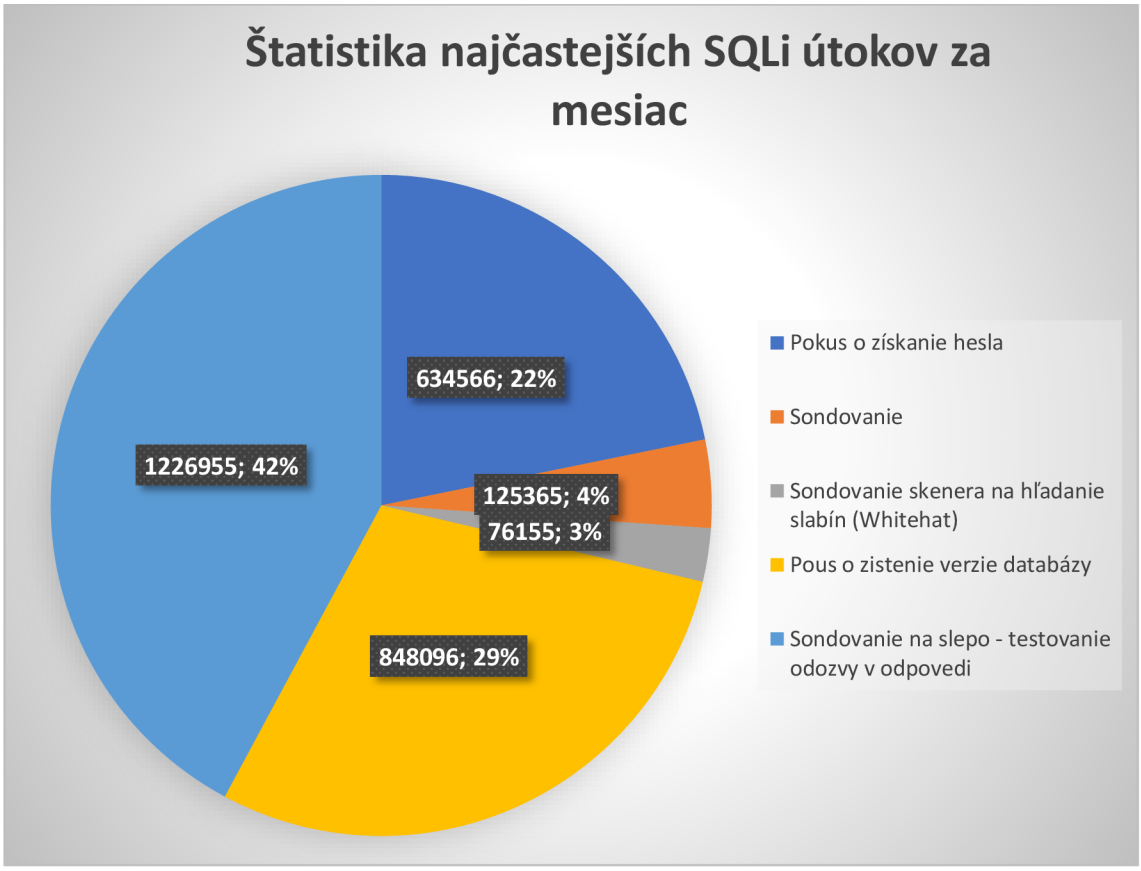
Cieľom tejto podkapitoly je prezentácia SQLi útočných metód na praktických ukážkach, vďaka ktorým uvidíme ako fungujú.

Tab. 5: Praktická ukážka SQL injection metód (vlastné).

Metódy SQLi	Stručný popis	Príklad
Priame SQLi metódy		
Union dotaz	Útočník vloží dotaz s originálnym SQL dotazom za použitia union kľúčových slov.	root´ union all select number from cards -- Dotaz vráti záznamy z tabuľky users, záznamy z tabuľky cards. Operátor UNION je vhodné použiť s kľúčovým slovom all, pretože bez neho databáza vráti DISTINCT (rozdielne záznamy)
SQLi na základe chyby	Útočník zlomyselne infikuje vstup v dotaze aby vytvoril nejaké logické chyby za účelom získania informácií.	SELECT*FROM tbl1 GROUP BY FLOOR (rand() *2) having min(0); SELECT*FROM x GROUP BY CONCAT_WS('~',version (),FLOOR (rand(0)*2)) having min(0);
Koment na konci riadka	Útočník použije polovicu vstupu a pridá -- na koniec jeho poľa, vďaka čomu je zvyšok informácií vnímaný ako komentár.	SELECT * FROM articles WHERE UNION SELECT * FROM TABLE user;-- root'; SELECT usename as email FROM pg_user -- root'; SELECT column_name as email FROM information_schema.columnsWHERE table_name =
Vložené komentáre	Pri útoku môžu slúžiť na odkomentovanie časti dotazu ako klasické komentáre. Tieto vložené komentáre sa v jazyku SQL uzatvárajú do reťazcov.	D/**/ROP TA/*xx*/BLE use/*xx*/rs Ukážka vyššie maskuje dotaz DROP TABLE users SELECT /*!32300 1/0, */ 1 FROM users Ukážka vyššie vráti chybu ak je verzia SQL serveru vyššia ako 3.23.02.
Slepé SQLi metódy		
SQLi založená na čase	Útočník zbiera informácie na základe pozorovania časovej stopy odpovede databázy.	SELECT*FROM products WHERE id=1-SLEEP(20) SELECT*FROM products WHERE id=1-BENCHMARK(100000000, rand()) SELECT * FROM products WHERE id=1; WAIT FOR DELAY '00:00:20' SELECT * FROM products WHERE id=1; IF SYSTEM_USER='ba' WAIT FOR DELAY '00:00:20'
SQLi založená na boolean	Útočník pošle databáze pravdivú alebo nepravdivú otázku a určí odpoveď na základe spätnej väzby aplikácie aby videl či db server odpovie rovnako akoby tomu bolo za normálnych okolností.	SELECT*from table_name WHERE id=1' AND 1=0 SELECT*from table_name WHERE id=1 SELECT *from table_name WHERE id=1' AND (length(database())) = 8

2.3.2 Štatistika najčastejších SQLi útokov

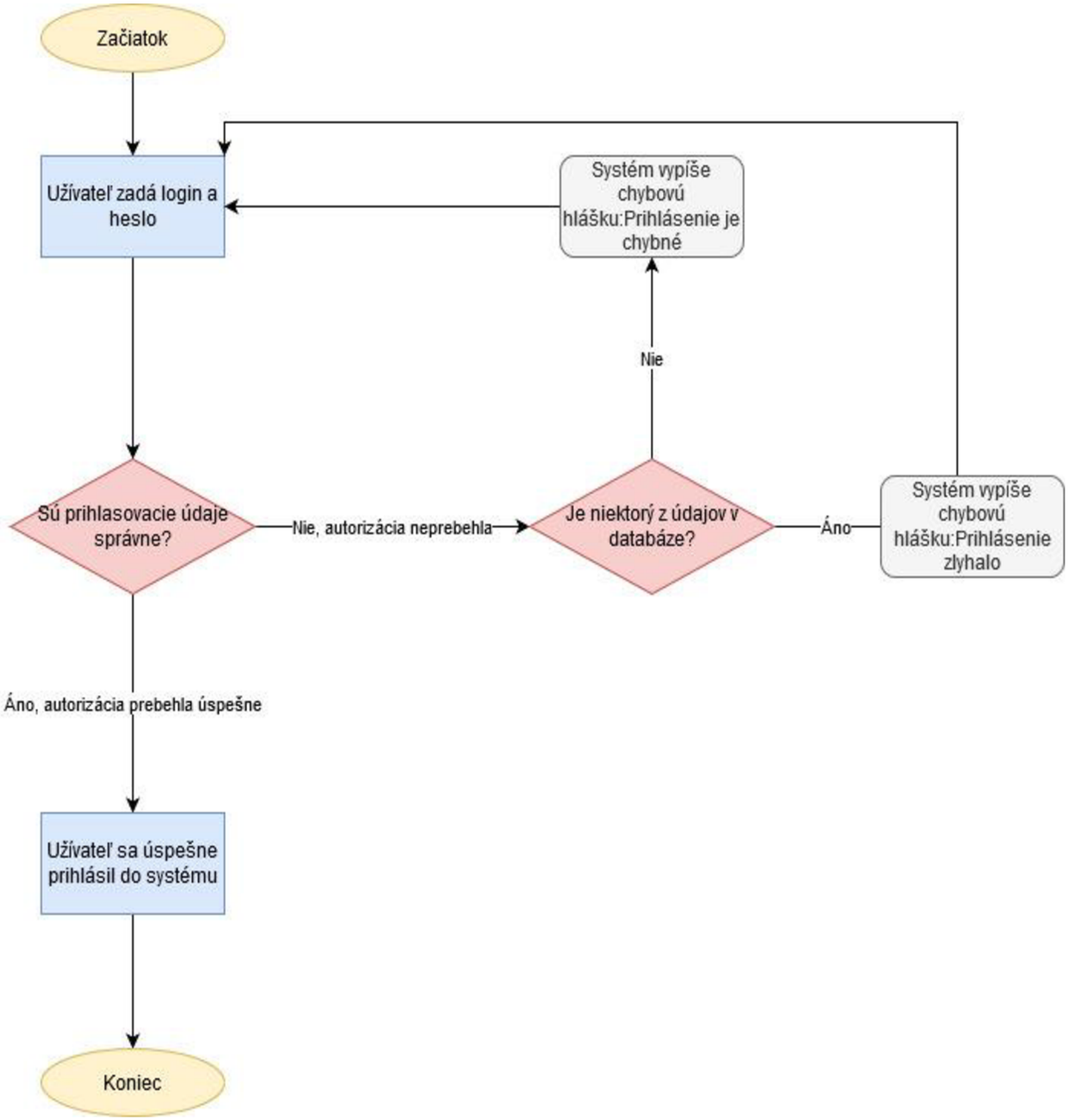
Táto štatistika mi pomôže pri výbere SQLi útokov na ktoré sa mám najviac zamerať v rámci testovania a aj pri prioritizácii spôsobov obrany proti jednotlivým SQLi metódam.



Obr. 16: Graf SQLi incidentov za mesiac (vlastné).

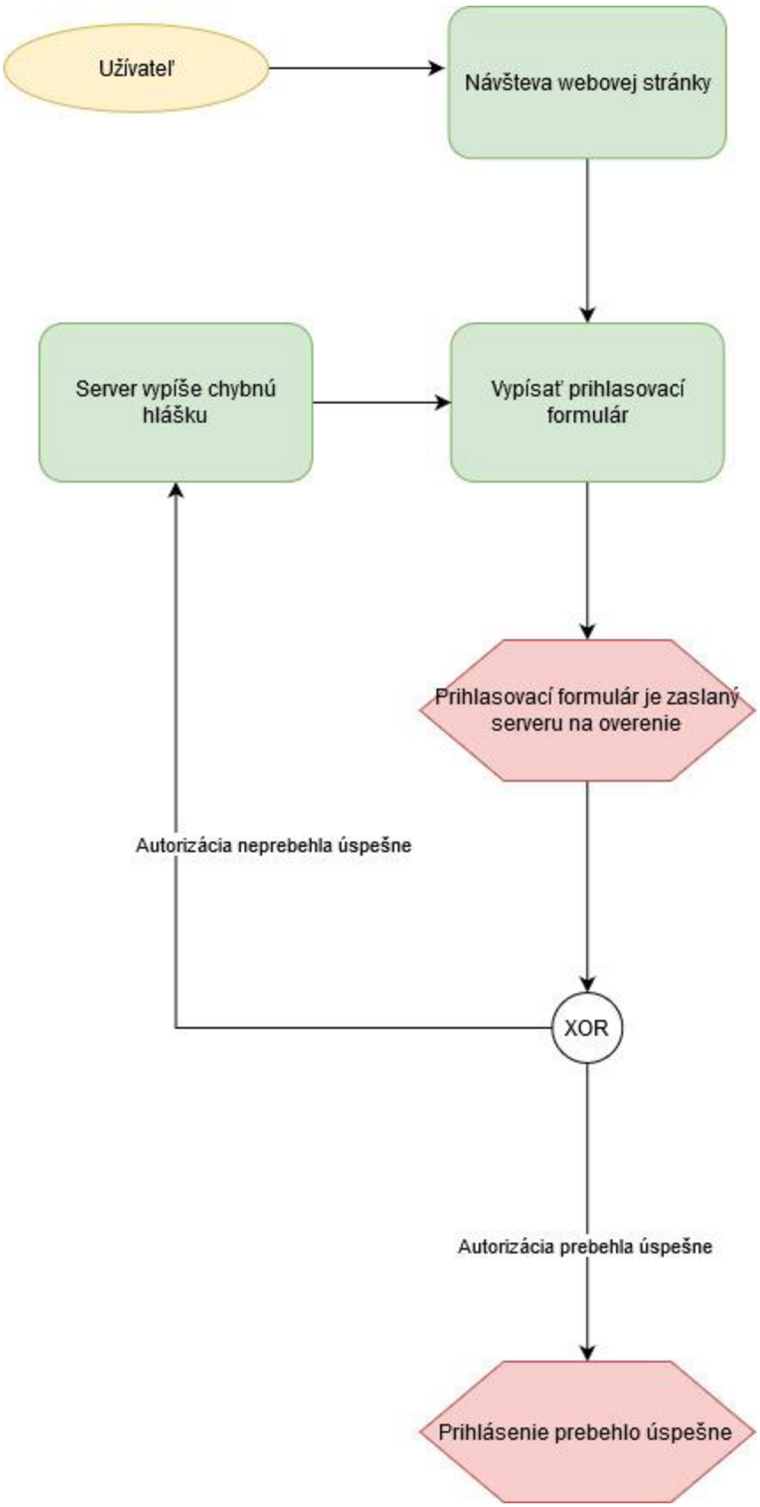
Ako môžeme vidieť na obr. 11, tak je veľmi znepokojujúce v akom pomere sú etický hackeri 3% v porovnaní s neetickými 97%. Z grafu vidím, že až 42% útokov zo sledovaného obdobia spadá pod SQLi útoky na slepo, konkrétne SQLi založenom na odozvy odpovede, v rámci práce mám túto metódu pomenovanú ako „SQLi založené na čase“.

2.3.3 Proces prihlásenia užívateľa na Subjekt A



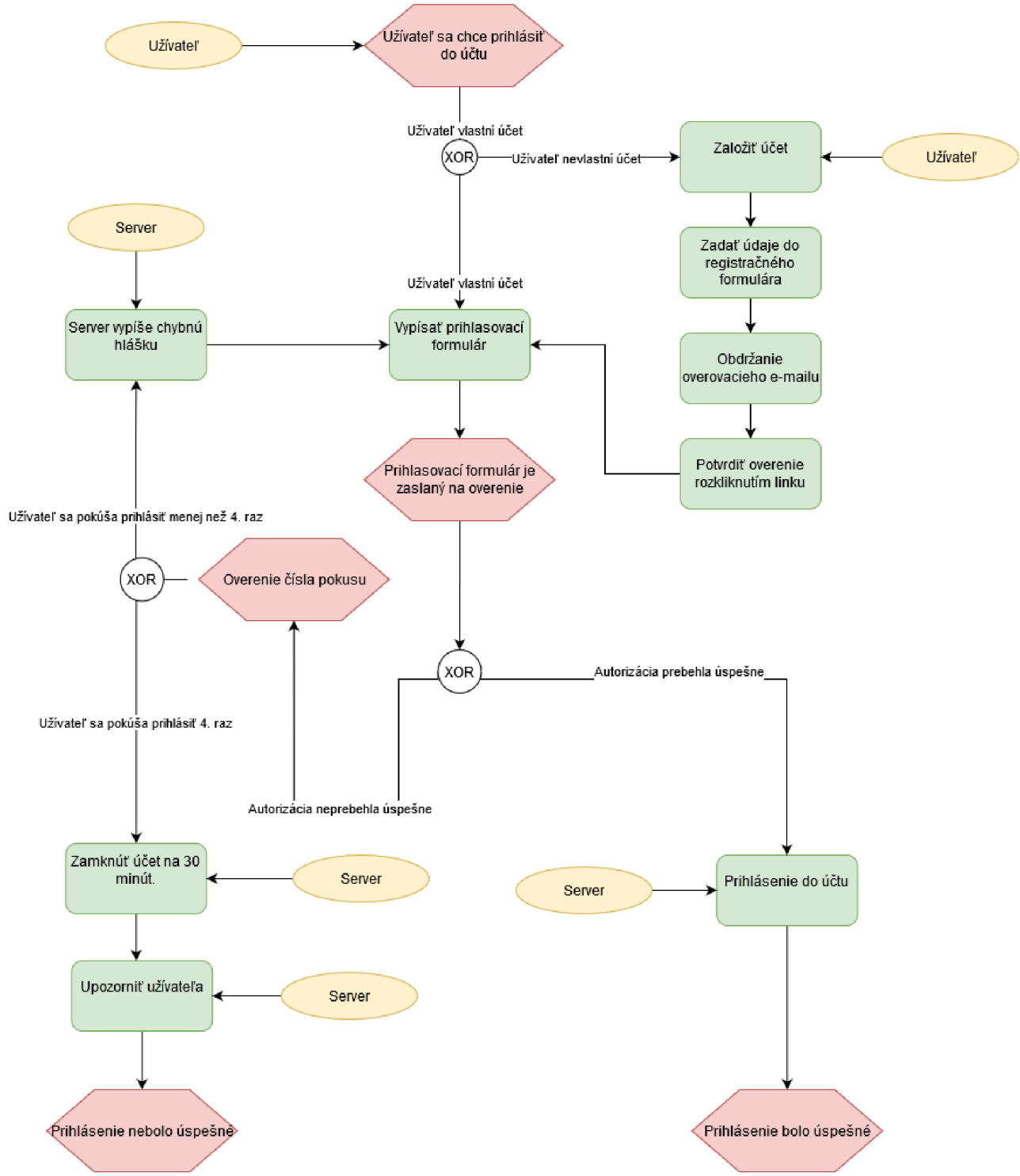
Obr. 17: Proces prihlásenia užívateľa na Subjekt A (vlastné).

2.3.4 Proces prihlásenia užívateľa na Subjekt B



Obr. 18: Proces prihlásenia užívateľa do testovacieho Subjektu B (vlastné).

2.3.5 Proces prihlásenia užívateľa na Subjekt C



Obr. 19: Proces prihlásenia užívateľa na Subjekt C (vlastné).

Tab. 6:Popisná tabuľka procesov prihlásenia užívateľa na Subjekt A,B,C (vlastné).

Subjekty testovania	Subjekt A	Subjekt B	Subjekt C
Názov procesu	Prihlásenie užívateľa do systému	Prihlásenie užívateľa do systému	Prihlásenie užívateľa do systému
Cieľ procesu	Prihlásenie užívateľa do systému	Prihlásenie užívateľa do systému	Prihlásenie užívateľa do systému
Vstup procesu	Vyplnený prihlasovací formulár	Vyplnený prihlasovací formulár	Vyplnený prihlasovací formulár
Výstup procesu	Úspešné alebo neúspešné prihlásenie sa do systému	Úspešné alebo neúspešné prihlásenie sa do systému	Úspešné alebo neúspešné prihlásenie sa do systému
Špecifikácia procesu	Podľa úspešnosti autorizácie, ktorá je overovaná na základe prihlasovacieho formulára je užívateľovi umožnené alebo zamietnute prihlásenie sa do systému	Podľa úspešnosti autorizácie, ktorá je overovaná na základe prihlasovacieho formulára je užívateľovi umožnené alebo zamietnute prihlásenie sa do systému	Podľa úspešnosti autorizácie, ktorá je overovaná na základe prihlasovacieho formulára je užívateľovi umožnené alebo zamietnute prihlásenie sa do systému
Vlastník procesu	Server	Server	Server
Role	Server, užívateľ	Server, užívateľ	Server, užívateľ
Zákazník procesu	Užívateľ s vytvoreným účtom, útočník	Užívateľ s vytvoreným účtom, útočník	Užívateľ s vytvoreným účtom, útočník
Podmienky	Vytvorený užívateľský účet	Vytvorený užívateľský účet	Vytvorený užívateľský účet
Metriky	–	–	–
Informačné systémy			
Software	Webová stránka, Databáza	Webová stránka, Databáza	Webová stránka, Databáza
Dokumenty	Prihlasovací fromulár, Registračný formulár, Overovací e-mail	Prihlasovací fromulár, Registračný formulár, Overovací e-mail	Prihlasovací fromulár, Registračný formulár, Overovací e-mail

2.3.6 Komentár k procesu prihlasovania sa na jednotlivé stránky

V tejto časti budú zhrnuté výsledky pozorovania ku ktorým som dospela v rámci pokusov prihlasovania sa na jednotlivé webové aplikácie.

2.3.6.1 Komentár k prihlasovaniu sa na Subjekt A

Po návšteve webových stránok testovaného subjektu bolo realizovaných niekoľko pokusov a prihlásenie sa do užívateľského účtu. Z týchto pokusov bol vypozorovaný priebeh prihlasovania sa, ktorý vypadal nasledovne:

- Zadanie loginu a hesla
- Overenie autorizácie: „Sú prihlasovacie údaje správne?“
- Pokiaľ žiaden z prihlasovacích údajov nebol správny, tak server vypísal toto chybné hlásenie: “Prihlásenie zlyhalo“
- Ak bolo správne aspoň užívateľské meno, tak server vypísal nasledovnú hlášku: „Prihlásenie je chybné.“
- Pri pokuse, kde boli správne všetky prihlasovacie údaje prebehla autorizácia úspešne.
- Užívateľovi sa podarilo prihlásiť do systému.

2.3.6.2 Komentár k prihlasovaniu sa na Subjekt B

Po návšteve webových stránok testovaného subjektu bolo realizovaných niekoľko pokusov a prihlásenie sa do užívateľského účtu. Z týchto pokusov bol vypozorovaný priebeh prihlasovania sa, ktorý vypadal nasledovne:

- Zadanie loginu a hesla
- Overenie autorizácie: „Sú prihlasovacie údaje správne?“
- Pokiaľ akýkoľvek údaj nebol správny, tak server vypísal chybné hlásenie.
- Následne bol užívateľ vrátený k ďalšiemu pokusu vypísať prihlasovací formulár, čo sa opakovalo do momentu kým neboli všetky údaje vypísané správne.
- Pri pokuse, kde boli správne všetky prihlasovacie údaje prebehla autorizácia úspešne.
- Užívateľovi sa podarilo prihlásiť do systému.

2.3.6.3 Komentár k prihlasovaniu sa na Subjekt C

Po návšteve webových stránok testovaného subjektu bolo realizovaných niekoľko pokusov a prihlásenie sa do užívateľského účtu. Z týchto pokusov bol vypozorovaný priebeh prihlasovania sa, ktorý vypadal nasledovne:

- Zadanie loginu a hesla
- Overenie autorizácie: „Sú prihlasovacie údaje správne?“
- Pokiaľ akýkoľvek údaj nebol správny, tak autorizácia neprebehla úspešne a server vypísal túto chybnú hlášku: “Vypadá to tak, že tvoje užívateľské meno alebo heslo je nesprávne, skús to znovu.“
- Server overil číslo pokusu prihlásenia, pokiaľ sa užívateľ snažil neúspešne prihlásiť menej ako 4 razy, tak ho to vrátilo na prihlasovací formulár.
- Ak sa užívateľ pokúšal neúspešne prihlásiť viac ako 3 razy, tak pred prístupom do formulára mu dal server vyplniť captcha aby overil, či sa snaží prihlásiť človek.
- Pri pokuse, kde boli správne všetky prihlasovacie údaje prebehla autorizácia úspešne.
- Užívateľovi sa podarilo prihlásiť do systému.

V rámci jednotlivých pokusov sa mi okrem procesu prihlasovania podarila vypozorovať ešte jedna vec. Pokiaľ sa podarilo náhodne zvolené užívateľské meno trafiť správne počas toho čo autorizácia neprebehla úspešne a zároveň tento užívateľ nebol prihlásený 6 a viac mesiacov, tak server vypísal nasledovnú chybnú hlášku: Z bezpečnostných dôvodov žiadame ľudí, ktorí nepristúpili k svojmu účtu 6 a viac mesiacov, aby sa buď prihlásili pomocou prezývky a hesla, alebo aby resetovali svoje heslo, aby sa mohli prihlásiť pomocou svojho e-mailu a hesla. Táto chybná hláška poskytuje informáciu o tom, že náhodne zvolené užívateľské meno je správne.

2.4 Predstavenie a analýza vybraných nástrojov na skenovanie webu

V rámci tejto časti budú predstavené a stručne rozanalyzované nasledujúce nástroje a na základe analýzy potom budú vybrané tie, kt. použijem v praktickej časti.

- Sqlmap
- Sqlninja

2.4.1 SQLmap (Kali Linux)

Sqlmap budem vo svojej práci využívať primárne na zbieranie citlivých dát z databázy ako napríklad: užívateľské meno, heslo, e-mail, adresa a pod. Tento nástroj je k dispozícii na operačnom systéme Kali Linux. Prostredníctvom tohto nástroja sa budem pokúšať nájsť SQLi, manipulovať s databázou skrz SQL dotazy alebo urobiť jej dump. S využitím injekcie dokáže získať interaktívny shell, alebo dokonca poskytnúť Virtual Network Computing reláciu naspäť k útočníkovi, v tomto prípade mne (2).

V nasledujúcich príkladoch týkajúcich sa SQLmap ukazujem parametre GET a POST, pretože sú to najbežnejšie typy SQLi a z toho dôvodu ich používam aj v praktickej časti. Oba zmieňované HTTP útoky ukazujú ako veľmi záleží na správnej konfigurácii požiadavku (ak by nebol správne nakonfigurovaný, tak sa útok pravdepodobne nezdarí) (2).

Obr. 20: SQLmap (16).

Príklad na parameter GET– V rámci tohto príkladu sa predpokladá, že parameter GET je práve to miesto, kde je s URL spojená zraniteľnosť SQLi. Cieľom je otestovať všetky parametre a utvrdiť sa v tom, že je nájdená zraniteľnosť SQLi skutočná a vyvrátiť možnosť, že by sa jednalo o falošné poplachy, kt. autor videl u skenovacích nástrojov, z čoho prichádza k záveru, že jediná metóda ako zaručiť, že sa skutočne jedná o použiteľný nález, je validácia (2).

- V prvom kroku zistíme či je SQL injekcia platná (pokiaľ áno, tak výsledkom bude uvítacie hlásenie):
  sqlmap – u „http://.com/info.php?user =test&pass=test“ -b

- V druhom kroku budeme získavať databázové užívateľské meno:

sqlmap – u „http://site.com/info.php?user =test&pass=test“ -- current-user

- V ďalšom kroku si napíšeme príkaz na získanie interaktívneho shell:
  sqlmap – u „http://site.com/info.php?user =test&pass=test“ --os-shell

- Niekoľko typov a trikov od autora, ktoré budem aplikovať v praktickej časti:
  - Pokiaľ nastane situácia, kedy bude potrebné špecifikovať to, na aký typ databáze sa má zaútočiť a pokiaľ usudzujeme, že má injekcia šancu, ale SQLmap nič nenašiel, tak je odporúčané nastaviť prepínač:
    --dbms= [typ databázy].
  - Autor spomína, že pokiaľ potrebujeme nejaký nález autentizovanej SQL injekcie, tak sa máme prihlásiť do webu prostredníctvom webového prehliadača a vziať cookie (je možné ho odobrať priamo z nástroja Burp). Definícia cookie za pomoci prepínača -- data= [COOKIE].
  - Za predpokladu, že si stále nevieme rady, tak je tu ešte možné skúsiť príkaz: sqlmap –wizard (2).

Príklad na parameter POST – tento parameter je veľmi podobný parametru GET, ich hlavný rozdiel tkvie v tom, že ako sa bude predávať zraniteľný parameter. Nebude sa nachádzať v URL, ako tomu bolo v predchádzajúcom príklade a to z toho dôvodu, že parametre POST sa predávajú v dátovej sekcii. Toto je možné úplne bežne vidieť u predávania užívateľských mien a hesiel a to preto, lebo webové servery obvykle logujú parametre GET a my nechceme, aby server zaznamenal tieto údaje do logu.

- V prvom kroku zistíme či je SQL injekcia platná (pokiaľ áno, tak výsledkom bude uvítacie hlásenie):

sqlmap – u „http://site.com/info.php“ --data= „user=test&pass=test“ –b

Obr. 21: Platnosť SQLi (2).

- V druhom kroku budeme získavať databázové užívateľské meno:

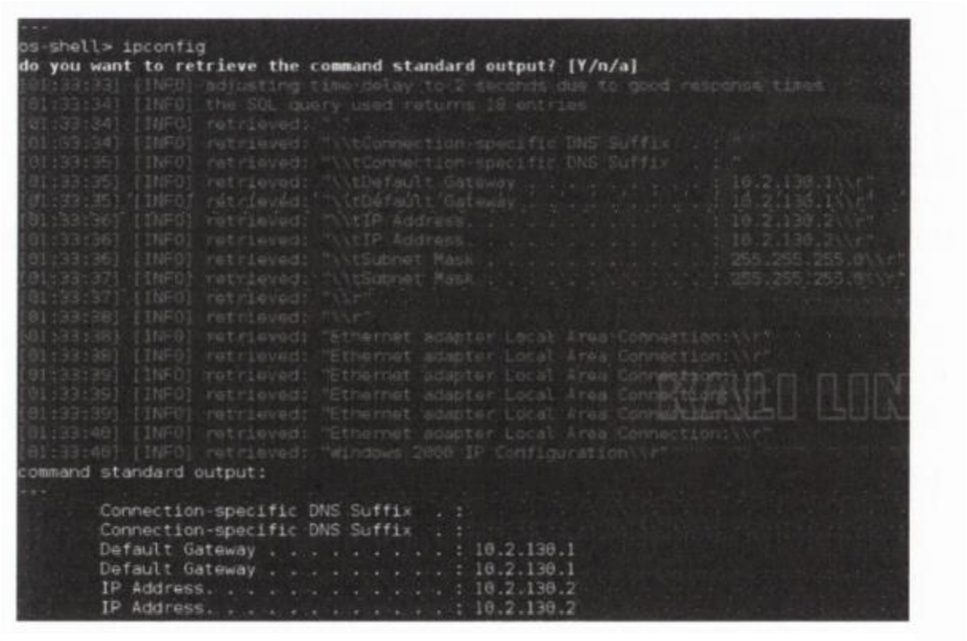
```
sqlmap – u „http://site.com/info.php“ --data= „user=test&pass=test“
-- current-user
```

Obr. 22: Získanie databázového užívateľského mena (2).

- V ďalšom kroku si napíšeme príkaz na získanie interaktívneho shell:

```
n sqlmap  u „http://site.com/info.php“ --data= „user=test&pass=test“
--os-shell
```

Obr. 23: Príkaz na získavanie interaktívneho shellu (2).



Obr. 24: Príklad na spustenie príkazu ipconfig (2).

Za predpokladu, že sa nám v poslednom kroku podarí získať prístup k nejakému shellu OS, ako databázový užívateľ, tak budeme mať úplný prístup k príkazovému riadku. V nasledujúcom príklade sa autorovi podarilo nájsť zraniteľnú SQLi, získať shell OS a spustiť príkaz ipconfig (2).

2.4.2 Sqlninja (Kali Linux)

Sqlninja je výborný SQLi nástroj pre nahrávanie shellov a obchádzanie sieťových IDS, ktorý tiež používam vo svojej práci na detekciu a to z toho dôvodu, že Sqlmap nemusí niektoré zraniteľnosti nájsť a dostatočná kontrola v oblasti bezpečnosti je dôležitá. Jeho hlavným rozdielom oproti Sqlmap tkvie v tom, že tu je potrebné definovať zraniteľnú premennú na rozdiel od predchádzajúceho nástroju. V Sqninja sa používa nasledujúci príkaz na injektovanie premennej :

SQL2INJECT

Obr. 25: Sqlninja príkaz na injektovanie premennej (vlastné).

Pred použitím nástroja je nutné vytvoriť konfiguračný SQL súbor, ktorý bude obsahovať všetky potrebné informácie o URL, typu http metódy, cookie relácie a agentov prehliadača.

Tieto informácie získam nasledovne:

- Nainštalujem trial verziu Burpu zo stránok: https://bit.ly/3bHOsu4
- Spustím Burp a zapnem pozastavovanie proxy na požiadavku v kt. sa predáva zraniteľné pole
- Zachytávanie požiadavku užívateľa do /wfLogin.aspx, následná identifikácia parametrov Post
- Vďaka vyššie zmieneným krokom získam informácie, kt. sa požadujú pri injekciách s týmto nástrojom

2.5 Analýza SQLi príkazov

2.5.1 Útoky na SQL prostredníctvom príkazu INSERT

Ako útočník máme možnosť využiť v náš prospech zle napísané príkazy jazyka SQL. Nasledujúci kód vkladá údaj do tabuľky SQL:

```
$sth - > prepare(“ INSERT INTO data (name) VALUES('$name')“;
$sth - > execute();
```

Útočník má možnosť do formulára vložiť dáta tak, aby premenná $name mala nasledovné hodnoty:

```
Bea' ); DROP TABLE data; INSERT INTO funct (bar) VALUES (' lol
```

Výsledná funkcia bude vypadať nasledovne:

```
$sth - > prepare(“ INSERT INTO data (name) VALUES('Bea')“; DROP TABLE data;
INSERT INTO  funct (bar) VALUES ('lol') ");
```

Takýmto spôsobom útočník vymaže tabuľku data a do tabuľky funct vloží vlastné dáta.

Je výhodné používať pozičné parametri. Namiesto vloženia zadanej hodnoty použijeme ? ako indikátor hodnoty. Následne samotnú hodnotu predáme ako parameter metódy execute (10).

Tab. 7: Analýza príkazu insert (10)

Analýza príkazu INSERT	
Rozšírenosť	5
Náročnosť	6
Dopad	5
Celkové riziko	5

2.6 Analýza rizík

Ide o prvý krok v procese znižovania rizík. Tieto riziká by mohli vzniknúť ak by sa správcovia webových aplikácii nedostatočne starali o ich informačnú bezpečnosť a nerobili by pravidelné kontroly prostredníctvom rôznych skenovacích nástrojov a ďalšie protiopatrenia.

Tab. 8: Hrozby pre informačnú bezpečnosť (vlastné).

Hrozby pre informačnú bezpečnosť			
P.č.	**Názov hrozby**	**Scenár**	**Protiopatrenia**
Hrozby pre dôvernosť			
1.	Predstieranie identity používateľ	Predstieranie identity je možné použiť na obídenie autentizácie a s tým spojených bezpečnostných funkcií. Tento scenár môže viesť k problémom s dôvernosťou, vždy keď takéto predstieranie umožní prístup k citlivým informáciam.	Riadenie a audit logického prístupu- toto samotné riadenie nieje schopné rozlíšiť autorizovaný prístup od neautorizovaného, ale používanie mechanizmov riadenia sa predpokladá, že vie znížiť dopad. Kontrola a analýza auditných záznamov môže zistiť vyskytujúce sa neautorizované aktivity.
2.	Škodlivý kód	Škodlivý kód môže viesť ku strate dôvernosti, napr. skrz zachytenie a odkrytie hesiel.	Včasné oznámenie všetkých neobvyklých udalostí môže obmedziť škody v prípade útokov škodlivého kódu. Na detekovanie pokusov o získanie prístupu do systému alebo siete možno použiť prostriedky na monitorovanie prístupu.
3.	Zlyhanie softvéru	Zlyhanie softvéru môže ohroziť dôvernosť, ak daný softvér chráni dôvernosť, môže ohroziť dostupnosť.	Každý kto si všimne nefunkčnosť/poruchu softvéru by to mal oznámiť zodpovednej osobe, aby sa mohli čo najskôr vykonať potrebné kroky. Niektorým zlyhaniam sa dá predísť kompletným testovaním softvéru pred jeho používaním, iným zlyhaniam vieme predísť prostredníctvom riadenia zmien softvéru.
Hrozby pre dostupnosť			
4.	Deštruktívny útok	Informácie môžu byť zničené deštruktývnimi útokmi.	Správca vrátane všetkých zamestnancov by si mali byť vedomí následkov úmyselného či neúmyselného zničenia informácií. Všetky médiá by mali byť chránené pred neautorizovaným prístupom, použitím osobnej zodpovednosti za všetky médiá. Dôležité je vytváranie záloh všetkých dôležitých súborov, dát.
5.	Zlyhanie komunikačných prostriedkov a služieb	Zlyhanie komunikačných služieb ohrozuje dostupnosť informácií prenášaných prostredníctvom týchto služieb	Implementácia zdvojených komponentov komunikačných služieb môže byť využitá na zníženie pravdepodobnosti zlyhania komunikačných služieb. V závislosti na prijateľnom trvaní výpadku možno použiť záložné prostriedky.
6.	Technické zlyhania	Technické zlyhania môžu spôsobiť nedostupnosť informácií, ktoré sú v týchto sieťach uložené a spracovávané.	Prevádzkové procedúry, systémové plánovanie a správna konfigurácia siete by sa mali používať na minimalizáciu rizík technických zlyhaní.
7.	Neautorizovaný prístup k počítačom, dátam, službám a aplikáciám	Neautorizovaný prístup k počítačom, dátam, službám a aplikáciám môže byť hrozbou pre dostupnosť týchto informácií, ak je možné neautorizované zničenie.	Preverovanie a analyzovanie auditných záznamov môže zistiť neautorizované aktivity vykonávané ľuďmi s prístupovými právami do systému. Kvôli sťaženiu neautorizovanému prístupu by sa malo aplikovať oddelovanie sietí.

8.	Používanie neautorizovaných programov a dát	Používanie neautorizovaných programov a dát ohrozuje dostupnosť informácií uložených a spracovávaných na danom systéme, ak sú tieto programy a dáta používané na mazanie / odstraňovanie informácií, alebo ak obsahujú zlomyselný kód(môže byť vložený v hrách a rôznych aplikáciách sťahovaných z internetu)	Všetci zamestnanci by si mali byť vedomí, že by nemali implementovať žiaden softvér bez autorizácie/schválenia bezpečnostného manažér IT alebo inej osoby zodpovednej za bezpečnosť daného systému. Zálohy by sa mali používať na obnovenie akýchkoľvek informácií, služieb a prostriedkov, kt. boli poškodené alebo stratené. Riadenie logického prístupu by malo zaistiť, aby len autorizované osoby mohli použiť softvér na spracovávanie a mazanie dát. V neposlednom rade by sa malo klásť dôraz na to, aby všetky programy a dáta boli pred použitím preverené z hľadiska možnej prítomnosti škodlivého kódu.
9.	Užívateľská chyba	Chyby užívateľov môžu ohroziť dostupnosť informácií.	Všetci užívatelia by mali byť riadne informovaní / zaškolení z hľadiska predchádzania užívateľským chybám pri spracovávaní informácií. Vzdelávanie by malo zahŕňať aj zaškolenie v definovaných procedúrach pre špecifické činnosti, ako sú prevádzkové alebo bezpečnostné procedúry. Predošlá generácia záloh môže byť použitá na obnovu informácií, kt. boli zničené následkom užívateľských chýb.
Hrozby pre integritu			
10.	Technické zlyhanie	Technické zlyhanie v sieti môže yničiť integritu akýchkoľvek informácií, kt. sú uložené alebo spracovávané v tejto sieti.	Aby sme sa vyhli zlyhaniu IS alebo siete, tak by sme mali využívať manažment konfigurácií a zmien, rovnako ako kapacitný manažment. Na zaistenie bezproblémového chodu systému/siete sa ako opatrenie odporúča používať aktuálnu dokumentáciu a pravidelnú údržbu zariadení. Zálohy by sa mali používať na obnovu akýchkoľvek informácií, ktoré boli poškodené.
11.	Neautorizovaný prístup k počítačom, dátam, službám a aplikáciám	Neautorizovaný prístup k počítačom, dátam, službám a aplikáciám môže byť hrozbou pre integritu týchto informácií, ak je možná neautorizovaná modifikácia.	Preverovanie a analyzovanie log záznamov môžu zistiť neautorizované aktivity vykonávané ľuďmi s prístupovými právami do systému. Na ochranu integrity informácií, ktoré sú uložené alebo prenášané možno použiť kryptografické prostriedky.
12.	Používanie neautorizovaných programov a dát	Používanie neautorizovaných programov a dát ohrozuje integritu informácií uložených a spracovaných v príslušnom IS.	Všetci zamestnanci by si mali byť vedomí, že by nemali inštalovať a používať žiaden softvér bez povolení od osoby zodpovednej za bezpečnosť. Kontrolovanie a analyzovanie logov môžu zistiť neautorizované aktivity. Všetky programy a dáta by mali byť preverené z hľadiska prítomnosti škodlivého kódu.

2.7 Ako postupujem pri SQL injection útoku

2.7.1 Prieskum webovej aplikácie

Prvou fázou SQL injection útoku je prieskum aplikácie. Útočník (ja) v nej preskúmavam jej prezentačnú vrstvu – zameriam sa na HTML formuláre, ktoré obsahujú prvky na vstupy dát. V rámci tejto fáze si na každej stránke testovacieho subjektu skúsim vytvoriť užívateľský účet alebo si skúsim k nejakému dohodnúť prístup za účelmi testovania, vďaka čomu budem schopná získať o aplikácií viac potrebných informácií a vytvoriť si procesový diagram prihlasovania užívateľa. Pri prieskume webovej aplikácie mi vo veľkej miere pomôže prehliadač. Skopírujem si do Google vyhľadávača link stránky na ktorú sa zameriava, pripíšem k nej „/php?id=“ a začnem vyhľadávať odkaz v ktorom budem napríklad vidieť konkrétne id užívateľa. Môže sa mi veľmi ľahko stať, že mi toto vyhľadávanie nevyjde na prvý raz a budem musieť odkaz prepisovať na základe informácii, ktoré už o aplikácii mám a toho aké sa snažím nájsť. Mojím cieľom je nájsť také vstupy, kt. obsah bude následne použitý v SQL dotaze a teda potencionálne náchylný na SQL injection útok. V tejto fáze pre zefektívnenie svojej práce budem používať sqlmap pretože mi značne pomôže a zautomatizuje hľadanie jednotlivých zraniteľností (12, vlastné).

2.7.1.1 Hlavné zdroje pri zbieraní informácií:

- komunikácia so zadávateľom úlohy na testovanie konkrétnej aplikácie,
- dokumentácia k aplikácii (pri testovacích subjektoch, kde ju mám k dispozícii),
- vyhľadávanie si histórie k realizovaným útokom na aplikáciu (hackerone.com, github.com),
- prieskum aplikácie z užívateľského účtu,
- prieskum aplikácie prostredníctvom Google vyhľadávača (Google hacking),
- prieskum testovacieho subjektu prostredníctvom sqlmap.

2.7.2 Testovanie náchylnosti na jednotlivé chyby

Druhou fázou je testovanie náchylností webových aplikácií na možnosť SQLi. Prostredníctvom skenera (sqlmap) by som mala byť schopná objaviť niekoľko kategórii bežných slabín webovej aplikácie. Výber skenovacieho nástroja je popísaný v kapitole: „2.3 Predstavenie a analýza vybraných nástrojov na skenovanie webu“, táto časť je zameriavaná len na zisťovanie vstupov náchylných na SQLi, takže tu budem používať len techniky, kt. nemenia dáta a zároveň mi sú schopné poskytnúť informácie o zraniteľnostiach databázy (12,2,).

2.7.3 Samotný SQLi útok

V tretej fáze mám k dispozícii výstup z druhej, kt. pozostáva zo zoznamu vstupov, kt. sú náchylné na SQLi. Na základe zámeru útoku si potom zvolím konkrétnu metódu SQLi a aplikujeme ju prostredníctvom zvoleného nástroja, ktorým je sqlmap. V tejto časti sa mi môže veľmi ľahko stať, že nenájdem žiadnu slabinu, čo môže znamenať dve veci, buď by som mala zvážiť prekonfigurovanie svojho požiadavku, alebo sa tam skutočne nemusí nachádzať SQLi slabina.

2.8 Najčastejšie vyskytované metódy SQL injection

V rámci rozanalyzovania spôsobov obrany proti SQLi sa budem opierať o informácie vyplývajúce z grafu v kapitole 2.3.2 a o kapitolu 1.7. Z tejto štatistky zahrnutej v grafe mi ako najčastejšie metódy SQLi útokov vychádza týchto päť:

- Blind SQLi – testovanie odozvy v odpovedi (42%)
- Pokus o zistenie verzie databázy (29%)
- Pokus o získanie hesla (22%)
- Sondovanie (4%)
- Sondovanie skenovacieho nástroja za účelom hľadania slabín (3%) – táto zložka štatistiky spadá pod etický hacking (whitehat) a z toho dôvodu ju v tejto časti nebudem analyzovať.

V základe sa proti týmto metódam je možné brániť tak, že keď už pri vytváraní kódu sa vývojár bude držať pravidiel, ktoré sú rozobrané v kapitole prehlaď pravidiel pre vytváranie bezpečného kódu od autora na kt. je odkaz pod číslom 4. Z týchto informácii bude vychádzané pri tvorbe návrhu bezpečnostného odporúčania pri obrane proti SQLi metódam.

3 VLASTNÝ NÁVRH RIEŠENIA

Hlavnou témou tejto diplomovej práce je vytvorenie praktickej ukážky používania metód SQL injection, oboznámenie sa s technikami obrany proti SQL injection a na základe toho vytvorenie návrhu zavedenia obrany proti nim. Prvá časť návrhu riešenia sa venuje nastaveniu prostredia. Nasledujúca časť je venovaná samotnému testovaniu webových aplikácií zvolených testovacích subjektov, toto testovanie je zhodnotené v ďalšej kapitole práce a na základe neho je vytvorený návrh bezpečnostných opatrení.

3.1 Nastavenie prostredia

V rámci penetračného testovania budú nakonfigurované dve rôzne prostredia, prvé s Windows 10 a druhé s Kali Linuxom, kt. je prispôsobený na penetračné testovanie. Zmyslom tejto časti je si vytvoriť taký základový systém o kt. sa vie že bude v testoch konzistentný.

Predtým ako započne prevádzkovanie virtuálneho stroja a inštalácia nástrojov potrebných k penetračným testom, tak je dôležité sa presvedčiť, že je k dispozícii taký počítač, kt. zvládne všetko potrebné. V tejto časti je dôležité postupovať tak aby bolo možné udržať počítač bez infekcie škodlivým softwarom.

Tab. 9: Požiadavky na hardware (vlastné).

Hardware	Požiadavky na hardware	Hardware k dispozícii	Splnenie požiadavok
RAM	8 GB a viac	8 GB	☑
Velkosť disku	500 GB a viac (SSD je plus)	500 GB SSD	☑
Procesor	i7 Intel Quad Core	Intel Core i7-7700HQ	☑
	WMware Workstations/Fusion/Player/Virtual Box	Virtual box	☑

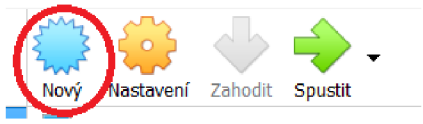
3.1.1.1 Inštalácia Kali Linuxu:

V predchádzajúcej podkapitole bolo overené či notebook určený na testovanie splňuje minimálne požiadavky na hardware pre penetračné testovanie, čo splňuje. Táto kapitola sa venuje jednotlivým krokom ako postupovať pri inštalácii Kali Linuxu na Virtual Box.

Distribúciu Kali Linux som sťahovala z tejto adresy : https://www.kali.org/downloads/

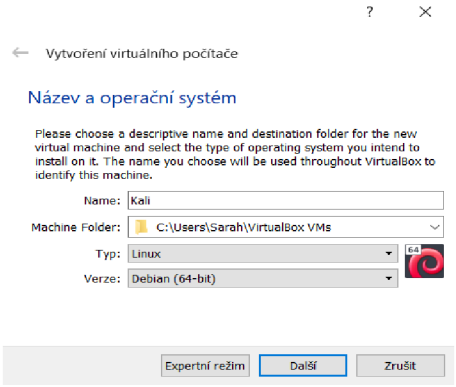
Hneď ako bude stiahnutý ISO súbor Kali Linux, tak je potrebné otvoriť Virtual Box a pokračovať podľa nasledujúcich bodov:

- Bola vybraná možnosť v červenom kruhu.



Obr. 26: Tvorba Linuxu vo VB (vlastné).

- Po vyskočení okna, aké je možno vidieť vpravo na obr. 23, boli všetky kolónky vyplnené tak, ako sú na tom istom obrázku.



Obr. 27: Tvorba virtuálneho pc (vlastné).

- Bola zvolená možnosť ďalší a vybratá veľkosť pamäte, odporúčané minimum sú 4 GB, v tomto prípade polovica z celkovej pamäte, čo na toto konkrétne testovanie bude stačiť.
- V nasledujúcom kroku bola zvolená možnosť Vytvoriť virtuálny hard disk a následne potvrdená. Výber typu – VDI (VirtualBox Disk Image).
- Miesto na fyzickom hard disku bolo dynamicky alokované.
- V ďalšom kroku bolo vyhľadané umiestnenie inštalačného súboru Kali Linuxu a bol vytvorený virtuálny počítač, čo spustilo inštaláciu.
- Ďalej je nutné vybrať možnosť inštalovať (install) a prejsť do kroku, kde sa vyberá jazyk, v akom to chce inštalujúca osoba mať a následne to potvrdí.
- Meno počítača bolo nechané prednastavené ako aj názov domény, kt. vyhodilo v nasledovnom kroku.
- Nastavenie užívateľa a hesla – zvoliť heslo a zopakovať zvolené heslo (ďalej).
- Rozdelenie disku – tu sa volí spôsob rozdelenia -> Asistovane – použiť celý disk.
- Rozdelenie disku (vybrať disk pre rozdelenie).
- Rozdelenie disku (Spôsob rozdelenia) -> výber prvej možnosť (Všetky súbory v jednej oblasti), potvrdenie zmeny a následná voľba možnosti (Ukončiť rozdeľovanie a zapísať zmeny na disk) – potvrdenie.
- Inštaluje sa systém.

3.2 Realizácia testovania

V tejto časti záverečnej práce je v prvom rade otestované, či bolo dobre nastavené prostredie sqlmap. Toto overenie je vykonané tak , že je útočené na testový web Acunetix. Pokiaľ jednotlivé príkazy budú fungovať a bude možné získať užívateľské meno s heslom, tak z toho vyplýva, že je všetko nastavené správne a testovanie ostatných vybraných testovacích Subjektov B,C,D môže začať.

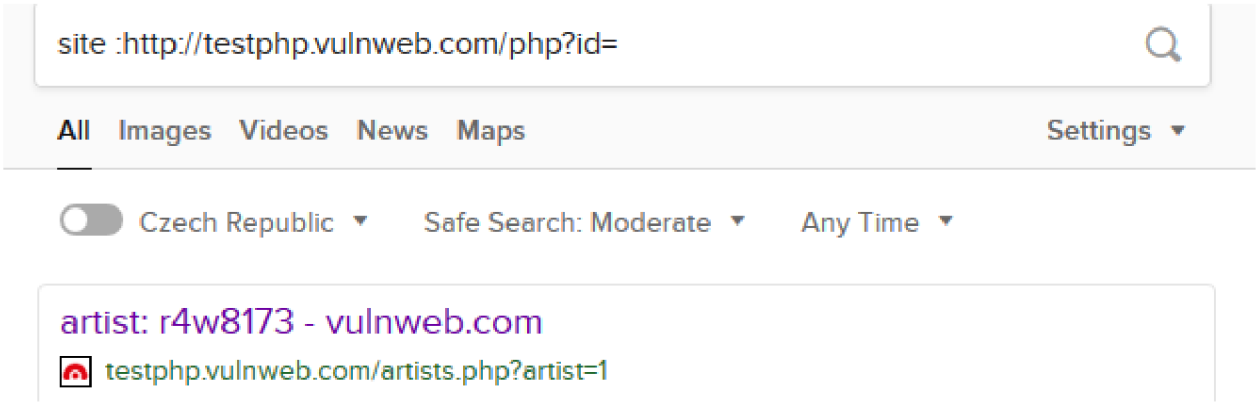
3.2.1 Testovanie webovej aplikácie Subjekt A

Pri testovaní webu na začiatok výrazne pri práci pomôže technika nazývaná Google hacking. Odkaz vybranej webovej stránky na testovanie(Subjekt A, B,C) je skopírovaná a vložený do Google vyhľadávača. Bude to vypadať nasledovne: site: názov stránky/php?id=

site :http://testphp.vulnweb.com/php?id=

Obr. 28:Upravený odkaz s cieľom hľadať ID (vlastné).

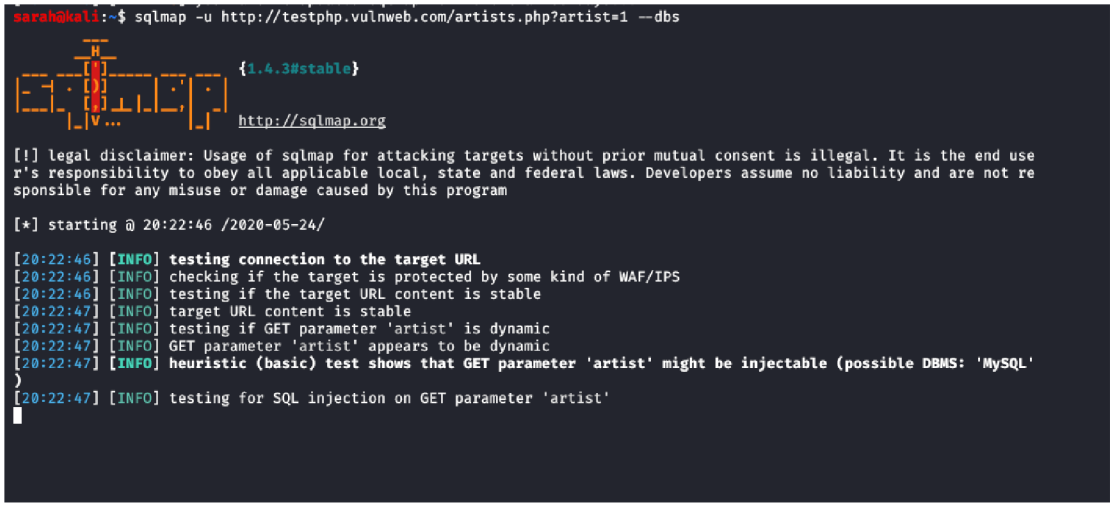
Po zadaní údajov upravených ako z obr.22 do prehliadača by mi to malo pomôcť nájsť link, ktorý mi pomôže při útoku na testovací Subjekt A, B, C.



Obr. 29: Hľadanie vhodného odkazu prostredníctvom Google vyhľadávača (vlastné).

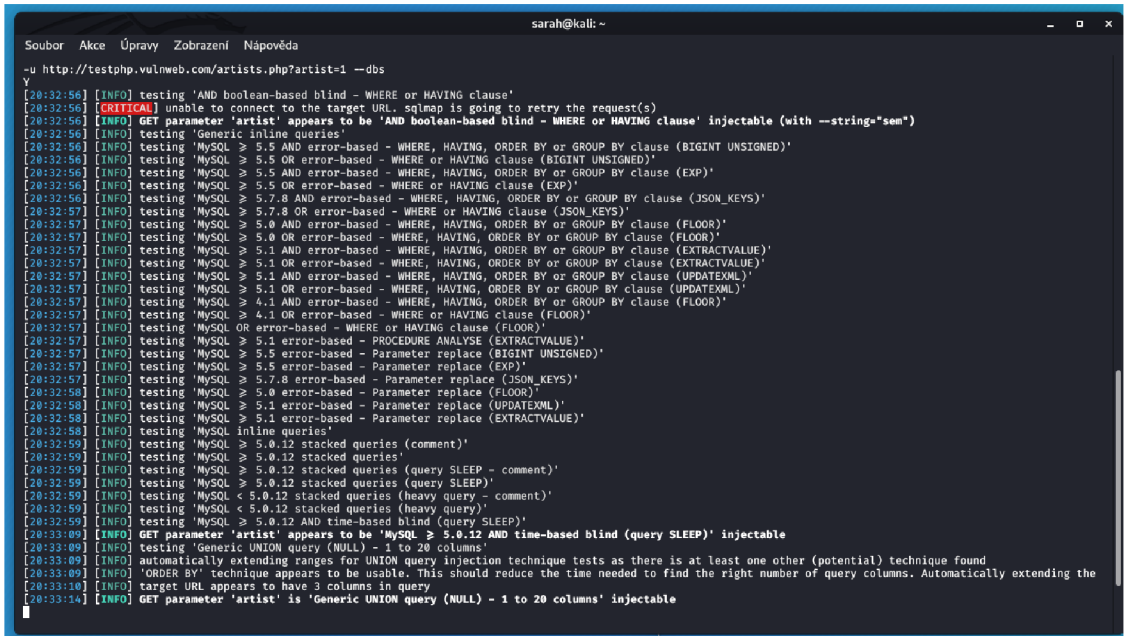
Aby použité testovacie subjekty mohli naďalej ostať pseudonymizované, tak pri realizácii tohto kroku bola použitá testovacia stránka od Acunetix, kde prostredníctvom Googlu bolo možné nájsť odkaz, ktorý je schopný pomôcť pri realizácii útoku.

V ďalšom kroku bol použitý príkazový riadok v ktorom bol otvorený nástroj, kt. sa volá sqlmap a tam bol vložený skopírovaný odkaz.



Obr. 30:Sqlmap- hľadanie databázy (vlastné).

Ako je vidieť na obr.27, tak bolo možné nájsť MYSQL databázu.



Obr. 31:Testovanie databáze (vlastné).
V nasledujúcom kroku bolo spustené testovanie pre MYSQL databázu.

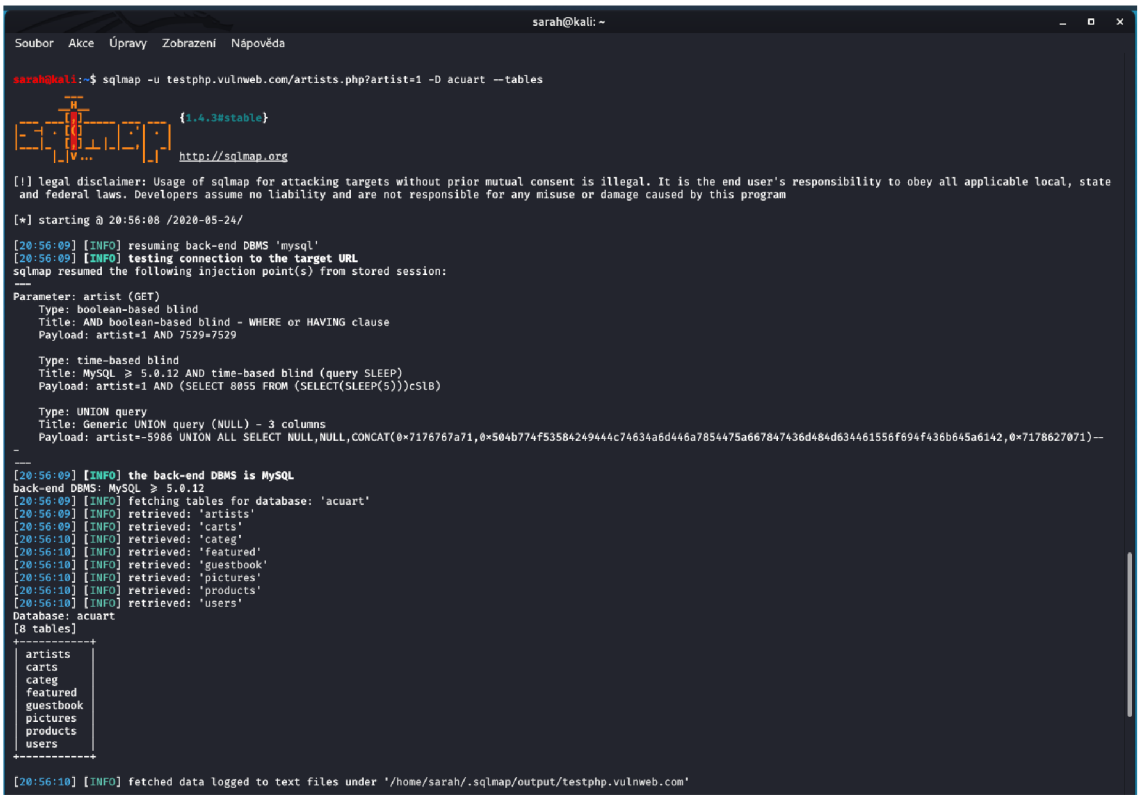
Obr. 32: Zraniteľnosť v databáze (vlastné).

Bola nájdená slabina databázy, kt. je vidieť na obr.29, vďaka tejto slabine by malo byť realizovateľné nájsť užívateľské meno a heslo. Bude sa uskutočňovať pokus o zmenu povolenia databázy prostredníctvom SQLi, aby som bolo možné získať užívateľské dáta.

Obr. 33: Zistenie konkrétneho názvu v databáze (vlastné).

Bolo zistené, že databáza sa volá acuart a v ďalšom kroku sa bude realizovať pokus o zistenie počtu tabuliek v tejto databáze, čo sa uskutoční prostredníctvom tohto príkazu :

Obr. 34: Zisťovanie počtu a názvov stĺpcov v databáze acuart (vlastné).

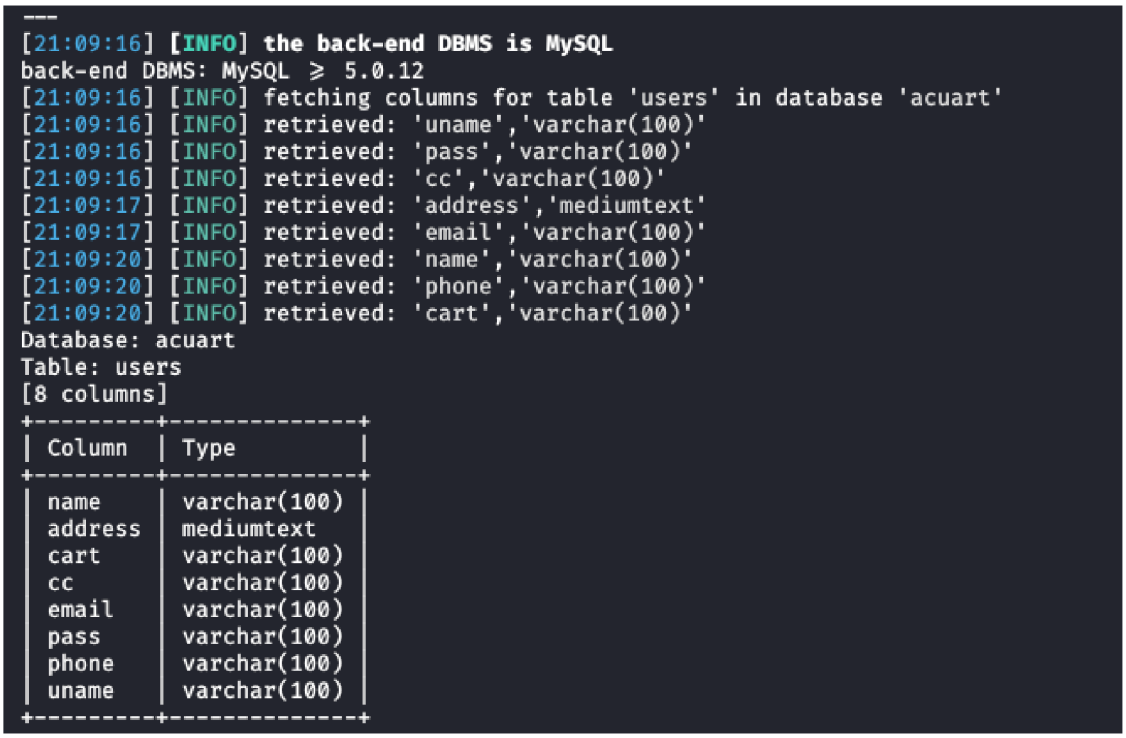


Obr. 35: Úspešné nájdenie výpisu tabuliek v databáze (vlastné).

Bolo zistené, že databáza acuart obsahuje 8 tabuliek, kt. názvy je vidieť na obr. 32. Vzhľadom k tomu, že cieľom je nájsť informácie o užívateľských menách a heslách, tak na preskúmavanie bude zvolená tabuľka users a vykoná sa to prostredníctvom nasledujúceho príkazu, kt, je vidieť na obr. 33:

```
sarah@kali:~$ sqlmap -u testphp.vulnweb.com/artists.php?artist=1 -D acuart -T users --columns
```

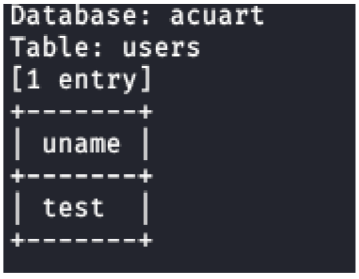
Obr. 36: Príkaz na získanie stĺpcov z tabuľky (vlastné).



Obr. 37: Nájdený výpis stĺpcov y tabuľky users (vlastné).

Sqlmap po zadaní príkazu z obr. 33 sprístupnila výstup na obr.35, vďaka čomu sú teraz známe názvy stĺpcov v tabuľke users.

Obr. 38: Príkaz na zistenie konkrétneho užívateľského mena (vlastné).



Obr. 39: Získanie užívateľského mena (vlastné).

Ako výstup bolo nakoniec získané užívateľské meno: test. V ďalšom kroku bude realizovaný pokus o nadobudnutie k nemu príslušného hesla a uskutoční sa to prostredníctvom príkazu z obr. 37:

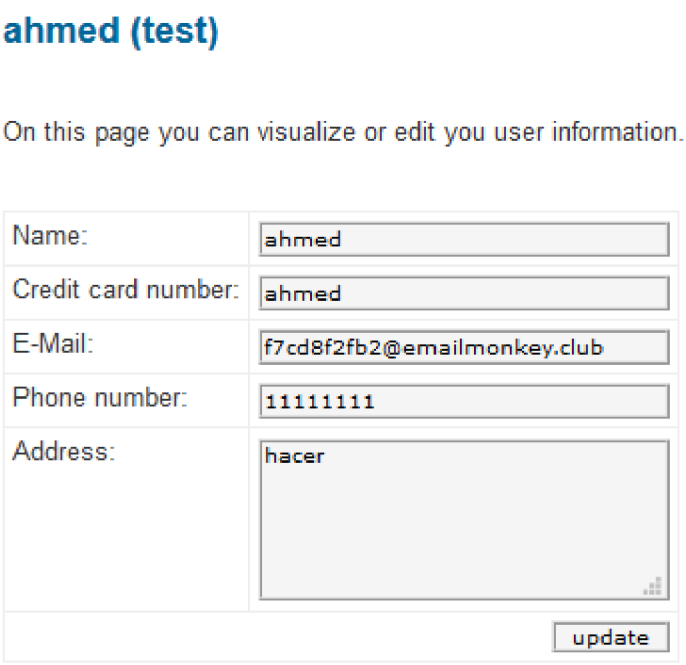
```
sarah@kali:~$ sqlmap -u testphp.vulnweb.com/artists.php?artist=1 -D acuart -T users -C pass --dump
```

Obr. 40: Príkaz na získanie užívateľského hesla (vlastné).

```
Database: acuart
Table: users
[1 entry]
+------+
| pass |
+------+
| test |
+------+
```

Obr. 41: Získanie užívateľského hesla (vlastné).

Posledným príkazom bolo získané aj heslo: test. Vzhľadom k tomu, že prostredníctvom sqlmap bolo nájdené užívateľské meno a aj k nemu príslušné heslo v databáze, tak bude nasledovať pokus o prihlásenie so získanými užívateľskými údajmi.



Obr. 42: Užívateľské informácie (17)

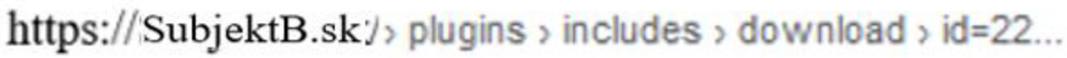
Vzhľadom k tomu, že sa prostredníctvom Blind SQLi podarilo získať užívateľské meno, heslo a pri žiadnom príkaze nevyhodilo chybu typu, že príkaz neexistuje, tak to vyhodnocujem tak, že je prostredie sqlmap nainštalované a nastavené správne.

3.2.2 Testovanie Subjektu B

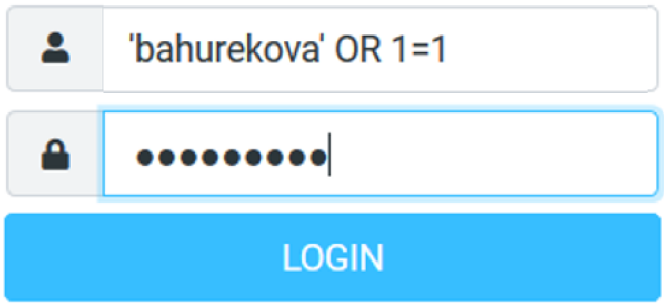
Pri testovaní tohto webu napriek tomu, že je vo WordPresse bude skúsené šťastie s Google hackingom a bude sa realizovať vyhľadávanie nejakého užitočného odkazu. Skopíruje sa odkaz webovej stránky Subjektu A, ktorá bola vybratá na testovanie a vloží sa do Google vyhľadávača. Bude to vypadať nasledovne: site:

Obr. 43:Upravený odkaz Subjektu B (vlastné).

V celej časti testovania bude v práci cenzurovaný názov tejto stránky tak, že bude nahrádzaný ako „SubjektB“. Na prvý pohľad to pôsobí tak, že webová aplikácia má celkom ošetrené vyhľadávanie informácií prostredníctvom Googlu za účelom útoku, respektíve pri takto sformulovanom odkaze to vyhľadávalo len ID dokumentov na siahnutie, ktoré sú na stránkach školy verejne dostupné, čo vypadalo ako je vidieť na obr. 41.



Obr. 44:Odkaz Subjektu B nájdený na Google s č. ID(vlastné).



Obr. 45: Prihlasovanie sa na účet školy (vlastné).

Na obr. 42 bola skúšaná metóda útoku, kt. je v práci rozpísanú pod obr.6, boli testované rôzne variácie tohto spôsobu, ale ani raz nebolo prihlásenie na užívateľský účet úspešné.

Po niekoľkých prepisovaniach odkazu a dlhého hľadania, ktoré neviedlo takmer k žiadnym informáciám bolo realizované prihlásenie pod užívateľským menom, kt. mi bolo sprístupnené organizáciou za účelom zbierania informácií. Vďaka ďalšej analýze stránky prišiel nápad skúsiť získať administrátorské ID a upraviť odkaz nasledovne:

https://SubjektB.sk/wp-login.php?id=

Obr. 46:Upravený odkaz s účelom hľadania ID (vlastné).

Keď bol tento novo upravený odkaz zadaný do vyhľadávača, tak boli nájdené nejaké 4 ID. Následne prostredníctvom sqlmap bude otestované, či tieto získané informácie budú nejak prospešné pri útoku.

Obr. 47: Testovanie pripojenia na zvolenú URL s ID administrátora zadaným vo vstupe (vlastné).

Ako je vidieť na obr. 44, tak v tomto jednom prípade bezpečnostné riešenie WordPressu nezastavilo útok priamo. Je to z toho dôvodu, že sa odkaz podarilo upraviť do formátu

v akom by bol keby sa administrátor s týmto ID prihlasoval normálne. Avšak túto časť má WordPress ošetrenú tak, že chce nastaviť svoje vlastné cookie. Zrovna v tomto prípade by sa možno dalo dostať prostredníctvom príkazu alebo programu ďalej, ale toto prihlasovanie spadá pod samotný WordPress a nie pod stránku školy, ktorá mi dala povolenie na testovanie, takže týmto spôsobom bohužiaľ v testovaní nie je možné pokračovať z právneho hľadiska.

Obr. 48: Testovanie pripojenia na zvolenú URL s neupraveným odkazom webovej stránky (vlastné).

V rámci všetkých nasledujúcich príkazov, ktoré boli vyskúšané bola odpoveď prostredníctvom sqlmap nasledovná. Stálo v nej, že je webová aplikácia chránená nejakým druhom WAF/IPS, čo je vlastne bezpečnostné riešenie, ktoré chráni spojenie medzi klientom a serverom. Monitoruje prenos z a na webovú aplikáciu.

3.2.3 Testovanie Subjektu C

V tejto kapitole bude testovaná webová aplikácia, ktorá v spolupráci s hackerone vypísala odmenu za hľadanie bezpečnostných bugov. Na toto testovanie som obdržala pozvánku, ktorá mi dáva právomoc niekoľko dní útočiť na ich webovú aplikáciu za účelom hľadania Blind SQLi.

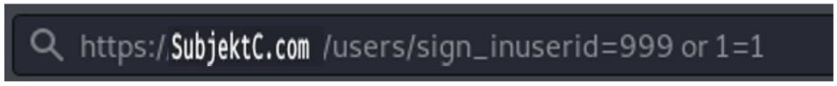
Pri Subjekte C v rámci zbierania informácii boli použité nasledovné metódy :

- Google Hacking
- Dokumentácia z Github stránok
- Nasledovné skenovacie nástroje:
    - WAFW00F
    - sqlmap
    - nikto
    - sqlmap

Pred začiatkom vyhľadávania bola nastavená vpn v príkazovom riadku Kali Linux. Nainštalovaná bola podľa návodu, kt. je na stránkach nordvpn a na pripojenie boli použité tieto príkazy: „nordvpn login“ po zadaní prihlasovacích údajov nasledoval príkaz : „nodvpn connect“.

Obr. 49: Pripojenie na nordvpn (vlastné).

Prostredníctvom Google hackingu boli vyhľadávané nejaké užitočné odkazy, ktoré by mohli poskytnúť informácie o bezpečnostných dierach, zdrojový kód, ktorý používa stránka. Na obrázkoch nižšie je niekoľko úprav odkazu podľa, kt. boli hľadané bezpečnostné diery a na základe kt. sa aj podarilo zozbierať pár informácií.



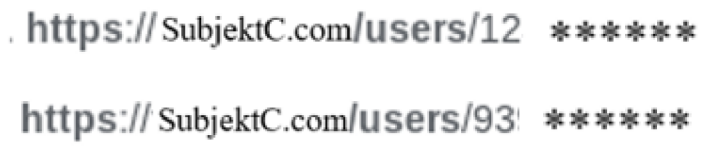
Obr. 50: Upravený odkaz na vyhľadávanie ID 1 (vlastné).

Obr. 51: Upravený odkaz na vyhľadávanie ID 2 (vlastné).

Obr. 52: Upravený odkaz na vyhľadávanie ID 3 (vlastné).

Obr. 53: Odkaz na zdrojový kód stránky (vlastné).

Obr. 54: Odkaz obsahujúci ID dvoch užívateľov (vlastné).



Obr. 55: Odkazy obsahujúce ID užívateľov (vlastné).

Zbieranie informácií prostredníctvom metódy Google hacking poskytlo tieto informácie: zdrojový kód stránky, ID štyroch užívateľov webovej stránky.

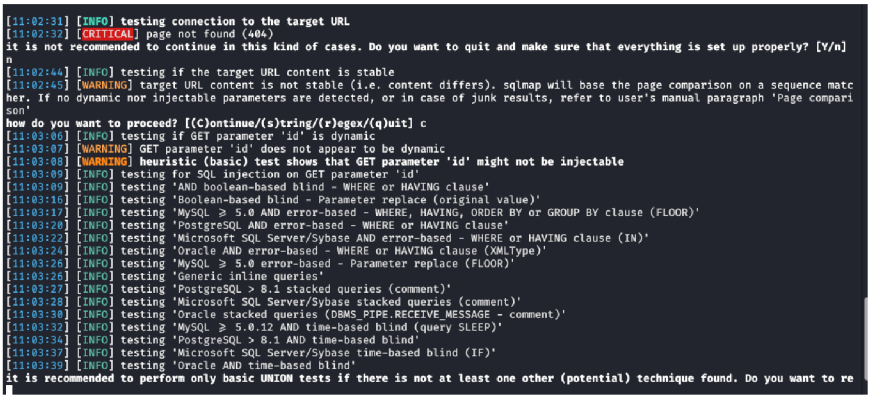
V ďalšom kroku bolo overované pomocou nástroja WAFW00F či je webová aplikácia chránená nejakým druhom WAF/IPS. Spomínaný nástroj je súčasťou prostredia Kali

Linux. Na obrázku 51 je do príkazového riadku zadané wafw00f na spustenie tohto nástroja a pripísaný odkaz cielenej webovej stránky.

Obr. 56: Wafw00f zisťuje, či webová aplikácia nie je chránená vlastním firewallom (vlastné).

Skenovací nástroj na obr. 51 na základe analýzy kódu odpovede, čo je za normálnych okolností „200“ určil, že webová aplikácie je chránená nejakým druhom firewallu pretože kód modifikovanej odpovede je „403“.

Informácií bolo zozbieraných na útok dostatok a z toho dôvodu bol použitý nástroj sqlmap. Bola použitá URL s ID nájdená prostredníctvom Google, zadaný príkaz vypadal v tomto tvare: sqlmap – u „(skopírovaná URL, kt. obsahuje ID)“



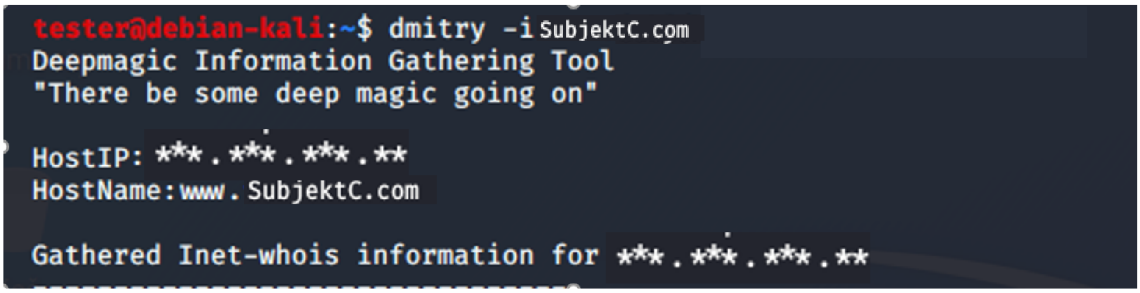
Obr. 57: Kontrola vstupných parametrov(vlastné).

Je to jeden z najjednoduchších príkazov, kt. sa používa v sqlmap, prostredníctvom tohto príkazu boli kontrolované vstupné parametre, aby sa zistilo, či nie sú náchylné na SQLi. Funguje to tak, že sa odosielajú rôzne druhy užitočných dát ohľadom vstrekovania a kontroluje sa výstup. V rámci tohto procesu je niekedy možné identifikovať vzdialený OS systém, názov databáze a verziu. Počas testovania zvoleného subjektu nebolo možné tieto informácie zistiť, sqlmap podal informáciu o tom, že stránka nebola nájdená,

prebehla kontrola stability obsahu URL, kt. bola následne vyhodnotená ako nestabilná. Boli otestované jednotlivé parametre, kt. sqlmap vyhodnotila tak, že nenašla žiaden zraniteľný. Ďalej bolo doporučené urobiť základné UNION testy a po vykonaní všetkých testov sqlmap vypísal hlášku z obr. 58, kde všetky otestované parametre vyhodnotil ako pravdepodobne nezraniteľné na SQLi.

Obr. 58: Hláška od sqlmap ohľadom parametrov (16)

V ďalších krokoch boli vyskúšané všetky doporučené príkazy, kt. boli vo formáte aký bol ukázaný na začiatku testovania tohto subjektu v sqlmap, len pri príkazoch level, thread a pod sa to musela určiť hodnota a napísať príkaz v tvare : sqlmap -u (URL) --level=(číselná hodnota napr. 5). vykazovali podobné výsledky z kt. nešlo dostať moc užitočných informácií na pomoc pri realizácií SQLi. Na základe toho bolo rozhodnuté a ďalšom zbieraní informácií prostredníctvom nástroja nikto, predtým však je potrebné zistiť IP cielenej webovej aplikácie. Lokalizácia IP adresy bola zistená prostredníctvom nasledujúceho príkazu zadaného do príkazového riadka v Kali Linux.

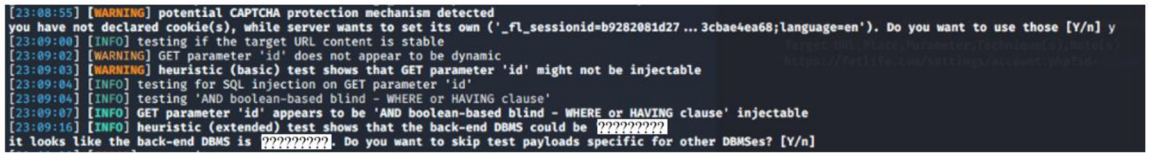


Obr. 59:Príkaz na lokalizovanie IP (vlastné)

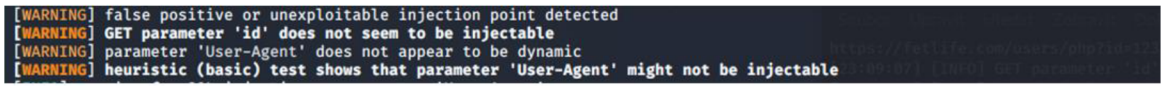
Vo výsledku je host IP adresa z dôvodu anonymizácie tohto údaju cenzurovaná hviezdičkami. Zistená adresa bola použitá ako vstupný parameter do príkazu v nástroji nikto.

Obr. 60: Príkaz v nástroji nikto na vyhľadávanie informácií (vlastné).

Získavanie IP adresy a následné spúšťanie nástroja nikto bolo naraz vykonávané na 2 zariadeniach a to z toho dôvodu, že na jednom bola zapnutá vpn a druhom bola vidieť moja skutočná IP adresa, kt. v tej dobe bola pridaná do whitelistu. Do nástroja bola zadaná host IP adresa a skenovanie portu 80. Nikto prešiel vyše 7000 položiek a ako výstup bolo niečo okolo 2000 položiek vrátane komentára. Niektoré z týchto údajov neposkytovali žiadnu užitočnú informáciu, avšak po prejdení všetkých sa podarilo niečo, čo by mohlo byť pri útoku prospešné nájsť. ??????????



Obr. 61: Z testu vyšiel back-end DBMS (vlastné).



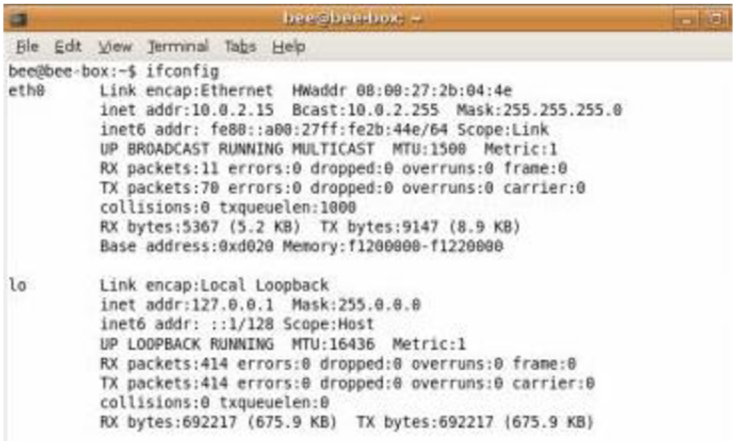
Obr. 62: Falošne nájdená SQLi zraniteľnosť (vlastné).

Na základe detailnejšieho testovania bol pozitívny výsledok na slepú SQLi vyvrátený. Po úprave vstupov v príkaze s cestou k administrátorskej zložke, kt. bola nájdená skrz nástroj nikto boli vykonané príkazy „--crawl= 5,6“ , --level =5, -- data, --schema, --risk,.. Prostredníctvom týchto príkazov bola zo zariadenia, kt. bolo na white liste v administrátorskej zložke nájdená URL prostredníctvom ktorej sa vymazáva účet.

Vo výstupe z nikto bolo niekoľko veľmi užitočných informácií, kt. už nebolo možné otestovať lebo niekedy pri začiatku ďalšieho testovania skončilo povolenie.

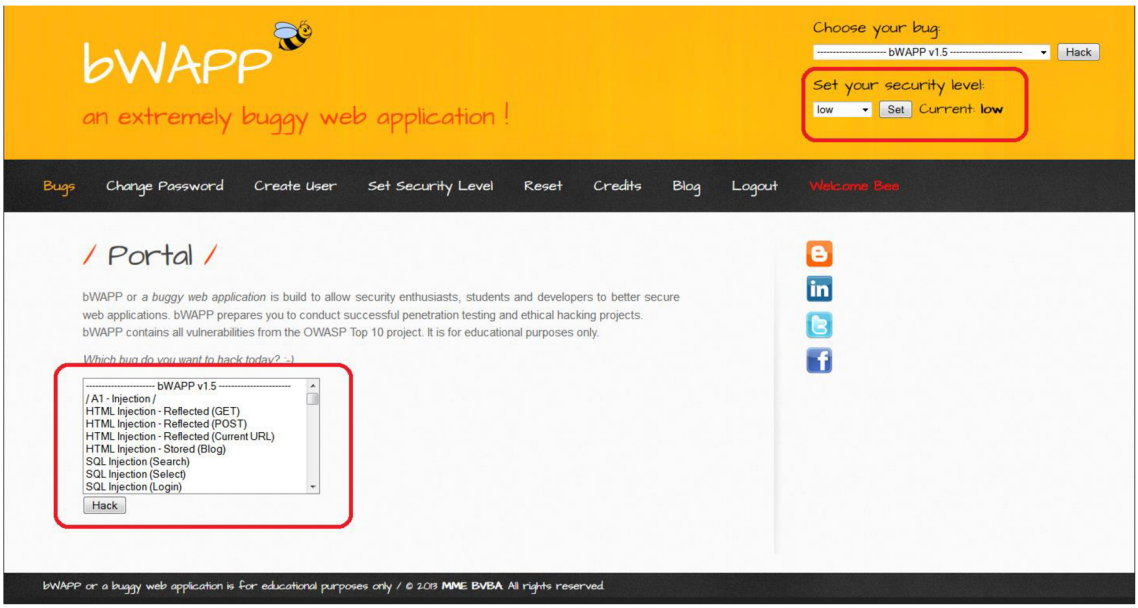
3.2.4 Testovanie Subjektu D

Subjekt D je webová aplikácia, ktorá bola vyvinutá na precvičovanie si SQLi metód v súlade so zákonom. Aplikácia je nainštalovanú do VB veľmi podobným spôsobom, ako prebiehala inštalácia Linuxu, pred nastavovaním samotnej aplikácie je potrebné v príkazovom riadku bee si skontrolovať nastavenia siete prostredníctvom príkazu ifconfig.

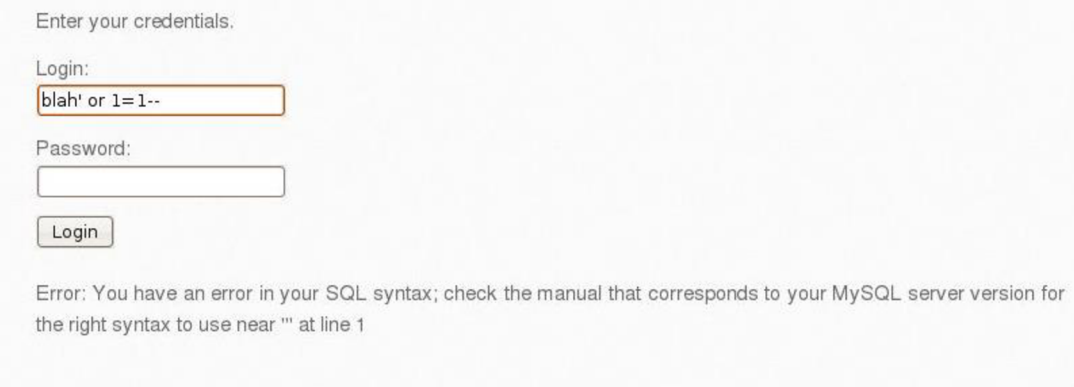


Obr. 63: Príkaz ipconfig (vlastné).

Než započne samotné testovanie, tak je potrebné v menu aplikácie nastaviť úroveň bezpečnosti a zraniteľnosť na akú sa chce útočník pri hackovaní zameriavať.



Obr. 64: bWAPP menu (24).



Na získavanie informácií o užívateľskom formuláre boli do vstupu vkladané základné SQLi operátori :

- '--
- ' or 'a'='a
- ' or 'a'='a'--
- ' or '1'='1
- ' or 1=1--

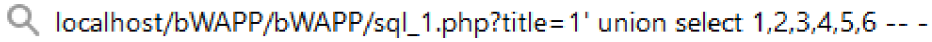
Obr. 65: Základné SQLi operátori (24).

Po zadávaní jednotlivých operátorov v prípadoch kedy nebolo možné sa skrz nich dostať do užívateľského účtu, tak server poskytoval chybné správy, kt. obsahovali užitočné informácie. Do účtu sa podarilo dostať, keď namiesto loginu boli dosadené tieto operátori ' or '1'='1 alebo ' or 'a'='a. V prípade kedy SQL dotaz bol injektovaný to vypadalo nasledovne:

SELECT * FROM table WHERE username='' or 'a'='a' AND password='bug'

Obr. 66: Príkaz na vstrekovanie užívateľa (vlastné).

Za predpokladu, že nie je užívateľské meno známe ale vlastné alebo na základe logu so zlými pokusmi o prihlásenie získané heslo zadáme do vstupu, tak je možné sa bez problémov prihlásiť.



Obr. 67: Úprava odkazu za pomoci union select (vlastné).

Na základe jednotlivých chybných hlášok bol odkaz upravený do tvaru, kt. je vidieť na obr. 65, táto úprava mala úspešné výsledky a sprístupnila tabuľku, kt. je vidieť na obr. 66.

Title	Release	Character	Genre	IMDb
2	3	5	4	Link

Obr. 68: Tabuľka, kt. vyšla ako výstup na základe zvoleného union select (vlastné)

Odkaz, kt. ukázal tabuľku zo systému bol skopírovaný do poznámkového bloku, kde vypadal tak ako je ho vidieť na obr. 67.

```
http://localhost/bWAPP/bWAPP/sqli_1.php?title=1%27%20union%20select%201,2,3,4,5,6--%20-
```

Obr. 69: Skopírovaný odkaz, kde bol použitý select union (vlastné).

Na základe výstupu z tabuľky obr. 66, kde charakter =5 bol prepísaný odkaz do nasledujúceho tvaru:

localhost/bWAPP/bWAPP/sql_1.php?title=1' union select 1,2,3,4,database(),6 -- -

Server na to reagoval tak, že poskytol z databázy informáciu o položke, kt. mala hodnotu bwapp. Takýmto spôsobom boli z databázy vybrané aj názvy tabuliek (table_name) dosadeným namiesto čísla 5 a dopísaním do odkazu: from information_schema.tables--.

Title	Release	Character	Genre	IMDb
2	3	CHARACTER_SETS	4	Link
2	3	COLLATIONS	4	Link
2	3	COLLATION_CHARACTER_SET_APPLICABILITY	4	Link
2	3	COLUMNS	4	Link
2	3	COLUMN_PRIVILEGES	4	Link
2	3	ENGINES	4	Link
2	3	EVENTS	4	Link
2	3	FILES	4	Link
2	3	GLOBAL_STATUS	4	Link
2	3	GLOBAL_VARIABLES	4	Link
2	3	KEY_COLUMN_USAGE	4	Link
2	3	OPTIMIZER_TRACE	4	Link

Obr. 70: Výstup z table_name (vlastné).

localhost/bWAPP/bWAPP/sql_1.php?title=1' union select 1,2,3,4,group_concat(table_name),6 from information_schema.tables--

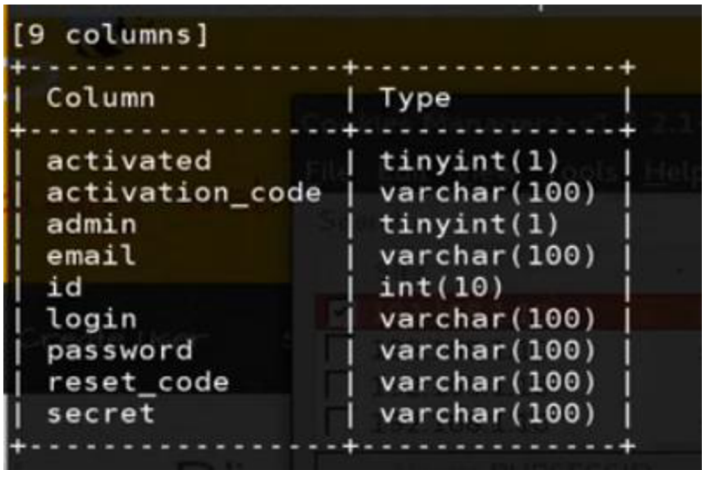
Obr. 71: Odkaz rozšírený o group_concat (vlastné).

Title	Release	Character	Genre	IMDb
2	3	blog,heroes,movies,users	4	Link

Obr. 72: Výstup na základe odkazu z obr. 68 (vlastné).

Obr. 73: Blind boolean SQLi (vlastné).

Po zadaní vstupu ako je to zobrazené na obr. 70 poslal server späť informačnú hlášku o tom, že film sa nachádza v databáze. Informácie kt. boli získané v rámci upravovania odkazu pre testovací Subjekt D by bolo možné získať aj prostredníctvom sqlmap, ako tomu bolo ukázané pri testovaní Subjektu A a z toho dôvodu sem tento typ postupu nebol doložený druhý raz. Pre predstavu toho aký by bol výstup z databázy bWAPP a tabuľky users je doložená tabuľka z sqlmap na obr. 72.



Obr. 74: Tabuľka users (vlastné).

3.3 Zhodnotenie testovania zvolených webou

Táto časť diplomovej práce sa venuje zhodnoteniu výsledkov jednotlivých testovacích subjektov.

3.3.1 Subjekt A

Ako prvý subjekt v rámci testovania SQLi metód bola zraniteľná webová aplikácia patriaca spoločnosti Acunetix. Tento web má účelne vytvorené slabiny aby si na ňom etický hackeri mohli zadarmo vyskúšať nastavenia svojich nástrojov, niektoré SQLi metódy, prípadne príkazy.

Na tejto stránke boli mnou uskutočnené tieto metódy vyhľadávania informácií:

- prieskum aplikácie z užívateľského účtu,
- prieskum aplikácie prostredníctvom Google Hackingu,
- prieskum testovacieho subjektu prostredníctvom sqlmap.

Vzhľadom k tomu, že je to stránka určená na testovanie, tak prostredníctvom Google vyhľadávača bolo možné nájsť odkaz aj s ID užívateľa veľmi rýchlo, čo pri skutočných webových stránkach nie je až tak jednoduché.

Získaný odkaz bol použitý do príkazu v sqlmap, vďaka čomu bola nájdená databáza. V ďalšom kroku bolo spustené testovanie, ktoré mi malo pomôcť nájsť zraniteľnosti. Nájdená zraniteľnosť bola voči Blind SQLi – testovanie odozvy v odpovedi, ktorá v rámci mesačnej štatistiky na najčastejšie realizované SQLi metódy bola vyhodnotená ako 1. a predstavovala 42% z analyzovaných útokov. Následne bol úspešne realizovaný útok s cieľom získať užívateľské meno a heslo. Prostredníctvom jednotlivých príkazov bol možný prístup k nižším vrstvám databázy, až z nej nakoniec bolo získané užívateľské meno a heslo, čo sa overilo prostredníctvom prihlásenia sa na užívateľský účet.

3.3.2 Subjekt B

Základná škola, Subjekt B má svoje webové stránky vytvorené vo WordPresse, takže ak je vynechaná táto časť bezpečnosti, kde by mali byť preškolení zamestnanci, študenti a podobne, tak sa o zabezpečenie a ochranu informácii webových stránok stará WordPress. Počas testovania sa po analýze stránok, informácií z Googlu nebolo možné prostredníctvom sqlmap dostať v rámci databázy školy v podstate nikam. Bohužiaľ nebolo možné sa dostať ani po krok, kde by bolo realizované pripojenia sa na databázu prostredníctvom nástroja. Sqlmap sprostredkovala informáciu o tom, že WordPress webovú aplikáciu chráni nejakým druhom WAF/IPS, toto bezpečnostné riešenie bolo schopné odraziť všetky moje útoky prostredníctvom sqlmap. Na stránkach WordPress je písané, že tento Firewall na ochranu webovej aplikácie obsahujú všetky verzie WordPress od verzie 4.6 a vyššie.

3.3.3 Subjekt C

Webová aplikácia Subjektu C je webová stránka, kt. spadá pod sociálne siete. Cieľom testovania bolo hľadať potenciálnu zraniteľnosť typu Blind SQLI. Počas testovania boli výrazne zaťažené útoky na užívateľský vstup skrz Captcha. Napriek niekoľkým zisteným užívateľským ID túto informáciu nešlo využiť na vstup do účtu. Neskôr bol skúšaný útok na tieto účty prostredníctvom sqlmap, kde proti realizácii tohto útoku sa realizovala WAF, Captcha, autentizačný token. Po obrovskom množstve rôznych úprav útoku bolo jasné, že má táto webová aplikácia ošetrený vstup zo strany serveru aj klienta. Na základe detailného skenovania odkazu skrz sqlmap so zameraním na ID klienta a vyhodnotení výstupov z tohto nástroja boli urobené závery, že v tejto oblasti nie je možné realizovať žiadnu SQLi. Vzhľadom k tomu, že ma server nezablokoval úplne a mala som možnosť aspoň skúšať rôzne príkazy aj keď výsledky z nich nepoukazovali na nijakú SQLi chybu. Po prihliadnutí na vyššie zmienené bolo usúdené, že je potrebné sa vrátiť zberbe informácií aby bol akýkoľvek pokus o útok aspoň potencionálne realizovateľný. K tomuto účelu bol zvolený nástroj nikto, kt. je podľa recenzií a mojich doterajších skúseností výborný na tento zámer. Pred jeho použitím bolo kľúčové zistiť HostIP, kt. bola zistená prostredníctvom príkazu dmitry -i www.SubjektC.com, zistená IP adresa sa skopírovala do nástroja nasledovne: nikto -h (vložená IP) -p 80. Skenovanie s nikto bolo zachytené nejakým skenerom webovej aplikácie, napriek tomu sa nástroju podarilo prejsť okolo7000 položiek a aspoň čiastočne vyhodnotiť okolo 2000. Vďaka týmto informáciám

bolo možné za pomoci sqlmap aspoň lustrovať v databáze po informáciách. Žiaľ k tomuto kroku som sa dostala až pár hodín pred skončením povolenia na testovanie, takže s určitosťou viem prehlásiť len to, že Blind SQLi mierené na užívateľa nie je možné vykonať na túto webovú aplikáciu. Či by pri pokračovaní testovania za pomoci novo získaných informácií bolo možné sa dostať k akýmkoľvek citlivým dátam z databázy určiť nie je možné, ale podľa doterajších výstupov zo skenera to nasvedčovalo tomu, že to nie je možné alebo je to možné veľmi ťažko pri dlhšej dobe skúšania o prenik do databázy. Ale v rámci toho čo bolo plne otestované je pozitívne, že vstup užívateľa je výborne zabezpečený, či už zo strany klienta alebo servera.

3.3.4 Subjekt D

Webová aplikácia bWAPP je určená na učenie sa/skúšanie/ testovanie jednotlivých SQLi metód. Aplikácia bola hostovaná na VB v prostredí Windows 10. V testovacom subjekte boli vyskúšané základné operátori a to ako na nich reaguje užívateľský formulár, v prípade zlého prihlásenie to aké informácie poskytuje server. Síce v analytickej časti táto aplikácia bola zhodnotená ako dobrá na testovanie jednotlivých SQLi metód, čo v podstate je, ale ponúka skúšanie týchto metód len na základnej úrovni. Trochu ma sklamalo ako fungovala z mojej vlastnej skúsenosti v prostredí Windows 10, kde mi prišla, že neponúka možnosť testovania týchto metód na úrovni, kt. by bola aspoň blízko k zabugovanej aplikácii v reálnom prostredí. Prostredníctvom testovania som si aj overila, že recenzie kt. dávali informáciu o tom, že táto aplikácia nefunguje úplne ako má vo VB a ovplyvňuje celú funkciu systému vo VB, takže ani Linux, kt. je jeho súčasťou nefunguje úplne ako by mal. Verzia kt. by bola dodatočne stiahnutá, nainštalovaná v Kali Linuxe by najskôr fungovala lepšie a pre niekoho kto si chce vyskúšať základné testovanie väčšej škály zraniteľností, než sú len tie kt. sú zamerané na SQLi túto aplikáciu bude schopný doceniť viac než ja.

3.3.5 Súhrn hodnotenia testovacích subjektov

V rámci tejto práce bolo realizované testovanie na štyroch testovacích subjektoch za účelom otestovania dvoch normálne fungujúcich webových stránok a zároveň bolo cieľom ukázať SQLi metódy v praxi, kde sa ich realizovanie niekedy podarí dotiahnuť do úspešného konca a pri veľmi dobre zabezpečených stránkach je veľmi pravdepodobné, že sa pokus o realizáciu takejto metódy nepodarí. Subjekt A bola slabo zabezpečená webová aplikácia v rámci Google hackingu bolo pri zbieraní informácií o tejto stránke úspešne využitých niekoľko logických operátorov(AND or +) a rozšírených operátorov (site:). Prostredníctvom sqlmap bola úspešne nájdená blind SQLi zraniteľnosť založená na časovej odozve. Prostredníctvom blind SQli metódy bolo z databázy získané užívateľské meno aj s heslom. Aplikácia nemala žiaden WAF a ani nemala zabezpečený užívateľský vstup či už zo strany klienta alebo serveru. Ak by to bola skutočná a nie testovaná stránka, tak by mala za sebou slušný incident skrz únik citlivých dát. Subjekt B bola webová aplikácia na platforme WordPress a patrila základnej škole, pri tomto testovacom subjekte bola tiež na zbieranie dát aplikovaná metóda Google hacking, vďaka čomu bolo nakoniec zistené administrátorské ID. Pri sqlmap skene sa nebolo možné bez ID dostať úplne nikam pretože pokus o skenovanie bolo bezpečnostným riešením na platforme WordPress zarazené hneď v zárodku. Bez ohľadu na spôsoby a počet pokusov bol vždy výsledok rovnaký. Prostredníctvom sqlmap bol vykonaný aj pokus o sken pri zadaní dobre sformulovaného odkazu s ID, WAF sqlmap nezarazil úplne ale aplikácia si chcela zadeklarovať vlastné Cookie, pravdepodobne by som sa tu bola schopná niekam dostať, ale moje povolenie na detailné testovanie stránky nezahŕňalo aj administrátorské účty a preto som v tomto kroku bola nútená skončiť. Riziko možnosti nabúrania administrátorského účtu môže ohroziť bezpečnosť celej aplikácie a preto by som správcovi stránky odporučila prejsť kapitolu 3.4.4.2 a 3.4.4.3, následne aplikovať odporúčania na svoj účet administrátora. To, že sa mi vôbec podarilo nájsť ID prostredníctvom nastavenia odkazu na základe toho, že viem ako sú automaticky generované niektoré názvy v databáze (vlastním WordPress účet) svedčí o tom, že niekto si zabudol premenovať kľúčové súbory. Z toho dôvodu odporúčam premenovanie súboru wp-login.php. Vzhľadom k nainštalovaným bezpečnostným pluginom a chrániacemu WAF si nemyslím, že je to až tak vážny problém, priemerný hacker by toho nevyužil určite, ale prečo dobrovoľne znižovať bezpečnosť ak to ide jednoducho vyriešiť.

3.4 Odporučenie bezpečnostných opatrení

Táto časť mojej diplomovej práce je venovaná návrhu bezpečnostných opatrení proti jednotlivým SQLi metódam, ktoré boli úspešne realizované v rámci testovania jednotlivých subjektov.

3.4.1 Ciel bezpečnostných opatrení

V tejto časti je dôležité si určiť účel týchto bezpečnostných opatrení. Databázy jednotlivých webových aplikácii uchovávajú obrovské kvantum dát. Tieto dáta z väčšej časti obsahujú veľmi citlivé osobné údaje, preto je dôležité urobiť maximum pri ochrane integrity, dôvernosti a dostupnosti týchto dát. Na webové aplikácie sú cielené útoky za účelom získania osobných dát, ich poškodenia, neoprávnenému prepisu a podobne zlomyseľných činov voči dátam v zacielenej databáze.

Cieľom odporučenia bezpečnostných opatrení je posilnenie ochrany dát a informácii proti SQL injection metódam, ktoré boli úspešne realizované na jednotlivé webové aplikácie počas testovania.

3.4.2 Výstupy

Na základe odporučenia bezpečnostných opatrení by zodpovedné osoby za informačnú bezpečnosť jednotlivých webových aplikácií mali byť schopný eliminovať jednotlivé nájdené zraniteľnosti voči SQLi metódam útoku.

3.4.3 Prínosy

Hlavnou myšlienkou týchto odporučení je pomoc pri zvýšení ochrany jednotlivých webových aplikácií voči SQLi útokom. V dnešnej dobe by sa malo k informačnej bezpečnosti pristupovať ozaj zodpovedne, keď sa len vezme do úvahy pomer aký v štatistike najčastejších SQLi útokov z kapitoly 2.3.2. zastupovali etický hackeri v pomere s tými neetickými, tak je to naozaj alarmujúce. Ak by také množstvo neetických hackerov zaútočilo na zle zabezpečenú webovú aplikáciu, tak vzhľadom k tomu, že by podľa GDPR mala zodpovedná osoba tieto informácie chrániť a byť za ne zodpovedná, tak by z to malo za následok slušnú katastrofu pre informačnú bezpečnosť.

3.4.4 Odporučenie bezpečnostných opatrení proti SQLi

Táto kapitola sa zameriava na odporučenia bezpečnostných opatrení na ochranu proti SQLi útokom. Prvá časť je venovaná obrane proti všetkým SQLi útokom. Druhá časť je venovaná platforme WordPress na obranu proti SQLi a tretia sa zameriava na obranu administrátorského účtu pred útokom na jeho zariadenie/ e-mail a prostredníctvom toho získania plného prístupu k databáze útočníkom.

3.4.4.1 Všeobecné odporúčania bezpečnostných opatrení proti SQLi pre ochranu databázy

Jedným z hlavných spôsobov ako sa chrániť proti SQL injection útokom je obranná metóda, ktorá zabezpečuje kontrolu vstupu užívateľa. Je niekoľko protiopatrení, ktoré by mali zabezpečiť užívateľský vstup v bezpečí. Dôležité pri zabezpečovaní bezpečnosti užívateľského vstupu je ho vnímať ako možné nebezpečie pokiaľ nie je dokázaný opak, z čoho vyplýva, že je potrebné tomuto vstupu preventívne nedôverovať a verifikovať.

Proces overovania na strane servera je tiež veľmi relevantný pri tom aby boli neutralizované potenciálne zlomyseľné príkazy, ktoré by mohli byť vložené do vstupného reťazca, za pomoci tohto procesu sa dá uistiť, že je realizovaný správny spôsob vstupu. Jednoduchá modifikácia tohto vstupu by mala ochrániť kód pred pridávaním únikového znaku (/)zlomyseľným užívateľom.

Je dobré ak je pridaná validácia zo strany klienta, ale nie je to tak výrazné opatrenie z hľadiska obrany ako je táto funkcia zo strany servera, pretože by sa to dalo obísť za pomoci pár nástrojov, akým je napríklad Burpsuit. Niektoré programovacie platformy obsahujú vstavané funkcie, ktoré automaticky vyhodnotia vstup používateľa ako škodlivý pri spätnom odoslaní stránky. Avšak aj takýto spôsob zabezpečenia môžu niektorí veľmi trpezliví hackeri obísť, z tohto dôvodu je kľúčové napriek predošlým protiopatreniam spúšťať vstup používateľa prostredníctvom vlastných bezpečnostných kontrolných postupov.

Lepšia alternatíva k únikovým znakom (/) je používanie príkazových parametrov, tie sú definované pridávaním zástupných symbolov mien v SQL príkazoch, kt. budú neskôr nahradené vstupom užívateľa.

Vyhnúť sa používaniu jednoduchých úvodzoviek v príkazoch SQL, ktoré zahŕňajú vstup bez reťazca môže tiež výrazne pomôcť ako obrana.

Výrazne podstatná je kontrola kódu každej stránky, či neobsahuje miesta, kde by boli skombinované príkazy, reťazce a obsah stránky so zdrojmi, ktoré môžu pochádzať od používateľa. Celkovo pri kontrole stránky a hľadaní potencionálnych SQL injection zraniteľností vie byť výrazne nápomocný nástroj sqlmap, ktorí pri hľadaní týchto slabín používajú mnohokrát aj hackeri.

Hlavnou súčasťou vývoja softvéru by malo byť zabezpečenie zdrojového kódu kvôli chybám a bezpečnostným dieram.

Dobrá, ale niekedy nákladná metóda na hľadanie zraniteľností je vypísanie odmeny za jej nájdenie s dodaním detailného reportu o tejto slabine. Celkom efektívne prostredie na takýto typ ošetrenie je spolupráca s hackerone, sú tam dve možnosti, buď dať svoju stránku k dispozícii pre skúšanie učiacim sa hackerom, čo je síce zadarmo, ale nemusí to byť tak prínosné ako za to vypísať odmenu. Síce je to nákladné, ale je to lacnejšie než sa potom snažiť riešiť škody napáchané únikom citlivých dát, čo môže mať pre niektoré organizácie zničujúce následky.

Jednou z najhorších chýb je používanie účtov ako root alebo sa na pripojenie webovej aplikácie k databázovému serveru. Prostredníctvom ohrozeného administrátorského účtu sa môže útočník dostať k celému systému. Škodlivými môžu byť aj neadministratívne účty, ktoré majú prístup do všetkých databáz v rámci servera. Na základe zmieneného je bezpečnejšie používať účet, ktorý má povolenia čítať-zapisovať a má prístup len ku konkrétnej databáze, takže v prípade hacknutia databázy sú napáchané škody len na jednej databáze.

Bez ohľadu na to koľko bolo zavedených predchádzajúcich opatrení, tak vždy je možnosť, že sa nejaká zraniteľnosť SQLi alebo parameter URL, ktorý umožňuje útočníkovi posielať na server ľubovoľné príkazy objaví a treba počítať aj s touto variantou. Bezpečnostné riešenie ako WAF je veľmi dobré pre jednanie s jednotlivými neznámymi hrozbami pre webovú aplikáciu. Tento bezpečnostný nástroj má na starosť premávku na aplikačnej vrstve, monitoruje http žiadosti a odpovede, ako sú menené medzi serverom a klientom, a ich skúmanie na škodlivé vzorce.

Šifrovanie citlivých dát v databáze by malo byť v obrane proti SQLi alfou a omegou, toto opatrenie by malo zabezpečiť, že ak aj hacker sa dostane k týmto dátam, tak z nich nebude schopný profitovať okamžite, čo môže poskytnúť čas na identifikovanie bezpečnostnej diery a jej zapečatenie, reaktívne protiopatrenia ako napríklad vynútenie obnovy hesla.

Najlepšie zabezpečenie citlivých dát je ich vôbec neskladovať ak to nie je nutné. Kedykoľvek sú uchovávané takto citlivé dáta, tak je dôležité zobrať v úvahu aké škody napácha to, ak skončia v nesprávnych rukách.

3.4.4.2 Odporúčania bezpečnostných opatrení proti SQLi pre ochranu databázy cielené na WordPress

WordPress ponúka veľmi dobrú ochranu proti SQLi, jeho súčasťou je odolné WAF riešenie a vo svojej zbierke neplatených bezpečnostných pluginov majú jadro bezpečnosti WordPress a pozostáva z nasledovného:

- Ochrana pred hrubými silami
- Monitorovanie výpadkov
- Bezpečné overenie
- Jednoduchá správa doplnkov
- Automatické aktualizácie doplnku
- Aktivita na webe

Platené bezpečnostné pluginy sú nasledovné:

- Filtrovanie spamu (podľa Akismet)
- Skenovanie škodlivého softvéru
- Automatické opravy

Vzhľadom k tomu, že WordPress prostredníctvom bezpečnostných pluginov ponúka dostatočnú obranu proti SQLi útokom, tak pre skutočnú ochranu proti nim by majiteľ webu/ správca mal dodržať tieto body:

1. Aby aplikácie neobsahovala diery, tak je dôležité mať nainštalované celé jadro bezpečnostných pluginov, ktoré sú ešte ako plus zadarmo.

2. Pri udržaní dobrej bezpečnosti aplikácie je kľúčové mať vždy najaktuálnejšiu verziu všetkých súčastí a pluginov WordPress.
3. Rozhodujúce je skrz bezpečnosť nedôverovať a neinštalovať do WordPress rozšírenia/pluginy z neznámych zdrojov. (WordPress je platforma, ktorá dovoľuje komukoľvek vytvárať a publikovať jeho rozšírenia pluginov. Pokiaľ by bol nainštalovaný plugin z neznámych zdrojov, tak je veľmi pravdepodobné, že si za jeho pomoci nevedomky vytvoríme bezpečnostnú dieru, ktorá napomôže útočníkovi zrealizovať útok).
4. Vždy treba skenovať nový plugin a zistiť jeho nedostatky pred inštaláciou!
5. Aj po aplikovaní vyššie zmienených bodov je potrebné si aspoň skrz sqlmap preskenovať stránku, aby sa videlo či tam nie sú nejaké SQLi zraniteľnosti, alebo sa na nejaký z vyššie uvedených opatrení nezabudlo, respektíve nebolo plne uplatnené.
6. Dobrá, ale niekedy nákladná metóda na hľadanie zraniteľností je vypísanie odmeny za jej nájdenie s dodaním detailného reportu o tejto slabine. Celkom efektívne prostredie na takýto typ ošetrenie je spolupráca s hackerone, sú tam dve možnosti buď dať svoju stránku k dispozícii pre skúšanie učiacim sa hackerom, čo je síce zadarmo, ale nemusí to byť tak prínosné ako za to vypísať odmenu.
7. Pokiaľ všetky zmienené body boli plne realizované, tak jediná bezpečnostná zraniteľnosť, ktorú je možné ovplyvniť môže vzniknúť v rukách správcu za predpokladu, že by nedodržal odporúčané opatrenia z kapitoly 3.4.4.3 a nasledujúce body pre ochranu časti spadajúcej pod administrátora vo WP.

Ochrana oblasti spadajúca pod administrátora vo WP:

1. Je potreba premenovať a stiahnuť WP zložku
2. Rozšíriť súbor Wp-config.php (obsahuje nastavenia a prístupové dáta do databázy), implementácia bezpečnostných kľúčov konfiguračného súboru, upraviť predvoľbu tabuľky na inú než je prednastavená
3. Ak je to možné, tak zašifrovať administrátorskú oblasť
4. Presunúť WP-config.php súbor na veľmi bezpečné miesto mimo aktuálnej inštalácie pretože obsahuje citlivejšie údaje než zvyšok súborov
5. Chrániť WP-config.php

6. Založiť si nový administrátorský účet a vymazať automaticky vygenerovaný administrátorský účet
7. Voľba silného hesla
8. Chrániť Wp-admin adresár
9. Nastaviť aby nebola podávaná chybná hláška
10. Obmedziť počet o chybné prihlásenie

3.4.4.3 Hlavné odporúčania bezpečnostných opatrení proti SQLi pre ochranu databázy cielené na správcu

Základné pravidlá k zabezpečeniu ochrany administrátorského účtu, ktorý má prístup k databáze. Tieto odporučenia platia pre všetkých administrátorov webových aplikácií, nie len pre tých z WordPress. Aby bol dobre chránený tento účet, tak je potrebné urobiť všetko pre zabezpečenie najvyššej ochrany e-mailu s ktorým je prepojený.

1. Voľba silného hesla, ktoré pozostáva z dôvernosti, zložitosti, dĺžky a unikátnosti
2. Zabezpečenie hesla : nepoužívať rovnaké heslo na viac miestach, nezapisovať si ho na dostupných miestach, nikdy s nikým nezdieľať heslo, používať manažéra hesiel
3. Nastavenie viacfaktorovej autentizácie
4. Zabezpečiť si a využívať antivírové programy, antimalware programy na ochranu mailu, ochrana proti ransomwaru...
5. Nikdy neotvárať neznáme odkazy
6. Zastavenie automatického preposielania e-mailov
7. Používať šifrovanie e-mailov
8. Chrániť e-mail pred neoprávneným získavaní údajov (napr. ochrana proti phishingu)
9. Ochrana e-mailu pred škodlivými súbormi pomocou bezpečnostných príloh ATP
10. Ideálne používať e-mailovú službu so zabezpečením na úrovni ProtonMail
11. Pri práci s citlivými údajmi nevyužívať verejnú wi-fi
12. Pravidelná modifikácia hesla
13. Neukladať heslo do prehliadača
14. Pre zvýšenie ochrany súkromia nainštalovať do prehliadača DuckDuckGo
15. Ochrana súkromia prostredníctvom VPN jedna z lepších je napr. NordVPN
16. Ochrana zariadenia z ktorého sa prihlasujeme

4 ZHODNOTENIE

Táto časť je venovaná zhodnoteniu výsledkov celkového testovania a na základe toho vytvoreného návrhu odporučení bezpečnostných opatrení.

Boli prevedené 4 rozsiahle testovania s účelom hľadania SQLi zraniteľností. Subjekt A a D predstavovali testované webové aplikácie, ktoré nemali žiadnu ochranu prostredníctvom WAF alebo podobného bezpečnostného riešenia. Počas testovania týchto aplikácií bol vykonaný Google hacking na získavanie informácii, boli hľadané dotazy prostredníctvom logických operátorov a rozšírených operátorov za účelom identifikácie bezpečnostných zraniteľností. V týchto aplikáciách bolo po skenovaní nájdených niekoľko SQLi zraniteľností boolean-based blind, time-based blind a Union. Následne bola vybraná a úspešne vykonaná Blind SQLi metóda, kt. je založená na základe časovej odozvy, čo bolo účinné pri získavaní užívateľského mena a hesla. V Subjekte D boli realizovaná niektorá z verzií zo všetkých troch typov SQLi.

Testované Subjekty B a C predstavovali skutočné webové stránky, na ktorých testovanie bolo udelené povolenie, avšak aby tieto testy mohli byť v práci ukázané, tak subjekty museli byť pseudonymizované a časť informácií cenzurovaná. Počas testovania Subjektu B bol taktiež realizovaný Google hacking a na skenovanie aplikácie využitá sqlmap. Po realizovaní pokusov o niekoľko útokov so zameraním na užívateľa nebolo možné sa dostať nikam pretože WAF WordPress tieto pokusy zastavil v zárodku. V rámci ďalšieho zbierania informácií bolo nájdených niekoľko administrátorských ID, pričom bolo s sqlmap začaté skenovanie, avšak povolenie, kt. som dostala od správcu nezahrňovalo možnosť testovať aj túto oblasť. Testovaniu Subjektu C bolo venovaného najviac času a hľadanie zraniteľností bolo primárne cielené na Blind SQLi. Počas testovania bolo overené výborné zabezpečenie vstupu užívateľa zo strany klienta aj serveru. Okrem metód kt. boli využité aj na ostatné testovacie subjekty, tak v tomto prípade bolo ku koncu testovania, kedy som bola presunutá do whitelistu siahnuté aj po nástroji nikto. Tento nástroj poskytol obrovské množstvo informácií na základe ktorých boli upravované vstupné príkazy, bohužiaľ keď som sa začala napriek prítomnosti WAF potencionálne niekam dostávať, tak mi skončilo povolenie na testovanie, kt. bolo udelené len do konkrétneho dátumu. Detailnejšie je to rozpísané v kapitole 3.3.

Napriek vyššie zmienenému ohľadom Subjektu B, na základe vykonaných krokov bolo zistené potencionálne bezpečnostné riziko týkajúce sa administrátorského účtu, návrh 3.4.4.2 a 3.4.4.3 podrobne zahŕňa postup ako chrániť administrátorský účet vo WordPress aby sa niečomu takémuto úplne predišlo. Ak by Subjektom B bol návrh plne aplikovaný, tak by to malo prínos v tom, že by ani administrátorské ID nemalo byť tak ľahko vyhľadané prostredníctvom Google hackingu ako sa to bolo realizovať v práci.

Odporučenie bezpečnostných opatrení je rozdelený do troch kategórii: Všeobecné opatrenia proti SQLi, Odporúčania opatrení proti SQLi mierené na WP, Odporúčania pre ochranu databázy proti SQLi mierené na správcu. Tieto odporúčania sú založené na výsledkoch testovania, niekoľkých rôznych existujúcich postupov obrany proti SQLi a vlastných skúsenostiach v tejto oblasti.

Vo všeobecnosti implementácia odporučenia bezpečnostných opatrení by mala výrazne pomôcť pri ochrane webovej aplikácii a databáze pred SQLi útokmi.

K prínosom odporúčaní patrí aj zníženie potencionálnych bezpečnostných incidentov, ktoré sa v kyberpriestore objavujú a majú tendenciu v dnešnej dobe rásť.

Práca ako taká mala nasledujúce prínosy, z testov vyplýva informácia o efektivite zabezpečenia jednotlivých subjektov proti SQLi. Testovaním bola vyvrátená väčšina hrozieb z analytickej časti pre Subjekt B a zároveň bolo odporučené bezpečnostné opatrenie pre administrátora, kt. nevzniknú finančné náklady. Ďalej boli vyvrátené bezpečnostné hrozby z analytickej časti pre Subjekt C, čim mu nevznikli finančné náklady za platenie testovania a opravy bezpečnostnej hrozby. Nasledujúcim prínosom práce je priblíženie tematiky metód SQLi z teoretického a aj detailnejšie spracovaného praktického hľadiska, skrz čo si tieto metódy môže čitateľ vyskúšať aj v jednej z odporúčaných webových aplikácií sám. Keďže je v práci spracovaná aj príprava prostredia, respektíve nainštalovanie Kali Linux, kt. obsahuje niekoľko základných nástrojov, tak čitateľ už nemusí tieto nástroje inštalovať dodatočne ako by to bolo potrebné napríklad v prostredí Windows. Práca môže slúžiť ako pomocná literatúra pre začínajúcich etických hackerov, nie je určená pre akúkoľvek pomoc pri neetickom a neoprávnenom hackovaní systému a pokiaľ sa to čitateľ rozhodne nerešpektovať, tak za to autor v akomkoľvek spôsobe nenesie zodpovednosť.

ZÁVER

Cieľom diplomovej práce bolo oboznámiť sa s SQL injection útokom, jeho metódami a so spôsobmi obrany proti napadnutiu databázovej vrstvy programu z teoretického aj praktického hľadiska.

Do práce bolo zahrnuté aj testovanie SQLi na vybraných webových stránkach, ktoré na základe žiadosti ich správcov boli v práci pseudonymizované. V prvej časti práce som sa zameriavala na teoretické východiská práce, ktoré sú potrebné pre správny postup pri analytickej, praktickej časti a návrhu bezpečnostných odporúčaní. V analytickej časti som vyhodnotila súčasný stav testovacieho prostredia a analyzovala testovacie webové aplikácie, na základe čoho som si potom jednu zvolila do práce ako testovací subjekt. Vytvorila som SWOT analýzu len pre tie testovacie subjekty, kt. patrili nejakej skutočnej organizácii, podstatným výstupom boli bezpečnostné hrozby pre jednotlivé webové aplikácie. Následne som spracovala ukážky SQLi metód, z ktorých bude časť použitá vo vlastnom návrhu riešenia v rámci tvorenia príkazov. Štatistiku najčastejších SQLi útokov som vytvorila za účelom získania informácie v percentách, aby som v praktickej časti vedela na aké typy sa mám najviac zamerať. Proces prihlasovania sa do užívateľského účtu jednotlivých testovacích subjektov som vytvorila na základe niekoľkých prihlasovaní sa do jednotlivých aplikácií. Vytvorené procesové diagramy mi poskytli do praktickej časti veľmi cennú informáciu o tom, akým spôsobom reaguje server na užívateľský vstup. Ďalej som urobila analýzu vybraných nástrojov na skenovanie, SQL príkazov, rizík a plánovaného postupu útoku. V tretej časti práce som sa najprv venovala nastaveniu testovacieho prostredia. Po nainštalovaní prostredia som začala testovanie jednotlivých subjektov. Vyhodnotila som výsledky testovania a vytvorila návrh bezpečnostných opatrení v obrane proti SQLi metódam, ktoré boli otestované v praktickej časti.

ZOZNAM POUŽITEJ LITERATÚRY

(1) CLARKE, Justin. SQL Injection Attacks and Defense [online]. Ilustrované vydanie. Elsevier, 2012 [cit. 2020-02-29]. ISBN 9781597499637.

(2) ONDRÁK, Viktor, Petr SEDLÁK a Vladimír MAZÁLEK. Problematika ISMS v manažerské informatice. Brno: Akademické nakladatelství CERM, 2013. ISBN 978-80-7204-872-4.

(2) KIM, Peter a Jan POKORNÝ. Hacking: praktický průvodce penetračním testováním. Brno: Zoner Press, 2015. ISBN 978-80-7413-313-8. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:1da4fe20-c0a1-11e8-bc37-005056827e51

(3) J. Abirami, R. Devakunchari and C. Valliyammai, "A top web security vulnerability SQL injection attack —Survey,"2015 Seventh International Conference on Advanced Computing (ICoAC), Chennai, 2015

(4) HOWARD, Michael a David LEBLANC. Bezpečný kód: [techniky a strategie tvorby bezpečných webových aplikací]. Brno: Computer Press, 2008. s. 37. ISBN 978-80-251-2050-7. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:3d20f4b0-bd54-11e4-854f-5ef3fc9ae867

(5) SMEJKAL, Vladimír. Kybernetická kriminalita. Plzeň: Vydavatelství a nakladatelství Aleš Čeněk, s.r.o., 2018. s. 129. ISBN 978-80-7380-720-7. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:370de239-9b71-4b57-a665-02ec9d055009

(6) Núkib. NÚKIB [online]. [cit. 2020-04-1]. Dostupné z: https://www.nukib.cz/

(7) SCAMBRAY, Joel a Mike SHEMA. Hacking bez tajemství: webové aplikace. Brno: Computer Press, 2003. s. 5. ISBN 80-7226-769-8. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:a8196880-5557-11e3-9ea2-5ef3fc9ae867

(8) GROFF, James R. a Paul N. WEINBERG. SQL: kompletní průvodce. Brno: CP Books, 2005. s. 30. ISBN 80-251-0369-2. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:39672dd0-fd7e-11e8-95ba-5ef3fc9bb22f

(9) KIM, Peter a Jan POKORNÝ. Hacking: praktický průvodce penetračním testováním. Brno: Zoner Press, 2015. s. 12. ISBN 978-80-7413-313-8. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:461a9430-e425-11e8-9984-005056825209

(10) HATCH, Brian, James LEE a George KURTZ. Hacking bez tajemství: Linux. Brno: Computer Press, 2003. s. 447. ISBN 80-7226-869-4. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:7e049ee0-77dc-11e4-9605-005056825209

(11) SCAMBRAY, Joel a Mike SHEMA. Hacking bez tajemství: webové aplikace. Brno: Computer Press, 2003. s. 135. ISBN 80-7226-769-8. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:b108da70-5557-11e3-9ea2-5ef3fc9ae867

(12) Stuttard, D. a Pinto, M.:The Web Application Hacker's Handbook, Wiley Publishing,Inc., 2008, 978-0-470-17077-9

(13) Collaborative Engineering v inovačním cyklu (2012 : Zlín, Česko) a Ivan MAŠÍN. Collaborative Engineering v inovačním cyklu: sborník přednášek z mezinárodní konference = Collaborative Engineering in the Innovation Cycle : international conference proceedings : Zlín 2012. V Liberci: Technická univerzita, 2012. ISBN 978-80-7372-938-7. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:11e218e0-f0c2-11e5-8d5f-005056827e51

(14) KRAJÁČ, Petr a Jaroslav CHALOUPKA. Inovace: inovační poradce jako samostatná odbornost. Brno: BIC Brno, 2014. ISBN 978-80-260-5801-4. Dostupné také z: https://kramerius5.nkp.cz/uuid/uuid:d5dda000-ebcb-11e8-bc37-005056827e51

(15) ŽÁK, David. Databázové systémy II [online]. In: . [cit. 2020-05-30]. Dostupné z: https://slideplayer.cz/slide/2361270/

(16) Sqlmap [online]. [cit. 2020-05-30]. Dostupné z: http://sqlmap.org/

(17) Test site for Acunetix WVS [online]. [cit. 2020-05-30]. Dostupné z: http://testphp.vulnweb.com/

(18) Blind SQL Injection. OWASP [online]. [cit. 2020-05-30]. Dostupné z: https://owasp.org/www-community/attacks/Blind_SQL_Injection

(19) Error based SQL injection attack. All things in moderation [online]. [cit. 2020-05-30]. Dostupné z: https://hydrasky.com/network-security/error-based-sql-injection-attack/

(20) SQL injection UNION attacks. PortSwigger web security [online]. [cit. 2020-05-30]. Dostupné z: https://portswigger.net/web-security/sql-injection/union-attacks

(21) What is SQL Injection (SQLi) and How to Prevent It. Acunetix [online]. [cit. 2020-05-30]. Dostupné z: https://www.acunetix.com/websitesecurity/sql-injection/

(22) Blue Hat Hacker Law and Legal Definition. USLegal [online]. [cit. 2020-05-30]. Dostupné z: https://definitions.uslegal.com/b/blue-hat-hacker/

(23) MCQUADE, Samuel. Understanding and Managing Cybercrime. Pearson, 2005. ISBN 13: 978-0205439737.

(24) BWAPP [online]. [cit. 2020-05-30]. Dostupné z: http://itsecgames.com/

(25) Kali Docs. Kali [online]. [cit. 2020-05-30]. Dostupné z: https://www.kali.org/docs/

(26) GitHub [online]. [cit. 2020-05-30]. Dostupné z: https://github.com/

(27) Hackerone [online]. [cit. 2020-05-30]. Dostupné z: https://www.hackerone.com

(28) KOLOUCH, Jan a Pavel BAŠTA. CyberSecurity [online]. Milešovská 5, 130 00 Praha 3: CZ.NIC, z. s. p. o., 2019 [cit. 2020-05-30]. ISBN 978-80-88168-34-8. Dostupné z: https://knihy.nic.cz/files/edice/cybersecurity.pdf#%5B%7B%22num%22%3A1169%2C%22gen%22%3A0%7D%2C%7B%22name%22%3A%22Fit%22%7D%5D

(29) What is the difference between application server and web server? [online]. In: . [cit. 2020-05-30]. Dostupné z: https://stackoverflow.com/questions/936197/what-is-the-difference-between-application-server-and-web-server

(30) BANGA, Swapnil. Web Application Architecture: Definition, Models, Types, and More. Hackr.io [online]. 28 Apr, 2020 [cit. 2020-05-30]. Dostupné z: https://hackr.io/blog/web-application-architecture-definition-models-types-and-more

(31) AEC [online]. [cit. 2020-05-30]. Dostupné z: https://www.aec.cz/cz/ztisku/matej-kacic-bezpecnostni-rizika-pro-webove-aplikace-it-systems-2016.pdf

ZOZNAM TABULIEK

ZOZNAM OBRÁZKOV

ZOZNAM SKRATIEK

SQLi	Structured query language injection
SQL	Structured query language
HW	Hardware
VB	VirtualBox
SSD	Solid State Drive
HDD	Hard Disk Drive
RAM	Random Access Memory
GB	Gigabyte
OS	Operating System
DB	Database
DBRS	Databázový riadiaci systém
DPO	Data protection officer
GDPR	General Data Protection Regulation
WAF	Web Application Firewall
URL	Uniform Resource Locator