VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2021 Peter Kopec

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ZABEZPEČENÁ ONLINE DATABÁZE PRO SBĚR DAT
ONLINE DATABASE FOR SECURE DATA COLLECTION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

Peter Kopec

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Marek Mikulec

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Peter Kopec ***ID:*** 211570
Ročník: 3 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

Zabezpečená online databáze pro sběr dat

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je implementovat systém, který umožní bezpečným způsobem přenést medicínská data prostřednictvím veřejné sítě internet do online databáze vlastního návrhu. Student realizuje průzkum existujících řešení umožňujících zabezpečený přenos dat a jejich uložení v podobě databáze. Následně bude vhodné řešení implementováno a bude proveden přenos dat (řečové záznamy, obrazové záznamy…). Součástí řešení bude rovněž návrh databázového modelu pro medicínská data. Na výsledném systému budou popsány bezpečnostní mechanismy zajišťující spolehlivý a bezpečný přenos dat a bude demonstrována funkčnost navrženého systému. Databáze bude dostupná z internetu. V rámci bakalářské práce bude implementován zabezpečený přenos medicínských dat s ohledem na připravený databázový model. Data budou uložena a precizně zabezpečena, přístup k datům bude možný pouze pro autorizované uživatele. Kladně bude hodnoceno využití pokročilých metod zabezpečení, například homomorfního šifrování, rozdělení dat, pseudonymizace, OpenID a podobně.

DOPORUČENÁ LITERATURA:

[1] ZHANG, R.; LIU, L. Security models and requirements for healthcare application clouds. 2010 IEEE 3rd International Conference on cloud Computing. IEEE, 2010. s. 268-275. DOI: 10.1109/CLOUD.2010.62.

[2] DUBOVITSKAYA, A., et al. Secure and trustable electronic medical records sharing using blockchain. IAMIA annual symposium proceedings. American Medical Informatics Association, 2017. s. 650.

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 31.5.2021

Vedoucí práce: Ing. Marek Mikulec

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Táto bakalárska práca sa venuje navrhnutiu, problematike a implementácií zabezpečenej online databáze pre zber dát, dostupnej z internetu. Databáza, ktorá je dostupná z internetu a obsahuje osobné údaje alebo iné cenné údaje musí byť dobre zabezpečená, pretože nechceme aby mohlo dojsť k zneužitiu týchto údajov neoprávnenou osobou. V teoretickej časti vyberáme vhodné aplikácie pre náš systém a rozoberáme teoretickú funkcionalitu týchto aplikácií. Aplikácie sú vyberané podľa funkcií, ktoré poskytujú, celkovej zložitosti a podpory online komunity. Časť práce sa venuje analýze únikov dát zo zdravotníckych zariadení za rok 2019 a 2020 a pár ďalším únikom z iných odvetví. Vďaka tejto analýze poznáme dôvody únikov dát a sme schopní sa viac zamerať na tieto slabiny a poukázať na problémy. Praktická časť práce sa venuje návrhu a implementácií praktického riešenia pomocou aplikácií, ktoré sme vybrali v teoretickej časti. V našom prípade sa jedná o MYSQL databázu, FLASK backend s Gunicorn WSGI a NGINX web server. Nakoniec robíme analýzu zabezpečenia tohoto riešenia za pomoci najbežnejších slabín podľa OWASP a za pomoci sieťového skeneru NMAP.

KĽÚČOVÉ SLOVÁ

autorizácia, bezpečnosť, databáza, neopravnený užívateľ, server, šifrovanie, zabezpečenie

ABSTRACT

This bachelor thesis deals with the design and implementation of a secure online database for data collection, which is accessible from the Internet. A database that is accessible from the Internet and contains personal data or other valuable data must be well secured, because we do not want this data to be misused by an unauthorized person. To begin with, we select the appropriate applications for our system and analyze their functionality. The applications are selected based on the features they provide, the overall complexity and support of their online community. Part of the work is devoted to the analysis of data leaks from medical facilities in 2019 and 2020 and a few other leaks from other industries. Thanks to this analysis, we know the reasons for the data leakage and we are able to focus more on these weaknesses and point out the problems. The next part of the work is devoted to the design and implementation of a practical solution using applications that we selected at the beginning. In our case it is a MYSQL database, FLASK backend with Gunicorn WSGI and NGINX web server. Finally, we analyze the security of this solution using the most common vulnerabilities according to OWASP and the NMAP network scanner.

KEYWORDS

authentication, database, encryption, safety, security, server, unauthorized user

KOPEC, Peter. *Zabezpečená online databáze pro sběr dat*. Brno, 2020, 37 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Marek Mikulec

VYHLÁSENIE

Vyhlasujem, že svoju bakalársku prácu na tému „Zabezpečená online databáze pro sběr dat“ som vypracoval samostatne pod vedením vedúceho bakalárskej práce, s využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil autorské práva tretích osôb, najmä som nezasiahol nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákonníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som podakoval vedúcemu diplomovej práce pánovi Ing. Markovi Mikulcovi za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Obsah

Zoznam obrázkov

Úvod

Táto práca sa venuje vytvoreniu a zabezpečeniu online Databáze pre zber dát. Skladá sa z dvoch časti, teoretickej a praktickej.

Teoretická časť sa venuje teoretickému vysvetleniu pojmov ako je databáza, web framework, približuje základné útoky na databázu a rozoberá príčiny najväčších únikov dát za rok 2020 a 2021. Taktiež analyzuje zabezpečenie praktického riešenia.

Praktická časť obsahuje zabezpečený funkčný web server s databázou. Pomocou web rozhranie poskytnutého web serverom sa vieme autorizovať a následne prezerať dáta, ktoré sa nachádzajú v databáze.

Žiadne dostupné riešenia neboli nájdené nakoľko sa jedná o špecifické využitie Webu a databáze na ukladanie dát.

Ciele práce:

1. Vytvoriť vhodný databázový model a web server, ktorý bude poskytovať dáta z databáze a bude riadiť prístup k dátam.
2. Ochrániť prenášané a uložené dáta pred nežiadúcemu čítaniu alebo úprave dát.
3. Umožniť nahravanie súborov na server a ich následne zapezpečenie.
4. Vylepšiť zabezpečenie.

Kapitola:

1. Popisuje čo je to databáza a aké tipy poznáme.
2. Popisuje čo je to Web framework a tiež popisuje jednotlivé komponenty nášho web serveru.
3. Rozoberá úniky dát zo zdravotníckych zariadení.
4. Predstavuje vybrané aplikácie, predstavu o fungovaní a opisuje priložené riešenie.
5. Opisuje implementované zabezpečenie a verifikuje zabezpečenie priloženého riešenia.

1 Databáza

V tejto časti si povieme čo je to databáza a aké typy databáz poznáme.

1.1 Čo je databáza

Databáza je štrukturovaná kolekcia informácií alebo dát. Databáza je zvyčajne ovládaná 'Database Management System' (DBMS). Väčšina databáz používa 'Structured Query Language' (SQL) na zapisovanie a čítanie dát [1].

SQL je programovací jazyk používaný takmer vo všetkých relačných databázach na dopytovanie, manipuláciu a definovanie údajov a na zabezpečenie riadenia prístupu. SQL bol prvýkrát vyvinutý v IBM v 70. rokoch minulého storočia, pričom spoločnosť Oracle bola hlavným prispievateľom, čo viedlo k implementácii štandardu SQL ANSI. SQL vyvolalo mnoho rozšírení od spoločností ako IBM, Oracle a Microsoft. Aj keď je dnes SQL stále veľmi rozšírené, začínajú sa objavovať nové programovacie jazyky [1].

1.2 Typy databáz

K dispozícii je mnoho typov databáz, ktoré budú popísané v nasledujúcom zozname.

- **Relačné databázy** V 80. rokoch sa stali dominantnými relačné databázy. Položky v relačnej databáze sú usporiadané ako skupina tabuliek so stĺpcami a riadkami. Technológia relačných databáz poskytuje najefektívnejší a najflexibilnejší spôsob prístupu k štruktúrovaným informáciám [1].
- **Objektovo orientované databázy** Informácie v objektovo orientovanej databáze sú predstavované vo forme objektov, ako v objektovo orientovanom programovaní [1].
- **Distribuované databázy** Distribuovaná databáza pozostáva z dvoch alebo viacerých súborov umiestnených na rôznych serveroch. Databáza môže byť uložená na viacerých počítačoch, umiestnených na rovnakom fyzickom mieste alebo rozptýlená v rôznych sieťach [1].
- **Dátové sklady** Dátový sklad, centrálne úložisko údajov, je typ databázy špeciálne navrhnutej na rýchle dotazy a analýzy [1].
- **NoSQL databázy** NoSQL alebo nepríbuzná databáza umožňuje ukladanie a manipuláciu s neštruktúrovanými a polo štruktúrovanými údajmi (na rozdiel od relačnej databázy, ktorá definuje, ako musia byť zložené všetky údaje vložené do databázy). NoSQL databázy sa stali populárnymi, pretože webové aplikácie sa stávali bežnejšími a zložitejšími [1].

- **Databázy grafov** Databáza grafov ukladá údaje z hľadiska entít a vzťahov medzi entitami [1].
- **Databázy OLTP** Databáza OLTP je rýchla a analytická databáza určená na veľký počet transakcií uskutočňovaných viacerými používateľmi [1].

Toto je iba niekoľko z desiatok typov databáz, ktoré sa dnes používajú. Ostatné menej bežné databázy sú prispôsobené veľmi konkrétnym vedeckým, finančným alebo iným účelom. Okrem rôznych typov databáz, zmeny a pokrok v technológiách ako napríklad cloud a automatizácia, poháňajú databáze úplne novými smermi [1].

Niektoré z najnovších databáz zahŕňajú:

- **Open source databázy** Open source databázový systém je taký, ktorého zdrojový kód je open source; takýmito databázami by mohli byť databázy SQL alebo NoSQL [1]. Open source znamená že zdrojový kód je verejne prístupný a vyvijaný komunitou.
- **Cloudové databázy** Cloudová databáza je zbierka údajov, štruktúrovaných aj neštruktúrovaných, ktorá sa nachádza na súkromnej, verejnej alebo hybridnej cloudovej výpočtovej platforme. Existujú dva typy cloudových databázových modelov: tradičný a databáza ako služba (DBaaS). V prípade DBaaS administratívne úlohy a údržbu vykonáva poskytovateľ služieb [1].
- **Multimodelová databáza** Multimodelové databázy kombinujú rôzne typy databázových modelov do jedného integrovaného koncového zariadenia. To znamená, že môžu obsahovať rôzne typy údajov [1].
- **Databáza dokumentov / JSON** Databázy dokumentov, ktoré sú určené na ukladanie, načítanie a správu informácií zameraných na dokumenty, sú moderným spôsobom ukladania údajov vo formáte JSON, a nie v riadkoch a stĺpcoch [1].
- **Samoriadiace databázy** Najnovší a najprevratnejší typ databázy, samo riadiace databázy (známe tiež ako autonómne databázy) sú založené na cloudových technológiách a pomocou strojového učenia automatizujú vyladenie, zabezpečenie, zálohovanie, aktualizácie a ďalšie bežné úlohy správy databáz, ktoré tradične vykonávajú ich spravcovia [1].

1.2.1 Čo je MySQL databáza

MySQL je open source systém správy relačných databáz založený na SQL. Bol navrhnutý a optimalizovaný pre webové aplikácie a môže bežať na akejkoľvek platforme. Keď sa na internete objavili nové a odlišné požiadavky, MySQL sa stala plat-

formou voľby pre webových vývojárov a webové aplikácie. Pretože je navrhnutý na spracovanie miliónov dotazov a tisícok transakcií, je MySQL obľúbenou voľbou pre podniky elektronického obchodu, ktoré potrebujú spravovať viac prevodov peňazí. Flexibilita na požiadanie je primárnou vlastnosťou MySQL [1].

MySQL je DBMS, ktorý stojí za niektorými z najlepších webových stránok a webových aplikácií na svete, vrátane Airbnb, Uber, LinkedIn, Facebook, Twitter a YouTube [1].

1.3 Čo je Database Managment System (DBMS)

Databáza zvyčajne vyžaduje komplexný databázový softvérový program známy ako systém správy databáz (DBMS). DBMS slúži ako rozhranie medzi databázou a jej koncovými používateľmi alebo programami, ktoré umožňuje používateľom vyhľadávať, aktualizovať a spravovať, ako sú informácie organizované a optimalizované. Systém DBMS tiež uľahčuje dohľad a kontrolu nad databázami a umožňuje rôzne administratívne operácie, ako je monitorovanie výkonu, ladenie, zálohovanie a obnova [1].

2 Web

Web application framework je typ frameworku špeciálne navrhnutým na pomoc vývojárom pri vytváraní webových aplikácií. Tieto frameworky zvyčajne poskytujú základné funkcie bežné pre väčšinu webových aplikácií, ako je správa relácií používateľov, data persistence a šablónové systémy. Použitím vhodného frameworku môže vývojár často ušetriť značné množstvo času pri vytváraní webových stránok [2].

2.1 Data persistence

Stálosť údajov, po anglicky Data persistence. Persistence je „pokračovanie účinku po odstránení jeho príčiny“. V súvislosti s ukladaním údajov do počítačového systému to znamená, že údaje prežijú aj po ukončení procesu, pomocou ktorého boli vytvorené. Inými slovami, aby sa dátové úložisko mohlo považovať za trvalé, musí zapisovať do energeticky nezávislého úložiska [3]. Stálosť údajov je pre nás veľmi dôležitá keďže údaje o pacientoch a ich vyšetreniach chcem zachovať aj v prípade výpadku elektrického prúdu. Preto používame databázu ktorá ukladá dáta na disk počítača na ktorom beží.

2.2 Flask

Flask je lightweight WSGI web aplication framework. Je navrhnutý tak, aby bolo začatie rýchle a ľahké s možnosťou rozšírenia až na zložité aplikácie. Začal ako jednoduchý obal okolo Werkzeug a Jinja a stal sa jedným z najpopulárnejších web application frameworks pre Python [4].

Flask ponúka návrhy, ale nepresadzuje žiadne závislosti ani rozloženie projektu. Je na vývojárovi, aby si vybral nástroje a knižnice, ktoré chce používať. Komunita poskytuje mnoho rozšírení, ktoré uľahčujú pridávanie nových funkcií [4].

- WSGI je rozhranie brány webového servera. Je to špecifikácia, ktorá popisuje, ako webový server komunikuje s webovými aplikáciami a ako môžu byť webové aplikácie vzájomne spojené za účelom spracovania jednej žiadosti [5].
- Werkzeug je WSGI web apllication knižnica [6].
- Jinja je moderný a návrhársky prívetivý šablónovací jazyk pre Python, ktorý bol vytvorený podľa šablón Django [7].

2.3 Gunicorn

Gunicorn „Green Unicorn“ je Python WSGI HTTP Server pre UNIX. Je to pre-forked model. Server Gunicorn je všeobecne kompatibilný s rôznymi webovými rozhraniami, jednoducho implementovaný, nenáročný na zdroje servera a pomerne rýchly [8].

2.4 Nginx

NGINX je bezplatný vysoko výkonný HTTP server s open source kódom a reverse proxy serverom, ako aj proxy server IMAP / POP3. NGINX je známy pre vysoký výkon, stabilitu, bohatú sadu funkcií, jednoduchú konfiguráciu a nízku spotrebu zdrojov. Na rozdiel od tradičných serverov NGINX sa nespolieha na vlákna procesoru pri vybavovaní požiadaviek. Namiesto toho používa oveľa škálovateľnejšiu (asynchrónnu) architektúru riadenú udalosťami [9].

3 Úniky informacií

V nasledujúcej kapitole sú uvedené jedny z najväčších únikov dát za posledné roky a dôvody prečo dané dáta unikli. Jednotlivé prípady rozdelíme do skupín:

- Fyzické zabezpečenie - chyba z reálneho sveta, napr. nestrážená miestnosť so serverom.
- Ľudská chyba - chyba spôsobená ľudským konaním, napr. poslanie prihlasovacích údajov cez email.
- Software zabezpečenie - chyba z virtuálneho sveta, napr. nezaheslovaný prístup na server.

3.1 Úniky zdravotných údajov

Rozpis najväčších únikov zdravotných údajov z nemocníc a iných zdravotníckych zariadení za rok 2020 a 2021.

3.1.1 10 najväčších únikov za rok 2020

1. **HEALTH SHARE OF OREGON** Krádež notebooku vlastneného dodávateľom prepravy Health Share of Oregon [10]. Tento únik zaradíme do Fyzickej bezpečnosti.
2. **FLORIDA ORTHOPAEDIC INSTITUTE** Útok ransomware na ortopedický inštitút na Floride potenciálne porušil údaje asi 640 000 pacientov [10]. Spôsob akým sa ransomware dostal do siete nie je známy. Tento únik zaradíme do Ľudskej chyby ale taktiež do Software zabezpečenia.
3. **ELITE EMERGENCY PHYSICIANS** Poskytovateľ, ktorý je teraz známy ako Elite Emergency Physicians, bol zahrnutý do rozsiahleho bezpečnostného incidentu, ktorý spočíval v nesprávnej likvidácii záznamov pacientov [10]. Tento únik zaradíme do Fyzickej bezpečnosti.
4. **MAGELLAN HEALTH** Hackeri získali prístup využitím phishingovej schémy sociálneho inžinierstva, Hackeri sa vydávali za klienta Magellan Health, päť dní pred nasadením ransomvéru [10]. Tento únik zaradíme do Ľudskej chyby.
5. **BJC HEALTH SYSTEM** Spoločnosť BJC Healthcare so sídlom v Missouri začala upozorňovať 287 876 pacientov z 19 pridružených nemocníc, že ich údaje boli po úspešnom phishingovom útoku zneužité [10]. Tento únik zaradíme do Ľudskej chyby.
6. **BENEFIT RECOVERY SPECIALISTS** Hacker získal prístupové údaje zamestnanca, aby získal prístup k systémom poisťovateľa kde nasadil malvér [10]. Tento únik zaradíme do Ľudskej chyby.

7. **AMBRY GENETICS** Tento incident potencialne ohrozil 232 772 pacientov. Vyšetrovanie odhalilo, že hacker získal prístup k e-mailovému účtu zamestnanca, úradníci však uviedli, že neboli schopní určiť, či útočník bol schopný získať prístup alebo exfiltrovať údaje obsiahnuté v účte [10]. Tento únik zaradíme do Ludskej chyby ale taktiež do Software zabezpečenia kedže nevieme presne ako došlo k ziskaniu prístupu na účet.
8. **PIH HEALTH** Prvotné porušenie bezpečnosti bolo zistené v júni 2019, keď po úspešnom phishingovom útoku bolo napadnutých niekoľko e-mailových účtov zamestnancov a potenciálne k nim mal prístup hacker [10]. Tento únik zaradíme do Ludskej chyby.
9. **BST & CO. CPAS** BST & CO. CPAS bol napadnutý hackerskou skupinou Maze ktorá do siete nasadila ransomware [10]. Tento únik zaradíme do Software zabezpečenia.
10. **AVEANNA HEALTHCARE** Spoločnosť Aveanna Healthcare v Gruzínsku začala vo februári upozorňovať 166 077 pacientov na možné porušenie spôsobené úspešným phishingovým útokom [10]. Tento únik zaradíme do Ludskej chyby.

3.1.2 10 najvačších unikov za rok 2019

1. **AMCA DATA BREACH** Začiatkom mája 8-K podanie u Komisie pre cenné papiere a burzy odhalilo, že predajca fakturačných služieb American Medical Collection Agency bol hacknutý po dobu ôsmich mesiacov [11]. Tento únik zaradíme do Software zabezpečenia.

2. **DOMINION NATIONAL** Poisťovateľ Dominion National ohlásil deväťročný hack na svojich serveroch, čo potenciálne narušilo údaje 2,96 milióna pacientov [11]. Software zabezpečenie.

3. **INMEDIATA HEALTH GROUP** Nesprávne nakonfigurovaná databáza bola objavená v januári, keď úradníci zistili, že funkcia vyhľadávacieho modulu umožňuje indexovanie interných webových stránok používaných na obchodné operácie [11]. Radíme do Software zabezpečenia.

4. **UW MEDICINE** Medicínska univerzita z Washingtonu začala upozorňovať 974 000 pacientov, že ich údaje boli vystavené online počas troch týždňov v dôsledku nesprávne nakonfigurovaného servera [11]. Radíme do Software zabezpečenia.

5. **WOLVERINE SOLUTIONS GROUP** Útoku ransomvéru na Wolverine Solutions Group došlo v septembri 2018 [11]. Nevieme povedať ako presne došlo k útoku ransomware ale útok môžeme zaradiť do Ľudskej chyby a Software zabezpečenie

6. **OREGON DEPARTMENT OF HUMAN SERVICES** V januári cielený phishingový útok spôsobil, že deväť zamestnancov odpovedalo na škodlivé e-maily a poskytli svoje používateľské údaje [11]. Radíme do Ľudskej chyby.

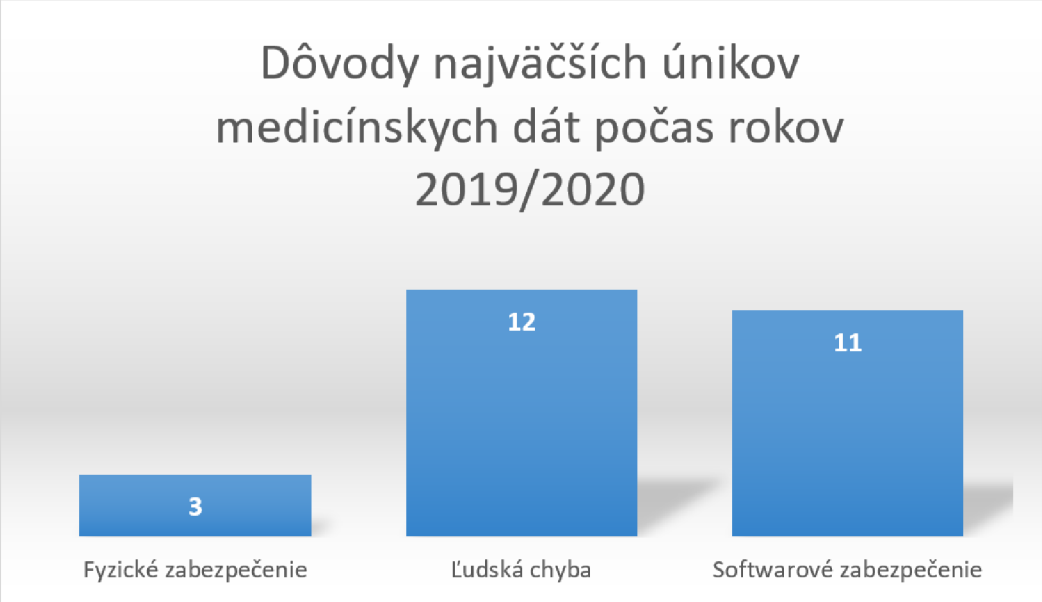
7. **COLUMBIA SURGICAL SPECIALIST OF SPOKANE** Na webe špecialistu nie je žiadne verejné oznámenie, ale údajne to bol útok ransomvérom [11]. Nevieme povedať ako presne došlo k útoku ale väčšina útokov ransomware je spôsebená stiahnutím škodlivého suboru takže tento útok môžeme zaradiť do Ľudskej chyby a Software zabezpečenie.

8. **UCONN HEALTH** Osobné a zdravotné údaje asi 326 629 pacientov s UConn Health boli potenciálne porušené potom, čo sa v decembri niekoľko zamestnancov stalo obeťou phishingových útokov [11]. Radíme do Ľudskej chyby.

9. **NAVICENT HEALTH** Neoprávnená tretia strana získala prístup k e-mailovým účtom zamestnancov Navicent Health a v júli 2018, čo potenciálne vypustilo údaje 278 016 pacientov. Spoločnosť začala pacientov upozorňovať v marci, osem mesiacov neskôr [11]. Tento únik zaradíme do Ľudskej chyby ale taktiež do Software zabezpečenia kedže nevieme presne ako došlo k získaniu prístupu.

10. **ZOLL SERVICES** Predajca zdravotníckych pomôcok ZOLL Services informoval 277 319 pacientov o porušení ich osobných a lekárskych údajov spôsobenom chybou migrácie servera [11]. Môžme zaradiť do Fyzického zabezpečenia ale aj do Software zabezpečenia.

3.1.3 Sčítanie únikov zdravotných údajov



Obr. 3.1: Sumarizácia únikov dát

Graf 3.1 vychádza z útokov, ktoré boli uvedené v kapitolách 3.1.1 a 3.1.2. Niekoľko prípadov rátame do ľudskej chyby, ako aj do chyby v software zabezpečení nakoľko incident bol spôsobený ľudským správaním, ale lepšie software zabezpečenie by tomu mohlo predísť. Taktiež prípady kde nevieme ako bol presne ransomware do siete zanesený. Ako môžeme vidieť na grafe vyššie, až v polovici prípadov sa jednalo o ľudskú chybu najčastejšie kvôli phishingu. Chyby v software zabezpečení sú na tom podobne v počte prípadov.

3.2 Ostatné úniky údajov

Facebook Security Breach Exposes Accounts of 50 Million Users Facebook uviedol, že útočníci využili dve chyby vo funkcii stránky „Zobraziť ako“, ktorá umožňuje používateľom skontrolovať, aké informácie o nich môžu vidieť iní ľudia. Táto funkcia bola navrhnutá tak, aby používateľom poskytla kontrolu nad ich súkromím. Spoločnosť uviedla, že tieto chyby boli spojené s chybou v programe nahrávania videí na Facebooku pri oslavách narodenín, čo bola softvérová funkcia zavedená v júli 2017. Táto chyba umožnila útočníkom ukradnúť takzvané prístupové tokeny - digitálne kľúče, ktoré umožňujú prístup na účet [12].
Tu môžeme vidieť príklad toho, ako rôzne extra funkcie v systéme môžu odhaliť slabiny v našom zabezpečení, a tak umožniť vstup útočníkovi. V jednoduchých

systémoch vieme lepšie ustriehnuť fungovanie jednotlivých funkcií.

Antheus Tecnologia Biometric Data Breach Spoločnosť zanedbala ochranu databázy v cloude heslom alebo jej správne šifrovanie. Toto je takmer určite výsledok ľudskej chyby zamestnancov IT [13].
V danom článku bolo niekoľko prípadov tohoto istého typu, kde nezabezpečená databáza bola voľne dostupná na internete.

Nintendo Is the Latest Victim of Credential Stuffing in Media Spoločnosť Nintendo uviedla, že napadnuté údaje boli „získané nezákonne inými prostriedkami, ako sú naše služby“. To výrazne naznačuje, že dotknutí používatelia nepoužívali jedinečné ID a heslá [13].
Tu môžeme vidieť príklad toho, že nezáleží na tom, ako veľmi máme chránené heslá v našom systéme pokiaľ ľudia používajú tie iste heslá aj na iných menej zabezpečených stránkach zkade môžu byt ukradnuté. Dvoj faktorové prihlasovanie by malo predísť tomuto problému.

4 Riešenie

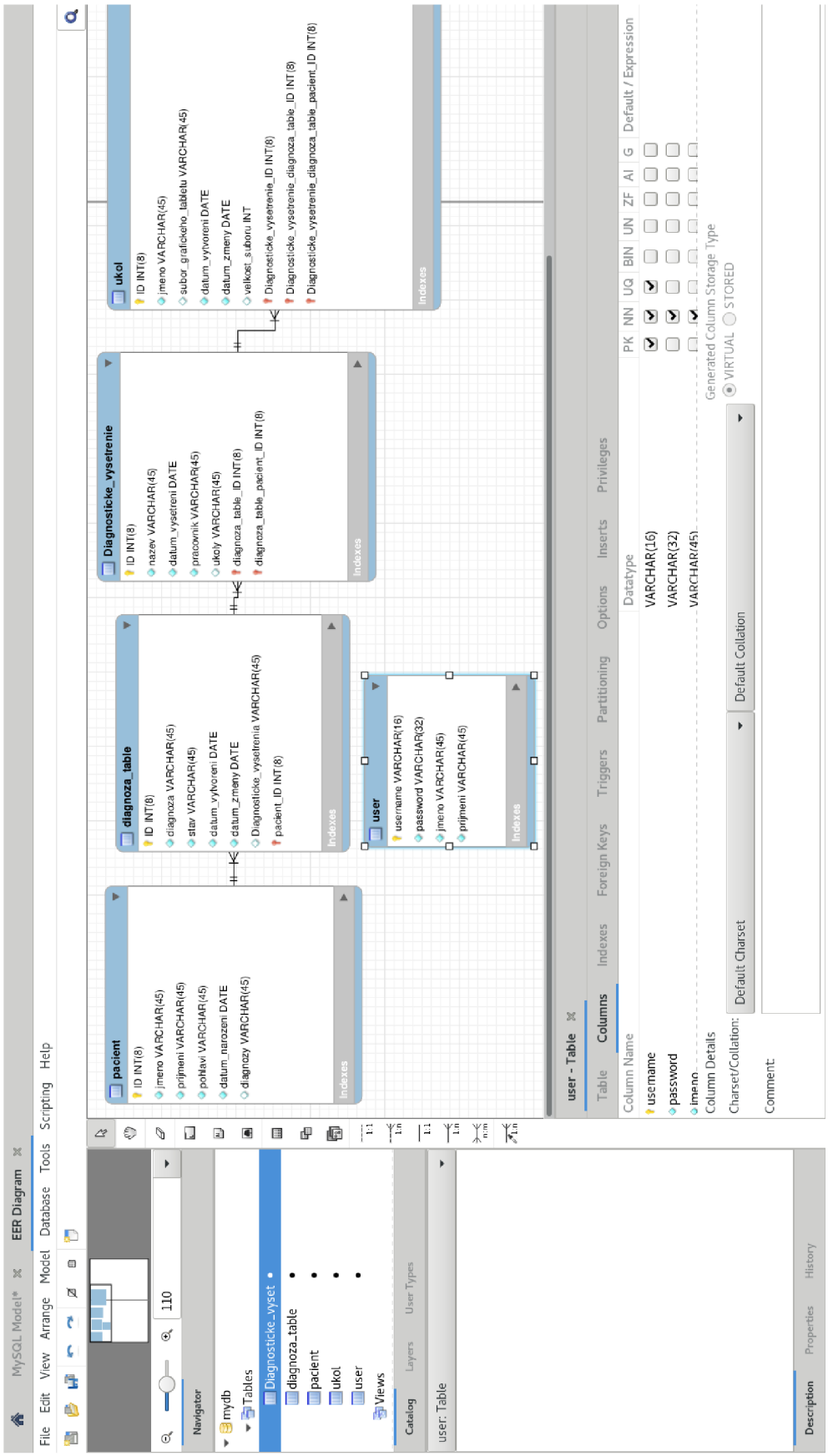
V tejto kapitole budeme rozoberať navrhnuté riešenie.

4.1 Voľba aplikácií

Zvolená bola MySQL databáza a Flask web framework spolu s Gunicorn python WSGI serverom a Nginx web server.
Mysql je open source databáza pri použití podľa GNU General Public License, ktorú používajú aj veľké stránky ako Facebook, Twitter, Youtube, ako už bolo spomenuté. Momentálne sa jedná o jednu z najpoužívanejších databáz. Taktiež podporuje ľahké programovanie pomocou MySQL workbench, kde stačí vytvoriť tabuľky, vložiť stĺpce pomenovať ich, vybrať si dátový tip a zvoliť si špeciálne funkcie ako napr: Primary Key, Not Null, Unique key, Binary, Unsigned, Zero-Filled, Auto Increment, Generated a Default value. Posledne si môžme ešte doplniť asociácie medzi tabuľkami pomocou GUI, viď obr. 4.1.
Framework Flask bol zvolený na základe skúseností z predchádzajúcich projektov, taktiež je vhodný na jednoduché stránky, čo je pre túto potrebu vhodné keďže samotné web rozhranie nemusí byť rozsiahle. Databáza aj web server budú bežať na Linux systéme. Linux je open source, stabilný, nenáročný, dobre zabezpečiteľný operačný systém a programuje sa v jazyku Python, ktorý je celkom jednoduchý s veľkou podporou na internete.
Flask taktiež podporuje nasadenie na rôznych systémoch alebo cloudoch, ako je napríklad OpenShift, alebo pomocou vlastných serverov, ako je Nginx, Apache a iné [14].
Nginx bola zvolená z dôvodu, že sa jedná o rýchly bežne používaný web server, ale taktiež, preto, lebo ako už bolo spomenuté jedná sa o reverse proxy, čo je jeho hlavná úloha v našom prípade. Nakoľko už máme vybranú Nginx a Flask tak Gunicorn je logická voľba keďže sa jedná o bežnú trojicu.

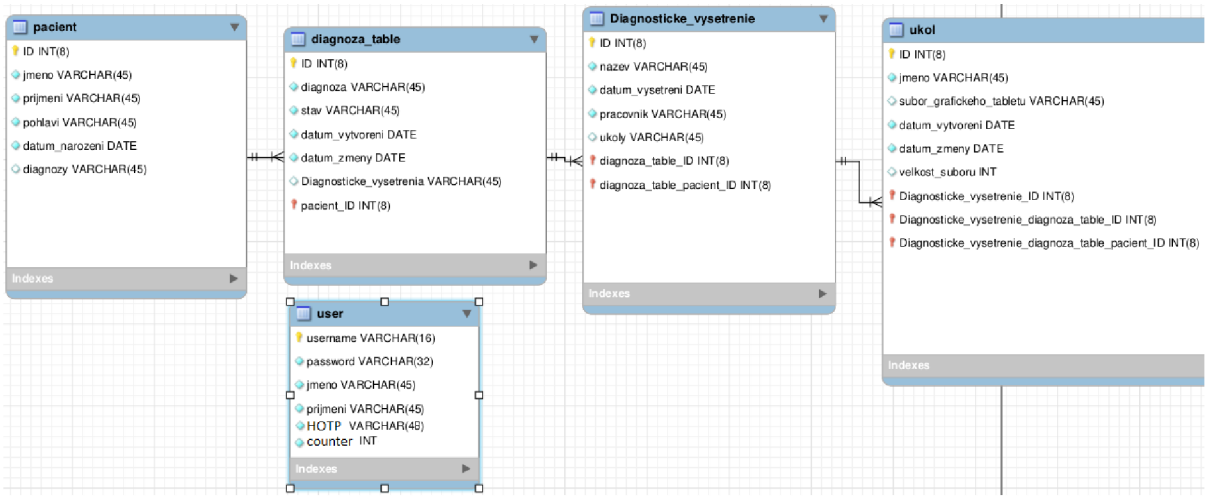
4.2 Predstava o fungovaní

Mame MySQL databázu, ktorá beží nezávisle v systéme. Zároveň s ňou beží Gunicorn, ktorý spúšťa Flask aplikáciu, ktorá pristupuje k databáze. Nginx nám poskytne web rozhranie, pomocou ktorého si vyklikáme čo by sme chceli, dovolí / zprístupní web server. Flask aplikácia spracuje informácie z web rozhrania a pristúpi k databáze a vráti nám požadované dáta, ktoré nám zobrazí na web rozhraní alebo vloží dáta, vytvorí nového pacienta, task alebo vyšetrenie.



Obr. 4.1: Ukážka rozhrania MySQL workbench

4.3 Model databáze



Obr. 4.2: Model databáze

Na obr. 4.2 môžeme vidieť päť tabuliek. Každá tabuľka ma jeden jedinečný „Primary key“, ktorý slúži na presné spárovanie dát, napr. ktorý pacient má akú diagnózu, vyšetrenie a úkol a všetky ostatné možné kombinácie. Tabuľka `user` nie je asociovaná so žiadnou inou, slúži len na ukladanie prihlasovacích údajov do databáze. Ostatné tabuľky slúžia na ukladanie informácií o pacientoch a ich vyšetreniach. Asociácie medzi spomenutými tabuľkami sú nasledovné: každý pacient `pacient` môže mať viac diagnóz `diagnoza_table`, každá diagnóza môže mať viac vyšetrení `Diagnosticke_vysetrenie`, a každé vyšetrenie môže mať viacero `úkolov`.

4.3.1 Pridanie užívateľa

Z bezpečnostných dôvodov nie je možné pridávanie nového užívateľa pomocou web rozhrania, ale len pomocou priameho prístupu k databáze. K tomuto účelu bol vytvorený skript, ktorému zadáme krstné meno, priezvisko, heslo a jedinečné užívateľské meno vygeneruje za nás zo zadaných údajov.

4.4 Server

Ďalšou časťou riešenia je server, ktorý ma niekoľko funkcií. Jedny z najdôležitejších je poskytovanie web rozhrania spolu s potrebnými dátami z databáze a riadenie prístupu, čiže prihlasovanie užívateľov keďže nechceme len tak hocikomu poslať údaje o pacientoch. Aby sme predišli odchyteniu týchto dát pri prenose komunikácie medzi užívateľom, webovým prehliadačom a serverom tak sa na komunikáciu využíva

HTTPS protokol. Aktuálne používa certifikát podpísaný nami vytvorenou autoritou. Certifikát tejto autority vieme nainštalovať na všetky zariadenia keďže sa nejedná o server pre verejnosť. Ale nakoľko používame webový server, ako je Nginx vieme si vyžiadať aj o ssl certifikát podpísaný globálne uznávanou autoritou.
Keby sme chceli vieme vynechať Nginx a použiť iba Gunicorn v lokálnej sieti, ktorý nie je vhodný pre užívateľov, ktorý majú pomalé pripojeninie lebo gunicorn ma obmedzený počet vlakien a kým neodošle všetky dáta dalej nemože toto vlákno zavriet a pokračovat. Ďalší problém by vznikol so získavaním verejných ssl certifikátov.

4.4.1 Ochrana prenášaných dát

Ochranu dát môžeme chápať ako to, že prenášané dáta sa dostanú z bodu A do B bez poškodenia, čiže klient dostane dáta v rovnakom stave, ako ich server odoslal. Toto nám zabezpečí internetový protokol TCP, ktorý sa bežne používa na prepravu paketov. Tak isto ochranu môžeme chápať ako prevenciu voči neželanému čítaniu alebo úprave dát. Na toto slúži TLS protokol, ktorý šifruje komunikáciu medzi klientom a serverom. Okrem protokolu TLS vieme použiť aj SSL protokol, ale ten je už zastaralý a nie je bezpečný [15].

4.4.2 Virtuálna privátna sieť

Pokiaľ len vybraná skupina ľudí potrebuje prístup na náš server je preferované mať server v privátnej sieti, aby nebol viditeľný z internetu. Pre prístup do tejto privátnej siete by sme použili Virtuálnu privátnu sieť VPN. Použitím VPN pre voliteľné získanie prístupu k serveru z internetu by sme získali ďalšiu vrstvu zabezpečenia pri prenose dát cez internet.

4.4.3 Prihlasovanie

Prihlasovanie na stránke je jednou z najcitlivejších funkcií keďže každý kto má prístup k sieti má možnosť sa skúsiť prihlásiť a po úspešnom prihlásení už nič nebráni prehliadaniu dát v databáze. Na obr. 4.3 môžeme vidieť, ako vyzerá prihlasovanie na strane servera.
Táto funkcia podporuje dve metódy GET a POST. Pri GET metóde server jednoducho vráti login.html šablónu, a to nastane, keď chceme stránku zobraziť. Keď sa chceme prihlásiť tak ideme do POST časti. Na začiatku skontroluje či políčko Meno, Heslo alebo Token nie sú prázdne. Ak sú tak server pošle stránku naspäť s chybovou správou že meno, heslo alebo token su nesprávne. Keď prejdeme cez kontrolu prázdnych políčok tak server od databáze vypýta heslo a token seed podľa zadaného mena. Ak databáza nevráti žiadneho user tak server vráti chybovú správu ako v predošlom

prípade. Keďže od databáze chceme dostať užívateľa podľa `username` musí sa jednať o jedinečný údaj v databáze lebo ak by nebol tak táto funkcia `cursor.fetchone()` by vrátila chybu. Využitím SQL injection nie je možné sa prihlásiť. Keď nám databáza vráti užívateľa tak pomocou funkcie `check_hash()` skontrolujeme či zadané heslo a heslo v databáze sú rovnaké. Pomocou `check_token()` skontrolujeme token. Ak sú rovnaké, server vytvorí `session` s názvom užívateľa a prehliadač si to zapamätá. Ak nie sú rovnaké, server vráti chybovú hlášku.

```
@app.route('/login', methods=['GET', 'POST'])
#@limiter.limit("3/second")
def login():
    """ Handle logging in """
    if request.method == 'POST':

        # Get login and password from form
        r_login = request.form['login']
        r_password = request.form['password']
        r_token = request.form['token']

        # Check for empty strings
        if not r_login or not r_password or not r_token:

            return render_template('login.html', invalid=True)

        # Fetch entry from database
        cursor.execute("SELECT  * FROM user WHERE username = %s;",(r_login,))')
        user = cursor.fetchone()

        # If user is not in the database
        if not user:
            return render_template('login.html', invalid=True)

        # Check password hash
        match = check_hash(r_password, user['password'])
        hotp = check_token(r_token, user['HOTP'], user['counter'])
        if match is True and hotp != 0:
            # Start new session
            session.clear()
            session['user_id'] = user['username']
            cursor.execute("UPDATE user SET counter= %s WHERE username= %s;",
                    (str(hotp),r_login))
            conn.commit()
            #session.permanent = True
            return redirect(url_for('index'))
        else:
            return render_template('login.html', invalid=True)

    return render_template('login.html')
```

Obr. 4.3: Ukážka `login` funkcie

Bez vytvorenej `session` server vráti vždy `login` stránku, viď obr. 4.4. Tento kus kódu sa nachádza na začiatku každej `app.route` ktorá označuje každú stránku akú vieme navštíviť na našom web serveri.

```
@app.route('/')
def index():

    if session:

        cursor.execute("SELECT * FROM pacient")

        pacient = cursor.fetchall()

        return render_template('index.html', pacient=pacient)
    else:
        return redirect(url_for('login'))
```

Obr. 4.4: Kontrola existencie `session`

4.4.4 Autentizačný server

V prípade, že by sme mali viac aplikácií, ktoré vyžadujú prihlasovanie, bolo by vhodné použiť autentizačný server ako je napriklad Kerberos alebo Keycloak. Kerberos aj Keycloak sú open source, čo znamená, že sú voľne dostupné a so širokou podporou komunity. V prípade použitia autentizačného servera by sa stačilo prihlásiť jeden krát a autentizovanie do všetkých podporovaných aplikácií by riešil autentizačný server za užívateľa. Avšak využitie ďalšej služby by mohlo znížiť bezpečnosť keďže sa jedná o ďalšiu aplikáciu, ktorá môže byť využitá na získanie neoprávneného prístupu, čím by útočník mohol získať prístup do viacerých služieb.

5 Bezpečnosť riešenia

V tejto kapitole si popíšeme zabezpečenie nášho systému.

5.1 Lokálna

Mysql služba beží v systéme a dá sa do nej pristúpiť len pomocou hesla. Fľask bude poznať heslo, aby mohol databázu otvoriť, ale toto heslo nemá taký vplyv na online bezpečnosť, keďže databáza nie je priamo prístupná z internetu. Heslo, ktoré je uložené vo Fľask aplikácia je v plaintexte, ale nik nebude mať práva čítať súbor. Heslá užívateľov v databáze sú uložené v HASH podobe pomocou algoritmu SHA 512. Databáza implementuje šifrovanie a všetky súbory, ktoré sa ukladajú na server majú obmedzenú veľkosť. Server povoľuje len určité koncovky súborov a po nahraní súborov sa súbor zašifruje.

5.2 Fyzická

Ako sme mohli vidieť v sekcii, kde sme rozoberali úniky dát tak nie len software zabezpečenie je dôležité. Tak isto je dôležité fyzicky ochrániť všetky prístupové body a zariadenia pred použitím ne autorizovanou osobou. Nesmieme ani zabudnúť na osoby, ktoré majú prístup k nášmu systému. Takéto osoby musia poznať základné bezpečnostné pravidlá a taktiež aj bezpečnostné pravidlá organizácie, pre ktorú pracujú. Podľa týchto pravidiel sa musia následne správať.

5.2.1 Antivírusový program

Na všetkých zariadeniach, ktoré majú prístup do internej siete a zároveň do internetu bude bežať antivírusový program. Takýto program by mal byť schopný ochrániť systém pred väčšinou zlomyseľných kódov, ako je malware a ransomware. Malware a ransomware boli v minulosti zodpovedné za veľa incidentov ohľadom úniku dát.

5.2.2 Data encrpytion

Databáza implementuje službu „Data at rest encryption“. Táto služba využíva dva typy kľúčov, „tablespace key“ ktorý šifruje tabuľku a nemení sa a druhý je „master encryption key“, ktorý šifruje „tablespace key“ a môže sa zmeniť. Keď aplikácia alebo autorizovaný užívateľ chce pristúpiť k dátam tak InnoDB dešifruje dáta [16]. Pri nahrávaní súborov server overuje koncovku súboru, ak sa koncovka nenachádza v takzvanom „white list“ tak server odmietne súbor. Po úspešnom nahraní súboru

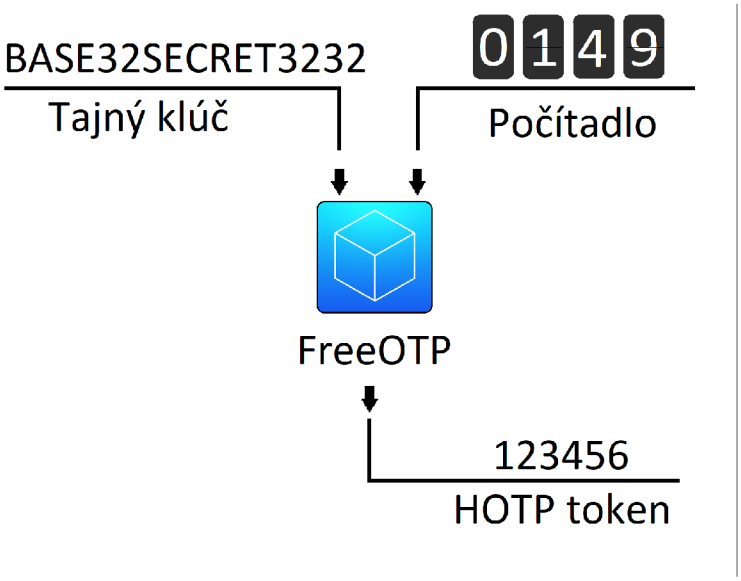
je súbor zašifrovaný pomocou python knižnice Fernet. Fernet na šifrovanie využíva AES v CBC móde so 128 bitovým kľúčom.
Fernet je implementácia symetrickej kryptografie overenou autentifikáciou [17].
InnoDB je univerzálny úložný modul, ktorý vyvažuje vysokú spoľahlivosť a vysoký výkon [18].

5.3 Online

Na spomenutom web rozhraní prebieha autentizácia pomocou mena, hesla a HOTP tokenu. Po úspešnom autentizovaní budeme oprávnený vidieť zoznam pacientov a vkladať a čítať ostatné dáta. Pre ochranu dát nie je možné nič vymazať pomocou web rozhrania, ale len pomocou priameho prístupu k databázy. Login je chránený voči SQL injection útoku. Systém, na ktorom beží Databáza a Web server je chránený, aby útočník nemohol získať prístup a dačo poškodiť. Samozrejme celá komunikácia prebieha cez moderný protokol TLS v 1.3, ktorý používa na šifrovanie AES 256 v GCM móde. V priloženom riešení používame `self singet` certifikáty, ktoré sú podpísané nami vytvorenou autoritou. Server obslúži jednu IP adresu momentálne len 3 krát za jednu sekundu. Toto by malo predísť DOS a útoku hrubou silou. Server odhlasuje užívateľov, ktorí sú neaktívni po určitú dobu, aktuálne je to jedna hodina. Cookies, kde je uložená session užívateľa, ktorú vytvoril server sú podpísané tajným kľúčom, ktorý pozná len web server a tým pádom pri zmene dát v cookies podpis nebude sedieť a server vie, že im nemôže veriť.

5.3.1 HOTP token

HOTP token je heslo, ktoré sa dá použiť len jeden krát. Vstup na generovanie tohto hesla je tajný kľúč a číslo s počítadla, ktoré sa zvýši o hodnotu 1 pri každom použití. Tieto hodnoty môže spracovať fyzické zariadenie alebo ako v našom prípade FreeOTP aplikácia, ktorú si vieme nainštalovať na smartphone. Tajný kľúč nie je rovnaký ako heslo, ktoré používa užívateľ. Token slúži ako plávajúca časť hesla pri dvoj faktorovom prihlasovaní [19]. Schéma HOTP s použitím aplikácie FreeOTP je znázornené na obr. 5.1.



Obr. 5.1: Ukážka HOTP tokenu

5.4 Verifikácia zabezpečenia

Top 10 zraniteľností databáze podľa [20] a top 10 zraniteľností webových aplikácií [21]. Krátke vyjadrenie k niektorým z nich, ktoré považujeme za najdôležitejšie.

5.4.1 SQL injection

Vkladanie a upravovanie SQL queries. Ako bolo spomenuté v sekcii 4.4.3 Login je odolný voči tomuto útoku nakoľko Server si od databáze podľa mena vypýta heslo, heslo zadane na `Login` stránke prevedie hash funkciou a porovná ich.

5.4.2 PEBKAC

Niektorý užívateľ môže mať slabé heslo a útočník získa prístup. Proti tomuto sa vieme chrániť tak, že prikážeme, aby heslo obsahovalo určitý počet znakov a taktiež aj špeciálne znaky. Taktiež spomenutý HOTP token nám zvyšuje ochranu v prípade slabšieho alebo uniknutého hesla.

5.4.3 Nepotrebné práva užívateľov

Niektorý užívateľ môže mať práva vykonávať niečo, čo nepotrebuje. Nakoľko všetci užívatelia majú rovnako obmedzené pravidlá nepovažujem túto zraniteľnosť za prob-

lém. Ale stále je tu miesto na vylepšenie, ako napríklad každému pacientovi prideliť určitých doktorov a len tí by mohli pristupovať ku dátam, ktoré sa týkajú daného pacienta.

5.4.4 Povolené nepotrebné funkcie

Jedná sa o funkcie, ktoré nevyužívame, sú zapnuté a firewall akceptuje ich komunikáciu. Preto náš server má vo firewall povolené len služby, ktoré potrebujeme, ako sú HTTPS, HTTP a SSH. HTTP máme povolené len, aby sme mohli klienta presmerovať na HTTPS. SSH je služba, ktorá slúži na vzdialený prístup a je chránená menom a heslom alebo RSA kľúčom. Keďže máme povolené to, čo potrebujeme tak táto zraniteľnosť nie je prítomná.

5.4.5 Nezašifrované dáta

Databáza a aj súbory, ktoré ukladáme na server sú šifrované. Tak isto prenos dát ku klientovi prebieha zabezpečeným protokolom TLS. Táto zraniteľnosť, teda nie je prítomná.

5.4.6 Cross-Site Scripting (XSS)

XSS umožňuje útočníkovi podstrčiť svoj vlastný javascriptový kód, čo môže napríklad viesť k získavaniu citlivých údajov návštevníkov stránky, obchádzaniu bezpečnostných prvkov aplikácie a phishingu. Ako ochranu sme použili CSP header vďaka čomu vieme nastaviť dôveryhodné zdroje javascriptových kódov.

5.4.7 Pokazená autentifikácia

Nesprávne implentovaná alebo pokazená služba na prihlasovanie. V našom prípade jediná možnosť, ako sa dá prihlásiť by mala byť tak, že zadáme správny token a zároveň zadáme do políčka heslo slovo, ktoré má rovnaký hash, ako je hash hesla užívateľa, čo nepožujeme za implementovanú alebo pokazenú službu prihlasovania. Takže túto zraniteľnosť nepožujem za problém.

5.5 Sken systému

Pri tomto skene budeme uvažovať, že server je verejne dostupný alebo, že útočník získal prístup do našej siete.

5.5.1 Nmap

Na oskenovanie portov nášho systému sme použili nástroj Nmap za pomoci GUI zvaného Zenmap.

Nmap („Network Mapper“) je bezplatný open source nástroj na zisťovanie a auditovanie sietí. Nmap bol časopismi „Linux Journal“, „Info World“, „LinuxQuestions.Org“ a „Codetalker Digest“ označený za „bezpečnostný produkt roka“. Bol dokonca uvedený v dvanástich filmoch, napríklad Matrix Reloaded, Die Hard 4, Girl With the Dragon Tattoo či Bourneovo ultimátum [22].

Na obr. 5.2 môžeme vidieť dva skeny pre porovnanie. Na obrázku vľavo je sken s vypnutým firewall a na obrázku vpravo je sken so zapnutým firewall.

Port	Protocol	State	Service	Version
22	tcp	open	ssh	OpenSSH 8.0 (protocol 2.0)
80	tcp	open	http	nginx 1.14.1
443	tcp	open	http	nginx 1.14.1
31365	udp	open|filtered	unknown	
56141	udp	open|filtered	unknown	
111	tcp	open	rpcbind	2-4 (RPC #100000)
3306	tcp	open	mysql	MySQL (unauthorized)
5000	tcp	open	http	Gunicorn 20.0.4
111	udp	open	rpcbind	2-4 (RPC #100000)
443	udp	open|filtered	https	
1012	udp	open|filtered	sometimes-rpc1	
1014	udp	open|filtered	unknown	
2967	udp	open|filtered	symantec-av	
5353	udp	open	mdns	DNS-based service discovery
8001	udp	open|filtered	vcom-tunnel	
21060	udp	open|filtered	unknown	
33030	udp	open|filtered	unknown	
33717	udp	open|filtered	unknown	
36458	udp	open|filtered	unknown	
49360	udp	open|filtered	unknown	

Port	Protocol	State	Service	Version
22	tcp	open	ssh	OpenSSH 8.0 (protocol 2.0)
80	tcp	open	http	nginx 1.14.1
443	tcp	open	http	nginx 1.14.1
136	udp	open|filtered	profile	
162	udp	open|filtered	snmptrap	
518	udp	open|filtered	ntalk	
965	udp	open|filtered	unknown	
16420	udp	open|filtered	unknown	
16739	udp	open|filtered	unknown	
17424	udp	open|filtered	unknown	
17468	udp	open|filtered	unknown	
18156	udp	open|filtered	unknown	
19165	udp	open|filtered	unknown	
19647	udp	open|filtered	unknown	
20326	udp	open|filtered	unknown	
20762	udp	open|filtered	unknown	
21358	udp	open|filtered	unknown	
24279	udp	open|filtered	unknown	
31365	udp	open|filtered	unknown	
34796	udp	open|filtered	unknown	
42434	udp	open|filtered	unknown	
49211	udp	open|filtered	unknown	
54807	udp	open|filtered	unknown	
56141	udp	open|filtered	unknown	

Obr. 5.2: Scan portov systému

Stav portov, ktoré sú v stave „open|filtered“ nebol jednoznačne určený nástrojom Nmap. Podľa nastavení nášho firewall sú povolené iba porty 22, 80 a 443, čo sa zhoduje s výsledkom skenu.

5.5.2 Zraniteľnosti zistených služieb

Zraniteľnosti boli zisťované pomocou CVE.
Poslaním programu CVE je identifikovať, definovať a katalogizovať verejne zverejnené chyby zabezpečenia v oblasti kybernetickej bezpečnosti [23].

- **OpenSSH 8.0**
  Na OpenSSH bola nájdená iba jedna zraniteľnosť CVE-2019-6111. Táto zraniteľnosť je použiteľná iba pri používaní funkcie SCP na prenos dát. Nakoľko neohrozuje vzdialený prístup a ani neumožňuje útočníkovi získať vzdialený prístup len z faktu, že služba SSH je povolená, tak nepožujem túto zraniteľnosť za ohrozujúcu.

- **nginx 1.14.1**
  Na nginx boli nájdené tri zraniteľnosti CVE-2018-16845, CVE-2018-16844 a CVE-2018-16843. Všetky tri zraniteľnosti sa týkajú doplňujúcich modulov. Nakoľko ani jeden z týchto modulov nepoužívame, nevzniká nám žiadne riziko z týchto zraniteľností.

Záver

Práca sa zaoberala teoretickým úvodom do použitých aplikácií, opisovaním zabezpečenia, opisom navrhnutého riešenia a prieskumom zdravotných údajov z nemocníc a iných zariadení.

Cieľom bolo vytvoriť vhodný databázový model a web server, ktorý bude poskytovať dáta z databáze a bude riadiť prístup k dátam. Bol vytvorený databázový model a web server ktorý spĺňa potreby. Web server pomocou prihlasovania užívateľov dokáže riadiť prístup k dátam a je nám schopný zobraziť dáta z databáze.

Ďalším cieľom bolo ochrániť prenášané a uložené dáta pred nežiadúcemu čítaniu alebo úprave dát. Uložené dáta sú chránené databázou keďže na prístup k databáze je potrebné meno a heslo. Prenášané dáta sú chránené použitím https protokolu. Ďalšia a vylepšená ochrana bude pridaná v budúcnosti.

Ďalším cieľom bolo umožniť nahrávanie súborov a následne ich zabezpečiť. Bola implementovaná funkcia pre nahrávanie súborov a ich následne šifrovanie.

Ďalším cieľom bolo vylepšiť zabezpečenie. Bolo pridané šifrovanie databáze, pri prihlasovaní bol pridaný HOTP token čo je druhý faktor prihlasovania. Firewall blokuje komunikáciu pre všetky služby, ktoré nepoužívame. Taktiež server implementuje ochranu pred DoS útokom.

Literatúra

[1] *Oracle: what-is-database*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://www.oracle.com/database/what-is-database.html>.

[2] *Web_archive: Web_application_framework*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://web.archive.org/web/20150723163302/http://docforge.com/wiki/Web_application_framework>.

[3] *datastax: data persistence*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://www.datastax.com/blog/what-persistence-and-why-does-it-matter>.

[4] *Flask: flask*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://palletsprojects.com/p/flask>.

[5] *WSGI: wsgi*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://wsgi.readthedocs.io/en/latest/what.htm>.

[6] *Werkzeug: Werkzeug*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://werkzeug.palletsprojects.com/en/1.0.x/>.

[7] *Jinja: Jinja*[online]. Z dňa 29.11.2020. Dostupné z URL: <https://jinja.palletsprojects.com/en/2.11.x/>.

[8] *Gunicorn: Gunicorn*[online]. Z dňa 5.4.2021. Dostupné z URL: <https://gunicorn.org/>.

[9] *Nginx: Nginx*[online]. Z dňa 5.4.2021. Dostupné z URL: <https://www.nginx.com/resources/wiki/>.

[10] *The 10 Biggest Healthcare Data Breaches of 2019*[online]. Z dňa 9.2.2021. Dostupné z URL: <https://healthitsecurity.com/news/the-10-biggest-healthcare-data-breaches-of-2020-so-far>.

[11] *The 10 Biggest Healthcare Data Breaches of 2019*[online]. Z dňa 9.2.2021. Dostupné z URL: <https://healthitsecurity.com/news/the-10-biggest-healthcare-data-breaches-of-2019-so-far>.

[12] *Facebook Security Breach Exposes Accounts of 50 Million Users*[online]. Z dňa 11.2.2021. Dostupné z URL:

<https://www.nytimes.com/2018/09/28/technology/facebook-hack-data-breach.html>.

[13] *The 11 Biggest Data Breaches of 2020*[online]. Z dňa 11.2.2021. Dostupné z URL:
<https://auth0.com/blog/the-11-biggest-data-breaches-of-2020-so-far/#L6--Wishbone-Data-Breach-Puts-Young-Users-at-Risk>.

[14] *Flask: deployment*[online]. Z dňa 5.4.2021. Dostupné z URL:
<https://flask.palletsprojects.com/en/0.12.x/deploying/>.

[15] *SSL vs TLS* [online]. Z dňa 21.4.2021. Dostupné z URL:
<https://www.globalsign.com/en/blog/ssl-vs-tls-difference>.

[16] *MYSQL: Data at rest encryption*[online]. Z dňa 21.4.2021. Dostupné z URL:
<https://dev.mysql.com/doc/refman/5.7/en/innodb-data-encryption.html>.

[17] *10 Web Application Security Risks*[online]. Z dňa 21.4.2021. Dostupné z URL:
<https://cryptography.io/en/latest/fernet/>.

[18] *10 Web Application Security Risks*[online]. Z dňa 21.4.2021. Dostupné z URL:
<https://dev.mysql.com/doc/refman/5.6/en/innodb-introduction.html>.

[19] *HOTK*[online]. Z dňa 21.4.2021. Dostupné z URL:
<https://www.onelogin.com/learn/otp-totp-hotp>.

[20] *Top 10 Database Security Issues to Avoid*[online]. Z dňa 29.11.2020. Dostupné z URL:
<https://webuilddatabases.com/top-10-database-security-issues-to-avoid>.

[21] *10 Web Application Security Risks*[online]. Z dňa 9.2.2021. Dostupné z URL:
<https://owasp.org/www-project-top-ten/>.

[22] *Nmap : about*[online]. Z dňa 18.4.2021. Dostupné z URL:
<https://nmap.org/>.

[23] *CVE : about*[online]. Z dňa 18.4.2021. Dostupné z URL:
<https://cve.mitre.org/about/index.html>.

Zoznam symbolov, veličín a skratiek

AES	Advance Encryption Standard
CBC	Cipher Block Chaining
CSP	Content Security Policy
CVE	Common Vulnerabilities and Exposures
DBMS	Database Management System
DOS	Denial Of Service
GUI	Graphical User Interface
HOTP	HMAC-based One-Time Password
HTTPS	Hypertext Transfer Protocol Secure
Nmap	Network Mapper
SQL	Structured Query Language
VPN	Virtual Private Network
WSGI	Web Server Gateway Interface
XSS	Cross-Site-Scripting

Zoznam príloh

A Riešenie - príloha

Odkaz na git kde je uložená práca:
`https://github.com/kamastom/prakticka_bakalarka`
Heslá, klúče aj certifikáty sú len ilustračné a pred použitím by mali byť zmenené.

A.0.1 Databáza

Jedná sa o SQL skript, ktorý je určený pre mysql databázu. Súbor `mysqldb` obsahuje len model databáze, `db_with_data` obsahuje model databáze aj s testovacími dátami.

A.0.2 Pridanie užívateľa

Jedná sa o `adduser` python3 skript ktorý je určený na pridávanie nových užívateľov do tabuľky user a vytvorenie jedinečného mena (username) užívateľa pomocou ktorého sa bude prihlasovať do systému.

A.0.3 Web Server

Obsahuje potrebné HTML šablóny, css a js kódy pre stránky, ďalej obsahuje `app` python3 kód servera a potrebné certifikáty pre HTTPS spolu s certifikátom našej autority.