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

KOMPLEXNÍ ŘEŠENÍ WAKE-ON-LAN PRO KONKRÉTNÍ POČÍTAČOVOU SÍŤ
COMPLEX WAKE-ON-LAN SOLUTION FOR PARTICULAR COMPUTER NETWORK

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR
Peter Sakala

VEDOUCÍ PRÁCE
SUPERVISOR
Ing. Jan Jeřábek, Ph.D.

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Peter Sakala ***ID:*** 164388
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Komplexní řešení Wake-on-LAN pro konkrétní počítačovou síť

POKYNY PRO VYPRACOVÁNÍ:

V rámci semestrálního projektu nastudujte problematiku Wake-on-LAN, souvisejících technologií a protokolů a také systém přihlašování přes eduID či VUT login v režimu externí aplikace. Navrhněte webovou aplikaci, která poběží ve k tomu účelu vhodném a vámi připraveném virtualizovaném operačním systému na jednom ze serverů UTKO. Navrhněte vhodnou strukturu databáze MAC adres, popř. uživatelů, která bude součástí vytvořeného řešení. Řešení rozpracujte. V rámci navazující bakalářské práce celý systém naprogramujte, nasaďte do provozu včetně přihlašování přes systém eduID a důkladně otestujte, zejména z pohledu funkčnosti a použitelnosti nejen přes klasické desktop prohlížeče, ale i přes mobilní prohlížeče (Android, iOS, ...). Ošetřete vytvořené systémy tak, aby bylo možné je považovat za bezpečné, aktualizované a připravte stručnou dokumentaci včetně webové stránky s uživatelským návodem na použití.

DOPORUČENÁ LITERATURA:

[1] FOROUZAN, Behrouz A. TCP/IP protocol suite. 4th ed. Boston: McGraw-Hill Higher Education, 2010, xxxv, 979 s. ISBN 978-0-07-337604-2.

[2] Česká akademická federace identit eduID.cz: Technická sekce [online]. Cesnet, ©2014 [cit. 2015-09-11]. Dostupné z: https://www.eduid.cz/cs/tech/index.

Termín zadání: 1.2.2016 ***Termín odevzdání:*** 1.6.2016

Vedoucí práce: Ing. Jan Jeřábek, Ph.D.
Konzultant bakalářské práce:

doc. Ing. Jiří Mišurec, CSc., *předseda oborové rady*

ABSTRAKT

Cieľom tejto práce je na základe získaných informácií navrhnúť webovú aplikáciu, ktorá bude slúžiť na vzdialené zobúdzanie počítačov v rámci Ústavu telekomunikací FEKT VUT v Brně. Teoretická časť sa venuje technológii WoL (Wake on LAN) a súvisiacim technológiám a protokolom. Ďalej sa venuje systému jednotného prihlasovania vo webových službách, funkcii federácie eduID.cz a súčasným webovým technológiám. Praktickou časťou je návrh aplikácie vrátane štruktúry databázy na ukladanie potrebných dát, jej realizácia, nasadenie a otestovanie.

KĽÚČOVÉ SLOVÁ

LAN, WAN, Wake on Lan, Magic Packet, jednotné prihlasovanie, SAML, certifikát, OAuth, webová aplikácia, ASP.NET, MVC, Entity Framework, SQL, Let's Encrypt

ABSTRACT

The goal of this thesis is based in designing a web application from obtained information, which will be used to remotely wake up computers within the Department of Telecommunications FEEC BUT. The theoretical part of the thesis is focused on WoL technology, and related technologies and protocols. Moreover, it is focused on a single sign-on system in web services, functions of eduID.cz federation and current web technologies. The practical part is the actual designing of the application including the database structure required for storing data, its realization, use and testing.

KEYWORDS

LAN, WAN, Wake on Lan, Magic Packet, Single Sign-On, SAML, certificate, OAuth, web application, ASP.NET, MVC, Entity Framework, SQL, Let's Encrypt

SAKALA, Peter *Komplexní řešení Wake-on-LAN pro konkrétní počítačovou síť*: bakalárska práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 64 s. Vedúci práce bol Ing. Jan Jeřábek, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Komplexní řešení Wake-on-LAN pro konkrétní počítačovou síť“ vypracoval samostatne pod vedením vedúceho bakalárskej práce, 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 zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Janu Jeřábkovi, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a konštruktívne návrhy k práci.

Brno . . . . . .

. . . . . .

podpis autora

SIX research centre
sensor, information and communication systems

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výskum popísaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .
podpis autora

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

2007–13
OP Výzkum a vývoj pro inovace

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM VÝPISOV

ÚVOD

V dnešnej dobe sa počítače využívajú takmer všade – v domácnostiach, vo firmách, na univerzitách a pod. Každý z nich je používaný na iný účel a preto sa stávajú prípady, keď potrebujeme mať prístup ku konkrétnemu počítaču (napr. práca z domu – prístup k špeciálnemu softvéru, ktorý je k dispozícii len v práci). Pre tento účel sa využívajú technológie zabezpečujúce vzdialený prístup k počítačom, najčastejšie cez VPN. Základnou podmienkou na fungovanie týchto technológií je, aby bol cieľový počítač (ten na ktorý vzdialene pristupujeme) zapnutý a komunikoval v sieti. Na splnenie tejto podmienky je potrebná technológia, ktorá zabezpečí, aby sa počítač zapol v požadovanom okamihu. Takouto technológiou je v súčastnosti výhradne Wake on LAN, ktorej sa venuje táto práca.

Technológia jednotného prihlasovania na webe (SSO) je v súčasnosti čím ďalej, tým viac používaná hlavne z dôvodu zvýšenia používateľského komfortu. Používatelia si tak nemusia pamätať množstvo prihlasovacích údajov, ale používajú na prihlásenie do internetových služieb jednotné prihlasovacie údaje, ktoré spravuje nejaká organizácia (napr. Google, Facebook). V tejto práci sú popísané rôzne možnosti aplikácie jednotného prihlasovania, súvisiace štandardy, federáciu eduID.cz a jednotné prihlasovanie pre externé aplikácie na VUT v Brně.

Pre vytvorenie webovej aplikácie dnes existujú rôzne technológie a platformy, ktoré podporujú aj mobilné zariadenia. Aplikácia je tak vhodne zobrazená na veľkých aj na malých obrazovkách. Praktická časť tejto práce zahŕňa návrh modernej webovej aplikácie, ktorá bude poskytovať používateľom službu Wake on LAN, pričom bude návrh zameraný na kompatibilitu s rôznymi druhmi zariadení a zabezpečenie, s využitím jednotného prihlasovania pomocou VUT konta. Okrem návrhu je cieľom túto aplikáciu naprogramovať, uviesť do prevádzky a otestovať jej funkciu.

1 WAKE ON LAN

Wake on LAN (WoL) je termín označujúci v informatike technológiu, ktorá umožňuje zapnutie alebo prebudenie počítača cez počítačovú sieť. Vytvorila ju v apríli roku 1997 aliancia AMA (Advanced Manageability Alliance) tvorená spoločnosťami Intel a IBM.[1]

Technológia WoL pracuje na 2. vrstve (Ethernet) nezávisle na protokole 3. vrstvy (IP, IPX atď.) vďaka špeciálnemu rámcu „**Magic packet**“, ktorý je spravidla poslaný programom spusteným na inom počítači v tej istej lokálnej sieti, prípadne cez internet. Okrem WoL existuje podobná technológia cielená na bezdrôtové siete nazvaná WoWLAN (Wake on WLAN). Kvôli rôznym obmedzeniam sa ale neujala tak ako WoL (z dôvodu nutnosti trvalého pripojenia k bezdrôtovej sieti, a teda zvýšenej spotreby).

1.1 Využitie

WoL sa využíva napr. vo väčších spoločnostiach alebo univerzitách, kde sa vyžaduje vzdialený prístup k PC bez toho, aby musel byť počítač sústavne zapnutý. Vtedy sa zapína na diaľku v prípade potreby. Aktívne WoL síce vyžaduje malé množstvo energie naviac, v konečnom dôsledku môže spoločnosť na energii ušetriť, pretože sú dané PC zapnuté iba vtedy, keď je to potrebné. WoL sa využíva aj na zariadeniach, ktoré poskytujú služby na vyžiadanie (on-demand).

Táto technológia nerieši opätovné vypnutie PC. To ale v prípade, že máme vzdialený prístup do operačného systému (OS) daného PC (napr. vzdialená pracovná plocha), nie je problém a teda vypnutie inicializujeme pomocou OS.[2]

1.2 Magic Packet

Magic Packet je štandardný Ethernet rámec (frame), ktorý obsahuje zdrojovú a cieľovú MAC adresu (Media Access Control) – to môže byť adresa stanice, ale väčšinou to je broadcast adresa.[3] V dátovej časti musí kdekoľvek (spravidla sú to jediné dáta v rámci) obsahovať synchronizačný tok (stream) – 6 bajtov s hodnotou 255 (FF_{16}). Za ním nasleduje 16× zopakovaná MAC adresa cieľového PC. Ako nepovinná časť Magic Packetu je SecureOn heslo (6 bajtov), ktoré ale nepodporuje mnoho sieťových kariet. Tabuľka 1.1 graficky znázorňuje obsah Magic Packetu, ktorý je neskôr zabalený do Ethernet rámca.[4] Na obrázku 1.1 je znázornený jednoduchý vývojový diagram algoritmu pre vytvorenie Magic Paketu. V prílohe A.5.1 je pomocná trieda v jazyku C# (použitá v praktickej časti), ktorá dokáže vytvoriť Magic Packet a poslať ho pomocou protokolu UDP do lokálnej siete (na základe IP a MAC adresy).

IP adresa bola v tomto prípade zahrnutá z dôvodu určenia výstupného rozhrania na serveri a využíva sa subnet broadcast.

Tab. 1.1: Magic packet (dátová časť) a veľkosti jeho častí v bajtoch

Synchronizácia	16×cieľová MAC adresa	Heslo (nepovinné)
6	96	0, 4 alebo 6

Z dôvodu, že sieťová karta v režime WoL nemá aktívny celý protokolový zásobník (full protocol stack), nezáleží na použitom protokole sieťovej a transportnej vrstvy. Rámec sa preto často odosiela ako UDP datagram s odporúčaným cieľovým portom 9 (discard), 7 (echo) alebo ľubovoľný od 0 do 65 535, keďže sa nekontroluje. Zdrojový port je dynamický. Magic Packet je možné odoslať aj priamo pomocou Ethernetu ako EtherType 0x0842.

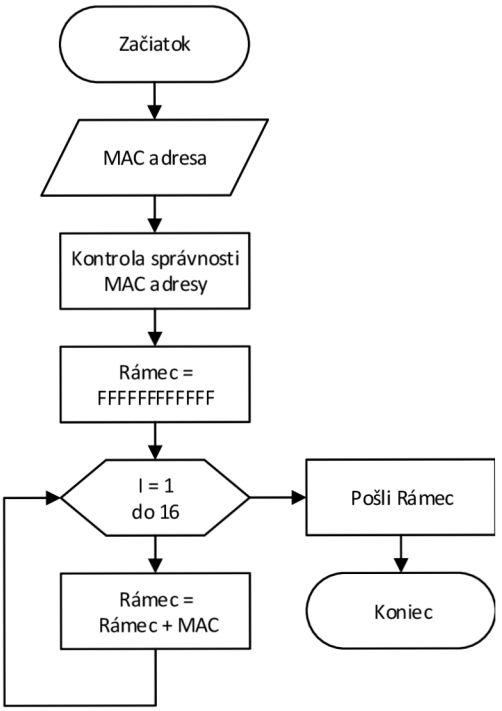
Výpis 1.1: Magic Packet – detail dátovej časti (Wireshark)

```
0000    ff ff ff ff ff ff 00 1f c6 88 87 cf 00 1f c6 88
0010    87 cf 00 1f c6 88 87 cf 00 1f c6 88 87 cf 00 1f
0020    c6 88 87 cf 00 1f c6 88 87 cf 00 1f c6 88 87 cf
0030    00 1f c6 88 87 cf 00 1f c6 88 87 cf 00 1f c6 88
0040    87 cf 00 1f c6 88 87 cf 00 1f c6 88 87 cf 00 1f
0050    c6 88 87 cf 00 1f c6 88 87 cf 00 1f c6 88 87 cf
0060    00 1f c6 88 87 cf
```

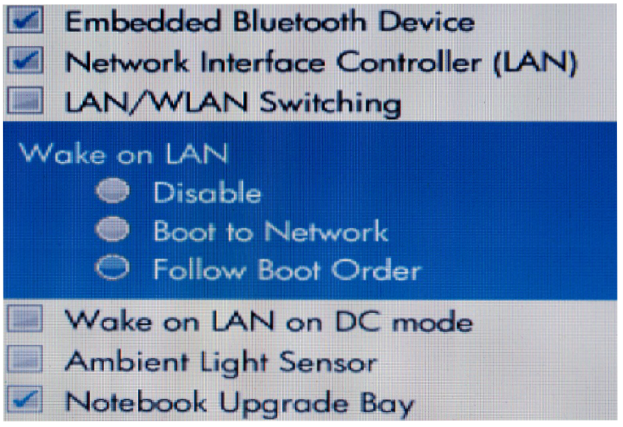
1.3 Požiadavky

Základná požiadavka, ktorú musí klient spĺňať na fungovanie WoL je podpora základnej dosky a sieťovej karty (NIC) PC. Táto technológia už z podstaty fungovania vyžaduje trvalo napájanú sieťovú kartu počítača, ktorá ho dokáže zapnúť/zobudiť. Podpora WoL sa zväčša zapína v nastavení BIOS/UEFI v časti Power Management (obr. 1.2) a často sa dá ovládať aj v možnostiach ovládača sieťového adaptéru v operačnom systéme (obr. 1.3). Po aktivácii tejto funkcie zostáva po vypnutí PC sieťová karta naďalej aktívna (LED indikátor linky zostane svietiť) čo znamená aj mierne zvýšenie spotreby sieťového adaptéra.

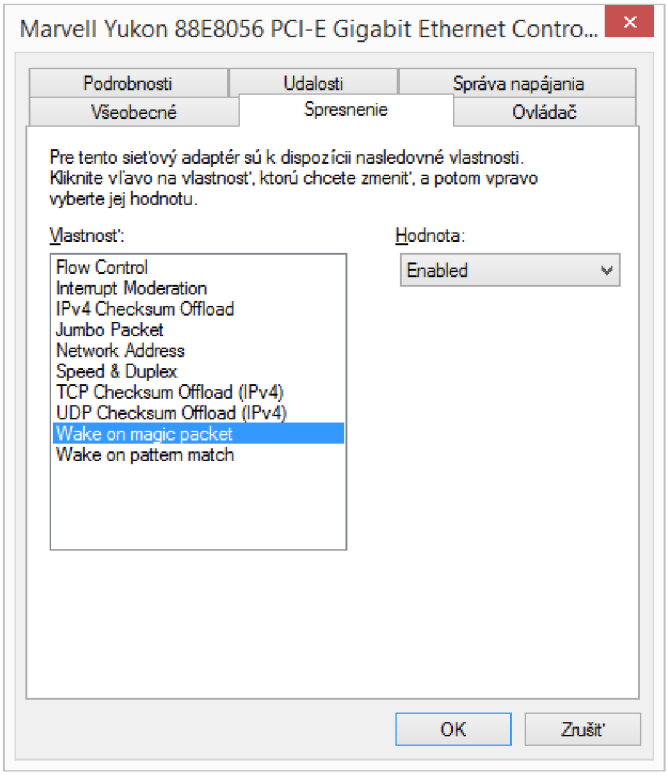
Technológia WoL nie je závislá na nainštalovanom OS, pracuje len na úrovni firmvéru sieťovej karty.[5] To nám umožňuje spúšťať akýkoľvek druh počítača (klientské PC, server a pod.). Staršie sieťové karty potrebovali priame prepojenie tzv. „WAKEUP-LINK“ so základnou doskou (3-pinový kábel), v súčasnosti toto prepojenie nie je potrebné, pretože „oživovací signál“ aj nepretržité napájanie je ošetrené priamo zbernicou PCI/PCI-e.



Obr. 1.1: Jednoduchý vývojový diagram vytvorenia Magic Packetu



Obr. 1.2: Nastavenie podpory WoL v UEFI



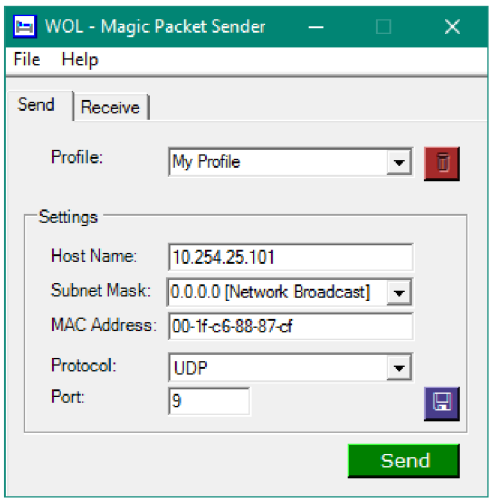
Obr. 1.3: Nastavenie podpory WoL v ovládači sieťového adaptéru vo Windowse

1.4 Použitie

Proces prebudenia vzdialeného PC je veľmi jednoduchý. Zväčša stačí použiť špeciálnu aplikáciu (obr. 1.4), ktorá dokáže vytvoriť a poslať Magic Packet alebo je takouto funkcionalitou vybavený smerovač/prepínač. Existuje množstvo hotových riešení (aplikácie, webové stránky), pomocou ktorých sa dajú posielať Magic Packety, no nie sú vhodné na použitie v tejto práci (komplexnosť riešenia).

1.4.1 LAN

V lokálnych sieťach je potrebné v aplikácii definovať iba MAC adresu cieľového PC. Aplikácia vytvorí rámec Magic Packet a pošle ho broadcastom do celej LAN (cieľová MAC adresa FF:FF:FF:FF:FF:FF).[5] Vo výpise 1.2 je zachytená táto komunikácia pomocou Wiresharku.



Obr. 1.4: Ukážka jednoduchej aplikácie WoL

Výpis 1.2: Magic Packet ako broadcast (Wireshark)

```
Frame 13: 144 bytes on wire (1152 bits),
    144 bytes captured (1152 bits) on interface 0
Ethernet II, Src: HewlettP_41:b1:c1 (e4:11:5b:41:b1:c1),
    Dst: Broadcast (ff:ff:ff:ff:ff:ff)
Internet Protocol Version 4, Src: 10.254.25.1,
    Dst: 255.255.255.255
User Datagram Protocol, Src Port: 52376 (52376),
    Dst Port: 9 (9)
Wake On LAN, MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
    Sync stream: ffffffffffff
    MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
```

1.4.2 Internet a rozsiahle LAN

Ak je potrebné posielať Magic Packet mimo LAN – napríklad cez internet alebo medzi podsiete (subnet) v rámci LAN – nastáva problém, ako smerovať broadcast cez smerovače. Tie totiž oddeľujú broadcast domény, tzn. že cez nich broadcastové pakety neprejdú. V tomto prípade existujú dve možnosti[5]:

Unicast

Magic Packet je poslaný na konkrétnu IP cieľového PC, takže prejde smerovačmi až k poslednému, ku ktorému je priamo pripojená sieť, v ktorej je dané PC pripojené.

Nastáva ale problém, že tento PC nemá aktívny protokol sieťovej vrstvy (IP) a tak chýba záznam v ARP tabuľke smerovača (záznam je platný iba krátky čas po vypnutí PC). Smerovač síce vyšle ARP request (broadcast), nedostane ale odpoveď. Najjednoduchšie riešenie je statický ARP záznam na smerovači, ktorým definujeme na akú MAC adresu má smerovaný paket s cieľovou IP adresou poslať. Vo výpise 1.3 je zachytená táto komunikácia pomocou Wiresharku.

Výpis 1.3: Magic Packet ako unicast (Wireshark)

```
Frame 359: 144 bytes on wire (1152 bits),
    144 bytes captured (1152 bits) on interface 0
Ethernet II, Src: HewlettP_41:b1:c1 (e4:11:5b:41:b1:c1),
    Dst: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
Internet Protocol Version 4, Src: 10.254.25.1,
    Dst: 10.254.25.101
User Datagram Protocol, Src Port: 53857 (53857),
    Dst Port: 9 (9)
Wake On LAN, MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
    Sync stream: ffffffffffff
    MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
```

Smerovaný broadcast

V prípade použitia smerovaného broadcastu (SDB – Subnet Directed Broadcasts) je cieľová IP adresa posledná v danej podsiete (napr. 10.0.0.255 v sieti 10.0.0.0/24). Rámec prejde cez všetky smerovače v ceste, kde posledný smerovač tento rámec upraví na broadcast a pošle do celej cieľovej siete. Z dôvodu zneužívania tejto techniky k útokom je táto možnosť na smerovačoch a bránach firewall vypnutá alebo blokovaná. Je ju preto nutné dodatočne povoliť (ak to sieťový prvok podporuje). Vo výpise 1.4 je zachytená táto komunikácia pomocou Wiresharku.

Výpis 1.4: Magic Packet ako smerovaný broadcast (Wireshark)

```
Frame 1117: 144 bytes on wire (1152 bits),
    144 bytes captured (1152 bits) on interface 0
Ethernet II, Src: HewlettP_41:b1:c1 (e4:11:5b:41:b1:c1),
    Dst: Routerbo_0e:0b:6d (4c:5e:0c:0e:0b:6d)
Internet Protocol Version 4, Src: 10.254.25.1,
    Dst: 172.16.0.255
User Datagram Protocol, Src Port: 59245 (59245),
    Dst Port: 9 (9)
Wake On LAN, MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
    Sync stream: ffffffffffff
    MAC: AsustekC_88:87:cf (00:1f:c6:88:87:cf)
```

2 JEDNOTNÉ PRIHLASOVANIE VO WEBOVÝCH SLUŽBÁCH

V dnešnej dobe je na internete nespočetné množstvo internetových služieb a ku každej sa prihlasuje osobitným účtom. Tento problém rieši technológia jednotného prihlásenia alebo Single Sign-On (SSO).[6, 7] Táto technológia pracuje na princípe univerzálneho kľúča. Používateľ má svoje prihlasovacie údaje uchované na jednom centrálnom mieste a u ostatných služieb sa prihlasuje práve týmito prihlasovacími údajmi. Služby, u ktorých sa používateľ prihlasuje, nemajú prístup k prihlasovacím údajom. Často býva tento mechanizmus rozšírený o možnosť jednotnej správy používateľského účtu (napr. osobné údaje).

Výhody:

- oveľa vyšší komfort pri používaní množstva služieb – používateľ si pamätá iba jedno heslo,
- minimalizácia rizika prezradenia hesla (ak sa použije dostatočne silné),
- možnosť použitia dvojfaktorovej autentifikácie – stačí, že ju podporuje poskytovateľ identít,
- šetrí čas zadávaním prihlasovacích údajov na rovnakú identitu,
- redukuje náklady IT podpory spojené s heslami.

Nevýhody:

- väčší bezpečnostný dopad pri krádeži identity – útočník získa prístup ku všetkým službám,
- poskytovateľ identít má informácie o prihlásení do služieb.

Známi poskytovatelia identít:

- Facebook, Google, Seznam . . .
- mojeID – služba založená na štandarde OpenID vytvorená združením CZ.NIC,
- eduID.cz – česká akademická federácia identít.

Známe štandardy na poskytovanie SSO

- Security Assertion Markup Language,
- OAuth,
- OpenID.

2.1 Security Assertion Markup Language

Security Assertion Markup Language (SAML) je otvorený štandard dátového formátu založený na XML, ktorý slúži na vymieňanie autentifikačných a autorizačných dát medzi účastníkmi, najmä medzi **poskytovateľom identít** a **poskytovateľom služieb**.[8] Vyvinul ho v roku 2001 OASIS Security Services Technical Committee. Posledná veľká aktualizácia bola publikovaná v marci 2005 kedy bola vydaná nová verzia štandardu SAML 2.0. Jednou z najdôležitejších úloh štandardu SAML je jednotné prihlasovanie na webe (SSO).

SAML štandardizuje veľký rozsah funkcií spojených s prijímaním, vysielaním a zdieľaním bezpečnostných informácií na:

- poskytnutie XML formátov pre bezpečnostné dáta používateľov a formátov na žiadanie a prenos informácií,
- určenie, ako tieto správy pracujú s protokolmi ako napr. SOAP (Simple Object Access Protocol),
- špecifikovanie presnej výmeny správ pre bežné použitie – napríklad web SSO (Single Sign-On),
- podporu množstva mechanizmov na zabezpečenie súkromia, vrátane možnosti určiť atribúty používateľa bez odhalenia jeho identity,
- popis, ako zaobchádzať s informáciami o identitách vo formátoch široko používaných technológií ako X.509, LDAP, DCE a XCML,
- formuláciu schémy metadát, ktoré umožňujú zúčastneným systémom vykomunikovať možnosti SAML, ktoré podporujú.

Špecifikácia SAML definuje tri roly:

- **používateľa**,
- **poskytovateľa identít**,
- **poskytovateľa služieb**.

SAML rieši prípad, keď používateľ požiada o službu od poskytovateľa služieb. Poskytovateľ služieb požiada a následne získa **tvrdenie** o identite (identity assertion) od poskytovateľa identít. Na základe tohto tvrdenia rozhodne poskytovateľ služieb o možnosti prístupu používateľa k službám ktoré poskytuje tento poskytovateľ. Pred tým ako poskytovateľ služieb dostane toto tvrdenie, poskytovateľ identít môže požadovať informácie od používateľa – zväčša to je meno a heslo – na autentifikáciu používateľa. Jeden poskytovateľ identít môže poskytovať SAML tvrdenia viacerým poskytovateľom služieb. Podobne aj naopak – jeden poskytovateľ služieb sa môže spoliehať a dôverovať tvrdeniam od rôznych nezávislých poskytovateľov identít.

SAML nešpecifikuje metódu autentifikácie používateľa u poskytovateľa identít,

takže môže použiť meno a heslo alebo inú formu vrátane multifaktorovej autentifikácie. Typickým zdrojom autentifikačných tokenov u poskytovateľa identít sú adresárové služby ako LDAP, RADIUS alebo Active Directory, ktoré umožňujú používateľom prihlásiť sa pomocou mena a hesla.

2.1.1 Štandard X.509 a certifikáty

X.509 je podľa odporúčania ITU-T[9] štandard v kryptografii pre systémy založené na verejnom kľúči (PKI, Public Key Infrastructure) pre jednoduché podpisovanie. Tento štandard okrem iného špecifikuje formát certifikátov, zoznamy odvolaných certifikátov a metódy platnosti kontroly certifikátov.

Certifikáty

V systéme X.509 certifikačné autority vydávajú certifikáty, ktoré potvrdzujú, že daný verejný kľúč patrí uvedenému vlastníkovi. Vlastník môže byť definovaný presným menom alebo alternatívnym označením ako je napríklad e-mailová adresa alebo DNS záznam. Organizácie môžu vydať svoje vlastné dôveryhodné koreňové certifikáty, ktoré platia iba v rámci podniku. Webové prehliadače majú zväčša predinštalované koreňové certifikáty známych poskytovateľov certifikátov (SSL certifikáty).

Štruktúra certifikátu podľa štandardu X.509 v3[10]:

- **Version Number** – verzia certifikátu,
- **Serial Number** – sériové číslo,
- **Signature Algorithm ID** – označenie algoritmu (ID),
- **Issuer Name** – vydavateľ,
- **Validity period** – platnosť:
  - **Not Before** – nepoužívať pred dátumom,
  - **Not After** – nepoužívať po dátume,
- **Subject Name** – vlastník verejného kľúča,
- **Subject Public Key info** – informácie o verejnom kľúči vlastníka:
  - **Public Key Algorithm** – algoritmus pre verejný kľúč,
  - **Subject Public Key** – verejný kľúč (dáta),
- **Issuer Unique Identifier** – unikátny identifikátor vydavateľa (nepovinný),
- **Subject Unique Identifier** – unikátny identifikátor vlastníka (nepovinný),
- **Extensions** – rozšírenia (nepovinný),
- **Certificate Signature Algorithm** – algoritmus pre certifikát (elektronický podpis),
- **Certificate Signature** – certifikát (elektronický podpis).

Bežné prípony súborov s certifikátmi X.509:

- **.DER** – (Distinguished Encoding Rules) zakódovaný certifikát,
- **.PEM** – (Privacy-enhanced Electronic Mail) DER certifikát zakódovaný pomocou Base64, nachádzajúci sa medzi riadkami „`---BEGIN CERTIFICATE---`“ a „`---END CERTIFICATE---`“
- **.P7B** a **.P7C** – PKCS#7 (Public-Key Cryptography Standards) štruktúra SignedData bez dát, iba certifikáty alebo CRL (Certification Revocation List,
- **.P12** – PKCS#12 môže obsahovať certifikát(y), verejné aj súkromné kľúče (chránené heslom),
- **.PFX** – predchodca PKCS#12, často obsahuje dáta vo formáte PKCS#12.

2.1.2 Metadáta v SAML 2.0

Metadáta zaisťujú v SAML 2.0 bezpečný prenos medzi poskytovateľom identít a poskytovateľom služieb.[11] SAML 2.0 poskytuje dobre definovaný a navzájom spolupracujúci formát metadát, ktoré entity môžu využiť pri zavádzaní procesu dôveryhodnosti (vzájomnou výmenou metadát).

Výpis 2.1: Ukážka štruktúry metadát IdP

```
<md:EntityDescriptor
    xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    entityID="https://idp.example.org/SAML2">
    <!-- insert ds:Signature element -->
    <!-- insert md:IDPSSODescriptor element (below) -->
    <!-- insert md:AttributeAuthorityDescriptor element (not shown) -->
    <md:Organization>
      <md:OrganizationName xml:lang="en">
        SAML Identity Provider
      </md:OrganizationName>
      <md:OrganizationDisplayName xml:lang="en">
        SAML Identity Provider @ Some Location
      </md:OrganizationDisplayName>
      <md:OrganizationURL xml:lang="en">
         http://www.idp.example.org/
      </md:OrganizationURL>
    </md:Organization>
    <md:ContactPerson contactType="technical">
      <md:SurName>SAML IdP Support</md:SurName>
      <md:EmailAddress>mailto:saml-support@idp.example.org</md:EmailAddress>
    </md:ContactPerson>
  </md:EntityDescriptor>
```

Keď IdP dostane element `<samlp:AuthnRequest>` od SP cez webový prehliadač, IdP sa pozrie do zoznamu služieb, ktorým dôveruje a overí si ju na základe metadát predtým, ako pošle odpoveď. Takisto pomocou týchto metadát IdP vie kam má presmerovať používateľa po úspešnej autentifikácii. SP overí podpis tvrdenia pomocou verejného kľúča IdP a tak SP vie, že IdP nie je podvrhnutý.

Výpis 2.2: Ukážka štruktúry metadát SP

```
<md:EntityDescriptor
    xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata"
    xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    entityID="https://sp.example.com/SAML2">
    <!-- insert ds:Signature element -->
    <!-- insert md:SPSSODescriptor element (see below) -->
    <md:Organization>
      <md:OrganizationName xml:lang="en">
        SAML Service Provider
      </md:OrganizationName>
      <md:OrganizationDisplayName xml:lang="en">
        SAML Service Provider @ Some Location
      </md:OrganizationDisplayName>
      <md:OrganizationURL xml:lang="en">
         http://www.sp.example.com/
      </md:OrganizationURL>
    </md:Organization>
    <md:ContactPerson contactType="technical">
      <md:SurName>SAML SP Support</md:SurName>
      <md:EmailAddress>mailto:saml-support@sp.example.com</md:EmailAddress>
    </md:ContactPerson>
  </md:EntityDescriptor>
```

2.1.3 Známe implementácie SAML

Kentor.AuthServices

Kentor Authentication Services je knižnica, ktorá pridáva podporu SAML 2.0 do webových aplikácií vytvorených na platforme ASP.NET a IIS, umožňujúca vystupovať danej aplikácii ako poskytovateľ služieb v rámci SAML 2.0.[12] Tento open source projekt vytvoril Anders Abel a ďalší prispievatelia.

Táto knižnica môže byť použitá tromi rôznymi spôsobmi:

- ako Http Modul načítaný do IIS pipeline, ktorý je kompatibilný s ASP.NET Web Forms aplikáciami,
- ako ASP.NET MVC kontrolér na lepšiu integráciu a zachytávanie chýb v aplikáciách,
- ako Owin Middleware na použitie s Owin Pipeline prípadne na integráciu s ASP.NET Identity.

Shibboleth

Shibboleth je open source softvérový balík založený na štandardoch na poskytovanie služieb jednotného prihlasovania medzi alebo v rámci organizácií.[13, 14] Patrí medzi projekty konzorcia Internet2. Implementuje široko používané štandardy identít, hlavne SAML na poskytovanie federačnej SSO a framework na výmenu atribútov.

SimpleSAMLphp

SimpleSAMLphp je aplikácia napísaná v PHP, ktorá sa zaoberá overovaním.[15] Projekt vedie UNINETT, má veľkú používateľskú základňu, nápomocnú komunitu

a veľký počet externých prispievateľov. Hlavným cieľom projektu je poskytovanie podpory pre:

- **SAML 2.0** ako poskytovateľ služieb,
- **SAML 2.0** ako poskytovateľ identít.

Okrem toho naviac podporuje aj iné protokoly identít a frameworky, ako sú napr. Shibboleth 1.3, A-Select, CAS, OpenID, WS-Federation alebo OAuth, a je ľahko rozšíriteľný.

2.2 Štandard OpenID

OpenID je, ako už napovedá názov, otvorený štandard, ktorý popisuje decentralizovaný spôsob autentifikácie používateľov.[16] Umožňuje používať jeden používateľský účet pre viacero webových služieb od rôznych poskytovateľov. Vznikol v roku 2005 ako vedľajší produkt pri vývoji blogovacieho systému systému LiveJournal od spoločnosti Six Apart. Postupom času začali tento štandard podporovať spoločnosti ako Google, Microsoft, AOL, Flickr, WordPress a ďalší.

Používatelia si zvolia svojho OpenID poskytovateľa a potom môžu účet u tohto poskytovateľa využívať na weboch, ktoré akceptujú prihlasovanie pomocou OpenID (OpenID akceptor). Štandard OpenID poskytuje framework na komunikáciu, ktorá musí prebehnúť medzi poskytovateľom identít a týmto akceptorom OpenID. OpenID identifikátor má tvar bežného URL alebo XRI (Extensible Resource Identifier), ktorý si používateľ zvolí.

Tretia generácia OpenID (február 2014), nazvaná ako OpenID Connect, je autentifikačná vrstva nad OAuth 2.0 frameworkom. Práve toto zlepšilo použitie OpenID v mobilných aplikáciach a v API webových služieb bez nutnosti použiť rozšírenia, ktoré dovtedy boli na to potrebné.

Na štandarde OpenID je postavená aj česká služba **mojeID** (2010), ktorá je prispôsobená pre prostredie českého internetu.[17] Prevádzkuje ju združenie CZ.NIC.

Služba dovoľuje používateľom vytvoriť si a spravovať internetovú identitu (súbor osobných údajov doplnený o prihlasovacie metódy a údaje). Táto identita môže byť potom využitá pri prihlasovaní na webových stránkach, ktoré mojeID podporujú. Služba mojeID poskytuje tri stupne overenia identity:

- základné overenie pomocou mailu a sms,
- pomocou doručovacej adresy,
- validácia kontaktu – overená žiadosť o validáciu (osobne s dokladom totožnosti, úradne overeným podpisom alebo digitálnym podpisom).

2.3 OAuth

OAuth je jeden z ďalších otvorených štandardov slúžiaci na autentifikáciu a autorizáciu oproti aplikačným rozhraniam (API) rôznych služieb.[18] Myšlienka vznikla v roku 2006 pri vývoji implementácii OpenID v internetovej službe Twitter. Vývojári sa zhodli na tom, že neexistuje vhodný otvorený štandard na delegáciu prístupu k API podobných služieb. V roku 2007 bola založená OAuth discussion group s cieľom navrhnúť vhodný štandard. Prvá finálna verzia bola uvedená ešte v tomto roku ako OAuth Core 1.0.

V súčasnosti je OAuth vo verzii 2.0, ktorá je zameraná na jednoduchosť a zároveň ponúka špecifické autorizačné toky pre rôzne aplikácie. Tento štandard používajú firmy ako Facebook, Google a Microsoft na overovanie prístupu k ich API.

2.4 Federácia eduID.cz

Česká akademická federácia identít, ktorá je postavená na projekte Shibboleth od Internet2, poskytuje svojim členom prostriedky pre vzájomné využívanie identít používateľov pri riadení prístupu k sieťovým službám pri rešpektovaní ochrany osobných údajov.[7] Operátorom federácie je organizácia CESNET (Czech Education and Scientific NETwork). Koordinuje dianie vo federácii a vykonáva v nej federačnú politiku.

CESNET – združenie, ktoré založili v roku 1996 verejné vysoké školy a Akademie věd České republiky a ktoré buduje, prevádzkuje a rozvíja českú národnú elektronickú infraštruktúru určenú pre potreby vedy, výskumu, vývoja a vzdelávania.[19]

Operátor federácie sa zároveň stará o beh federácie ako celku – uskutočňuje registráciu členov, poskytuje podporu a rieši bezpečnostné incidenty. Organizácie, ktoré prešli procesom registrácie sa stávajú členmi federácie. Tieto môžu poskytovať dáta o svojich používateľoch alebo poskytovať sieťové služby a aplikácie. Podľa toho delíme komponenty federácie na:

- **poskytovateľov identít** (IdP) – napojené k používateľským databázam, vykonávajú autentizáciu a poskytujú informácie o používateľoch,
- **poskytovateľov služieb** (SP) – poskytujú online služby a zdroje používateľom federácie.

Hlavnou výhodou pre používateľov poskytovateľov identít je, že svojím jediným používateľským menom a heslom spravovaným domovskou inštitúciou získajú prístup k radu sieťových služieb. Taktiež medzi výhody patrí aj to, že správcovia aplikácií

neudržujú autentizačné dáta ani autentizáciu nevykonávajú a autentizácia používateľa prebieha vždy v kontexte domovskej organizácie, citlivé autentizačné údaje neopúšťajú domovskú sieť.

2.5 Jednotné prihlasovanie na VUT v Brně

Na tejto univerzite je použitá aplikácia SimpleSAMLphp na umožnenie jednotného prihlasovania podľa štandardu SAML 2.0. Táto aplikácia zabezpečuje fungovanie členstva vo federácii eduID.cz, kde predstavuje poskytovateľa identít aj poskytovateľa služieb. Aplikácia beží na adrese `https://www.vutbr.cz/SSO`, konkrétne:

- `https://www.vutbr.cz/SSO/saml2/idp` – poskytovateľ identít,
- `https://www.vutbr.cz/SSO/saml2/sp` – poskytovateľ služieb.

VUT v Brně je podľa webu Metadata Explorer Tool[20] ako poskytovateľ identít okrem eduID.cz aj vo federáciách eduGAIN, InCommon Federation, SWAMID Federation a UK Access Management Federation. Na tejto stránke sú k dispozícii voľne stiahnuteľné metadáta. V praktickej časti tejto práce boli použité IdP metadáta VUT v Brně.

3 WEBOVÁ PLATFORMA A DATABÁZA

Pri vytváraní webových stránok a aplikácií je v súčasnosti veľmi dôležitá kompatibilita s mobilnými zariadeniami, pretože sú využívané rovnako často (niekedy aj častejšie) ako klasické počítače. Z tohto dôvodu je potrebné zvoliť platformu, ktorá natívne podporuje vytváranie obsahu pre tieto zariadenia, hlavne čo sa týka komfortu používania na zariadeniach s malými dotykovými displejmi. Medzi tieto platformy patrí aj ASP.NET od Microsoftu.

3.1 Technológia ASP.NET

ASP.NET je webový framework (v rámci .NET) postavený na CLR (Common Language Runtime) vhodný na budovanie dynamických webových stránok a aplikácií použitím programovacích jazykov ako C# alebo Visual Basic.NET.[21, 22] Aktuálna verzia ASP.NET 4.6 vyšla v júli 2015.

Vlastnosti:

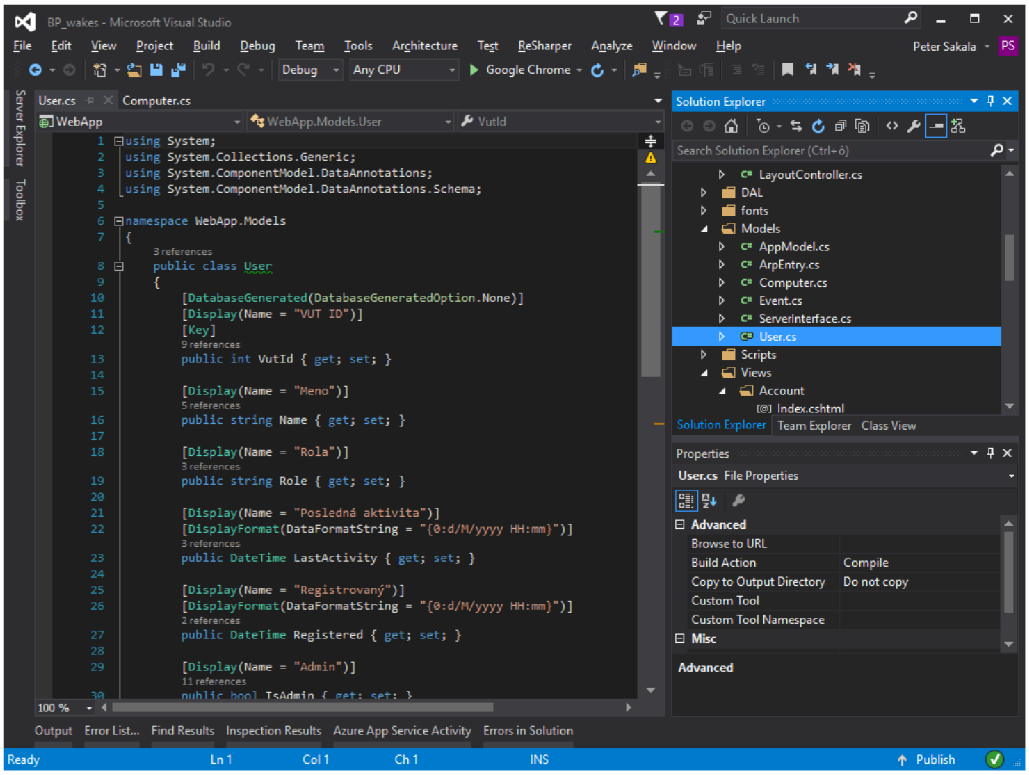
- veľká podpora od firmy Microsoft, oficiálna dokumentácia a ukážky použitia frameworku,
- integrované vývojové prostredie (IDE – Integrated Development Environment) – Visual Studio (obr. 3.1) aktuálne vo verzii 2015 (úplná verzia Community zadarmo),
- podpora veľkého počtu programovacích jazykov: C#, Perl, Python, Visual Basic.NET, JScript.NET, Managed C++,
- oproti skriptovacím jazykom ma výhodu v predkompilovaní aplikácií – rýchlejšie, možnosť zachytenia chýb pri vývoji.

ASP.NET používa v súčasnosti 2 modely:

- ASP.NET **Web Forms** a
- ASP.NET **MVC**.

Model Web Forms priniesol oproti pôvodnému ASP riešenie mnohých problémov vytvorením abstrakcie na vysokej úrovni nad bezstavovým webom a simulovaným stavovým modelom pre vývojárov. Boli predstavené koncepty ako self postback a ViewState. Najväčšia novinka bola, že pri použití Web Forms nie je potrebné písať kód na spracovávanie GET a POST metód, stačí ošetriť logiku daných prvkov na stránke – na základe udalostí (napr. kliknutie na tlačidlo). Tieto prvky sa dajú pridávať na stránku spôsobom „drag and drop“.

Naproti tomu model MVC (v rámci ASP.NET) je založený na rovnomennom architektonickom vzore MVC (Model View Controller).



Obr. 3.1: Ukážka vývojového prostredia Visual Studio 2015

3.1.1 ASP.NET MVC

ASP.NET MVC je ďalší webový framework v rámci ASP.NET a ako už bolo spomenuté, je založený na architektonickom vzore MVC, ktorý sa používa hlavne vo webových aplikáciách.[22] Tento vzor určuje rozdelenie aplikácie na tri časti:

- **Model** definuje biznis pravidlá, logiku a dáta. Pracuje nezávislo na ostatných dvoch častiach,
- **View** sa stará o zobrazenie alebo reprezentáciu dát z modelu,
- **Controller** spracováva požiadavky používateľa a stará sa o vstupnú logiku.

Hlavnou úlohou tohto vzoru je oddeliť tieto časti od seba. To umožňuje vývojárom zamerať sa iba na konkrétnu časť aplikácie.

Výhody ASP.NET MVC oproti Web Forms:

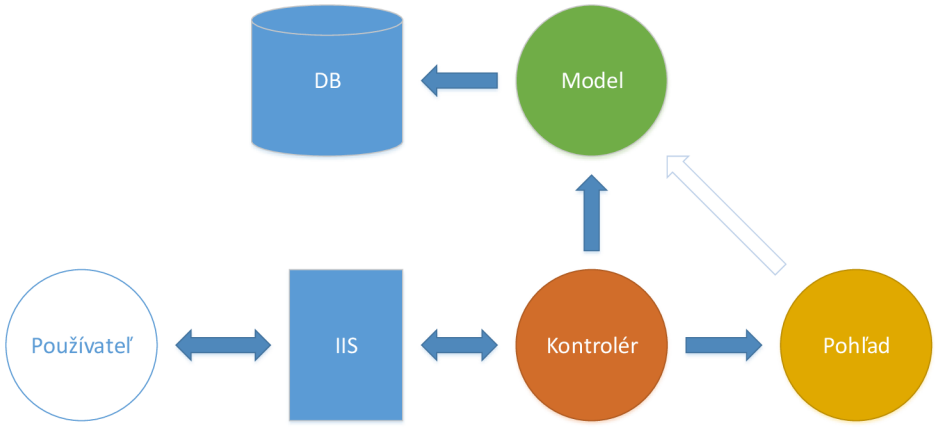
- Architektúra projektu – všetky súčasti sú logicky oddelené,

- Viacnásobná použiteľnosť kódu,
- Výkon – absencia ViewState, menšie veľkosti stránok,
- Úplná kontrola nad HTML,
- SEO (Search Engine Optimization) a URL routing – používateľsky prívetivé URL adresy,
- Rozšíriteľnosť,
- Existujúce funkcie z ASP.NET.

Nevýhodou ale je práve absencia programovania na základe udalostí a ViewState metódy, čo má za následok nutnosť väčších skúseností vývojára s vývojom webových aplikácii.

Ako to funguje (obr. 3.2):

1. Používateľ vytvorí požiadavku na nejaký zdroj na serveri (zadaním URL adresy do prehliadača).
2. Požiadavka dorazí ku správnemu kontroléru (vďaka routing engine).
3. Ak je to potrebné, kontrolér požiada model o dáta.
4. Model operuje v databáze (alebo v inom zdroji dát) a vracia dáta v požadovanom formáte naspäť do kontroléra.
5. Kontrolér určí správny pohľad (view), napr. zobrazenie tabuľky.
6. Kontrolér posunie dáta (z bodu 4) do určeného pohľadu (z bodu 5), ktorý bude vhodne týmito dátami vyplnený.
7. Kontrolér vráti pohľad naspať používateľovi, ktorému sa zobrazí v prehliadači.



Obr. 3.2: Grafické znázornenie modelu ASP.NET MVC

3.1.2 Front-end framework Bootstrap

Bootstrap je open source front-end knižnica, určená na vytáranie moderných webových stránok a aplikácií.[23] Zahŕňa v sebe HTML a CSS šablóny pre typografiu, formuláre, tlačidlá, navigačné a iné prvky prostredia a takisto JavaScript (JS) rozšírenia. Zameriava sa na jednoduchý vývoj dynamických stránok a webových aplikácií.

Aktuálne je vo verzii 3.3.6, v ktorej adoptuje filozofiu dizajnu mobile-first, to znamená, že pri vytváraní webu sa štandardne počíta s responzívnym dizajnom pre mobilné zariadenia. Responzívny dizajn umožňuje zobrazovať jednu webovú stránku na displejoch s rôznou veľkosťou uhlopriečky (smartfóny, tablety, PC). Bootstrap to zabezpečuje pomocou tzv. tekutej mriežky (fluid grid), kde sú uložené dané komponenty stránky a podľa toho, koľko priestoru je k dispozícii, zobrazí dané komponenty vedľa seba alebo pod sebou prípadne ich skryje.

3.2 Databáza

Databáza je množina štruktúrovaných dát alebo informácií uložených v počítačovom systéme.[24] Na získavanie informácií z databázy sa používa dopytovací jazyk, napr. SQL.

3.2.1 SQL

SQL (Structured Query Language) ako štruktúrovaný dopytovací jazyk slúži na manipuláciu s dátami v relačných databázach.[25] SQL príkazy sa delia nasledovne:

- na manipuláciu s dátami (SELECT, INSERT, UPDATE, DELETE, ...),
- na definíciu dát (CREATE, ALTER, DROP, ...),
- na úpravu prístupových práv (GRANT, REVOKE),
- na riadenie transakcií (START TRANSACTION, COMMIT, ROLLBACK).

Medzi relačné databázové systémy v súčasnosti patria napríklad MS SQL, MySQL, Oracle DB. Všetky systémy okrem štandardizovaného jazyka SQL využívajú aj svoje nadstavby (rozšírenia) napr. Transact-SQL (Microsoft), PL/SQL (Oracle) atď.

3.2.2 Entity Framework

Microsoft Entity Framework (EF), predtým ako súčasť .NET frameworku, je open source object-relational mapping (ORM) framework, ktorý dovoľuje vývojárom pracovať s relačnými dátami v databáze ako s objektami, čo eliminuje množstvo kódu

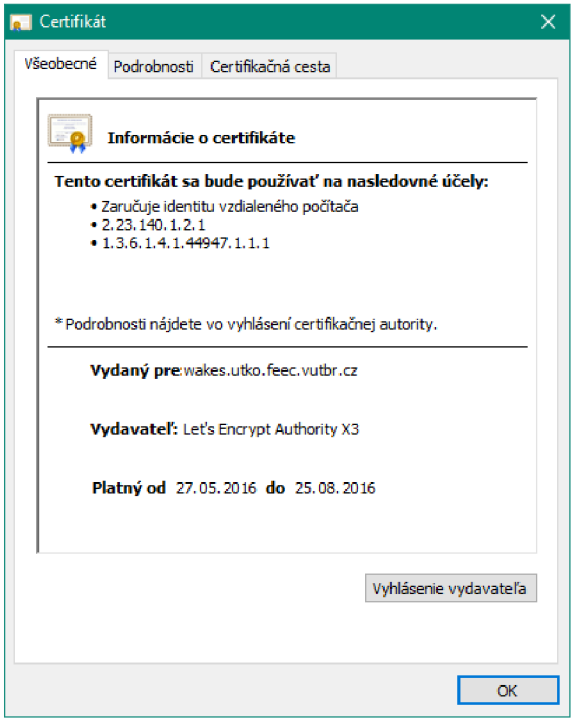
potrebného na prácu s databázou.[26] Pri použití EF vývojár zadáva dopyty pomocou LINQ (Language-Integrated Query) z ktorých následne dostane pevne definované objekty s ktorými môže manipulovať. Vývojárovi to dovoľuje venovať sa biznis logike danej aplikácie namiesto princípom prístupu k dátam.

3.3 Let's Encrypt certifikáty

Let's Encrypt je certifikačná autorita, ktorá poskytuje X.509 certifikáty zadarmo na TLS šifrovanie pomocou automatizovaného procesu dizajnovaného na eliminovanie súčasného zdĺhavého a komplexného procesu vytvárania, overovania, podpisovania, inštalácie a obnovenia certifikátov slúžiacich na zabezpečenie webových stránok (HTTPS).[27]

Projekt vznikol v roku 2014 na základe iniciatívy ľudí z Mozilla Foundation, Electronic Frontier Foundation (EFF) a University of Michigan. Od 12. apríla 2016 beží v plnej prevádzke a poskytuje certifikáty s maximálnou platnosťou 90 dní (obr. 3.3).

Existuje mnoho softvérových riešení na automatické obnovovanie certifikátov. Ich podstatou je nový protokol Automated Certificate Management Environment (ACME), na základe ktorého sa overí príslušnosť domény k danému serveru.



Obr. 3.3: Certifikát Let's Encrypt na serveri WAKES

4 WEBOVÁ APLIKÁCIA PRE UTKO

Vytvorená webová aplikácia pre Ústav telekomunikací slúži na vzdialené spúšťanie počítačov v tomto ústave, pričom je prístupná pre zamestnancov a doktorandov. Cieľom je umožniť vzdialený prístup k počítačom aj v prípade, že boli vypnuté. Počítače tak nebudú musieť byť zapnuté po celú dobu, ale iba vtedy ak ich budú chcieť používatelia využívať. Vo výsledku by mala aplikácie zvýšiť komfort používateľov a zároveň znížiť spotrebu energie.

Požadované vlastnosti:

- kompatibilita s desktopovými a mobilnými prehliadačmi – aplikácia má responzívny dizajn, prispôsobí sa veľkosti displeja použitého zariadenia,
- prihlasovanie sa do aplikácie pomocou VUT konta – nie je potrebná nová registrácia, používateľ si vystačí s už existujúcim VUT kontom,
- prístup z internetu – aplikácia bude prístupná odkiaľkoľvek,
- zabezpečený prenos (HTTPS) – kvôli bezpečnosti prenášaných dát medzi klientom a serverom,
- uchovávanie MAC adries počítačov v databáze – každý používateľ má pod svojim účtom zoznam svojich počítačov, ktoré chce zobúdzať.

4.1 Návrh dátového modelu (databáza)

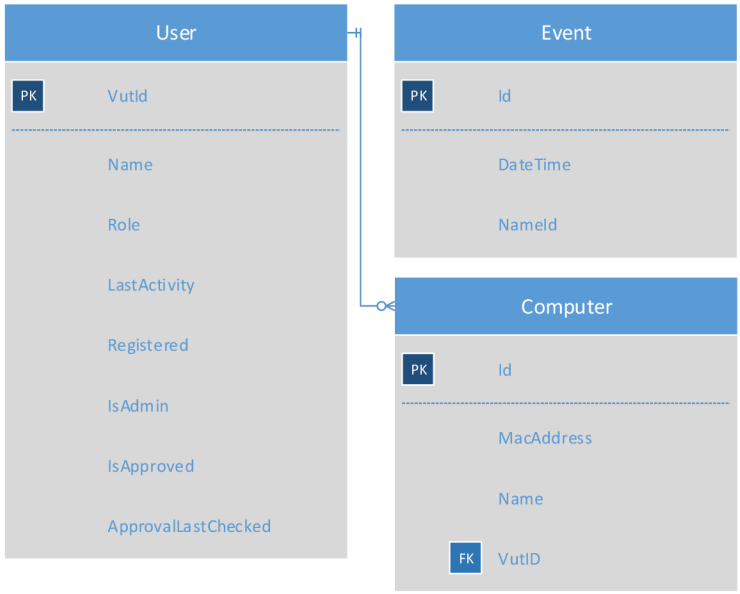
Vytvorená aplikácia na uchovávanie dát používa SQL databázu. Táto databáza je nasadená na jednej inštancii MS SQL Servera vo verzii Express, ktorá pobeží na tom istom serveri ako webový server (IIS).

Štruktúra databázy bola definovaná pomocou Entity Frameworku v ASP.NET MVC aplikácii (obr. 4.1). Modely, podľa ktorých sa vytvárajú tabuľky v databáze sú nasledovné:

- User – základné informácie o používateľovi (príloha A.1.1),
- Computer – názov a MAC počítača (príloha A.1.2),
- Event – Dáta o udalosti (príloha A.1.3).

Model User má v sebe zahrnutú kolekciu modelov Computer, to znamená, že používateľovi môžeme priradiť ľubovoľný počet počítačov – to zabezpečí previazanie tabuliek v databáze.

Vo vrstve prístupu k dátam resp. Data Access Layer (DAL) je definované (príloha A.3.1), ktoré modely sa ukladajú do databázy, prípadne akým spôsobom. Taktiež sa tu určuje názov databázového kontextu („WakesContext“).



Obr. 4.1: Štruktúra databázy aplikácie Wakes

Okrem týchto modelov boli definované aj iné, ktoré sa neukladajú v databáze a slúžia len na prenos informácií v rámci aplikácie.

AppModel (príloha A.1.4) definuje štruktúru dát získanú v Base Controller, pomocou ktorého sú niektoré pohľady vyplnené dátami (napr. pohľad Account/Index).

Model ServerInterface (príloha A.1.5) bol vytvorený na získavanie informácií o aktívnych sieťových rozhrania na serveri. Obsahuje v sebe aj potrebné metódy, ktoré naplnia list obsahujúci zoznam s informáciami o týchto rozhraniach.

Model ArpEntry (príloha A.1.6) bol vytvorený na získavanie MAC adries počítačov (pomocou záznamov protokolu ARP na serveri), na ktorých je spustená aplikácia a zároveň sú v rovnakej sieti ako server.

4.2 Návrh aplikácie

Aplikácia používa 6 kontrolérov (Controllers), z toho jeden je bázový, z ktorého dedia ostatné 4, a jeden je použitý na dynamickú úpravu horného menu v aplikácii.

Base controller (príloha A.2.1) bol vytvorený preto, že mnoho dát bolo v rámci aplikácie spoločných (napr. informácie o prihlásenom používateľovi, ktoré buď nájde v databáze alebo využije pomocnú triedu `UserHelper.cs` (príloha A.5.2)). Tento

kontrolér sa stará o získanie týchto dát a poskytuje ich kontrolérom, ktoré z neho dedia.

Home controller (príloha A.2.2) sa stará o všetky webové stránky, ku ktorým je prístup aj bez prihlásenia, napr. domovská stránka, stránka s návodom a pod.).

Account controller (príloha A.2.3) má na starosti iba stránku účtu používateľa a rieši jeho automatické vytváranie prípadne mazanie.

App controller (príloha A.2.4) riadi hlavnú funkciu aplikácie, tzn. zobrazuje počítače používateľov a umožňuje posielanie Magic Packetu vďaka pomocnej triede `WakeOnLan.cs` (príloha A.5.1).

Admin controller (príloha A.2.5) definuje možnosti administrácie pre používateľov, ktorí majú administrátorské práva. Umožňuje administrátorom rušiť účty, kontrolovať oprávnenie používania aplikácie alebo zobraziť denník (log).

Layout controller (príloha A.2.6) iba posiela dáta o aktuálnom používateľovi čiastočnému pohľadu (partial view), ktorý predstavuje menu. Umožňuje to zobrazenie iba relevantných položiek v menu.

Aplikácia je navrhnutá tak, že neprihlásenému používateľovi sa môžu zobraziť iba základné stránky. Po prihlásení sa načítajú údaje o používateľovi pomocou SAML a vďaka externému skriptu na serveri VUT overí príslušnosť používateľa k ústavu a jeho rolu (ako XML dáta). Ak je podľa týchto dát používateľ oprávnený používať túto aplikáciu, automaticky sa mu vytvorí účet po prvom spustení stránky „Moje PC“, kedy sa tieto údaje uložia do databázy (vytvorí sa účet). Toto oprávnenie sa kontroluje maximálne raz za 24 hodín. Registrovaný používateľ si môže ľubovoľne pridávať svoje počítače, nezávisle na ostatných používateľoch.

Administrácia umožňuje spravovať registrovaných používateľoch, zobrazovať informácie o nich alebo zobraziť denník. Po prístupe sa overuje oprávnenie všetkých používateľov, u ktorých bolo posledné overenie pred viac ako 5 dňami. K administrácii majú prístup len používatelia, ktorým boli pridelene práva (iným administrátorom).

V prílohe A.4.1 je ukážka pohľadu, konkrétne stránky so zoznamom počítačov používateľa.

4.3 Vytvorenie poskytovateľa služieb (SAML)

Z dôvodu použitia webovej platformy ASP.NET MVC bolo vybrané riešenie Kentor.AuthServices MVC. Tento softvér bol pridaný do aplikácie pomocou Visual Studia ako NuGet balík. Potom bol vygenerovaný self-signed certifikát (platný do roku 2040), ktorým sa podpisujú žiadosti od SP. Certifikát sa nainštaloval na server WAKES a v konfiguračnom súbore „Web.config“ sa definovalo sériové číslo certifikátu,

podľa ktorého ho aplikácia nájde. V tomto stave aplikácia vie vygenerovať metadáta SP (prístupné na adrese `https://wakes.utko.feec.vutbr.cz/AuthServices/`).

Tieto metadáta boli sprístupnené zodpovednému človeku na VUT v Brně, ktorý ich pridal do IdP. Zároveň bol v aplikácii podľa metadát definovaný tento IdP, pomocou ktorého sa vykonáva overovanie používateľov. Po dokončení týchto úkonov bolo funkčné prihlasovanie v aplikácii pomocou VUT konta.

4.4 Virtuálny server WAKES

Virtuálny server je spustený na virtualizačnej technológii od VMware. Nainštalovaný operačný systém je Microsoft Windows Server 2012 R2 Standard.

Vlastnosti:

- podpora použitých webových technológií – ASP.NET 4.6 v rámci IIS servera,
- podpora MS SQL servera – výhoda automatických aktualizácii pomocou Microsoft Update,
- jednoduchá správa systému – možnosť automatickej údržby a aktualizácií,
- technická podpora – Windows Server 2012 R2 má predĺženú technickú podporu do 10. 1. 2023 to znamená, že do tohto dátumu bude dostávať bezpečnostné záplaty.

Špecifikácia virtuálneho servera:

- počet jadier CPU: 2,
- RAM: 2 GB,
- HDD: 100 GB (dynamicky sa zväčšujúci),
- virtuálna sieťová karta (verejná IP).

4.4.1 Použité roly a softvér

Po nainštalovaní operačného systému boli pridané roly Web Server (IIS) s podporou .NET Framework 4.5 vrátane nástrojov na správu a FTP Server používaný na upload webovej aplikácie na server.

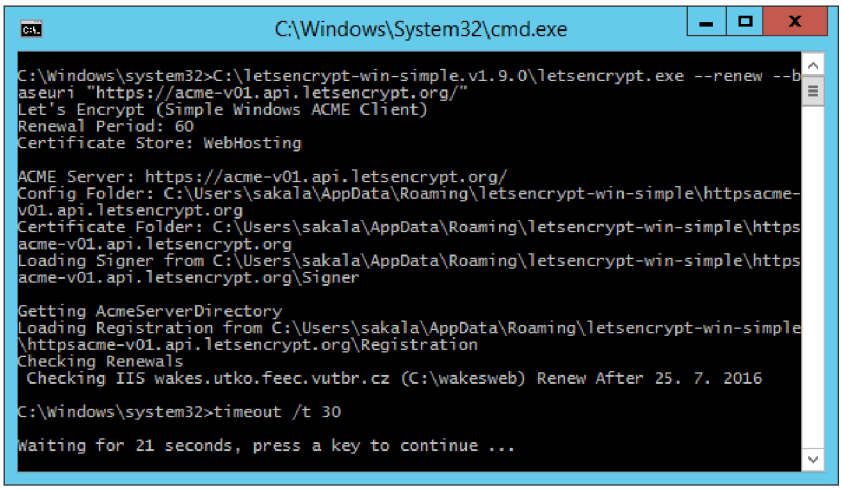
Nainštalovaný bol SQL Server Express 2014 s jednou inštanciou (s názvom MSSQLSERVER) na poskytovanie databázových služieb aplikácii. Ďalej bol nainštalovaný modul do webového servera – IIS URL Rewrite 2.0, ktorý umožňuje vytvoriť pravidlá na prepisovanie URL. Na automatické obnovenie certifikátu Let's Encrypt bol využitý nástroj letsencrypt-win-simple. Počas testovania webovej aplikácie boli využívane programy Wireshark na analýzu sieťovej prevádzky a LINQPad na kontrolu dát v databáze.

4.4.2 Nastavenie firewallu

Firewall v systéme Windows Server je v súčasnosti nastavený tak, aby sa na webovú aplikáciu (HTTP/HTTPS) dalo pripojiť iba zo siete VUT (147.229.0.0/16) a Eduroam v rámci VUT. V budúcnosti by sa to malo zmeniť na neobmedzený prístup z internetu. Okrem toho je povolený protokol RDP (na správu servera) a FTP (na upload aplikácie) zo siete VUT.

4.4.3 Automatické obnovenie SSL certifikátu

Pomocou nástroja letsencrypt-win-simple bol prvý certifikát manuálne vytvorený. Po nainštalovaní certifikátu nástroj automaticky vytvoril úlohu v Plánovači úloh, ktorá sa opakuje každý deň a overuje, či nie je aktuálny certifikát starší ako 60 dní (Renewal Period). Ak áno, certifikát sa obnoví. Na obrázku 4.2 je vidieť manuálne spustenú kontrolu platnosti certifikátu na serveri.

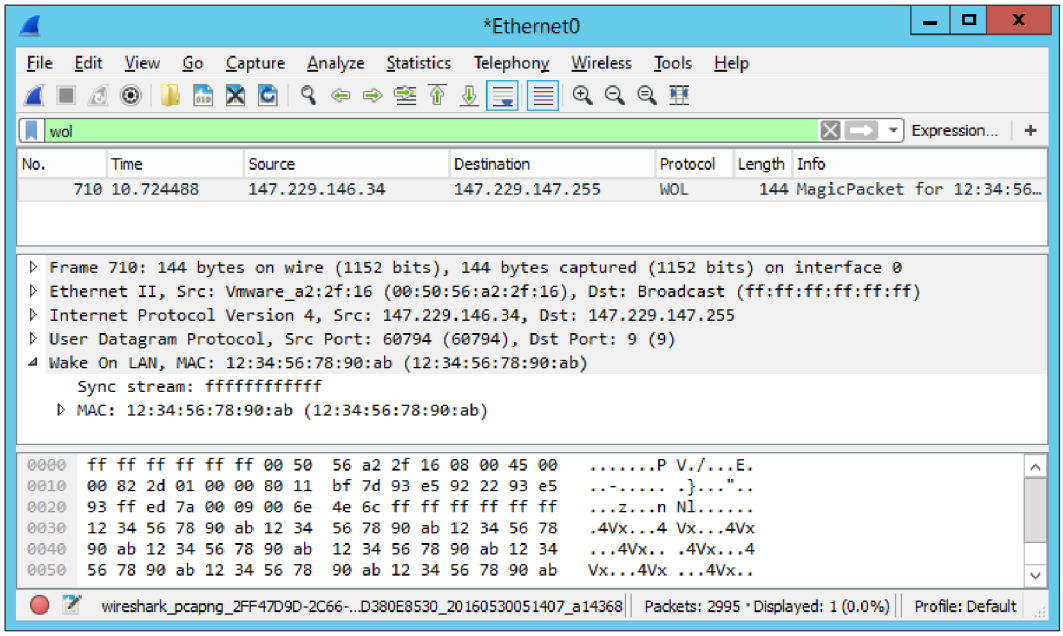


Obr. 4.2: Kontrola platnosti certifikátu Let's Encrypt

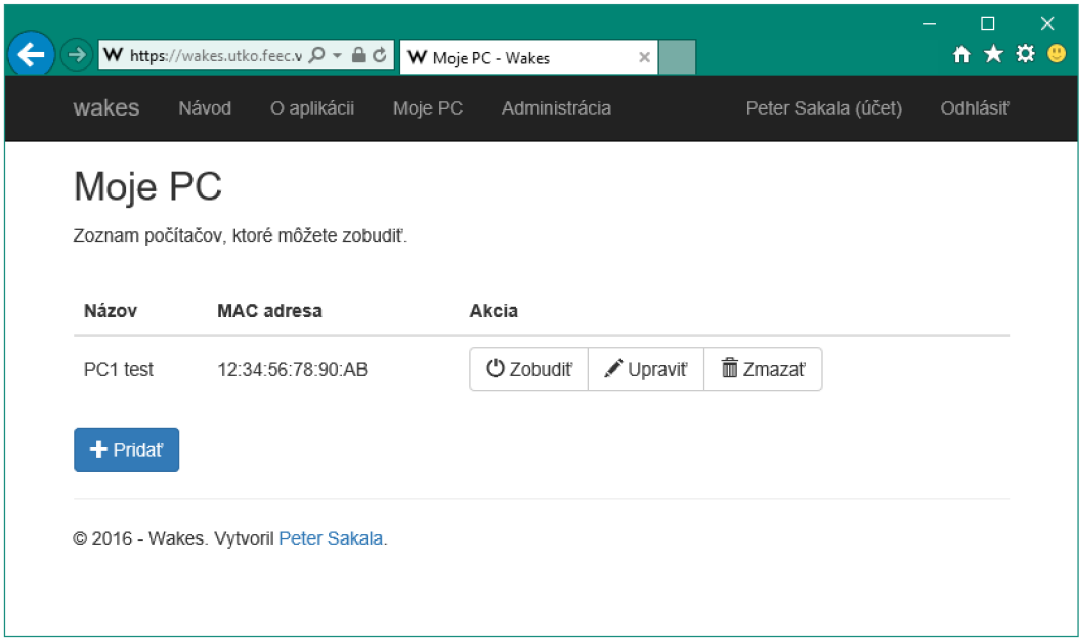
4.5 Výsledky testovania

Po vytvorení funkčnej verzie aplikácie bola táto spustená do testovacej prevádzky (pomocou brány firewall obmedzený prístup len na sieť VUT). Počas testovania sa do aplikácii zaregistrovalo niekoľko zamestnancov a doktorandov, ktorí aplikáciu vyskúšali. Aplikácia úspešné posiela Magic Pakety (obr. 4.3). Taktiež bolo úspešne otestované posielanie Magic Paketov cez viac ako jednu sieťovú kartu. Bola overená kompatibilita aplikácie s rôznymi operačnými systémami (Windows, Android, iOS) a veľkosťami displejov (smartfón, tablet, PC). Na obrázkoch 4.4 a 4.5 je na

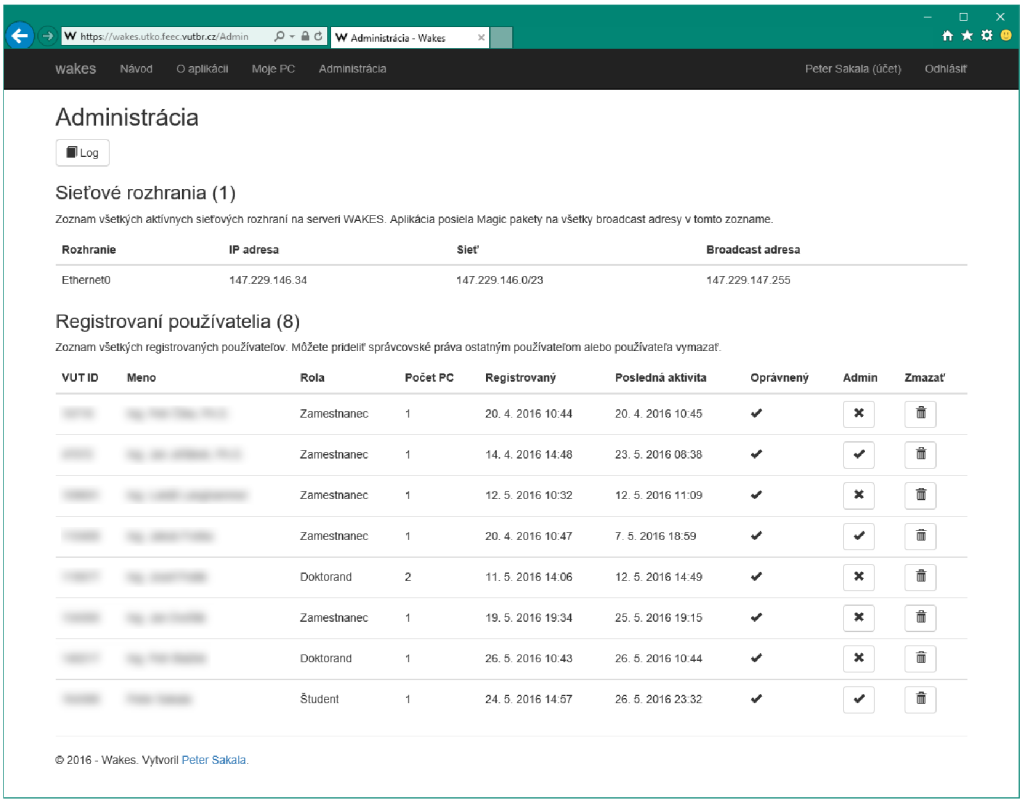
ukážku zobrazená stránka „Moje PC“ a „Administrácia“ vo webovom prehliadači na PC (Internet Explorer). Na obrázkoch 4.6 a 4.7 je ukážka aplikácie na mobilných operačných systémoch Windows 10 Mobile a Android 5.1.



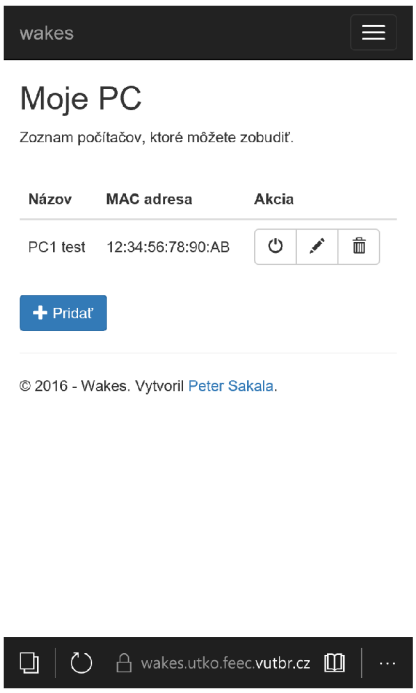
Obr. 4.3: Magic Packet zachytený pri teste aplikácie



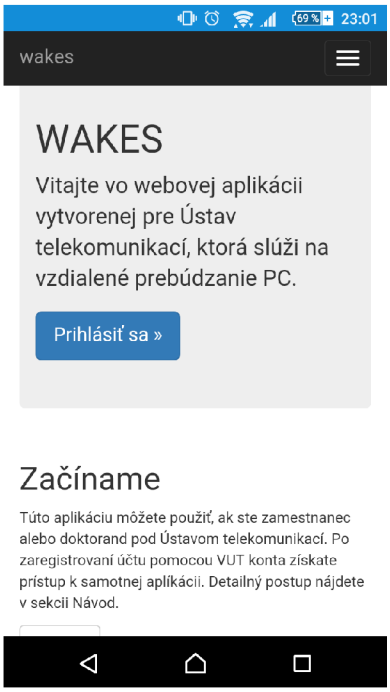
Obr. 4.4: Ukážka aplikácie zobrazenej na počítači



Obr. 4.5: Stránka administrácie



Obr. 4.6: Ukážka webovej aplikácie na smartfóne (Windows 10 Mobile)



Obr. 4.7: Ukážka webovej aplikácie na smartfóne (Android 5.1.1)

5 ZÁVER

V tejto bakalárskej práci je podrobne popísaná technológia Wake on LAN, princíp jej fungovania a jej využitie v praxi. Ďalej sú popísané požiadavky, ktoré musia byť splnené, aby sa dala táto technológia prakticky použiť. Taktiež sú popísané najčastejšie scenáre, kedy a akým spôsobom sa aplikuje technológia WoL, či už išlo o jednoduchú sieť LAN alebo internet. Aj keď táto technológia nie je nová, stále má reálne využitie v praxi, o čom vypovedajú aj výsledky tejto práce.

V časti zaoberajúcej sa jednotnému prihlasovaniu (SSO) je prehľad súčasných technológií a popis federácie eduID.cz ako aj fungovanie jednotného prihlasovania externých aplikácii v rámci VUT v Brně. Taktiež bol popísaný štandard SAML vo verzii 2.0, ktorý používajú mnohé riešenia jednotného prihlasovania ako napr. Kentor.AuthServices, Shibboleth alebo SimpleSAMLphp. V súvislosti s tým bol popísaný aj štandard X.509, ktorý definuje kryptografické systémy pre jednoduché podpisovanie a taktiež formát certifikátov.

Ďalšia časť tejto práce sa venuje webovým technológiám, ktoré boli neskôr použité v samotnom návrhu aplikácie v praktickej časti práce. Bola využitá moderná platforma ASP.NET MVC od Microsoftu v kombinácii s front-end frameworkom Bootstrap. Kombinácia týchto dvoch technológií umožňuje skonštruovať ľubovoľnú webovú aplikáciu s responzívnym dizajnom. Boli popísané aj databázové systémy a s tým súvisiaci Entity Framework.

Navrhnutá aplikácia pre Ústav telekomunikací, ktorá beží na virtuálnom serveri tohto ústavu, slúži na zobúdzanie počítačov patriacich pod tento ústav, najmä počítače v kanceláriách zamestnancov a doktorandov. Pri návrhu boli využité naštudované technológie. Prihlasovanie používateľov je riešené jednotným prihlasovaním VUT v Brně, čiže pomocou VUT konta. Zabezpečenie prenosu medzi webovým serverom a klientom je uskutočnené pomocou certifikátu Let's Encrypt, ktorý sa automaticky obnovuje. Aplikácia bola úspešne uvedená do prevádzky a úspešne otestovaná.

LITERATÚRA

[1] *IBM Announces Universal Management – Industry's Most Comprehensive Tools to Lower Total Cost of Ownership* [online]. 15. 4. 1998 [cit. 1. 12. 2015]. Dostupné z URL: <https://www-03.ibm.com/press/us/en/pressrelease/2705.wss>.

[2] *Magic Packet Technology – White Paper* [online]. 11. 1995 [cit. 1. 12. 2015]. Dostupné z URL: <http://support.amd.com/TechDocs/20213.pdf>.

[3] KUROSE, James F. a Keith W. ROSS. *Počítačové sítě.* 1. vyd. Brno: Computer Press, 2014, 622 s. ISBN 9788025138250. [cit. 1. 12. 2015]. Kapitola 5.4.1, Adresování linkové vrstvy a protokol ARP, s. 365–561.

[4] *WakeOnLAN – The Wireshark Wiki* [online]. Posledná úprava 3. 11. 2014 [cit. 1. 12. 2015]. Dostupné z URL: <https://wiki.wireshark.org/WakeOnLAN>.

[5] BOUŠKA, Petr. *Wake on LAN – lokální i vzdálený subnet* [online]. 10. 8. 2008 [cit. 1. 12. 2015]. Dostupné z URL: <http://www.samuraj-cz.com/clanek/wake-on-lan-lokalni-i-vzdaleny-subnet>.

[6] *Jak na internet – Jednotné přihlašování* [online]. [cit. 1. 12. 2015]. Dostupné z URL: <http://www.jaknainternet.cz/page/1189/jednotne-prihlasovani>.

[7] *Česká akademická federace identit eduID.cz: O federaci* [online]. Posledná úprava 30. 5. 2013 [cit. 1. 12. 2015]. Dostupné z URL: <https://www.eduid.cz/cs/index>.

[8] LOCKHART, Harold. *Demystifying SAML* [online]. 11. 9. 2005 [cit. 1. 12. 2015]. Dostupné z URL: <http://www.oracle.com/technetwork/articles/entarch/saml-084342.html>.

[9] TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU (ITU-T). *Information technology — Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks* [online]. 10. 2012 [cit. 1. 5. 2016]. Dostupné z URL: <https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-X.509-201210-I!!PDF-E&type=items>.

[10] INTERNET ENGINEERING TASK FORCE (IETF). *RFC 5280 - Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List*

(CRL) Profile [online]. 5. 2008 [cit. 1. 5. 2016]. Dostupné z URL: <https://tools.ietf.org/html/rfc5280>.

[11] ORGANIZATION FOR THE ADVANCEMENT OF STRUCTURED INFORMATION STANDARDS (OASIS). *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0* [online]. 15. 3. 2005 [cit. 1. 5. 2016]. Dostupné z URL: <https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf>.

[12] ANDERS, Abel. *GitHub – KentorIT/authservices: Saml2 Authentication services for ASP.NET* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://github.com/KentorIT/authservices>.

[13] PAVLÍK, Jiří. *Řekni Shibboleth :-) webové jednotné přihlašování nejen pro vzdálený přístup v knihovnách.* Knihovna [online]. 2009, roč. 20, č. 1, s. 110-112 [cit. 1. 12. 2015]. Dostupné z URL: <http://oldknihovna.nkp.cz/knihovna91/pavlik.htm>. ISSN 1802-8772.

[14] *Shibboleth* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://shibboleth.net/>.

[15] *SimpleSAMLphp* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://simplesamlphp.org/>.

[16] *Specifications & Developer Information | OpenID* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <http://openid.net/developers/specs/>.

[17] *MojeID – Hlavní výhody služby* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://www.mojeid.cz/page/1856/hlavni-vyhody-sluzby/>.

[18] *Documentation – OAuth* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <http://oauth.net/documentation/>.

[19] *Základní informace o sdružení CESNET* [online]. [cit. 1. 12. 2015]. Dostupné z URL: <https://www.cesnet.cz/sdruzeni/zakladni-informace-o-sdruzeni-cesnet>.

[20] RESEARCH AND EDUCATION FEDERATIONS (REFEDS) *Metadata Explorer Tool* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://met.refeds.org>.

[21] *Get Started with ASP.NET* [online]. [cit. 1. 12. 2015]. Dostupné z URL: <http://www.asp.net/get-started>.

[22] *WebForms vs. MVC – CodeProject* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <http://www.asp.net/get-started>.

[23] *Bootstrap – The world's most popular mobile-first and responsive front-end framework.* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <http://getbootstrap.com/>.

[24] Wikipédia a jej prispievatelia *SQL* [online]. Posledná úprava 10. 12. 2015 [cit. 12. 12. 2015]. Dostupné z URL: <https://en.wikipedia.org/w/index.php?title=Special:CiteThisPage&page=SQL&id=694609587>.

[25] *What is SQL?* [online]. [cit. 1. 12. 2015]. Dostupné z URL: <http://www.sqlcourse.com/intro.html>.

[26] *Entity Framework – Home* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://entityframework.codeplex.com/>.

[27] *Let's Encrypt – Free SSL/TLS Certificates* [online]. [cit. 1. 5. 2016]. Dostupné z URL: <https://letsencrypt.org/>.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

ARP	Address Resolution Protocol
BIOS	Basic Input Output System
DNS	Domain Name System
EF	Entity Framework
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IDE	Integrated Development Environment
IdP	Identity Provider
IP	Internet Protocol
LAN	Local Area Network
LINQ	Language-Integrated Query
MAC	Media Access Control
NIC	Network Interface Card
PCI	Peripheral Component Interconnect
PCIe	PCI Express
SAML	Security Assertion Markup Language
SDB	Subnet Directed Broadcasts
SOAP	Simple Object Access Protocol
SP	Service Provider
SSO	Single Sign-On
SQL	Structured Query Language
TLS	Transport Layer Security
TCP	Transmission Control Protocol

UDP	User Datagram Protocol
UEFI	Unified Extensible Firmware Interface
VPN	Virtual Private Network
WAN	Wide Area Network
WLAN	Wireless Local Area Network
WoL	Wake on LAN
WoWLAN	Wake on WLAN
WWW	World Wide Web

ZOZNAM PRÍLOH

A ZDROJOVÝ KÓD APLIKÁCIE

Súbory s príponou .cs sú napísané v jazyku C#.
Súbory s príponou .cshtml sú napísané v jazyku Razor.

A.1 Modely/Models

A.1.1 User.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace WebApp.Models
{
    public class User
    {
        [DatabaseGenerated(DatabaseGeneratedOption.None)]
        [Display(Name = "VUT ID")]
        [Key]
        public int VutId { get; set; }
        [Display(Name = "Meno")]
        public string Name { get; set; }
        [Display(Name = "Rola")]
        public string Role { get; set; }
        [Display(Name = "Posledná aktivita")]
        [DisplayFormat(DataFormatString = "{0:d/M/yyyy HH:mm}")]
        public DateTime LastActivity { get; set; }
        [Display(Name = "Registrovaný")]
        [DisplayFormat(DataFormatString = "{0:d/M/yyyy HH:mm}")]
        public DateTime Registered { get; set; }
        [Display(Name = "Admin")]
        public bool IsAdmin { get; set; }
        [Display(Name = "Oprávnený")]
        public bool IsApproved { get; set; }
        public DateTime ApprovalLastChecked { get; set; }
        public virtual ICollection<Computer> Computers { get; set; }

    }
}
```

A.1.2 Computer.cs

```
using System.ComponentModel.DataAnnotations;

namespace WebApp.Models
{
    public class Computer
    {
        public int Id { get; set; }

        [Display(Name = "MAC adresa")]
        [RegularExpression(
            "^(?:[0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}
            |(?:[0-9a-fA-F]{2}-){5}[0-9a-fA-F]{2}|
            (?:[0-9a-fA-F]{2}){5}[0-9a-fA-F]{2}$",
            ErrorMessage = "Nesprávne zadaná MAC adresa")]
        [Required(ErrorMessage = "Povinná položka")]
        public string MacAddress { get; set; }

        [Display(Name = "Názov")]
        [Required(ErrorMessage = "Povinná položka")]
        [StringLength(30, MinimumLength = 3, ErrorMessage = "Povolený počet znakov 3 až 30")]
        public string Name { get; set; }
    }
}
```

A.1.3 Event.cs

```
using System;
using System.ComponentModel.DataAnnotations;

namespace WebApp.Models
{
    public class Event
    {
        public int Id { get; set; }
        [Display(Name = "Dátum a čas")]
        public DateTime DateTime { get; set; }
        [Display(Name = "Popis udalosti")]
        public string Description { get; set; }
        [Display(Name = "Meno (VUT ID)")]
        public string NameId { get; set; }
    }
}
```

A.1.4 AppModel.cs

```
using System;
namespace WebApp.Models
{
    public class AppModel
    {
        public int VutId { get; set; }
        public string Name { get; set; }
        public string RoleName { get; set; }
        public bool IsAuthenticated { get; set; }
        public bool IsRegistered { get; set; }
        public bool IsApproved { get; set; }
        public bool IsAdmin { get; set; }
        public DateTime Registered { get; set; }
        public DateTime LastLoggedIn { get; set; }
    }
}
```

A.1.5 ServerInterface.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Net.NetworkInformation;

namespace WebApp.Models
{
    public class ServerInterface
    {
        private ServerInterface(UnicastIPAddressInformation ipInformation, string name)
        {
            IpAddress = ipInformation.Address;
            var addressInt = BitConverter.ToInt32(ipInformation.Address.GetAddressBytes(), 0)
                ;
            var maskInt = BitConverter.ToInt32(ipInformation.IPv4Mask.GetAddressBytes(), 0);
            var broadcastInt = addressInt | ~maskInt;
            var subnetInt = addressInt & maskInt;
            BroadcastAddress = new IPAddress(BitConverter.GetBytes(broadcastInt));
            SubnetAddress = new IPAddress(BitConverter.GetBytes(subnetInt));
            Prefix = ipInformation.PrefixLength;
            Name = name;
        }

        public IPAddress IpAddress { get; }
        public IPAddress BroadcastAddress { get; }
        public IPAddress SubnetAddress { get; }
        public int Prefix { get; }
        public string Name { get; }

        public static IEnumerable<ServerInterface> Interfaces => (
            from ni in System.Net.NetworkInformation.NetworkInterface.GetAllNetworkInterfaces
                ()
            where ni.NetworkInterfaceType != NetworkInterfaceType.Loopback
            where ni.OperationalStatus == OperationalStatus.Up
```

```
34              where ni.NetworkInterfaceType == NetworkInterfaceType.Ethernet
35              let name = ni.Name
36              let interfaceUnicastAddresses = ni.GetIPProperties().UnicastAddresses
37              from ip in interfaceUnicastAddresses
38              where ip.Address.AddressFamily == System.Net.Sockets.AddressFamily.InterNetwork
39              select new ServerInterface(ip, name)).ToList();
40      }
41  }
```

A.1.6 ArpEntry.cs

```
1   using System.Collections.Generic;
2   using System.Diagnostics;
3   using System.Linq;
4   using System.Net;
5   using System.Net.NetworkInformation;
6
7   namespace WebApp.Models
8   {
9       public class ArpEntry
10      {
11          private ArpEntry(IPAddress ip, PhysicalAddress mac)
12          {
13              IpAddress = ip;
14              MacAddress = mac;
15          }
16
17          public IPAddress IpAddress { get; }
18          public PhysicalAddress MacAddress { get; }
19
20          public static IEnumerable<ArpEntry> ArpEntries => (
21              from arp in GetArpResult().Split('\n', '\r')
22              where !string.IsNullOrEmpty(arp)
23              select (from piece in arp.Split(' ', '\t')
24                  where !string.IsNullOrEmpty(piece)
25                  select piece).ToArray()
26              into pieces
27              where pieces.Length == 3
28              select new ArpEntry(IPAddress.Parse(pieces[0]),
29                  PhysicalAddress.Parse(pieces[1].ToUpperInvariant()))).ToList();
30
31          private static string GetArpResult()
32          {
33              var p = Process.Start(new ProcessStartInfo("arp", "-a")
34              {
35                  CreateNoWindow = true,
36                  UseShellExecute = false,
37                  RedirectStandardOutput = true
38              });
39
40              if (p == null) return string.Empty;
41
42              var output = p.StandardOutput.ReadToEnd();
43              p.Close();
44              return output;
45          }
46      }
47  }
```

A.2 Kontroléry/Controllers

A.2.1 BaseController.cs

```
1   using System;
2   using System.Data.Entity;
3   using System.Linq;
4   using System.Web.Mvc;
5   using WebApp.DAL;
6   using WebApp.Models;
7
8   namespace WebApp.Controllers
9   {
10      public class BaseController : Controller
```

```
11      {
12          protected readonly WakesContext Db;
13          protected readonly AppModel Am;
14          protected readonly User CurrentUser;
15
16          protected BaseController()
17          {
18              Am = new AppModel {IsAuthenticated = System.Web.HttpContext.Current.User.Identity
                    .IsAuthenticated};
19
20              if (!Am.IsAuthenticated)
21              {
22                  Am.IsRegistered = false;
23              }
24              else
25              {
26                  Db = new WakesContext();
27                  Am.VutId = UserHelper.VutIdFromSaml;
28                  Am.IsRegistered = Db.Users.Any(u => u.VutId == Am.VutId);
29
30                  if (!Am.IsRegistered)
31                  {
32                      Am.Name = UserHelper.NameFromSaml;
33                      Am.RoleName = UserHelper.LoggedUserRoleName;
34                      Am.IsApproved = UserHelper.LoggedUserIsApproved;
35                  }
36                  else
37                  {
38                      CurrentUser = Db.Users.Find(Am.VutId);
39                      CurrentUser.LastActivity = DateTime.Now;
40
41                      if (CurrentUser.ApprovalLastChecked <= DateTime.Now.AddDays(-1) || !
                            CurrentUser.IsApproved)
42                      {
43                          CurrentUser.IsApproved = UserHelper.LoggedUserIsApproved;
44                          CurrentUser.Role = UserHelper.LoggedUserRoleName;
45                          CurrentUser.ApprovalLastChecked = DateTime.Now;
46                      }
47                      if (CurrentUser.Name != UserHelper.NameFromSaml)
48                      {
49                          CurrentUser.Name = UserHelper.NameFromSaml;
50                      }
51                      Db.Entry(CurrentUser).State = EntityState.Modified;
52                      Db.SaveChanges();
53
54                      Am.Name = CurrentUser.Name;
55                      Am.VutId = CurrentUser.VutId;
56                      Am.RoleName = CurrentUser.Role;
57                      Am.IsApproved = CurrentUser.IsApproved;
58                      Am.IsAdmin = CurrentUser.IsAdmin;
59                      Am.LastLoggedIn = CurrentUser.LastActivity;
60                      Am.Registered = CurrentUser.Registered;
61                  }
62              }
63          }
64
65          protected void Log(string descr) {
66          Db.Log.Add(new Event
67              {
68                  DateTime = DateTime.Now,
69                  Description = descr,
70                  NameId = $"{Am.Name} ({Am.VutId})"
71              });
72          }
73      }
74  }
```

A.2.2 HomeController.cs

```
1  using System.Linq;
2  using System.Web.Mvc;
3  using WebApp.Models;
4
5  namespace WebApp.Controllers
6  {
7      public class HomeController : BaseController
8      {
9          public ActionResult Index()
```

```
10          {
11              return View(Am);
12          }
13
14          public ActionResult About()
15          {
16              return View();
17          }
18
19          public ActionResult Manual()
20          {
21              var networks = ServerInterface.Interfaces.Select(network =>
22              $"{network.SubnetAddress}/{network.Prefix}").ToList();
23              return View(networks);
24          }
25      }
26  }
```

A.2.3 AccountController.cs

```
1  using System.Net;
2  using System.Web.Mvc;
3
4  namespace WebApp.Controllers
5  {
6      public class AccountController : BaseController
7      {
8          [Authorize]
9          public ActionResult Index()
10         {
11             return View(Am);
12         }
13
14         [Authorize]
15         public ActionResult Start()
16         {
17             if (Am.IsRegistered)
18             {
19                 if (CurrentUser.IsApproved)
20                 {
21                     return RedirectToAction("Index", "App");
22                 }
23             }
24             return RedirectToAction("Index");
25         }
26
27         [Authorize]
28         public ActionResult Unregister()
29         {
30             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
31
32             return View();
33         }
34
35         [ActionName("Unregister"), Authorize, HttpPost, ValidateAntiForgeryToken]
36         public ActionResult UnregisterConfirmed()
37         {
38             Db.Computers.RemoveRange(CurrentUser.Computers);
39             Db.Users.Remove(CurrentUser);
40             Log($"Používateľ {Am.Name} ({Am.VutId}) sa odregistroval.");
41
42             Db.SaveChanges();
43             return RedirectToAction("Index");
44         }
45     }
46 }
```

A.2.4 AppController.cs

```
1  using System;
2  using System.Data.Entity;
3  using System.Linq;
4  using System.Net;
5  using System.Net.NetworkInformation;
6  using System.Web.Mvc;
7  using System.Web.WebPages;
```

```
 8 using WebApp.Models;
 9
10 namespace WebApp.Controllers
11 {
12     public class AppController : BaseController
13     {
14         [Authorize]
15         public ActionResult Index()
16         {
17             if (!Am.IsRegistered)
18             {
19                 if (Am.IsApproved)
20                 {
21                     Register();
22                     return RedirectToAction("Index");
23                 }
24                 return RedirectToAction("Index", "Account");
25             }
26             if (!Am.IsApproved) return RedirectToAction("Index", "Account");
27
28             ViewBag.MacAvailable = MacOfCurrentPcAvailableAndNotUsed();
29             return View(CurrentUser.Computers.ToList());
30         }
31
32         private void Register()
33         {
34             Db.Users.Add(new User
35             {
36                 VutId = Am.VutId,
37                 Name = Am.Name,
38                 IsAdmin = false,
39                 Role = Am.RoleName,
40                 Registered = DateTime.Now,
41                 LastActivity = DateTime.Now,
42                 IsApproved = true,
43                 ApprovalLastChecked = DateTime.Now
44             });
45             Log($"Používateľ sa zaregistroval.");
46             Db.SaveChanges();
47         }
48
49         [Authorize]
50         public ActionResult Add()
51         {
52             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
53             ViewBag.MacLoaded = false;
54             ViewBag.MacAvailable = MacOfCurrentPcAvailableAndNotUsed();
55             return View(new Computer());
56         }
57
58         [Authorize]
59         public ActionResult AddThisPc()
60         {
61             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
62
63             ViewBag.MacLoaded = true;
64             ViewBag.MacAvailable = MacOfCurrentPcAvailableAndNotUsed();
65             return View("Add", new Computer {MacAddress = MacOfCurrentPc});
66         }
67
68         [Authorize, HttpPost, ValidateAntiForgeryToken]
69         public ActionResult Add([Bind(Include = "MacAddress, Name")] Computer computer)
70         {
71             ViewBag.MacLoaded = false;
72             ViewBag.MacAvailable = MacOfCurrentPcAvailableAndNotUsed();
73             if (!ModelState.IsValid) return View(computer);
74             TidyMacAddress(computer);
75             CurrentUser.Computers.Add(computer);
76             Log($"Používateľ pridal počítač \"{computer.Name}\" ({computer.MacAddress}).");
77             Db.SaveChanges();
78             return RedirectToAction("Index");
79         }
80
81         [Authorize]
82         public ActionResult Delete(int? id)
83         {
84             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
85             if (!id.HasValue) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
86             if (!ComputerExist(id)) return new HttpStatusCodeResult(HttpStatusCode.NotFound);
```

```
87
88              var computer = Db.Computers.Find(id);
89              ViewBag.ComputerName = computer.Name;
90              return View();
91          }
92
93          [ActionName("Delete"), Authorize, HttpPost, ValidateAntiForgeryToken]
94          public ActionResult DeleteConfirmed(int? id)
95          {
96              var computer = Db.Computers.Find(id);
97              Db.Computers.Remove(computer);
98              Log($"Používateľ vymazal počítač \"{computer.Name}\" ({computer.MacAddress}).");
99              Db.SaveChanges();
100             return RedirectToAction("Index");
101         }
102
103         [Authorize]
104         public ActionResult Edit(int? id)
105         {
106             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
107             if (!id.HasValue) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
108             if (!ComputerExist(id)) return new HttpStatusCodeResult(HttpStatusCode.NotFound);
109
110             return View(Db.Computers.Find(id));
111         }
112
113         [Authorize, HttpPost, ValidateAntiForgeryToken]
114         public ActionResult Edit([Bind(Include = "MacAddress, Name, Id")] Computer computer)
115         {
116             if (!ModelState.IsValid) return View(computer);
117
118             TidyMacAddress(computer);
119             Db.Entry(computer).State = EntityState.Modified;
120             Log($"Používateľ upravil počítač \"{computer.Name}\" ({computer.MacAddress}).");
121             Db.SaveChanges();
122             Db.SaveChanges();
123             return RedirectToAction("Index");
124         }
125
126         [Authorize]
127         public ActionResult WakeUp(int? id)
128         {
129             if (!Am.IsRegistered) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
130             if (!id.HasValue) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
131             if (!ComputerExist(id)) return new HttpStatusCodeResult(HttpStatusCode.NotFound);
132
133             var computer = Db.Computers.Find(id);
134             foreach (var iface in ServerInterface.Interfaces)
135             {
136                 WakeOnLan.SendWoLPacket(MacAddressFromString(computer.MacAddress), iface.
                        BroadcastAddress);
137             }
138             Log($"Používateľ poslal Magic Packet na počítač \"{computer.Name}\" ({computer.MacAddress
                    }).");
139             Db.SaveChanges();
140             return View(computer);
141         }
142
143         private bool ComputerExist(int? id)
144         {
145             return CurrentUser.Computers.Any(c => c.Id == id);
146         }
147
148         private static string TidyMacAddress(PhysicalAddress macAddress)
149         {
150             return string.Join(":", Enumerable.Range(0, 6).Select(i => macAddress.ToString().
                    Substring(i * 2, 2)));
151         }
152
153         private static void TidyMacAddress(Computer computer)
154         {
155             computer.MacAddress = TidyMacAddress(MacAddressFromString(computer.MacAddress));
156         }
157
158         private static PhysicalAddress MacAddressFromString(string macAddress)
159         {
160             macAddress = string.Join("", macAddress.Split('-', ':'));
161             macAddress = macAddress.ToUpper();
162             return PhysicalAddress.Parse(macAddress);
```

```
163         }
164
165         private string MacOfCurrentPc
166         {
167             get
168             {
169                 var userIp = IPAddress.Parse(Request.UserHostAddress);
170                 var arpEntries = ArpEntry.ArpEntries;
171                 var mac = string.Empty;
172                 foreach (var arpEntry in arpEntries)
173                 {
174                     if (!arpEntry.IpAddress.Equals(userIp)) continue;
175                     mac = TidyMacAddress(arpEntry.MacAddress);
176                     break;
177                 }
178                 return mac;
179             }
180         }
181
182         private bool MacOfCurrentPcAvailableAndNotUsed()
183         {
184             var mac = MacOfCurrentPc;
185             if (mac.IsEmpty()) return false;
186             return !CurrentUser.Computers.Any(c => c.MacAddress == mac);
187         }
188     }
189 }
```

A.2.5 AdminController.cs

```
1  using System;
2  using System.Data.Entity;
3  using System.Linq;
4  using System.Net;
5  using System.Web.Mvc;
6  using WebApp.Models;
7
8  namespace WebApp.Controllers
9  {
10     public class AdminController : BaseController
11     {
12         [Authorize]
13         public ActionResult Index()
14         {
15             if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
16
17             return View();
18         }
19
20         [Authorize]
21         public ActionResult Interfaces()
22         {
23             if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
24             return PartialView(ServerInterface.Interfaces);
25         }
26
27         [Authorize]
28         public ActionResult RegisteredUsers()
29         {
30             if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
31             CheckUserApproval();
32
33             return PartialView(Db.Users.ToList());
34         }
35
36         private void CheckUserApproval()
37         {
38             var flag = false;
39             var fiveDaysAgo = DateTime.Now.AddDays(-5);
40             foreach (var user in Db.Users.Where(user => user.ApprovalLastChecked <=
                   fiveDaysAgo))
41             {
42                 user.IsApproved = UserHelper.IsApproved(user.VutId);
43                 user.Role = UserHelper.GetRoleName(user.VutId);
44                 user.ApprovalLastChecked = DateTime.Now;
45                 Db.Entry(user).State = EntityState.Modified;
46                 flag = true;
47             }
```

```
48              if (flag) Db.SaveChanges();
49          }
50          [Authorize]
51          public ActionResult Log()
52          {
53              if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
54
55              ClearLog();
56
57              var log = from ev in Db.Log
58                        orderby ev.DateTime descending
59                        select ev;
60
61              return View(log);
62          }
63
64          private void ClearLog()
65          {
66              var yearAgo = DateTime.Now.AddYears(-1);
67              Db.Log.RemoveRange(Db.Log.Where(e => e.DateTime <= yearAgo));
68              Db.SaveChanges();
69          }
70
71          [Authorize]
72          public ActionResult AdminToggle(int? id)
73          {
74              if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
75              if (!id.HasValue) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
76              if (!UserExist(id)) return new HttpStatusCodeResult(HttpStatusCode.NotFound);
77
78              return View(Db.Users.Find(id));
79          }
80
81          [ActionName("AdminToggle")]
82          [Authorize]
83          [HttpPost]
84          [ValidateAntiForgeryToken]
85          public ActionResult AdminToggleConfirmed(int? id)
86          {
87              var user = Db.Users.Find(id);
88              user.IsAdmin = !user.IsAdmin;
89              Db.Entry(user).State = EntityState.Modified;
90              if (user.IsAdmin)
91                  Log($"Administrátor pridal administrátora {user.Name} ({user.VutId}).");
92              else Log($"Administrátor zrušil administrátora {user.Name} ({user.VutId}).");
93              Db.SaveChanges();
94
95              return RedirectToAction("Index");
96          }
97
98          [Authorize]
99          public ActionResult Delete(int? id)
100         {
101             if (!Am.IsAdmin) return new HttpStatusCodeResult(HttpStatusCode.Forbidden);
102             if (!id.HasValue) return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
103             if (!UserExist(id)) return new HttpStatusCodeResult(HttpStatusCode.NotFound);
104
105             var user = Db.Users.Find(id);
106             return View(user);
107         }
108
109         [ActionName("Delete")]
110         [Authorize]
111         [HttpPost]
112         [ValidateAntiForgeryToken]
113         public ActionResult DeleteConfirmed(int? id)
114         {
115             var user = Db.Users.Find(id);
116             Db.Computers.RemoveRange(user.Computers);
117             Db.Users.Remove(user);
118
119             if (id == Am.VutId)
120             {
121                 Log($"Používateľ sa odregistroval");
122                 Db.SaveChanges();
123                 return RedirectToAction("Index", "Account");
124             }
125             Log($"Administrátor vymazal používateľa {user.Name} ({user.VutId}).");
126             Db.SaveChanges();
```

```
127             return RedirectToAction("Index");
128         }
129
130         private bool UserExist(int? id)
131         {
132             return Db.Users.Any(u => u.VutId == id);
133         }
134     }
135 }
```

A.2.6 LayoutController.cs

```
1  using System.Web.Mvc;
2
3  namespace WebApp.Controllers
4  {
5      public class LayoutController : BaseController
6      {
7          public ActionResult Menu()
8          {
9              return PartialView(Am);
10         }
11     }
12 }
```

A.3 Vrstva prístupu k dátam/Data Access Layer

A.3.1 WakesContext.cs

```
1  using System.Data.Entity;
2  using System.Data.Entity.ModelConfiguration.Conventions;
3  using WebApp.Models;
4
5  namespace WebApp.DAL
6  {
7      public class WakesContext : DbContext
8      {
9          public DbSet<Computer> Computers { get; set; }
10         public DbSet<User> Users { get; set; }
11         public DbSet<Event> Log { get; set; }
12         public WakesContext() : base("WakesContext") {}
13
14         protected override void OnModelCreating(DbModelBuilder modelBuilder)
15         {
16             modelBuilder.Conventions.Remove<PluralizingTableNameConvention>();
17         }
18     }
19 }
```

A.4 Pohľady/Views (ukážka)

A.4.1 App/Index.cshtml

```
1  @model IEnumerable<WebApp.Models.Computer>
2  @{
3      ViewBag.Title = "Moje PC";
4  }
5  <h2>@ViewBag.Title</h2>
6  <p>Zoznam počítačov, ktoré môžete zobudiť.</p>
7  <br />
8  @if (Model.Any())
9  {
10     <table class="table">
11         <thead>
12         <tr>
13             <th>@Html.DisplayNameFor(model => model.Name)</th>
14             <th>@Html.DisplayNameFor(model => model.MacAddress)</th>
15             <th style="min-width: 135px">Akcia</th>
16         </tr>
17         </thead>
18         <tbody>
19         @foreach (var item in Model)
20         {
21             <tr>
22                 <td>@Html.DisplayFor(modelItem => item.Name)</td>
23                 <td>@Html.DisplayFor(modelItem => item.MacAddress)</td>
24                 <td>
25                     <div class="btn-group">
26                         <a href="@Url.Action("WakeUp", new {id = item.Id})" class="btn
                               btn-default">
27                             <span class="glyphicon glyphicon-off"></span>
28                             <span class="hidden-xs"> Zobudiť</span>
29                         </a>
30                         <a href="@Url.Action("Edit", new {id = item.Id})" class="btn
                               btn-default">
31                             <span class="glyphicon glyphicon-pencil"></span>
32                             <span class="hidden-xs"> Upraviť</span>
33                         </a>
34                         <a href="@Url.Action("Delete", new {id = item.Id})" class="btn
                               btn-default">
35                             <span class="glyphicon glyphicon-trash"></span>
36                             <span class="hidden-xs"> Zmazať</span>
37                         </a>
38                     </div>
39                 </td>
40             </tr>
41         }
```

```
42          </tbody>
43      </table>
44  }
45  else
46  {
47      <h4>V zozname zatiaľ nie je žiaden počítač. Pridajte ho kliknutím na tlačidlo nižšie.</
            h4>
48      <br/>
49  }
50  <div class="btn-toolbar">
51      <a href="@Url.Action("Add")" class="btn btn-primary">
52          <span class="glyphicon glyphicon-plus"></span><span> Pridať</span>
53      </a>
54      @if (ViewBag.MacAvailable)
55      {
56          <a href="@Url.Action("AddThisPc")" class="btn btn-success">
57              <span class="glyphicon glyphicon-plus"></span><span> Pridať tento počítač</
                    span>
58          </a>
59      }
60  </div>
```

A.5 Pomocné triedy

A.5.1 WakeOnLan.cs

```
using System;
using System.Net;
using System.Net.NetworkInformation;
using System.Net.Sockets;

namespace WebApp
{
    public static class WakeOnLan
    {   //funkcia na poslanie Magic Packetu
        public static void SendWoLPacket(PhysicalAddress macAddress, IPAddress ipAddress)
        {
            var macAddressString = macAddress.ToString();
            //konverzia retazca na byte
            var macAddressByte = new byte[6];
            for (var i = 0; i < 6; i++)
            {
                var t = macAddressString.Substring((i * 2), 2);
                macAddressByte[i] = Convert.ToByte(t, 16);
            }
            using (var client = new UdpClient())
            {
                //pošle paket na zadanú IP adresu a port 9 pomocou UDP
                client.Connect(ipAddress, 9);
                var packet = new byte[17 * 6];
                //synchronizačný stream
                for (var i = 0; i < 6; i++)
                {
                    packet[i] = 0xFF;
                }
                // 16x vloží MAC adresu
                for (var i = 1; i <= 16; i++)
                {
                    for (var j = 0; j < 6; j++)
                    {
                        packet[i * 6 + j] = macAddressByte[j];
                    }
                }
                //odoslanie Magic Packetu
                client.Send(packet, packet.Length);
            }
        }
    }
}
```

A.5.2 UserHelper.cs

```
using System.Collections.Generic;
using System.Linq;
using System.Security.Claims;
using System.Web;
using System.Web.WebPages;
using System.Xml.Linq;

namespace WebApp
{
    public static class UserHelper
    {
        private static readonly Dictionary<string, string> Role = new Dictionary<string,
            string>
        {
            {"S", "Študent"},
            {"Z", "Zamestnanec"},
            {"D", "Doktorand"}
        };

        public static int VutIdFromSaml
        {
            get
            {
                var identity = HttpContext.Current.User.Identity as ClaimsIdentity;
                if (identity != null)
```

```
25                  {
26                      return int.Parse((from id in identity.Claims
27                          where id.Type == "perid"
28                          select id.Value).Single());
29                  }
30                  else return 0;
31              }
32          }
33
34          public static string NameFromSaml
35          {
36              get
37              {
38                  var identity = HttpContext.Current.User.Identity as ClaimsIdentity;
39                  if (identity != null)
40                  {
41                      return (from id in identity.Claims
42                          where id.Type == "urn:oid:2.5.4.12"
43                          select id.Value).SingleOrDefault();
44                  }
45                  else return "";
46              }
47          }
48
49          public static string LoggedUserRoleName => GetRoleName(VutIdFromSaml);
50
51          public static string GetRoleName(int vutId)
52          {
53              var roleCode = GetRoleCode(vutId);
54              return roleCode.IsEmpty() ? "nonUTKO" : Role[roleCode];
55          }
56
57          private static string GetRoleCode(int vutId)
58          {
59              var address = $@"https://www.vutbr.cz/external/export.php";
60
61              var xdoc = XDocument.Load(address);
62              var element = xdoc.Element("data").Element("row");
63              if (element == null) return string.Empty;
64              return element.Element("ROLE").Value;
65          }
66
67          public static bool IsApproved(int vutId)
68          {
69              var roleCode = GetRoleCode(vutId);
70              return roleCode == "Z" || roleCode == "D" || vutId == 164388;
71          }
72
73          public static bool LoggedUserIsApproved => IsApproved(VutIdFromSaml);
74      }
75  }
```

B NÁVOD NA POUŽÍVANIE APLIKÁCIE

Kto môže používať túto aplikáciu: Prístup do aplikácie majú iba zamestnanci a doktorandi pod Ústavem telekomunikací FEKT VUT v Brně.

Z akých sietí je možné pripojiť sa do aplikácie: Prístup do aplikácie je v súčasnosti možný zo siete VUT (147.229.0.0/16) vrátane VPN alebo z WiFi siete Eduroam (iba v rámci VUT).

Ako sa prihlásiť do aplikácie: Prihlásite sa pomocou položky „Prihlásiť“ v menu. Budete presmerovaní na prihlasovaciu stránku VUT, kde sa pomocou VUT konta prihlásite. Po úspešnom prihlásení vás to vráti späť do aplikácie.

Ako sa zaregistrovať v aplikácii: Registrácia v aplikácii je veľmi jednoduchá. Stačí splňovať podmienky prístupu do aplikácie a prihlásiť sa pomocou VUT konta. Účet pre tento server sa vytvorí automaticky pri prvom prístupe na stránku „Moje PC".

Ktoré počítače je možné zobudiť: V súčasnosti je možné zobudiť počítače s podporou Wake on LAN, ktoré sú pripojené do niektorej z nasledujúcich sietí v rámci Ústavu telekomunikací: 147.229.146.0/23

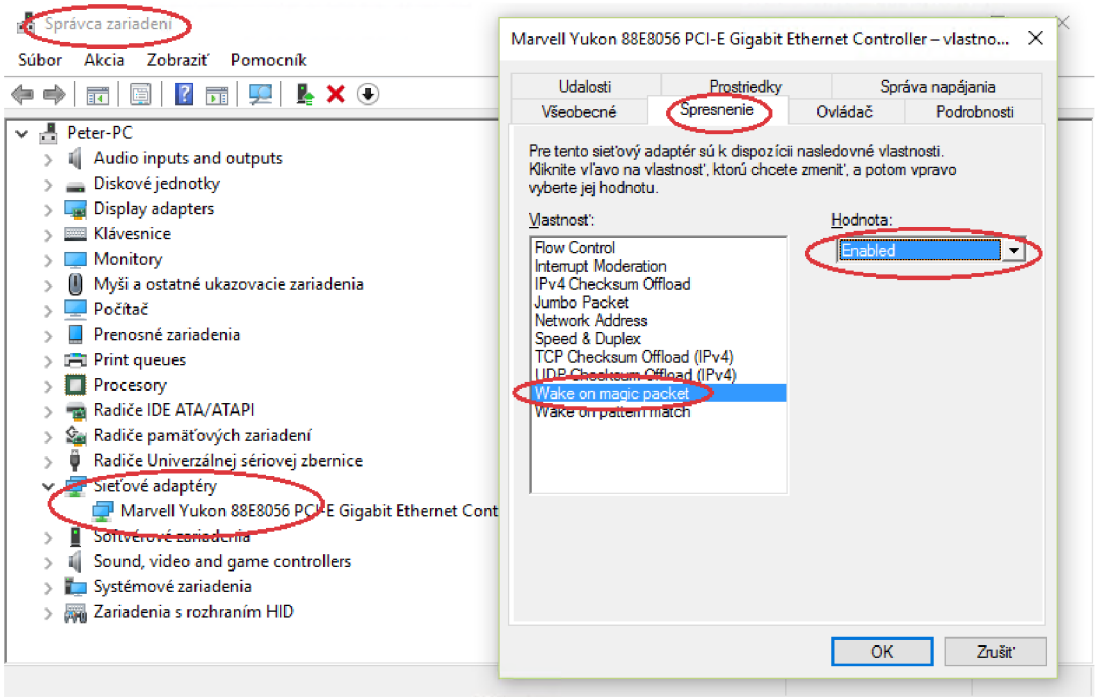
Ako pridať nový počítač: Na stránke „Moje PC“ kliknite na tlačidlo „Pridať“. Otvorí sa vám formulár kde je potrebné vyplniť názov a MAC adresu daného počítača (12 miestne hexadecimálne číslo s oddeľovačmi alebo bez). Potom stačí potvrdiť zadané údaje tlačidlom „Uložiť“. Ak sa pripájate z PC, ktorý chcete pridať do databázy, stačí kliknúť na tlačidlo „Pridať tento počítač“ vďaka čomu sa vám automaticky predvyplní MAC adresa daného PC. Po vyplnení názvu môžete počítač uložiť.

Ako zistiť MAC adresu počítača: MAC adresu zistíte podľa operačného systému nasledovne:

- Windows: v príkazovom riadku (Ctrl + R -> „cmd“) príkazom „getmac /v“ alebo „ipconfig /all“.
- Linux: v terminále (Ctrl + Alt + T) príkazom „/sbin/ifconfig“.
- MAC OS X: v terminále príkazom „networksetup -listallhardwareports“.

Ako povoliť Wake on LAN na počítači: Prvým krokom k sprevádzkovaniu Wake on Lan je povolenie tejto funkcie v BIOS-e (UEFI). Väčšinou nájdeme túto voľbu pod položkou Power Management v menu Set WakeUp Events, kde následne musíme povoliť Wake on PME alebo priamo pod voľbou Wake on LAN. Poznámka: Umiestnenie tejto voľby sa môže líšiť v závislosti od BIOS-u (UEFI), ktorým vaša základná doska disponuje (obr. 1.2).

Ďalším krokom je povolenie prebudenia PC priamo vo Windows v nastaveniach sieťovej karty (záleží na type karty a ovládača), do ktorých sa dostaneme cez správcu zariadení. V nastaveniach sieťovej karty, ktorou chceme PC prebudiť, vyberieme kartu Rozšírené (Advanced) a povolíme voľbu „Wake on Lan“ prípadne „Wake on Magic Packet“(obr. B.1).



Obr. B.1: Nastavenie podpory WoL vo Windowse

Ako prebudiť počítač: Ak má počítač povolenú funkciu Wake on LAN a pridali ste jeho MAC adresu do aplikácie, môžete ho prebudiť jednoduchým kliknutím na tlačilo „Zobudiť“ vedľa záznamu daného počítača na stránke „Moje PC“.

C OBSAH PRILOŽENÉHO DVD

- Elektronická verzia práce (PDF),
- Visual Studio riešenie (solution) „BP_wakes“, ktoré obsahuje zdrojové kódy webovej aplikácie Wakes,
- Skompilovaná webová aplikácia Wakes.