Implementace Wake on LAN v podnikové lokální síti

Tomáš Uhřík

2023

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2022/2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Tomáš Uhřík**
Osobní číslo:	**A20413**
Studijní program:	**B0613A140020 Softwarové inženýrství**
Forma studia:	**Prezenční**
Téma práce:	**Implementace Wake on LAN v podnikové lokální síti**
Téma práce anglicky:	**Implementation of Wake on LAN in the Local Network of Company**

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Nastudujte a popište principy fungování Wake on Lan.
3. Seznamte se specifiky podnikové lokální počítačové sítě.
4. Popište nastavení potřebné pro fungování Wake on Lan.
5. Připravte rozhraní pro buzení počítačů v podnikové lokální síti.
6. Věnujte pozornost zabezpečení navrženého řešení.

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. PETERSON, Larry L. a Bruce S. DAVIE. Computer networks: a systems approach. 5th ed. Burlington: Morgan Kaufmann, 2011. ISBN 9780123850591.
2. CHOU, Eric. Mastering Python networking: your one stop solution to using Python for network automation, DevOps, and SDN, 2020. ISBN 1839214678.
3. What Is Wake-on-LAN, and How Do I Enable It?. How-To Geek – We Explain Technology [online]. YATRITRIVEDI, 2021 [cit. 2022-11-20]. Dostupné z: https://www.howtogeek.com/70374/how-to-geek-explains-what-is-wake-on-lan-and-how-do-i-enable-it/
4. LIEBERMAN, Philip. White Paper: Wake on LAN Technology [online]. 2002 [cit. 2022-11-20]. Dostupné z: https://web.archive.org/web/20061019012133/http://www.liebsoft.com/index.cfm/whitepapers/Wake_On_LAN
5. AMD. Magic Packet Technology – white paper [online]. 1998. Dostupné z: https://web.archive.org/web/20041209061029/http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf

Vedoucí bakalářské práce: **doc. Ing. Martin Sysel, Ph.D.**
Ústav počítačových a komunikačních systémů

Datum zadání bakalářské práce: **2. prosince 2022**
Termín odevzdání bakalářské práce: **26. května 2023**

doc. Ing. Jiří Vojtěšek, Ph.D. v.r.
děkan

prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 7. prosince 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

Tomáš Uhřík, v. r.
podpis studenta

ABSTRAKT

Bakalářská práce se zabývá návrhem a implementací technologie Wake on LAN v prostředí podnikové lokální sítě. Cílem práce je analyzovat prostředí podnikové sítě a navrhnout a implementovat efektivní řešení pro využití technologie Wake on LAN, která umožní vzdálenou správu vypnutých počítačů v síti, včetně jejich spuštění a sledování stavu.

Teoretická část práce shrnuje všeobecné poznatky o technologii Wake on LAN. Následuje stručný popis použité technologie pro její implementaci v podnikové síti.

Praktická část popisuje aktuální stav infrastruktury podnikové sítě a samotnou implementaci Wake on LAN včetně tvorby grafického rozhraní pro možnost jednoduchého a intuitivního zapnutí počítače na dálku.

Klíčová slova: Wake on LAN, podniková síť, správa počítačů, úspora energie, vzdálený přístup, Django, Python

ABSTRACT

The bachelor thesis deals with the design and implementation of the Wake on LAN technology in a corporate local area network environment. The aim of the thesis is to analyze the enterprise network environment and to design and implement an effective solution for the use of Wake on LAN technology, which will allow remote management of shutdown computers in the network, including their startup and status monitoring.

The theoretical part of the thesis summarizes the general knowledge about Wake on LAN technology. This is followed by a brief description of the technologies used to implement it in an enterprise network.

The practical part describes the current state of the enterprise network infrastructure and the implementation of Wake on LAN, including the creation of a graphical interface to enable easy and intuitive remote computer power-up.

Keywords: Wake on LAN, enterprise network, computer management, power saving, remote access, Django, Python

PODĚKOVÁNÍ

Děkuji vedoucímu mé bakalářské práce panu doc. Ing. Martinu Syslovi, Ph.D. za odborné vedení, vstřícnost pří konzultacích a cenné rady a připomínky, jež umožnily úspěšné zpracování této práce. Rád bych poděkoval také všem odborníkům, kteří velice kladně reagovali na mé oslovení při zpracovávání bakalářské práce.

Děkuji obzvlášť svým rodičům za možnost vzdělávat se, obrovskou trpělivost a morální i psychickou podporu.

OBSAH

ÚVOD

Wake on LAN (WoL) je síťová technologie, která umožňuje vzdálené zapnutí nebo probuzení počítače přes síť. [1] Jedná se o užitečnou funkci, která snižuje celkovou spotřebu energie v podnicích a zároveň umožňuje správcům systémů efektivněji spravovat a kontrolovat počítače v síti bez nutnosti fyzického přístupu. WoL existuje již od 90. let a stal se standardní funkcí většiny moderních počítačů a síťových karet. [2]

Podnikové sítě, jako jsou ty na vysokých školách, v korporacích a vládních agenturách, obvykle obsahují velké množství počítačů, které je třeba spravovat a udržovat IT správci systému. Tito správci jsou zodpovědní za aktualizace a úkoly údržby systémů, které často vyžadují, aby byly počítače vzdáleně zapnuté nebo probuzené. Technologie WoL je pro tyto správce užitečným nástrojem, který jim umožňuje počítače spravovat efektivněji a šetřit čas a zdroje.

Kromě správců sítě to jsou také zaměstnanci, kteří mohou technologii WoL používat. Možnost zapnout svůj počítač vzdáleně bez nutnosti fyzického docházení do kanceláře zvyšuje flexibilitu a šetří čas. To může být obzvlášť užitečné pro pracovníky pracující v nepravidelných hodinách. Odpadá přitom nutnost permanentně zapnutého počítače v podnikové síti, což nejen značně šetří energii, ale také snižuje opotřebení hardwaru, což může prodloužit jeho životnost. K této možnosti je však vhodná tvorba grafického rozhraní, které umožní jednoduché zapnutí počítače na dálku i pro zaměstnance, kteří nemají v oblasti IT žádné hlubší znalosti. Zároveň je však nutná implementace bezpečnostních opatření. Například aby zaměstnanec neměl možnost (záměrně nebo nedopatřením) probudit počítač jiného zaměstnance.

Implementace technologie WoL v podnikové síti vyžaduje pečlivé plánování a konfiguraci. Správci systémů musí zajistit, že všechny počítače v síti jsou kompatibilní s WoL a že i síťová infrastruktura je připravena pro implementaci této technologie v potřebných částech sítě. Musí také zajistit, že je funkce WoL povolena na každém počítači a že jsou provedena nezbytná bezpečnostní opatření, která zabrání neoprávněnému přístupu do sítě.

Cílem bakalářské práce je prozkoumat technologii Wake on LAN, její výhody, nevýhody a možné problémy při její implementaci. Práce začne přehledem této technologie, včetně její historie, architektury a způsobu fungování. Poté se bude zabývat praktickou implementací technologie v konkrétní podnikové síti, včetně tvorby webového rozhraní pro možnost vzdáleného zapnutí počítače samotnými zaměstnanci podniku. Práce bude

také diskutovat dopad WoL na bezpečnost sítě a poskytne doporučení pro zmírnění případných rizik.

I. TEORETICKÁ ČÁST

1 WAKE ON LAN

Tato kapitola se zabývá technologií Wake on LAN (WoL). Kapitola začíná přehledem této technologie, včetně její historie, principu fungování, stavby magického paketu a jeho možnostmi zasílání po síti. Jsou zde zmíněny také problémy a možné limitace, které mohou nastat při pokusu implementace této technologie v podnikové síti.

1.1 Historie

V roce 1996 založily společnosti Intel a IBM alianci AMA (Advanced Manageability Alliance) s cílem vyvinout neproprietární nástroje založené na standardech pro zjednodušení správy počítačů. [2] V následujícím roce tato aliance představila technologii WoL, která umožňovala zapínat počítače na dálku prostřednictvím speciálního magického paketu. [3] V roce 1998 představila iniciativa Wired for Management, založená společností Intel, novou specifikaci, která měla pomoci omezit celkové náklady na vlastnictví podnikových počítačů. V rámci této iniciativy bylo požadováno, aby základní systémy měly možnost vzdáleného probuzení pomocí jedné ze tří technik: magický paket, filtrování paketů nebo wake-on-ring. [4] Tato iniciativa mimo jiné poskytovala další podporu pro WoL, ale od té doby byla nahrazena standardy Intelligent Platform Management Interface (IPMI) a Intel Active Management Technology (AMT).

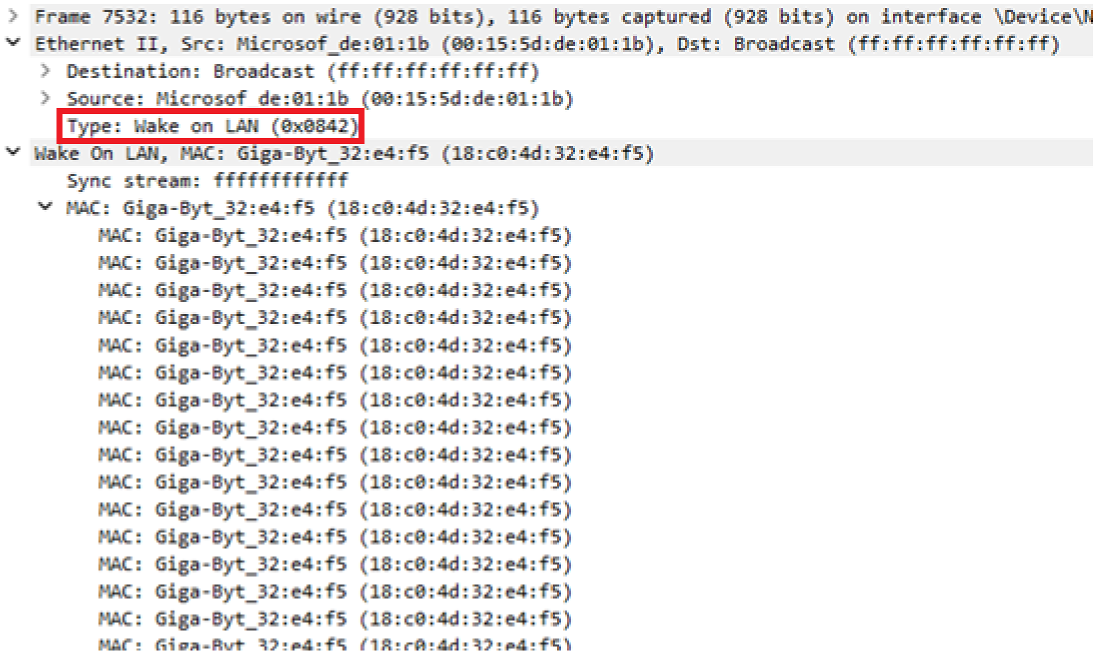
1.2 Princip fungování

V počítači, jehož hardware podporuje technologii WoL, a který je správně nakonfigurován pro fungování této technologie, zůstává jeho síťová karta (NIC) zapnutá, i když je systém zcela vypnutý (může být i v režimu spánku nebo hibernace), a čeká na speciální síťovou zprávu (rámec) zvanou magický paket. [5] Ponechání síťové karty v provozu sice spotřebovává určitou energii, ale tato spotřeba je mnohem nižší než běžná provozní spotřeba počítače. Magický paket je odesílán na datové vrstvě (vrstva 2 v OSI modelu) a je většinou vysílán na všechna připojená zařízení v dané síti pomocí broadcastu (MAC broadcast) – vrstva 3 v OSI modelu v tomto případě není použita. Magický paket obsahuje adresu MAC cílového počítače, což je identifikační číslo zabudované do každé karty síťového rozhraní (NIC) nebo jiného zařízení Ethernet v počítači, což umožňuje jeho jednoznačné rozpoznání a adresování v síti. [6] Vypnutý počítač s aktivovanou technologií WoL bude v režimu nízké spotřeby „naslouchat“ přes své síťové rozhraní příchozím paketům. Pokud je počítačem přijat magický paket, který je směrován na jeho MAC

adresu, vyšle síťová karta počítače signál napájecímu zdroji nebo základní desce, aby bylo provedeno probuzení systému. Technologie WoL byla původně vytvořena pro buzení počítačů pouze v lokální síti (LAN). Dnes se však lokální sítě velmi často dělí do několika podsítí, čímž se řeší problematika odesílání magických paketů i napříč podsítěmi, případně přes Internet (WOL over the Internet). [7] Pravidlem dnes není ani odesílání paketu pouze na datové vrstvě. Magické pakety mohou být zapouzdřeny v IP paketu a odesílány pomocí protokolu TCP nebo UDP do vzdálených podsítí. Při odesílání je doporučeno použít port 9 nebo 7, většinou se však port nekontroluje, tudíž lze teoreticky použít libovolný port v rozsahu 0 až 65535. Pokud jsou však firewallem nastavena bezpečnostní pravidla, která povolují pouze specifické porty nebo protokoly, může být zasílání magických paketů na jiném portu blokováno. Zdrojový port je dynamický. Ve většině situací se pakety WoL přenášejí v rámci stejné sítě i napříč sítěmi pomocí broadcastu (MAC i IP broadcast). Za určitých podmínek však lze využít i unicast (MAC i IP unicast).

1.3 Magický paket

Magický paket je speciální rámec, který obsahuje šest konstantních bajtů s hodnotou 255, hexadecimálně zapsaných jako FF:FF:FF:FF:FF:FF (označováno jako synchronizační stream), následovaných šestnáctkrát se opakující 48bitovou MAC adresou cílového počítače (bez oddělovačů). [8] Obvykle je odesílán na linkové vrstvě pomocí broadcastu jako EtherType 0x0842 (viz. Obr. 1). [9]



Obr. 1 Magický paket odeslaný na linkové vrstvě – Ethertype 0x0842

Vzhledem k tomu, že vypnutý počítač, který ‚naslouchá' příchozím paketům, kontroluje pouze výše uvedený řetězec reprezentující magický paket, může být paket zapouzdřen a odeslán prostřednictvím protokolů síťové a transportní vrstvy. [7] V tomto případě je často odesílán jako datagram protokolu UDP na port 0 (rezervované číslo portu), 7 (protokol Echo) nebo 9 (protokol Discard) (viz. Obr. 2). Protokol transportní vrstvy orientovaný na spojení, jako je TCP, je pro tuto úlohu méně vhodný, protože před odesláním uživatelských dat vyžaduje navázání aktivního spojení.

```
> Frame 198389: 144 bytes on wire (1152 bits), 144 bytes captured (1152 bits) on interface \De
> Ethernet II, Src: Microsof_de:01:1b (00:15:5d:de:01:1b), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
> Internet Protocol Version 4, Src: 192.168.0.36, Dst: 192.168.0.255
v User Datagram Protocol, Src Port: 49766, Dst Port: 9
    Source Port: 49766
    Destination Port: 9
    Length: 110
    Checksum: 0x0baa [unverified]
    [Checksum Status: Unverified]
    [Stream index: 571]
  > [Timestamps]
    UDP payload (102 bytes)
v Wake On LAN, MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
    Sync stream: ffffffffffff
  v MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
      MAC: Giga-Byt_32:e4:f5 (18:c0:4d:32:e4:f5)
```

Obr. 2 Magický paket odeslán jako datagram UDP protokolu na port 9

1.3.1 Omezení

Standardní magický paket má následující omezení:

- Vyžaduje MAC adresu cílového počítače.
- Neposkytuje potvrzení o doručení.
- Vyžaduje hardwarovou i softwarovou podporu funkce Wake-on-LAN na cílovém počítači.

1.4 Wake on LAN v rámci stejné sítě

1.4.1 Linková vrstva – broadcast

Dříve byl WoL paket standardně odesílán na linkové vrstvě pomocí broadcastu (MAC broadcast). Jde o jednoduché řešení, kdy k odeslání paketu stačí znát cílovou MAC adresu počítače. V rámci je poté nastavena šestnáctibitová hodnota (EtherType) na 0x0842. [9] Tato hodnota udává protokol dat zapouzdřených v datové části rámce – v tomto případě magický paket. Na takovém principu pracuje například nástroj `etherwake` v operačním systému Linux s distribucí Debian 11. [10]

1.4.2 Linková vrstva – unicast

Magický paket lze na linkové vrstvě odeslat také pomocí cílové unicast MAC adresy. Zde však mohou vznikat rizika se správným doručením paketu. Pokud by switch neměl ve své tabulce MAC adres cílovou MAC adresu počítače, switch by pravděpodobně místo unicastu přeposlal paket jako broadcast – tzv. unicast flooding. [11] Je však třeba poznamenat, že takové chování switchů se může lišit v závislosti na jejich konkrétních funkcích a konfiguraci. Některé switche mohou blokovat nebo omezovat přeposílání unicastových paketů jako broadcastové, aby minimalizovaly negativní dopady neoprávněného přeposílání paketů na síťovém rozhraní. [12] Pro odeslání magického paketu na linkové vrstvě pomocí unicast MAC adresy lze také použít nástroj `etherwake` v operačním systému Linux. [9]

1.4.3 Síťová a transportní vrstva – broadcast

Vzhledem k faktu, že se dnes používá technologie WoL i napříč různými sítěmi a podsítěmi, jsou vytvářeny programy, které magický paket zapouzdří do IP paketu a odesílají ho protokolem transportní vrstvy. [13] Tyto programy jsou pak často využívány

pro odesílání WoL paketů i v rámci stejné sítě. V případě odeslání jako broadcast (IP broadcast) je situace poměrně jednoduchá a bezproblémová. Cílová IP adresa je 255.255.255.255 a cílová MAC adresa je FF:FF:FF:FF:FF:FF. Je tedy potřeba znát pouze cílovou MAC adresu počítače pro sestavení magického paketu. Na tomto principu pracuje například nástroj `wakeonLan` operačního systému Linux distribuce Debian 11 nebo modul `wakeonLan` programovacího jazyku Python. [14][15]

1.4.4 Síťová a transportní vrstva – unicast

Magický paket lze odeslat také jako unicast (IP unicast). Může však nastat problém, kdy počítač odesílající magický paket nemá v ARP tabulce MAC adresu cílového počítače. Při vytváření rámce by tedy počítač odeslal ARP dotaz (broadcast). [16] Pokud je však cílový počítač vypnutý, tak nikdy na tento dotaz odesílající počítač nedostane odpověď a rámec nevytvoří. Použití unicast adresy by tedy bylo funkční pouze do doby, kdy by měl odesílající počítač záznam cílové MAC adresy ve své lokální ARP tabulce. [7]

1.5 Wake on LAN napříč sítěmi – Wake on Internet

V případě odesílání WoL paketu do jiné sítě je situace složitější, kdy se v cestě nachází jeden či více routerů. Hlavním omezením tohoto způsobu odesílání WoL paketů je, že broadcast pakety obecně nejsou směrovány. Router totiž od sebe broadcastové domény odděluje, tudíž se broadcast pakety skrze router nešíří. [17]

1.5.1 Unicast

Pro probuzení počítače v jiné síti lze použít IP unicast adresu. Takové řešení však bude pravděpodobně fungovat pouze po omezenou dobu po vypnutí počítače. Při odeslání unicastu do jiné sítě je paket poslán na bránu (gateway). Router ve své ARP tabulce vyhledá uloženou hodnotu cílové IP adresy a k ní příslušnou cílovou MAC adresu. Tato MAC adresa je poté vložena do rámce, který je následně odeslán. Problém nastává v případě, kdy router nemá záznam o příslušné IP a MAC adrese ve své ARP tabulce. Router sice odešle broadcast, aby zjistil, kterému počítači patří daná IP adresa, ale protože je cílový počítač vypnutý, tak nikdy nedostane odpověď. Paket je tedy zahozen. [5]

1.5.2 Směrovaný broadcast

Směrovaný broadcast (DB – IP directed broadcast) používá boradcastovou adresu dané podsítě. Cílový router poté upraví rámec na běžný broadcast a odešle ho na požadované

rozhraní. Tato technika je však standardně na routerech i firewallech zakázána vzhledem k náchylnosti na použití útoků jako je např. Smurf Attack, který způsobuje Denial of Services (DoS). [18] Je tedy třeba dbát na filtrování paketů tak, aby byly předávány pouze požadované pakety (např. WoL pakety). Toto se však týká pouze koncového routeru, kdy magický paket by měl ostatními routery mezi odesílajícím počítačem a koncovým routerem standardně projít.

1.5.3 Příjem magického paketu

Pokud je odesílající počítač ve stejné síti s cílovým počítačem, obvykle nedochází k problémům s příjmem paketu. Při odesílání přes Internet, zejména v případě, že se jedná o router implementující NAT (Network Address Translator), který je typicky nasazen ve většině domácností, je často třeba provést speciální konfiguraci routeru. Jednou z možností je nastavení pravidla přesměrování portů (port forwarding), kdy router naslouchá paketům na určitém portu a předává je do sítě cílovému počítači s konkrétní IP adresou (pro správnou funkci přesměrování portů by na cílovém počítači měla být nastavena statická IP adresa). [19] Magický paket je tedy odeslán na veřejnou IP adresu routeru a nakonfigurovaný port. Router paket poté přepošle na cílové síťové rozhraní počítače uvnitř sítě.

1.6 Hardwarové požadavky

Podpora funkce WoL je implementována na základní desce počítače a na kartě síťového rozhraní, a proto není závislá na operačním systému, který na hardwaru běží. [5] Některé operační systémy však mohou ovládat a ovlivňovat fungování WoL prostřednictvím ovladačů síťové karty nebo jiných funkcí. U starších základních desek, pokud není síťové rozhraní integrováno přímo na základní desce, může být vyžadováno připojení karty k základní desce dalším kabelem. Základní desky s integrovaným řadičem sítě Ethernet, který podporuje funkci WoL, kabel nepotřebují. Napájecí zdroj musí splňovat specifikace ATX 2.01. Starší základní desky musí být vybaveny také konektorem WAKEUP-LINK, kterým jsou propojeny pomocí 3pinového kabelu se síťovou kartou. Síťové karty a základní desky podporující WOL se standardem PCI 2.2 však takový kabel už nepotřebují. PCI verze 2.2 totiž podporuje PME (Power Management Events), takže se ‚budící' signál i požadované napájení v pohotovostním režimu šíří přímo po sběrnici PCI. [20] Většina dnešních moderních základních desek se sběrnicí PCI Express

s integrovanou síťovou kartou technologii WoL podporuje. Podrobnosti lze zjistit v dokumentaci pro daný hardware.

1.7 Konfigurace Wake on LAN

Pokud hardware podporující WoL nemá ve výchozím nastavení tuto funkci povolenu, je třeba funkci povolit v systému BIOS/UEFI. V některých případech je nutná i další konfigurace síťové karty přímo v operačního systému. Přesné kroky konfigurace závisí na konkrétním hardwaru v kombinaci s operačním systémem.

1.7.1 BIOS/UEFI

Funkce WoL musí být povolena v BIOS/UEFI, kde musí být zvoleno neustálé napájení síťové karty i po vypnutí počítače. Funkci je obvykle třeba povolit v sekci Power Management (Správa napájení). Ve starších systémech může být tato funkce pro aktivaci technologie WoL označována stejnou zkratkou jako WoL, v novějších systémech podporujících PCI verze 2.2 může být označována jako PME (Power Management Events), Wake on LAN nebo Power On By PCIE/PCI. Vždy záleží na konkrétním hardwaru. [21][22] Informace o aktivaci WoL lze obvykle nalézt v dokumentaci základní desky. Lze se také setkat s případem, kdy WoL bude naopak fungovat po pouhém vypnutí některé z jiných funkcí (např. zakázání funkce ErP), přitom nemusí být potřeba žádnou další funkci aktivovat. [23]

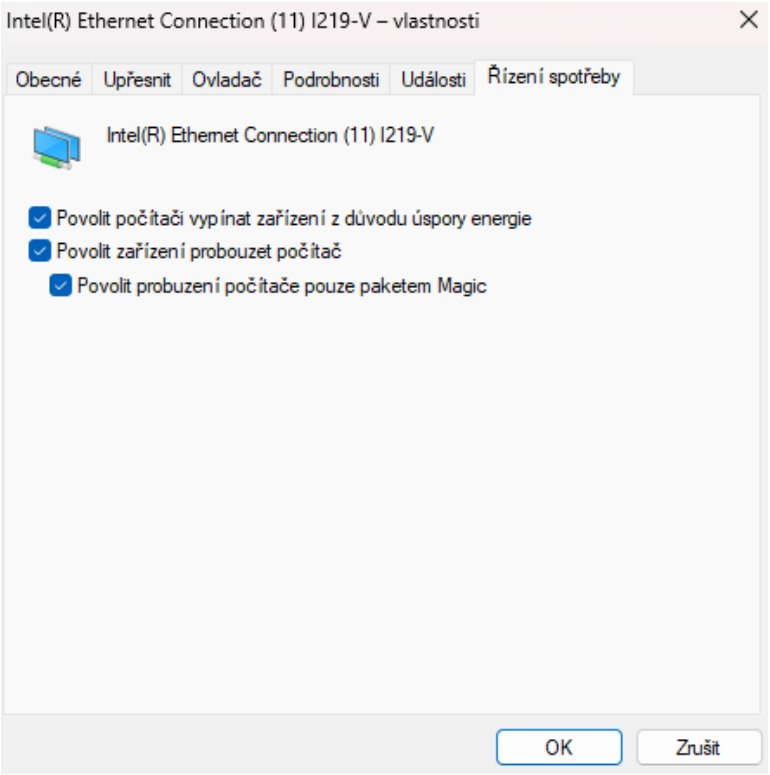
1.7.2 Operační systém

Konfiguraci síťové karty pro správnou funkci WoL je v některých případech nutno provést i v operačního systému. Nastavení se však můžou napříč různými systémy a jejich verzemi lišit.

1.7.2.1 Microsoft Windows

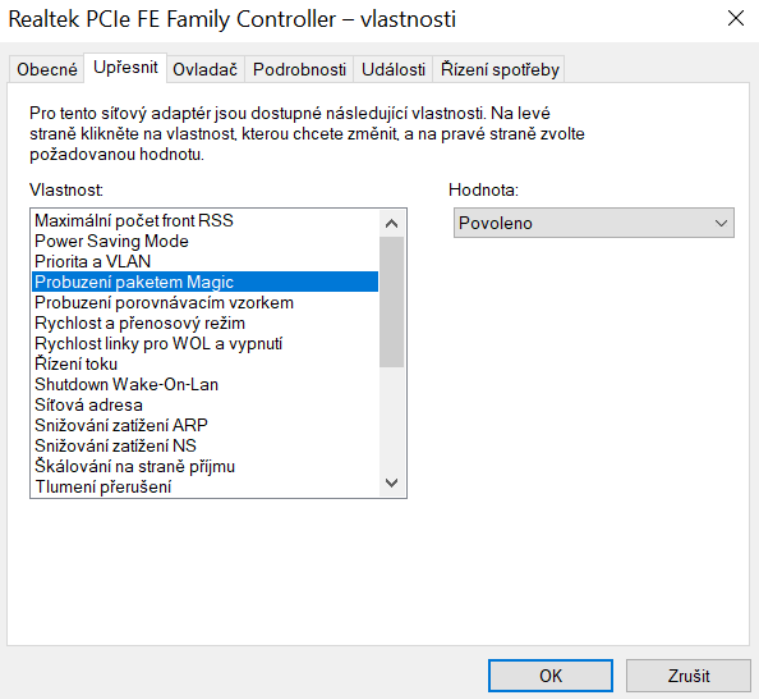
Novější verze systému Microsoft Windows integrují funkci WoL do Správce zařízení. Funkce je k dispozici na kartě Řízení spotřeby ve vlastnostech ovladače síťové karty. Zde je většinou potřeba povolit všechny tři následující funkce (viz Obr. 3) [22]:

- Povolit počítači vypínat zařízení z důvodu úspory energie.
- Povolit zařízení probouzet počítač.
- Povolit probuzení počítače pouze paketem Magic.



Obr. 3 Konfigurace WoL– záložka Řízení spotřeby

Nastavení je v některých případech nutné provést také na kartě Upřesnit. Zde se však názvy vlastností, které je nutno povolit pro funkci WoL mohou lišit. Nejčastěji je však nutné povolit vlastnost Probuzení paketem Magic (viz. Obr. 4).



Obr. 4 Konfigurace WoL– záložka Upřesnit

Možnost probuzení ze stavu hybridního vypnutí (S4) (známého také jako Fast Startup) nebo ze stavu kompletního vypnutí (S5) není v systémech Windows 8 a novějších podporována. Důvodem je změna chování operačního systému, která způsobuje deaktivaci WoL na síťovém adaptéru při přechodu do těchto stavů, protože je očekávána nulová spotřeba energie. [24] Standardně je WoL podporováno z nehybridního stavu hibernace (S4) (tj. když uživatel přímo požádá o hibernaci) nebo ze stavu spánku (S3). Některý hardware však může povolovat WoL ze stavů, které nejsou systémem Windows podporovány (Fast Startup a S5). Pro plnou podporu funkce WoL (jako je například možnost probuzení z vypnutého stavu – S5) může být však nutná také instalace nebo aktualizace úplné sady ovladačů od výrobce síťového zařízení.

1.7.2.2 Linux

Nastavení funkce WoL v operačním systému Linux se zdá být jednoznačnější než ve Windows. Aktivaci funkce lze provést jediným příkazem pomocí nástroje `ethtool` v rámci požadovaného síťového rozhraní. [25] Např. pokud má počítač síťovou kartu s názvem `enp3s0`, WoL lze aktivovat příkazem `ethtool -s enp3s0 wol g` (`-s` – nastavení rozhraní, `wol` – Wake on LAN, `g` – povoleno). Správné nastavení pak lze

zkontrolovat příkazem *ethtool enp3s0*, kde by ve výstupu u atributu *Wake-on* měla být hodnota g (povoleno, v opačném případě zde může být hodnota *d* – zakázáno) (viz. kapitola 4.2.1.2). Pokud by po zadání příkazu *ethtool název-rozhraní* nebyl ve výstupu atribut *Wake-on*, síťová karta pravděpodobně technologii WoL nepodporuje.

1.8 Problémy a omezení Wake on LAN v podnikové síti

Navzdory podpoře WoL a výhodám, které může tato technologie přinášet, existuje několik omezení, které je třeba při implementaci zohlednit. Mezi tato omezení patří systémové požadavky, absence potvrzení doručení magického paketu, zabezpečení sítě anebo nutnost cílových MAC adres pro odeslání magického paketu. Tato omezení mají v typické domácí síti menší vliv na implementaci než ve větších a komplexních sítích. Ve větších podnikových sítích, kde mohou být stovky nebo tisíce počítačů (často rozdělené do samostatných podsítí), mohou tato omezení implementaci WoL značně ztížit. [26]

1.8.1 Systémové požadavky

Implementace technologie WoL vyžaduje pro spolehlivou funkci odpovídající podporu v systému BIOS/UEFI, na síťové kartě a někdy i v operačního systému a u síťových prvků. To obvykle není problém u dnešních moderních systémů, počítačů a sítí. Pokud však podniková síť nemá podporu WoL (starší verze hardwaru a softwaru), změna hardwaru a softwaru pro její podporu může být potenciálně velmi nákladná. [26]

1.8.2 Potvrzení doručení magického paketu

S cílem minimalizovat spotřebu energie a dobu zpracování je technologie WoL založena na jednoduchém principu, který umožňuje NIC (síťová karta) zpracovávat magické pakety, když je počítač vypnutý (viz. kapitola 1.2), což má však za následek, že neposkytuje potvrzení o jejich doručení. Při skutečném používání WoL to obvykle nepředstavuje problém, nicméně mohou nastat potíže při řešení problémů s její implementací a testováním funkčnosti. Testování a analýzu správného doručení paketu na cílovém počítači lze provádět pouze, když je počítač zapnutý. Správné doručení paketu při zapnutém počítači však může být zavádějící, protože po jeho vypnutí nemusí být síťová karta počítače schopna paket přijmout. To se může dít zejména v operačním systému Microsoft Windows verze 8 a novější při zapnuté funkci Fast Startup, která může povolenou funkci WoL blokovat (viz. kapitola 1.7.2.1). Pro analýzu příjmu magického paketu lze využít například nástroj Wireshark. [8]

1.8.3 Problémy se zabezpečením

Pokud není síťový hardware nakonfigurován tak, aby umožňoval pouze provoz, který splňuje specifické bezpečnostní požadavky, může pakety WoL odesílat kdokoli ve stejné místní síti (LAN). Odeslání "neautorizovaného" paketu WoL sice neobchází standardní bezpečnostní opatření (jako jsou systémová hesla a síťové brány firewall), jakmile je však počítač zapnutý, potenciální útočníci mohou být schopni prohledat počítač skenováním různých zranitelností. Bez ohledu na to, zda je tedy WoL implementován, či nikoli, musí být nastaveny silné vnitřní bezpečnostní zásady sítě. [27]

1.8.4 Odesílání magických paketů napříč podsítěmi

Další potenciální bezpečnostní problém při implementaci WoL může nastat při pokusu o odeslání magického paketu v podniku z jedné podsítě do druhé použitím směrovaného broadcastu (viz kapitola 1.5.2).

1.8.5 Nutnost cílové adresy MAC

V rámci fungování WoL je vyžadována MAC adresa cílového počítače. V místní síti lze adresu MAC snadno zjistit pomocí základních příkazů jako je *ipconfig /all* a *getmac* (Windows) nebo *ip* popřípadě *ifconfig* (Linux). Použití této metody pro zjištění adresy větší skupiny počítačů ve velké síti může být však poměrně nepraktické a zdlouhavé. Složitější situace nastává při nutnosti zjistit větší počet MAC adres počítačů v jiné podsíti.

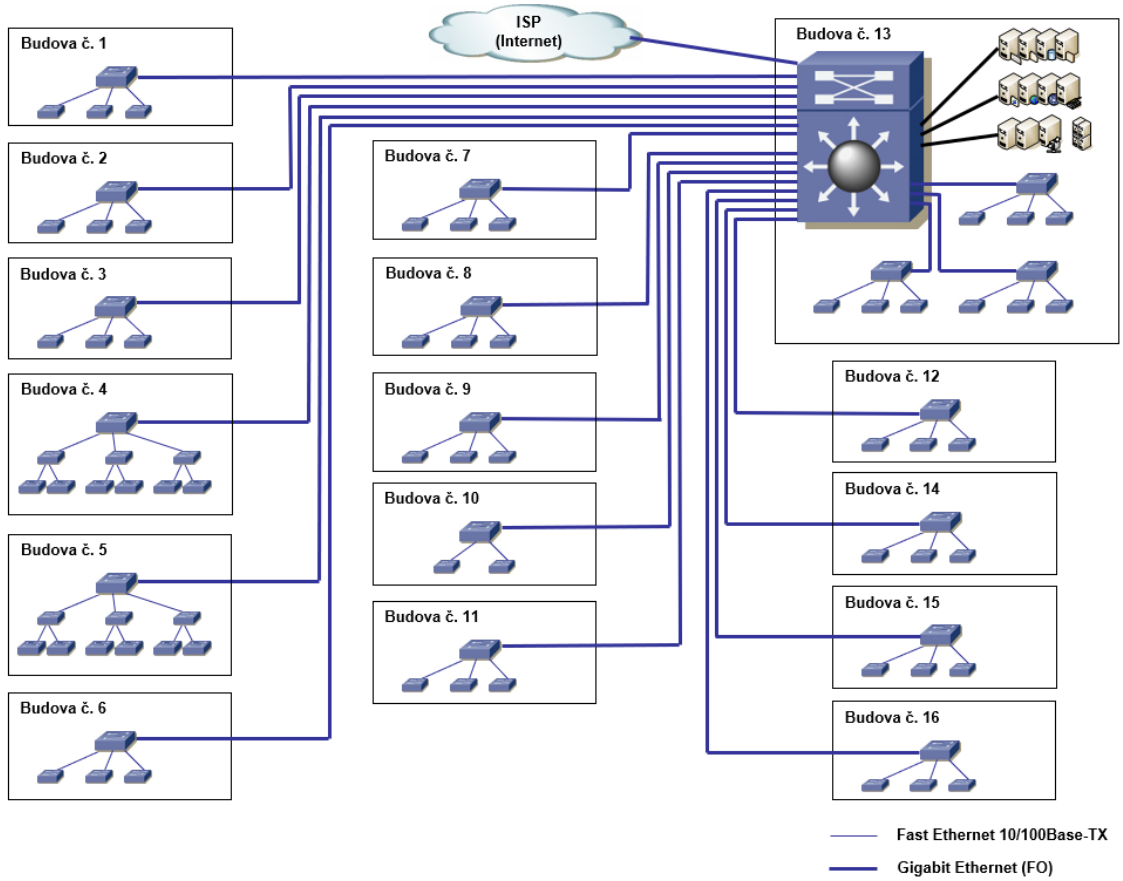
1.8.6 Probuzení počítače z různých stavů

V některých případech se počítač může probudit z jednoho stavu nízké spotřeby, ale ne z jiných. To znamená, že kvůli problémům s hardwarem a operačním systémem (může v některých případech blokovat WoL, i když je na hardwaru aktivována) se počítač může probudit ze stavu vypnutí (S5), ale neprobudí se z režimu spánku nebo hibernace a naopak (viz. kapitola 1.7.2.1).

II. PRAKTICKÁ ČÁST

2 INFRASTRUKTURA IMPLEMENTAČNÍ PODNIKOVÉ SÍTĚ

Tato kapitola popisuje aktuální stav IT infrastruktury podniku. Celý podnik tvoří 16 budov a je rozdělen na centrální část – budova č. 13 a dále na jednotlivá pracoviště, které jsou k budově č. 13 připojeny pomocí 10Gbit optických linek (viz. Obr. 5).



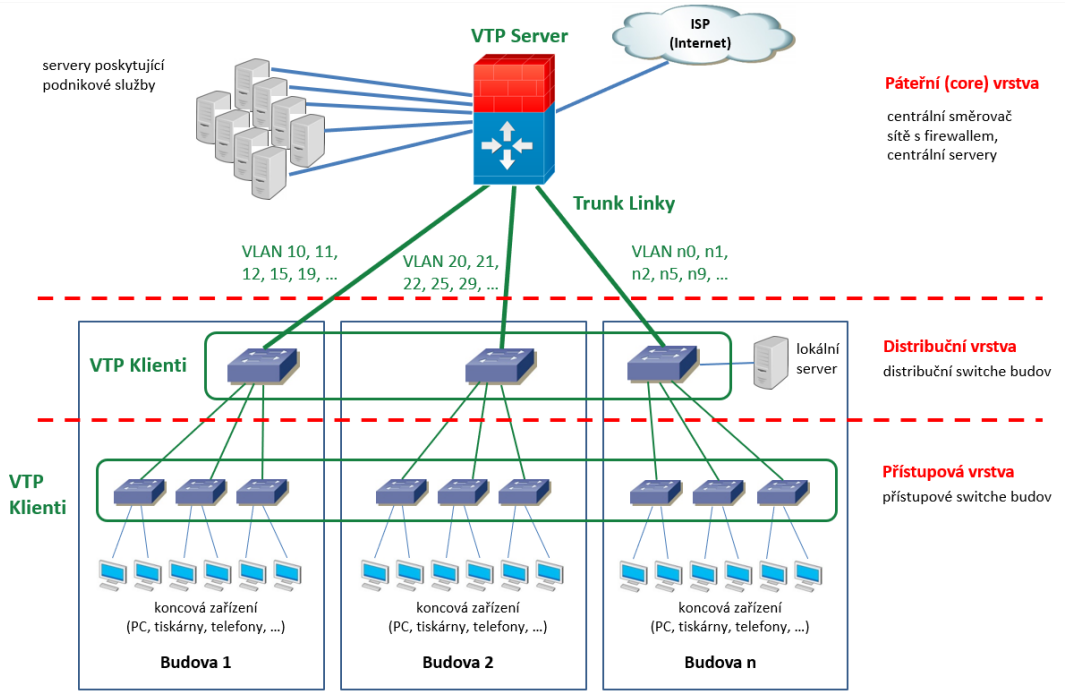
Obr. 5 Blokové schéma IP podnikové sítě a aktivních prvků [Zdroj: IT správce sítě]

2.1 Fyzická topologie

Tato topologie popisuje fyzické rozložení aktivních síťových prvků podniku. (viz. Obr. 5). Centrálním prvkem celé sítě je výkonný router umístěn v budově č. 13 napojen na ISP podnikové sítě pro přístup do Internetu. K routeru jsou přímo připojeny servery poskytující různé podnikové služby, z nichž některé mohou být přístupné i z Internetu. Z budovy č. 13 je v hvězdicové topologii optickými linkami rozvedena datová síť do všech ostatních budov podnikové sítě. Většina budov má jednu distribuční vrstvu tvořenou distribučním switchem. Výjimkami jsou budovy č. 4 a 5 disponující dvěma distribučními vrstvami

z důvodu rozlohy obou budov. Na distribuční switche jsou strukturovanou kabeláží (FastEthernet 10/100Base TX) napojeny switche přístupové, ke kterým jsou připojeny různá koncová zařízení. K distribučním switchům mohou být připojeny také lokální servery poskytující služby v rámci budovy.

2.2 Logická topologie



Obr. 6 Logická topologie podnikové sítě [Zdroj: IT správce sítě]

Logická topologie popisuje na základě schématu (viz. Obr. 6), jak jsou data v podnikové síti přenášena. Podniková síť je logicky členěna prostřednictvím VLAN (virtual local area network – virtuální síť). Důvodem tohoto dělení není pouze snaha o redukci přetížení sítě, ale také účelné rozdělení komunikace dle kategorie uživatelů a účelu (management, zaměstnanci, telefony, Wi-Fi, kamery, …) a podle budov. Pro správný přenos VLAN a snadnější správu switchů síť využívá VTP (VLAN Trunking Protocol). Centrální router pracuje v režimu server a každá změna nastavení VLAN provedena na serveru je přenesena na všechny switche typu klient, což značně usnadňuje management prvků. Pro správný přenos dat po síti z více VLAN jsou porty routeru a distribučních switchů nastaveny do režimu trunk, čímž se vytvoří trunk linky. Porty přístupových switchů jsou

nastaveny do modu access pro zařazení do jedné VLAN. Provoz mezi všemi VLAN je kontrolován a omezován centrálním firewallem.

2.3 IP adresace

Adresace sítě vychází z fyzického (budovy) i logického (dle kategorie a účelu) rozdělení na podsítě. Většina zařízení používá privátní IP adresy třídy A z rozsahu 10.0.0.0 – 10.255.255.255. Ostatním zařízením a serverům jsou přiděleny veřejné IP adresy, tudíž některé podnikové služby mohou být dostupné i z Internetu.

2.3.1 Adresace v rámci budovy

Následující příklad slouží pouze pro ilustraci principu adresace v síti. Nejedná se o adresy reálně použité v síti v rámci daných budov podniku:

- **síť**: 10.3.0.0/24
- **DNS server**: 10.172.80.63
- **gateway**: 10.3.0.1
- **broadcast**: 10.3.0.255

3 INSTALACE A KONFIGURACE SERVERU PRO BUZENÍ POČÍTAČŮ

Následující kapitola popisuje instalaci a konfiguraci serveru pro buzení počítačů v podnikové síti. Server je instalován a konfigurován jako virtuální stroj prostřednictvím systému Hyper-V. Cílem tohoto stroje bude dále obsluha webového serveru, který bude poskytovat uživatelům webovou aplikaci (viz. kapitola 9), prostřednictvím které bude server odesílat magické pakety pro buzení počítačů v síti. Cílem kapitoly není vytvořit podrobný návod instalace, nýbrž přiblížit vytvoření, instalaci a konfiguraci virtuálního stroje v prostředí Hyper-V. V rámci vytvoření virtuálního stroje je důležitá především síťová konfigurace serveru pro možnost odesílání magických paketů do různých VLAN sítě.

3.1 Server jako virtuální stroj

Pro účely serveru byl zvolen virtuální stroj, který poskytuje několik výhod a je dostatečným řešením pro odesílaní magických paketů. Jedná se poměrně o jednoduchou úlohou, která nevyžaduje velký výpočetní výkon nebo fyzický server, tudíž není nutnost pořizovat další hardwarové prostředky a lze využít stávající hardware a infrastrukturu. Toto řešení může tedy představovat významnou úsporu finančních prostředků. Virtualizace navíc poskytuje izolaci a oddělení mezi hostitelským systémem a virtuálním prostředím, kde bude server provozován, čímž se minimalizuje riziko nežádoucího ovlivnění ostatních systémů a aplikací. Virtuální stroj také nabízí lehkou škálovatelnost a flexibilitu, kdy lze jednoduše upravit výpočetní kapacitu nebo přidávat a odebírat přidělené zdroje, aby vyhovovaly potřebám serveru. [28] V případě serveru pro odesílání magických paketů se jedná především o přidávání a odebírání síťových zdrojů pro odesílání paketů do různých částí sítě.

3.2 Volba operačního systému

Při výběru vhodného operačního systému pro nasazení virtuálního stroje je nutné zvážit několik faktorů, jako je spolehlivost, flexibilita, bezpečnost, dostupnost funkcí a komunitní podpora. V tomto případě byl pro server zvolen operační systém Linux. Linux je open-source operační systém, což znamená, že je jeho zdrojový kód volně dostupný a může být dle potřeb dále upravován. Díky této volnosti lze mít nad systémem plnou kontrolu a přizpůsobit ho tak, aby splňoval požadavky na bezpečnost, síťovou konfiguraci nebo

správu zdrojů. Linux je také známý pro svou stabilitu a spolehlivost, což jsou klíčové faktory pro provozování jak virtuálního, tak fyzického serveru. [29] Linuxové distribuce jsou často optimalizovány právě pro serverové nasazení a mají dlouhou historii úspěšného provozu v podnikovém prostředí. Široká komunitní podpora navíc přispívá ke snazšímu odhalování a odstraňování chyb a zranitelností, což přispívá ke zvýšení systémové bezpečnosti a stability. Výběr tohoto operačního systému může však přinést i finanční výhody. Linux je distribuován pod svobodnou licencí, která umožňuje jeho bezplatné používání bez nutnosti zakoupení licenčních klíčů, což může představovat značné finanční úspory např. v porovnání s operačním systémem Windows. [30] Zásadním faktorem pro výběr Linuxu je také možnost jednoduché vzdálené správy a administrace prostřednictvím příkazové řádky. Tento aspekt je zvláště významný pro serverové nasazení, protože umožňuje vzdálenou správu serveru bez nutnosti grafického rozhraní. Příkazová řádka poskytuje mnoho pokročilých nástrojů a utilit pro automatizaci úloh, konfiguraci, monitorování a snadnou správu. Díky tomu je možné provádět různé operace a údržbu serveru efektivně a bez nutnosti fyzického přístupu k serverovému zařízení.

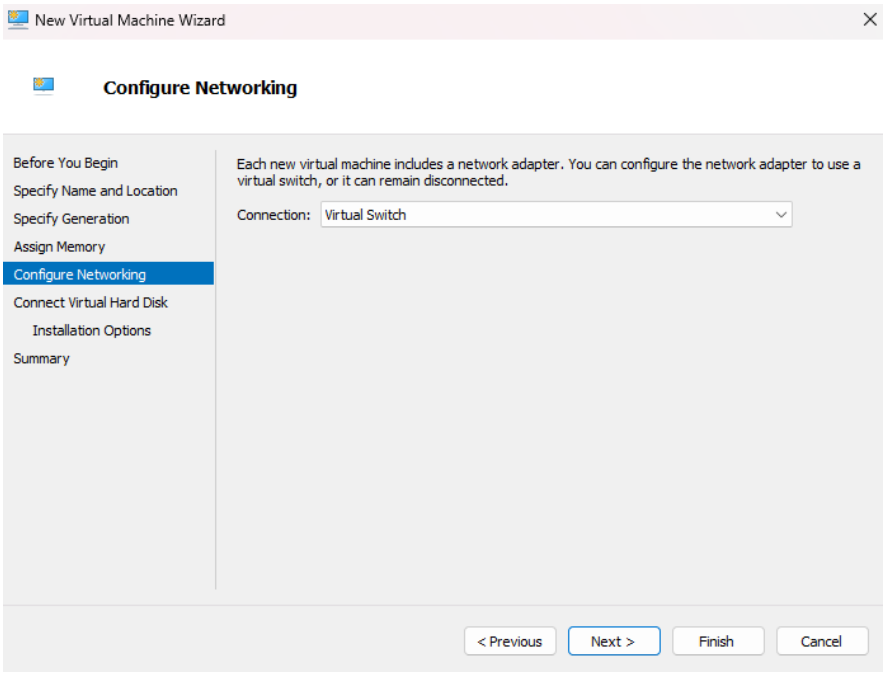
3.3 Instalace a konfigurace

Pro server byl zvolen operační systém Linux s distribucí Debian s rozhraním příkazové řádky. Byl stažen instalační obraz s označením *netinst*, který pro správnou instalaci vyžaduje funkční připojení k Internetu. [31]

3.3.1 Vytvoření virtuálního stroje v Hyper-V

Pro vytvoření nového virtuálního stroje v prostředí Hyper-V je v sekci *Actions* zvolena možnost *New -> Virtual Machine…,* čímž se rozběhne průvodce vytvořením virtuálního stroje:

- virtuální stroj byl pojmenován jako **WakeOnLan**,
- operační paměť nastavena při spuštění na **2048 MB**,
- virtuální adaptér byl připojen k virtuálnímu switchi s názvem **Virtual Switch** (viz. Obr. 7),



Obr. 9 Nastavení virtuálního switche

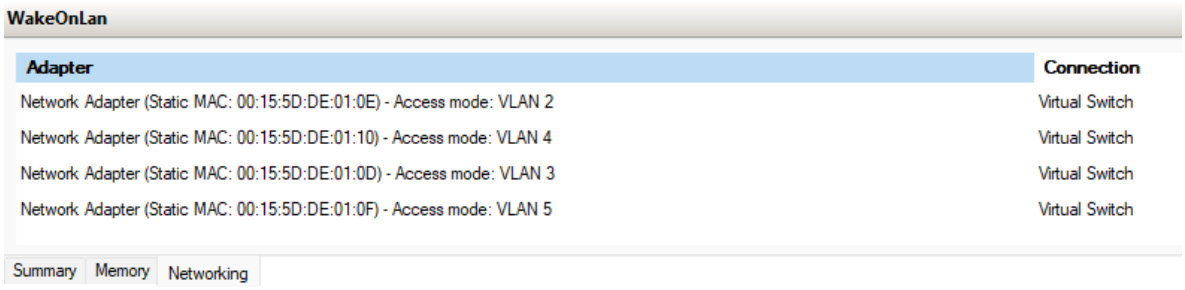
- byl vytvořen nový virtuální disk s názvem **WakeOnLan.vhdx**,
- byla zvolena možnost instalace operačního systému pomocí staženého `netinst` ISO obrazu,
- po úspěšném dokončení se server s daným jménem zobrazil v seznamu všech vytvořených virtuálních strojů.

3.3.2 Síťová konfigurace – Hyper-V

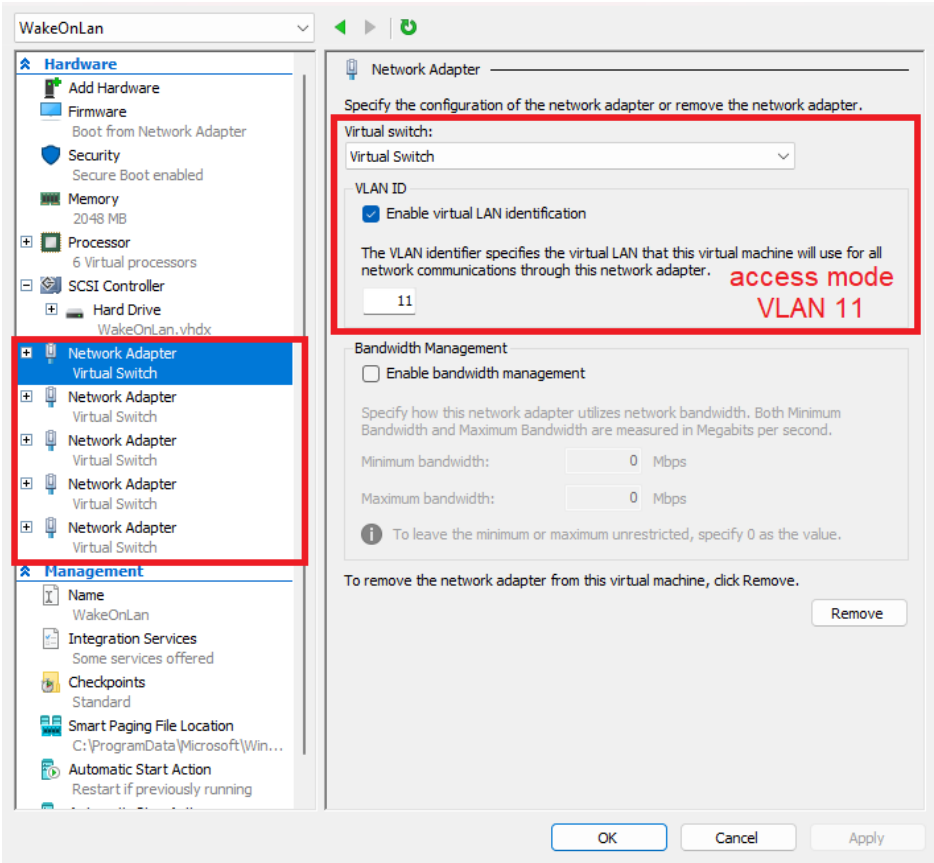
Konfigurace se týká nastavení síťových karet virtuálního stroje pro možnost odesílání magických paketů pro buzení počítačů do různých požadovaných VLAN. Byly vytvořeny dvě varianty, z nichž byla využita varianta 2.

3.3.2.1 Varianta 1

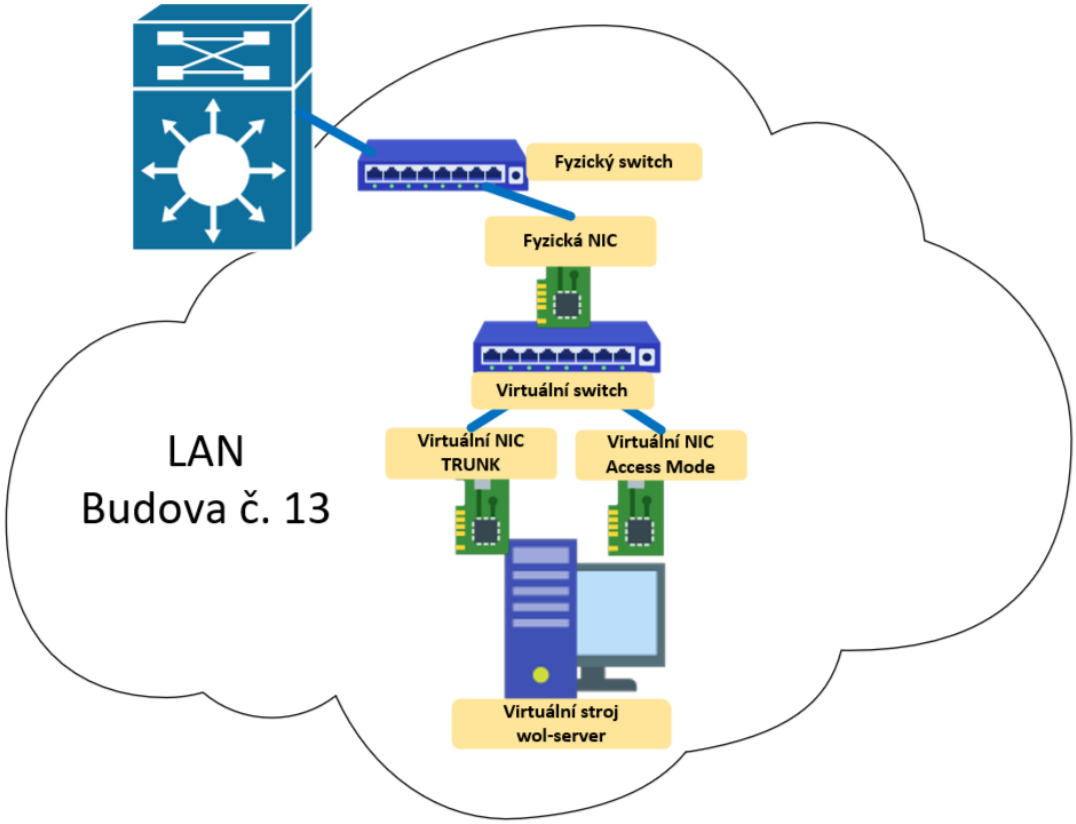
Konfigurace spočívá ve vytvoření samostatných virtuálních síťových adaptérů pro každou podsíť zvlášť nastavených do `access` modu a zařazených do jedné konkrétní VLAN (viz. Obr. 8 a Obr. 9). Pro každou novou podsíť je potřeba manuálně založit novou síťovou kartu a přiřadit jí dané číslo VLAN. Podniková síť má však několik desítek VLAN, takže manuální management např. 50 síťových karet by byl poměrně nepraktický.



Obr. 8 Přehled virtuálních adaptéru s nastavenými VLAN



Obr. 9 Virtuální adaptéry v režimu access v nastavení virtuálního stroje



Obr. 10 Síťová konfigurace virtuálního stroje

3.3.2.2 Varianta 2

Druhá varianta konfigurace pracuje se dvěma virtuálními síťovými kartami. První je nastavena do `access` modu a je zařazena do VLAN 511. Tato karta slouží především pro správu serveru přes vzdálené připojení prostřednictvím SSH. Druhá síťová karta je nastavena do režimu `trunk` a zařazena do více VLAN najednou pro možnost odesílání magický paketů (viz. Obr. 10). Nastavení `trunk` režimu však nelze provézt přímo v prostředí Hyper-V. Vše potřebné je nutno nastavit přes Windows Powershell s právy administrátora (viz. Obr. 11) [32]:

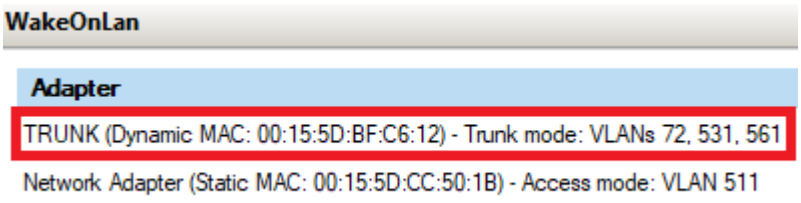
- po spuštění Powershellu vypíšeme seznam síťových adaptérů daného virtuálního stroje příkazem `Get-VMNetworkAdapterVlan –VMName "WakeOnLan"`,
- příkaz `$VMNetAdap = Get-VMNetworkAdapter -VMName "WakeOnLan"` uloží seznam adaptérů jako hodnoty do pole proměnné `$VMNetAdap`,
- pro přehlednost přejmenujeme potřebný adaptér na **TRUNK**: `Rename-VMNetworkAdapter $VMNetAdap[1] -newname "TRUNK"`,

- příkaz `Set-VMNetworkAdapterVlan -VMName "WakeOnLan" -VMNetworkAdapterName TRUNK -Trunk -AllowedVlanIdList "72,531,561" -NativeVlanId 0` nastaví adaptér do režimu *trunk* a zařadí ho do VLAN 72, 531 a 561

```
PS C:\Windows\system32> Get-VMNetworkAdapterVlan -VMName "WakeOnLan"

VMName    VMNetworkAdapterName Mode     VlanList
------    -------------------- ----     --------
WakeOnLan Network Adapter      Access   511
WakeOnLan Network Adapter      Untagged


PS C:\Windows\system32> $VMNetAdap = Get-VMNetworkAdapter -VMName "WakeOnLan"
PS C:\Windows\system32> Rename-VMNetworkAdapter $VMNetAdap[1] -newname "TRUNK"
PS C:\Windows\system32> Get-VMNetworkAdapterVlan -VMName "WakeOnLan"

VMName    VMNetworkAdapterName Mode     VlanList
------    -------------------- ----     --------
WakeOnLan Network Adapter      Access   511
WakeOnLan TRUNK                Untagged


PS C:\Windows\system32> Set-VMNetworkAdapterVlan -VMName "WakeOnLan" -VMNetworkAdapterName TRUNK -Trunk -AllowedVlanIdList "72,531,561" -NativeVlanId 0
PS C:\Windows\system32> Get-VMNetworkAdapterVlan -VMName "WakeOnLan"

VMName    VMNetworkAdapterName Mode   VlanList
------    -------------------- ----   --------
WakeOnLan Network Adapter      Access 511
WakeOnLan TRUNK                Trunk  0,72,531,561


PS C:\Windows\system32>
```

Obr. 11 Nastavení adaptéru *trunk* v Powershell

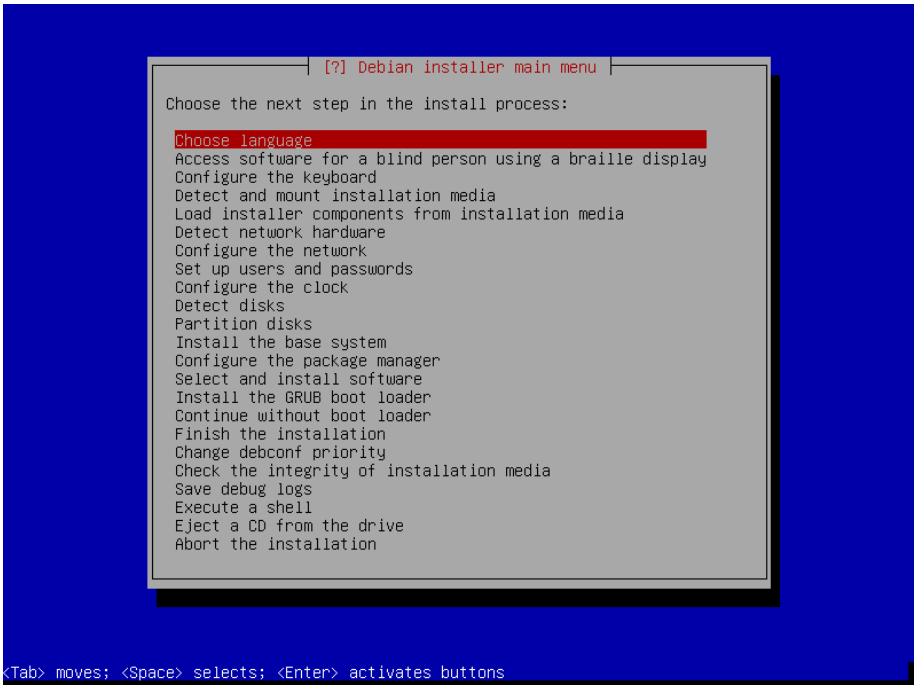
Změna konfigurace se následně zobrazí přímo v prostředí systému Hyper-V (viz. Obr. 12).



Obr. 12 Adaptér TRUNK s přehledem registrovaných VLAN

3.3.3 Instalace OS

Po spuštění virtuálního stroje se nabootuje instalační obraz a naběhne instalační nabídka. Z nabídky byla vybrána expertní instalace (viz. Obr. 13). Tento typ instalace vyžaduje veškerá nastavení samotným uživatelem. To zahrnuje například nastavení jména počítače, uživatelských účtů, hesel nebo volbu grafického rozhraní. V tomto případě nebylo nainstalováno žádné grafické rozhraní a se serverem se pracuje pouze prostřednictvím příkazové řádky. Byl navíc nainstalován SSH server pro možnost vzdálené správy serveru.



Obr. 13 Expertní instalace systému Linux Debian

Po dokončení instalace se načte rozhraní příkazové řádky s výzvou pro přihlášení do systému.

3.3.4 Síťová konfigurace – Linux

Pro využití *trunk* rozhraní připojeného k virtuálnímu stroji je potřeba adaptér nakonfigurovat přímo v operačním systému pro přístup ke konkrétním VLAN, které jsou na adaptéru registrovány. Pomocí textového editoru byla do souboru */etc/network/interfaces* přidána konfigurace (viz. Obr. 14) dle následujícího vzoru [33]:

```
auto [adapter-name].[VLAN]
    iface [adapter-name].[VLAN] inet static
    address [ip address]
```

Na obrázku níže lze vidět, že systém používá staré schéma pojmenování síťových rozhraní – *eth*. [34]

Obr. 14 Konfigurace adaptéru TRUNK v operačním systému

Příkazem *ping* bylo otestováno, že je v rámci dané VLAN možnost komunikovat s centrálním routerem sítě (viz. Obr. 15 a Obr. 16).

```
root@wol-server:~# ping -c 4 -I eth1.72 10.[redacted].1
PING 10.[redacted].1 (10.[redacted].1) from 10.[redacted] eth1.72: 56(84) bytes of data.
64 bytes from 10.[redacted].1: icmp_seq=1 ttl=255 time=0.381 ms
64 bytes from 10.[redacted].1: icmp_seq=2 ttl=255 time=0.345 ms
64 bytes from 10.[redacted].1: icmp_seq=3 ttl=255 time=0.359 ms
64 bytes from 10.[redacted].1: icmp_seq=4 ttl=255 time=0.325 ms

--- 10.[redacted].1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3051ms
rtt min/avg/max/mdev = 0.325/0.352/0.381/0.020 ms
```

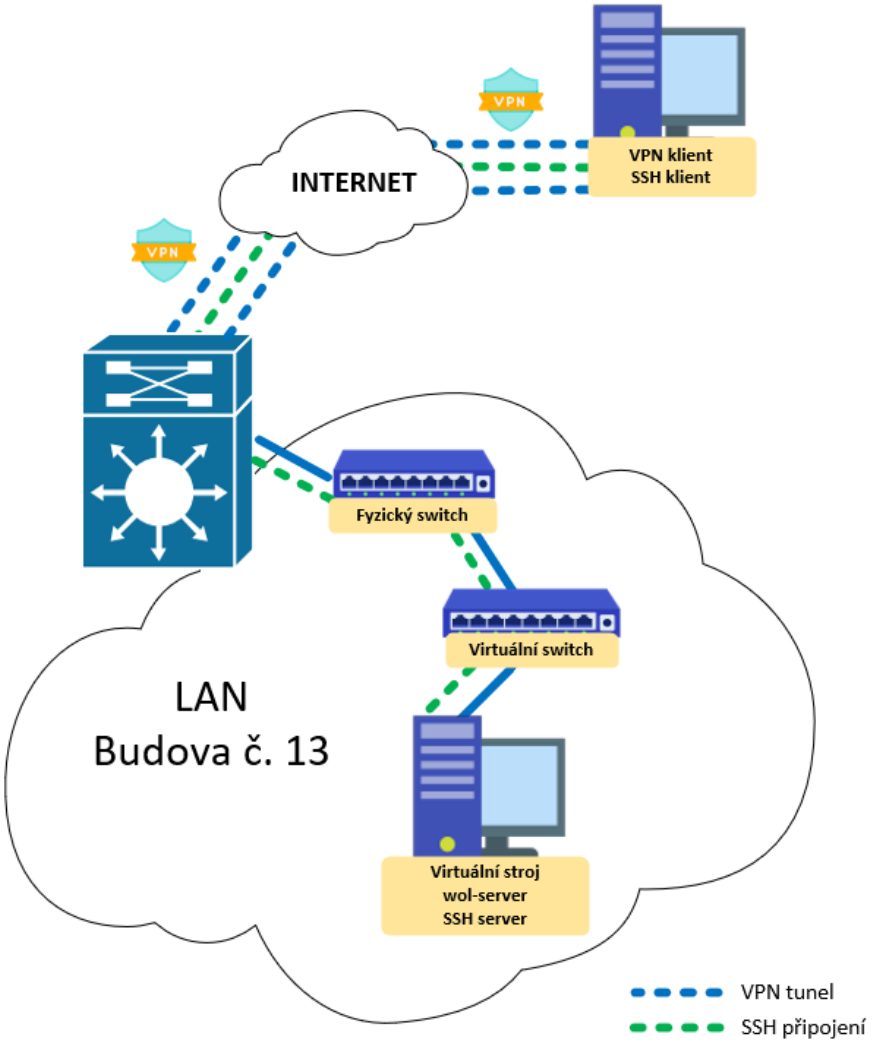
Obr. 15 Příkaz *ping* na bránu VLAN 72

```
root@wol-server:~# ping -c 4 -I eth1.561 10.[redacted].1
PING 10.[redacted].1 (10.[redacted].1) from 10.[redacted].4 eth1.561: 56(84) bytes of data.
64 bytes from 10.[redacted].1: icmp_seq=1 ttl=255 time=0.408 ms
64 bytes from 10.[redacted].1: icmp_seq=2 ttl=255 time=0.388 ms
64 bytes from 10.[redacted].1: icmp_seq=3 ttl=255 time=0.395 ms
64 bytes from 10.[redacted].1: icmp_seq=4 ttl=255 time=0.405 ms

--- 10.[redacted].1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3065ms
rtt min/avg/max/mdev = 0.388/0.399/0.408/0.008 ms
```

Obr. 16 Příkaz *ping* na bránu VLAN 561

3.4 Správa serveru

Server je vzdáleně spravován a veškeré změny na něm jsou prováděny prostřednictvím komunikačního protokolu SSH (viz. Obr. 17) za využití klientského softwaru PuTTY. Připojení k serveru je prováděno přes IP adresu (10.x.x.x) virtuální síťové karty nastavené do `access` modu. Druhá síťová karta v režimu `trunk` slouží pro odesílání magických paketů pro buzení počítačů v různých VLAN. Pro možnost komunikace se serverem je potřeba mít připojení do podnikového intranetu. K tomu podnik využívá VPN službu (viz. Obr. 17).



Obr. 17 Správa virtuálního stroje

4 INSTALACE A KONFIGURACE KLIENTA PRO TESTOVÁNÍ WAKE ON LAN

Následující kapitola popisuje instalaci a konfiguraci klientského počítače pro testování technologie Wake on LAN v síti. Cílem počítače je přijmout magický paket odeslaný virtuálním serverem a spustit se. Pro kontrolu by měl počítač po spuštění odeslat e-mail s informací o probuzení na testovací e-mailovou adresu. Podnikem byl poskytnutý starší počítač s hardwarem s defaultním nastavením BIOS/UEFI a naformátovaným diskem pro možnost instalace libovolného OS. Zapojení a instalace byla provedena v centrální budově č. 13.

4.1 Volba operačního systému

I přesto že většina počítačů v podnikové síti běží na systému Windows, byl pro testování zvolen operační systém Linux. Odeslání a příjem magického paketu totiž primárně nezávisí na typu operačního systému. Cílem navíc není otestovat příjem magického paketu konkrétním počítačem, nýbrž otestovat, zda magický paket bez problému projde sítí přes síťové prvky od serveru až k cílovému počítači, případně identifikovat a opravit vzniklé problémy při cestě paketu sítí. Testování příjmu paketu na systému Windows by mělo smysl v případě, že je k dispozici stejný typ počítače (typ hardwaru, verze OS a síťových ovladačů), který používají i zaměstnanci a tyto počítače by byly stejné v rámci celého podniku. Po úspěšném testování by se poté dalo s velkou pravděpodobností říct, že s danou konfigurací pro fungování WoL, bude možné probudit jakýkoliv počítač v podnikové síti. Počítače se však v rámci budov i jednotlivých místností mohou lišit jak typem hardwaru, tak verzí operačního systému nebo síťových ovladačů. Při použití operačního systému Windows navíc mohou vznikat problémy s příjmem paketu a probuzením počítače (viz. kapitola 1.8.6.) vzhledem k nejednoznačné konfiguraci síťového adaptéru pro WoL způsobené různým typem hardwaru. Tyto problémy jsou však důsledkem právě volby operačního systému, nikoliv špatnou konfigurací síťových prvků nebo serveru pro odesílání magických paketů, což může být při odstraňování problémů s příjmem paketu zavádějící. Naproti tomu operační systém Linux poskytuje nástroje (např. nástroj ethtool – viz. kapitola 1.7.2.2) i konfigurační soubory ve formátu textu, které umožňují snadnou a přehlednou konfiguraci síťových adaptérů pro účely příjmu magického paketu. Zásadním faktorem pro výběr Linuxu je také možnost jednoduché vzdálené správy a administrace

testovacího počítače prostřednictvím příkazové řádky stejně jako v případě serveru pro odesílání WoL paketů (viz. kapitola 3.2).

4.2 Instalace a konfigurace

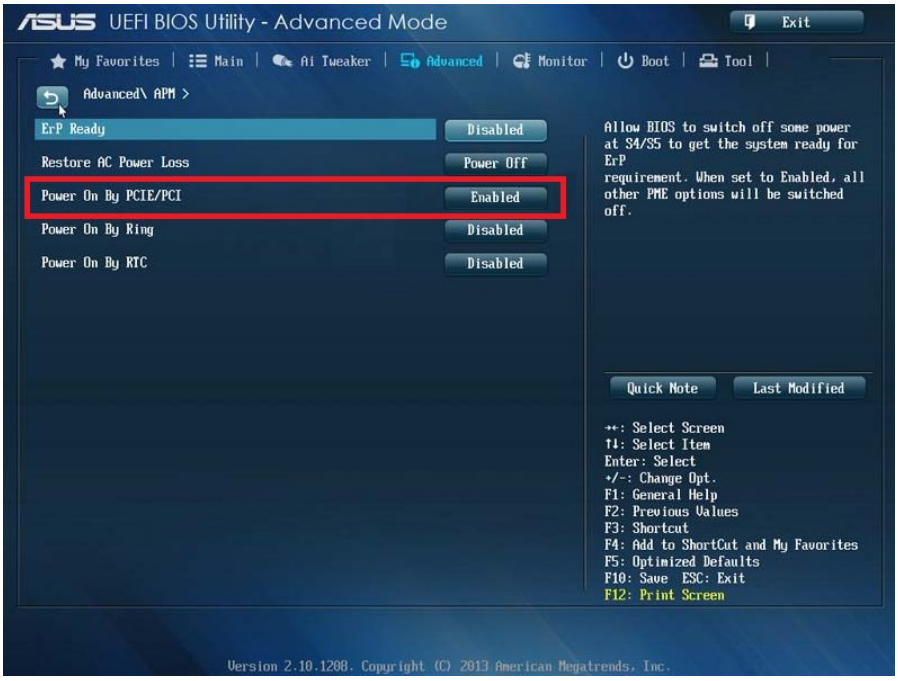
Pro klientský počítač byl zvolen operační systém Linux s distribucí Ubuntu verze 22.04.2 s grafickým rozhraním. [35] Instalační obraz byl pomocí nástroje balenaEtcher zapsán na přenosný USB disk. Po nabootování z USB byla instalace provedena přes sekvenci obrazovek se základními nastaveními systému (název počítače, vytvoření uživatele, …). Počítač byl nastavením IP adresy síťovému rozhraní zařazen do VLAN 531.

4.2.1 Konfigurace Wake on LAN

Pro příjem magického paketu síťovou kartou musí být povolena technologie WoL v BIOS/UEFI a v operačním systému.

4.2.1.1 BIOS/UEFI

Způsob povolení technologie Wake on LAN v BIOSu záleží na výrobci a konkrétním typu základní desky. Klientský počítač má základní desku ASUS B85M-E. Z dokumentace bylo zjištěno, že je pro tento typ desky potřeba v BIOSu v sekci *Advanced\ APM* povolit funkci *Power On By PCIE/PCI* (viz. Obr. 18) [36]. Je pravděpodobné, že u desky jiného výrobce bude potřeba technologii aktivovat přes jinou funkcionalitu pod jiným názvem.



Obr. 18 Povolení technologie WoL v BIOS/UEFI

4.2.1.2 Ubuntu

Aktivace WoL byla v operačním systému provedena pomocí nástroje *ethtool*, který slouží k úpravě parametrů síťových rozhraní v Linuxu [25]:

- Použitím příkazu *ip a* je zobrazen přehled síťových rozhraní a tím zjištěn název rozhraní – enp3s0 (viz. Obr. 19).

```
woluser@wol-client:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen
0
    link/ether 40:16:7e:70:9b:4c brd ff:ff:ff:ff:ff:ff
    inet 10.51.8.5/22 brd 10.51.11.255 scope global dynamic noprefixroute enp3s0
       valid_lft 21314sec preferred_lft 21314sec
    inet6 fe80::ab58:f480:d1b8:de00/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Obr. 19 Přehled informací o síťových rozhraních

- Použitím příkazu *sudo ethtool enp3s0 | grep Wake* je zjištěno, zda je technologie WoL na rozhraní aktivní – d (vypnuto), g (magický paket povolen) – u klientského počítače byla technologie defaultně vypnutá (viz. Obr. 20).

```
woluser@wol-client:~$ sudo ethtool enp3s0 | grep Wake
        Supports Wake-on: pumbg
        Wake-on: d
```

Obr. 20 Zobrazení stavu funkce WoL

- Použitím příkazu *sudo ethtool -s enp3s0 wol g* je funkce WoL na rozhraní aktivována (viz. Obr. 21).

```
woluser@wol-client:~$ sudo ethtool -s enp3s0 wol g
woluser@wol-client:~$ sudo ethtool enp3s0 | grep Wake
        Supports Wake-on: pumbg
        Wake-on: g
```

Obr. 21 Povolení funkce WoL pomocí *ethtool*

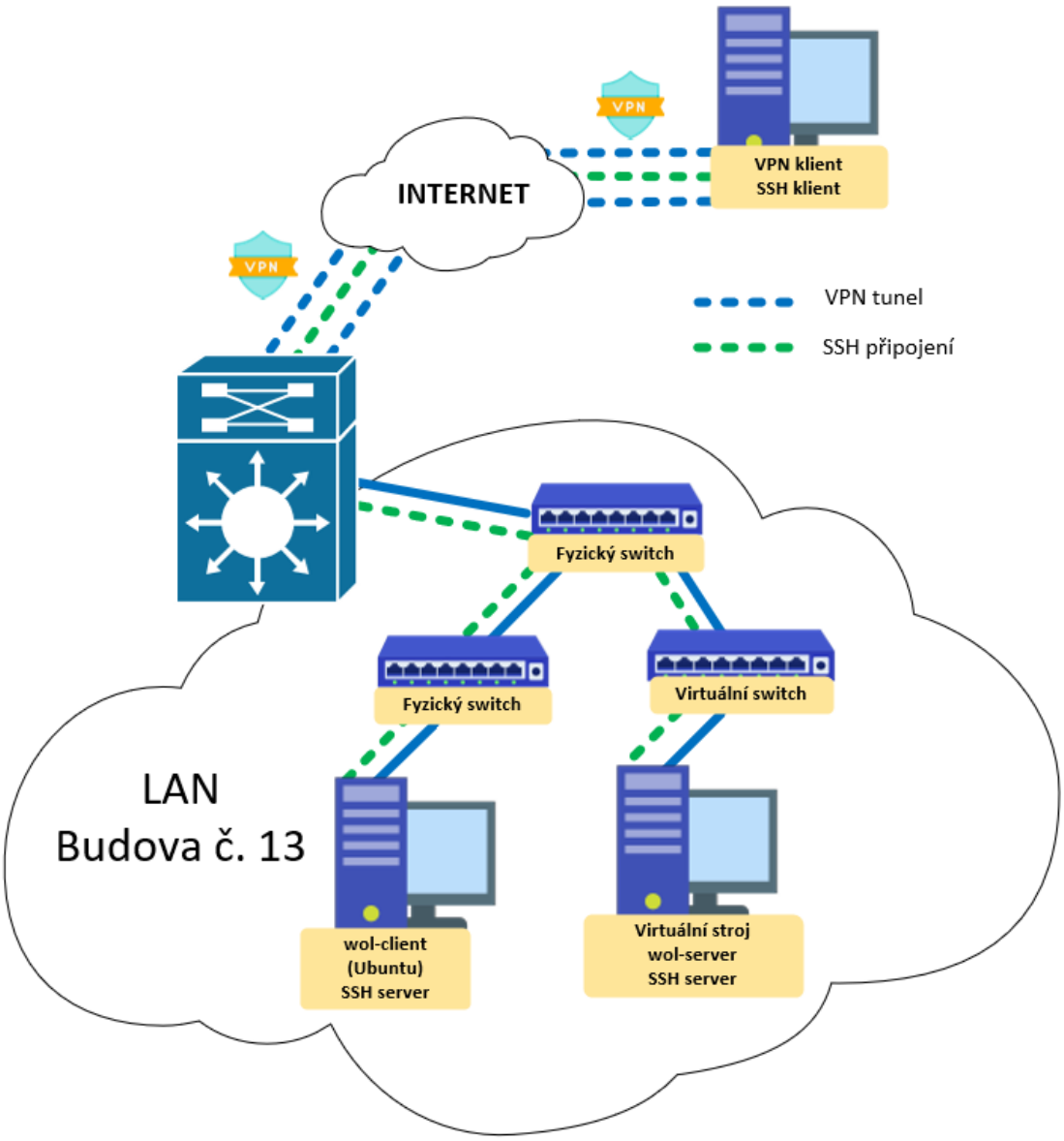
Je možné, že se při dalším restartu počítače nastavení neuloží. Proto byla ve složce */etc/systemd/system* vytvořena nová služba s názvem *wol.service* (viz. Obr. 22), která při každém startu počítače aktivuje technologii WoL na požadovaném rozhraní. Po uložení souboru s konfigurací byla služba aktivována příkazem *sudo systemctl enable wol.service*.

```
[Unit]
Description=Enable Wake On Lan

[Service]
Type=oneshot
ExecStart = /usr/sbin/ethtool --change enp3s0 wol g

[Install]
WantedBy=basic.target
```

Obr. 22 Nastavení služby *wol.service*

4.3 Správa klienta

Po fyzickém zapojení počítače do sítě a základním nastavení pro fungování WoL je dále klient spravován prostřednictvím SSH (viz. Obr. 23) a klientského softwaru PuTTY. Opět je potřeba připojení do intranetu podniku, které zajišťuje VPN služba (viz. Obr. 23).



Obr. 23 Správa klientského počítače

5 TVORBA A TESTOVÁNÍ SKRIPTU PRO BUZENÍ POČÍTAČŮ V PODNIKOVÉ SÍTI

Tato kapitola podrobně popisuje tvorbu a testování skriptu pro odesílání magických paketů bez grafického rozhraní. Cílem je otestovat správné odeslání magického paketu ze serveru na cílový testovací počítač v konkrétní podsíti. Skript je psán v programovacím jazyce Python a využívá modulů pro práci se sítí a soketovým programováním.

5.1 Python

Python je v dnešní době velmi populárním programovacím jazykem, a to z několika důvodů. Mezi ně patří jeho jednoduchá syntaxe, široká podpora knihoven a modulů, rychlost vývoje aplikací a široké spektrum využití od vývoje webových aplikací až po vědecké výpočty a analýzy dat. Podle TIOBE indexu patří tento programovací jazyk k nejpopulárnější v oblasti IT. [37] Má také mnoho knihoven a nástrojů pro síťové programování a práci s daty. Modul socket, kterou skript využívá pro síťovou komunikaci, je součástí standardní knihovny Pythonu, což usnadňuje práci s ní a zjednodušuje kód. Python také podporuje multiplatformnost, což znamená, že stejný kód může být spuštěn na různých operačních systémech. To je výhodné pro psaní skriptů, které musí být spustitelné na různých typech zařízení a operačních systémech. Celkově Python poskytuje jednoduchý, efektivní a přenositelný způsob psaní skriptů pro síťové operace a práci s daty, a proto byl zvolen pro pro tvorbu skriptu pro odesílání magických paketů v síti. [38]

5.2 Popis skriptu

Na základě MAC adresy vzdáleného počítače je ve skriptu vytvořen magický paket. Pomocí IP adresy počítače je poté vybráno ze všech dostupných rozhraní serveru jedno rozhraní, jehož IP adresa je ve stejné VLAN jako vzdálený počítač. Přes toto rozhraní je poté odeslán sestavený magický paket do dané VLAN. Odesílaní probíhá přes síťový protokol UDP na port 9. Paket lze odeslat na unicastovou adresu cílového počítače nebo broadcastovou adresu VLAN. Bylo tedy třeba určit, kterou adresu použít v závislosti na infrastruktuře sítě a síťové konfiguraci serveru.

5.2.1 Unicast vs Broadcast

Odesílání magického paketu probíhá v rámci stejné lokální podsítě. Router tedy do komunikace nezasahuje, čímž je eliminován problém, kdy router po 4 hodinách maže

záznamy z ARP tabulky. Unicast adresa by se tedy zdála jako lepší řešení s ohledem na nižší zátěž sítě. Může však nastat problém, kdy server odesílající magický paket nemá v ARP tabulce MAC adresu cílového počítače. Při vytváření rámce by tedy počítač odeslal ARP dotaz (broadcast). Pokud je však cílový počítač vypnutý, tak nikdy na tento dotaz server nedostane odpověď a rámec nevytvoří (viz. kapitola 1.5.1). Použití unicast adresy by tedy bylo funkční pouze do doby, kdy by měl server záznam cílové MAC adresy ve své lokální ARP tabulce.

Vzhledem k odesílání paketu ve stejné podsíti je vyřešen i problém s použitím směrového broadcastu, který používá boradcastovou adresu dané podsítě a je standardně na routerech i firewallech zakázána. Tato technika je totiž náchylná na použití útoků jako je např. Smurf Attack, který způsobuje Denial of Services (DoS) (viz. kapitola 1.5.2). Pokud jsou však server a cílový počítač ve stejné podsíti, je situace poměrně jednoduchá a bezproblémová. Při použití broadcastu není potřeba mít ani záznam o cílové MAC adrese v APR tabulce. Potenciálním rizikem může být větší zatížení sítě při více současných požadavcích pro odeslání magických paketů. Broadcastová adresa je však spolehlivějším řešením, a proto byla použita ve skriptu pro odeslání magického paketu do konkrétní podsítě.

5.3 Tvorba skriptu

5.3.1 Moduly

Skript využívá následující moduly:

- `socket`: Pro vytvoření a ovládání síťového socketu pro odesílání magických paketů. [39]
- `netifaces` (ve skriptu zkratka nif): Získání informací o síťových rozhraních serveru a jejich konfiguraci. [40]
- `binascii`: Pro převod MAC adresy na binární data. [41]
- `ipaddress` (ve skriptu zkratka ip): Pro práci s IP adresami. [42]

Tyto moduly jsou pomocí příkazu *import* načteny do programu pro přístup k jejich obsahu (viz. Obr. 24).

```
import socket
import netifaces as nif
import binascii
import ipaddress as ip
```

Obr. 24 Použité moduly pro tvorbu skriptu

5.3.2 Definice funkce

Samotný skript začíná definicí funkce *wake_on_lan(ip_address, ethernet_address)*, která má dva vstupní parametry: *ip_address* a *ethernet_address* (viz. Obr. 25). Parametr *ip_address* obsahuje IP adresu počítače, který má být probuzen, zatímco *ethernet_address* obsahuje fyzickou adresu jeho síťové karty. Na začátku skriptu jsou také definovány proměnné *iface_ip* (IP adresa rozhraní, ze kterého bude odeslán magický paket), *bind_interface* (název rozhraní, na které se naváže vytvořený soket pro odeslání paketu) a *interface_broadcast* (broadcastová adresa, na kterou bude paket odeslán) s hodnotou *None* (viz. Obr. 25). Funkce vrací hodnotu *True* v případě úspěšného odeslání magického paketu, v opačném případě vrací hodnotu *False*.

```
def wake_on_lan(ip_address, ethernet_address):
    iface_ip = bind_interface = interface_broadcast = None
```

Obr. 25 Definice funkce *wake_on_lan*

5.3.3 Převod MAC adresy na binární data

Pomocí modulu *binascii* je nejprve převedena fyzická adresa z hexadecimálního řetězce do binární podoby pomocí funkce *unhexlify()* a tato hodnota je uložena do proměnné *hwa*. Poté je sestaven samotný magický paket založený na této fyzické adrese. Paket je složen z posloupnosti bajtů: prvních 6 bajtů obsahuje hodnotu *0xff* (*'\xff' * 6*),

následuje 16krát zopakovaná binární hodnota fyzické adresy uložená v proměnné *hwa* (*hwa * 16*). Magický paket je tedy složen *jako* *b'\xff' * 6 + hwa * 16* a jeho hodnota je uložena do proměnné *msg* (viz. Obr. 26).

```
# Convert MAC address to binary data
hwa = binascii.unhexlify(ethernet_address)

# Build magic packet
msg = b'\xff' * 6 + hwa * 16
```

Obr. 26 Převod MAC adresy na binární data

5.3.4 Maska podsítě a broadcast adresa rozhraní

V další části se využitím funkce *interfaces()* z modulu *netifaces* získají všechny síťové rozhraní serveru, které jsou uloženy do proměnné *interfaces* a pomocí *for* cyklu jsou postupně procházeny hodnoty všech těchto rozhraní. Pro každé rozhraní se získá maska podsítě - *nif.ifaddresses(interface).get(nif.AF_INET)*:

1. Funkce *nif.ifaddresses(interface)* vrátí seznam slovníků, kde každý slovník popisuje jeden aspekt sítě pro dané rozhraní. Může se například jednat o IPv4 nebo IPv6 adresy, MAC adresy, síťové masky apod.
2. Pomocí metody *get(nif.AF_INET)* se získá seznam všech IPv4 adres pro dané rozhraní, konstanta *AF_INET* označuje typ sítě (v tomto případě IPv4).
3. Z tohoto seznamu se vybere první prvek, což je slovník obsahující informace o první IPv4 adrese v seznamu.
4. Z vybraného slovníku se pomocí metody *get('netmask')* získá maska podsítě.
5. Výsledná hodnota je uložena do proměnné *mask*.

Podobně jako maska podsítě je také získána broadcast adresa daného rozhraní. Místo parametru *'netmask'* je však v metodě *get* použitý parametr *'broadcast'* - *nif.ifaddresses(interface).get(nif.AF_INET)[0].get('broadcast')*. Hodnota je uložena do proměnné *interface_broadcast* (viz. Obr. 27).

```
# Get all network interfaces
interfaces = nif.interfaces()

for interface in interfaces:
    try:
        # Get subnet mask of the interface
        mask = nif.ifaddresses(interface).get(nif.AF_INET)[0].get('netmask')

        # Get broadcast address of the interface
        interface_broadcast = nif.ifaddresses(interface).get(nif.AF_INET)[0].get('broadcast')
```

Obr. 27 Získání masky a broadcast adresy

5.3.5 Určení sítě vzdáleného počítače

Po získání masky a broadcast adresy rozhraní je vytvořena IPv4 síť na základě IP adresy hosta a masky podsítě (proměnná mask) získané z daného síťového rozhraní (proměnná interface). V prvním kroku se IP adresa hosta a maska podsítě spojí do řetězce ve formátu *ip_address/mask*. Dále je vytvořena instance třídy *IPv4Network* z modulu *ipaddress* s použitím vytvořeného řetězce jako argumentu. Například, pokud má host IP adresu 192.168.1.100 a masku podsítě 255.255.255.0 (tj. /24), pak se vytvoří instance *IPv4Network('192.168.1.100/24')*, což reprezentuje IPv4 síť 192.168.1.0/24. Tato část kódu umožňuje snadno porovnat broadcast adresu vypočítanou z dané sítě s broadcast adresou získanou z daného síťového rozhraní, což je velmi důležité pro rozhodování, na jaké síťové rozhraní (do jaké VLAN) bude magický paket poslán. Hodnota získané sítě je uložena do proměnné *host_network_address* (viz. Obr. 28).

```
# Create IPv4 network from IP address of host and subnet mask of the interface
# e.g. 10.51.8.5/255.255.252.0 creates 10.51.8.0/22 network
host_network_address = ip.IPv4Network(ip_address + '/' + mask, strict=False)
```

Obr. 28 Vytvoření sítě z masky rozhraní a IP adresy počítače

5.3.6 Určení správného rozhraní pro odeslání magického paketu

Další část kódu (viz. Obr. 29) hledá příslušné síťové rozhraní (proměnná *interface*), na které bude odeslán magický paket. Pokud je definovaná broadcastová adresa (*if interface_broadcast is not None* – některá rozhraní nemusí mít ve svém slovníku definovanou broadcast adresu), tak je pomocí podmínky ověřeno, zda se shoduje

s broadcastovou adresou sítě, která byla vypočtena z IP adresy a masky sítě. V případě, že se shoduje, IP adresa tohoto rozhraní je uložena do proměnné *iface_ip* a název rozhraní do proměnné *bind_interface* pro další použití. Pokud není nalezena shoda, cyklus pokračuje dál ve vyhledávání správného rozhraní pro odeslání paketu. Pokud je tedy broadcast adresa rozhraní stejná jako broadcast adresa vypočítaná na základě IP adresy cílového počítače a masky podsítě, znamená to, že cílový počítač je v této podsíti a magický paket bude odeslán na broadcast adresu tohoto rozhraní.

```
    if interface_broadcast is not None:
        # If broadcast address of the interface equals broadcast address of created network from the host
        # PC belongs to VLAN of this interface (MP will be sent to the broadcast of this interface)
        if str(host_network_address.broadcast_address) == interface_broadcast:
            iface_ip = nif.ifaddresses(interface).get(nif.AF_INET)[0].get('addr')
            bind_interface = interface
            break
except (KeyError, ValueError, TypeError):
    continue
```

Obr. 29 Porovnání broadcastových adres

5.3.7 Odeslání magického paketu

Tato část skriptu (viz. Obr. 30) se stará o odeslání magického paketu pomocí protokolu UDP na broadcastovou adresu, která je specifická pro danou VLAN. Konkrétně se v této části skriptu používají funkce modulu *socket* k vytvoření a konfiguraci soketu, kterým bude paket odeslán:

1. *socket.socket(socket.AF_INET, socket.SOCK_DGRAM)* - vytvoří nový soket s protokolem IPv4 (konstanta *AF_INET*) a protokolem (konstanta *SOCK_DGRAM*) UDP.
2. *soc.setsockopt(socket.SOL_SOCKET, 25, bytes(bind_interface, 'utf-8'))* - nastaví soket na hodnotu 25, která odpovídá konstantě *SO_BINDTODEVICE*, která slouží k navázání soketu na konkrétní síťové rozhraní serveru. Argument *bind_interface* je název rozhraní, na které se soket váže, a je předán jako bytová posloupnost.
3. *soc.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)* - nastaví sokketovou volbu SO_BROADCAST na hodnotu 1, což umožní odesílat pakety na broadcastové adresy.

4. *soc.bind((iface_ip, 0))* - váže soket na IP adresu *iface_ip* a libovolný volný port.
5. *soc.sendto(msg, (interface_broadcast, 9))* - odesílá datagram obsahující magický paket na broadcastovou adresu *interface_broadcast* a port 9, což je standardní port pro protokol Wake-on-LAN (lze použít i port 7)
6. *soc.close()* - uzavírá socket po odeslání paketu.

```
if interface_broadcast and bind_interface and iface_ip:
    # Send packet to broadcast address of the specific VLAN (interface) using UDP port 9
    try:
        soc = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        soc.setsockopt(socket.SOL_SOCKET, 25, bytes(bind_interface, 'utf-8'))
        soc.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
        soc.bind((iface_ip, 0))
        soc.sendto(msg, (interface_broadcast, 9))
        soc.close()
        return True
    except (KeyError, IndexError, socket.error):
        return False
else:
    return False
```

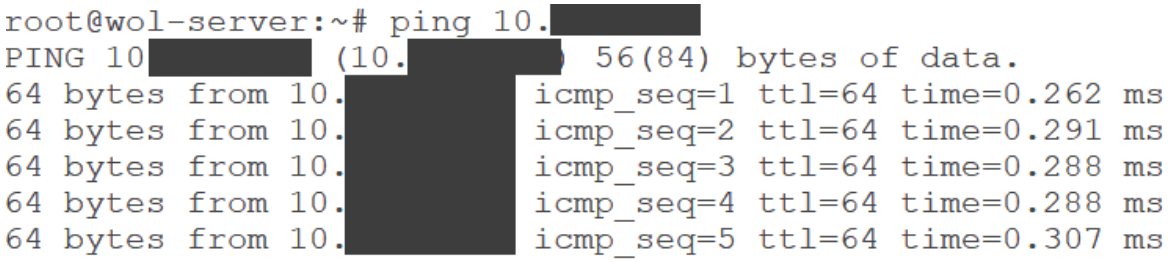
Obr. 30 Odeslání magického paketu

5.4 Testování skriptu

Skript byl na serveru pojmenován jako wakeonlan.py. Bylo do něho přidáno volání vytvořené funkce *wake_on_lan* s hodnotami IP adresy a MAC adresy testovacího počítače. Pomocí příkazu *chmod u+x* bylo uděleno skriptu právo na vykonávání pro uživatele, který skript vytvořil. [43] Skript byl vykonán příkazem *./wakeonlan.py* v aktuálním adresáři. Testovací počítač je ve VLAN 531, na základě jeho IP adresy je tedy ve skriptu vyhledáno příslušné rozhraní s broadcastovou IP adresou právě pro VLAN 531. Na tuto broadcast adresu je následně odeslán magický paket sestavený z MAC adresy cílového počítače.

5.4.1 Výsledek testování

Protože nelze ověřit příjem paketu při vypnutém klientském počítači, je po přibližně 20 vteřinách od odeslání paketu proveden z příkazové řádky příkaz *ping* na IP adresu testovacího počítače, který úspěšně vrací odpověď (viz. Obr. 31). Nevyskytly se žádné problémy při cestě paketu sítí.



Obr. 31 Příkaz ping na testovací počítač po odeslání paketu

6 TVORBA GRAFICKÉHO ROZHRANÍ PRO WAKE ON LAN

Tato kapitola popisuje tvorbu grafického rozhraní s implementací skriptu pro buzení počítačů v podnikové síti (viz. kapitola 8). Rozhraní je dostupné z virtuálním serveru (viz. kapitola 6) a bylo vytvořeno formou webové aplikace v aplikačním frameworku Python Django s využitím SQLite databáze. Cílem tohoto rozhraní je umožnit uživatelům registraci svého pracovního počítače do aplikace a následné vzdálené zapnutí tohoto počítače zasláním magického paketu. V úvodní části je stručný popis rozhraní, dále se kapitola v rámci tvorby rozhraní věnuje především backendové (serverové) části, do které patří např. definice databázového modelu, tvorba Django views s implementací WoL skriptu, zpracování dat, komunikace s databází, validace registračního formuláře a celkové zabezpečení aplikace. Cílem kapitoly není poskytnout podrobný návod, jak vytvořit přesnou kopii rozhraní, nýbrž nastínit a ukázat tvorbu důležitých aspektů aplikace pro její správný chod.

6.1 Aplikační framework Django

Tato podkapitola popisuje framework Django, což je webový aplikační framework napsaný v programovacím jazyku Python. Django nabízí kompletní sadu nástrojů a funkcí pro vývoj robustních a škálovatelných webových aplikací. Tento framework byl zvolen jako nástroj pro tvorbu grafického rozhraní kvůli výhodám, které poskytuje [44]:

- Rychlost a produktivita: Framework je známý pro svou vysokou produktivitu a rychlost vývoje. Poskytuje mnoho zabudovaných funkcí, konvencí pro správu databáze, formulářů, URL adres nebo zpracování uživatelského přihlášení a šablonovací systém (viz. kapitola 6.1.2.4). Má navíc k dispozici již vytvořený administrační systém a samostatný odlehčený webový server pro vývoj a testování.
- Přehlednost: Django funguje na programovacím principu DRY (Do not Repeat Yourself), díky kterému se programátoři mohou vyvarovat opakování kódu. Protože každý projekt, který je v Djangu vytvořen má stejnou základní strukturu, je pro ostatní programátory jednoduché se vyznat v cizích projektech
- Rozsáhlá dokumentace: Django má komplexní a přehlednou dokumentaci, která pokrývá všechny aspekty frameworku. Dokumentace obsahuje návody, příklady a reference, což usnadňuje pochopení a používání funkcí frameworku, čímž je usnadněn a zrychlen vývoj aplikace.

- Bezpečnost: Django má vestavěné bezpečnostní mechanismy, které chrání aplikaci před běžnými bezpečnostními hrozbami, jako jsou útoky typu Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF) nebo SQL injection. Framework poskytuje vrstvu ochrany, která umožňuje psát bezpečný kód a chránit data uživatelů.
- Škálovatelnost: Pomocí Djanga lze snadno zvládout nárust provozu aplikace pomocí technik jako je horizontální škálování, nasazení na více serverů a použití mezipaměťových úložišť. Framework také podporuje využití cloudových služeb, jako je Amazon Web Services (AWS) nebo Google Cloud Platform (GCP), což umožňuje snadno škálovat aplikaci podle potřeb.
- Velká komunita: Django má rozsáhlou komunitu vývojářů, kteří jej aktivně podporují a přispívají k jeho vývoji. Existuje mnoho diskuzních fór, komunitních webů, knih a tutoriálů, které pomáhají pří vývoji aplikace.

6.1.1 Historie

Django byl vyvinut společností Django Software Foundation a poprvé byl vydán jako open-source projekt v roce 2005. Od té doby se stal jedním z nejpopulárnějších webových frameworků díky svému výkonu, bezpečnosti a flexibilitě. [45]

6.1.2 Architektura

Architektura Django je založena na architektonickém vzoru Model-View-Controller (MVC), který rozděluje aplikaci do tří hlavních vrstev: modely (Models), pohledy (Views) a šablony (Templates). Každá vrstva má specifickou roli a odpovědnost, což usnadňuje organizaci a údržbu kódu. [46]

6.1.2.1 Modely (Models)

Modely v Django reprezentují datovou strukturu aplikace a jsou definovány v souboru models.py. V této vrstvě se definují třídy, které popisují entity a jejich atributy, jako jsou uživatelé, produkty, objednávky apod. Každá třída modelu je odvozena od třídy `django.db.models.Model` a obsahuje atributy, které představují pole v databázi. Modely také definují vztahy mezi entitami pomocí cizích klíčů a dalších polí. Django využívá ORM (Object-Relational Mapping) pro komunikaci s databází a poskytuje jednoduché a výkonné API pro manipulaci s daty (viz. 6.1.2.2). [47]

6.1.2.2 ORM (Object–relational mapping)

Django poskytuje ORM vrstvu, která umožňuje vývojářům pracovat s databází pomocí objektově orientovaného přístupu. ORM abstrahuje konkrétní databázový systém a poskytuje API (Application Programming Interface) umožňující vytvářet, manipulovat a přistupovat k objektům, které jsou mapovány na záznamy v databázi. [48]

6.1.2.3 Pohledy (Views)

Pohledy jsou zodpovědné za logiku zpracování požadavků a vykreslení výsledků do šablony a jsou definovány v souboru views.py. Každý pohled je definován jako funkce nebo třída, která je připojena k URL adrese a obsluhuje HTTP požadavky. Pohledy zpracovávají vstupní data, komunikují s modely (pomocí Django ORM) a vybírají odpovídající šablonu pro vykreslení výsledků. Pohledy také mohou provádět autentizaci, autorizaci a další úkony spojené s požadavky. [49]

6.1.2.4 Šablony (Templates)

Django poskytuje robustní a efektivní systém šablon pro tvorbu dynamických webových stránek. Tento systém využívá ‚templating' jazyk (tagy a filtry), který usnadňuje oddělení prezentace od logiky aplikace a umožňuje opakované použití šablon pro různé části webové stránky. Pomocí tagů a filtrů lze šablony naplnit daty získanými z databáze nebo jiných zdrojů. To umožňuje snadnou tvorbu dynamických stránek s opakujícími se prvky, podmíněným zobrazením a vkládáním dat. Django obsahuje několik základních prvků syntaxe šablonovacího jazyka [50]:

- Značky (tags): Tyto značky jsou ohraničeny *{% %}* a slouží k provedení určité akce nebo kontroly v šabloně. Například lze použít značku pro cyklus, podmínku, volání funkce atd.
- Filtry (filters): Filtry jsou ohraničeny závorkami *{{ }}* a slouží k úpravě výstupu proměnných v šabloně. Například lze použít filtr pro formátování času, převod textu na malá nebo velká písmena.
- Proměnné (variables): Proměnné jsou také ohraničeny *{{ }}* a představují data, která jsou v šabloně zobrazena. Tyto proměnné mohou být předány do šablony z pohledu (view) a použity k vykreslení dynamického obsahu.

6.1.3 Mapování

V Django se mapování vztahuje k procesu propojení URL cest s odpovídajícími pohledy (views) v aplikaci. Mapování umožňuje určit, jaké pohledy mají být volány při požadavcích na různé URL adresy a jejich parametry. V Django je mapování implementováno v souboru urls.py, který je součástí každé aplikace. V tomto souboru jsou definovány URL cesty a k nim přiřazeny odpovídající pohledy. Django používá regulární výrazy pro definici URL formátů, které slouží k mapování konkrétních URL adres na pohledy. [51]

6.2 SQLite

SQLite je relační databázový systém, známý pro svou jednoduchost, malou velikost a snadnou integraci do aplikací. Je navržen bez nutnosti externího databázového serveru, kdy data jsou ukládána do jediného souboru, nezávislém na platformě. [52] Tento databázový systém byl pro tvorbu aplikace zvolen vzhledem k jednoduchosti navržené databáze pro aplikaci (viz kapitola 6.4). Databáze má pouze jednu tabulku s pěti atributy pro uložení záznamu o počítači uživatele. Aplikace tedy nebude pracovat s komplexními relacemi (databázové tabulky) a nebude vytvářet složité dotazy na databázi. Neočekává se ani významný růst objemu dat a předpokládá se nízká až střední zátěž webu, pro kterou je SQLite dostačující i v rámci jediného virtuálního počítače, na kterém bude aplikace dostupná. [53] Není tedy nutné použití robustních databázových systémů jako PostgreSQL nebo MySQL, které navíc vyžadují instalaci a konfiguraci databázového serveru.

Django poskytuje abstrakci nad různými databázovými systémy, včetně SQLite, aby byla umožněna snadná práce s databází. Při použití SQLite v Django je konfigurace databáze definována v souboru settings.py v projektu aplikace. Zde je specifikováno SQLite jako databázový backend s cestou k SQLite souboru (viz. Obr. 32).

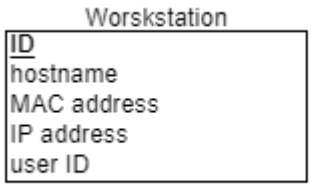
```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

Obr. 32 Konfigurace databáze

6.3 Databáze

Aplikace je napojena na SQLite databázi. Tato databáze má pouze jednu jednoduchou tabulku udržující záznamy o registrovaných počítačích všech uživatelů (viz. Obr. 33). Tabulka počítačů má následující databázové pole:

- ID: Jedinečný identifikátor počítače – sloupec v tabulce má celočíselnou hodnotu, nesmí obsahovat hodnotu NULL, musí být jedinečný pro každý záznam a bude automaticky generovat nové hodnoty o jedničku vyšší než předchozí maximum (`integer NOT NULL PRIMARY KEY AUTOINCREMENT`).
- Hostname: Název počítače – sloupec v tabulce má hodnotu řetězce s maximální délkou 50 znaků, nesmí obsahovat hodnotu NULL (`varchar(50) NOT NULL`)
- MAC address: MAC adresa počítače – sloupec v tabulce má unikátní hodnotu řetězce s maximální délkou 12 znaků, nesmí obsahovat hodnotu NULL (`varchar(12) NOT NULL UNIQUE`)
- IP address: IP adresa počítače – sloupec v tabulce má unikátní hodnotu řetězce s maximální délkou 15 znaků, nesmí obsahovat hodnotu NULL (`varchar(15) NOT NULL UNIQUE`)
- user ID: identifikátor uživatele, který počítač registroval – sloupec v tabulce má hodnotu řetězce s maximální délkou 128 znaků, nesmí obsahovat hodnotu NULL (`varchar(128) NOT NULL`)



Obr. 33 Databázová tabulka *Workstation*

6.4 Databázový model

V souboru models.py bylo za pomoci Python tříd a jejich atributů vytvořeno databázové schéma tak, jak bylo výše popsáno. Byl vytvořen Django model `Workstation` odvozený z třídy `django.db.models.Model` (viz. Obr. 34). Tento model má následující pole:

- *mac_address*: Pole je typu CharField s maximální délkou 12 znaků pro ukládání MAC adresy počítače. Používá validační funkci validate_mac_address pro ověření korektního formátu adresy. Pole je také nastaveno jako unikátní (každý počítač má unikátní MAC adresu). MAC adresa počítače je na základě její validace do databáze ukládána bez oddělovačů s délkou 12 znaků. Oddělovače slouží pouze pro lepší přehlednost a zlepšení čitelnosti a z hlediska adresy nemají žádnou hodnotu. Při použití MAC adresy pro odeslání magického paketu by oddělovače musely být stejně odstraněny.
- *ip_address*: Pole je typu CharField s maximální délkou 15 znaků pro ukládáni IP adresy počítače. Je zde použita validační funkce *validate_ip_address* z třídy *django.core.validators*. Toto pole je také nastaveno jako unikátní (každý počítač v síti má unikátní IP adresu).
- *hostname*: Pole je typu *CharField* s maximální délkou 50 znaků a je validováno funkcí *validate_hostname*.
- *user_id*: Pole je typu *CharField* pro uložení identifikátoru uživatele, který počítač zaregistroval do aplikace.

```
class Workstation(models.Model):
    mac_address = models.CharField(max_length=12,
                                   validators=[validate_mac_address],
                                   unique=True,
                                   error_messages={
        "unique": "Počítač s touto MAC adresou již existuje!"
    })

    ip_address = models.CharField(max_length=15,
                                  validators=[validate_ip_address],
                                  unique=True,
                                  error_messages={
        "unique": "Počítač s touto IP adresou již existuje!"
    })

    hostname = models.CharField(max_length=50, validators=[validate_hostname])
    user_id = models.CharField(max_length=128)

    def __str__(self):
        return self.mac_address
```

Obr. 34 Databázový model *Workstation* v Django

6.4.1 Validace polí

Pro pole databázového modelu počítače byly vytvořeny serverové validace (viz. Obr. 35), které jsou prováděny po odeslání dat v registračním formuláři a před uložením do databáze:

- *validate_mac_address*: Funkce kontroluje hodnotu MAC adresy ve formátu [A-F0-9] s délkou 12 znaků. V tomto případě lze tedy do databáze uložit pouze MAC adresu, která bude obsahovat velká písmena A-F nebo čísla 0-9 s přesnou délkou 12 znaků (bez jakýchkoliv oddělovačů).
- *validate_ip_address*: Funkce kontroluje správnou hodnotu IP adresy pomocí vestavěné Django validační funkce *validate_ipv4_address*.
- *validate_hostname*: Funkce kontroluje, zda název počítače obsahuje povolené znaky (alfanumerické znaky, mezery a pomlčky) a má délku od 3 do 50 znaků.

```
def validate_mac_address(value):
    mac_addr_regex = '^([A-F0-9]{12})$'
    if not re.match(mac_addr_regex, value):
        raise ValidationError('Nevalidní formát MAC adresy!')


3 usages
def validate_ip_address(value):
    try:
        validate_ipv4_address(value)
    except ValidationError:
        raise ValidationError('Nevalidní formát IP adresy!')


1 usage
def validate_hostname(value):
    hostname_regex = '^([a-zA-Z0-9 -]{3,50})$'
    if not re.match(hostname_regex, value):
        raise ValidationError('Nevalidní formát názvu počítače!')
```

Obr. 35 Validace pro pole modelu *Workstation*

6.4.2 Migrace

Migrace v Django slouží k synchronizaci definicí modelů se skutečným stavem databáze. Existuje několik příkazů pro práci s migrací a pro zpracování databázového schématu [54]:

- *migrate*: Použití a odstranění migrace.
- *makemigrations*: Vytvoří novou migraci na základě změn, které byly provedeny v modelech pomocí příkazu *migrate*.
- *sqlmigrate*: Zobrazí příkazy, které byly použity pro aplikaci změn v databázi.
- *showmigrations*: Zobrazí seznam všech migrací vytvořených v rámci projektu

Pro vytvoření migrace na základě vytvořeného modelu *PcForm* (viz. kapitola 6.4) byl použit v příkazové řádce v adresáři projektu příkaz *python3 manage.py makemigrations*. Pro aplikaci migrace na databázi byl proveden příkaz *python3 manage.py migrate*, čímž byla v databázi vytvořena tabulka pro ukládání počítačů (viz. kapitola 6.3). Příkazem *python3 manage.py sqlmigrate* byly zobrazeny SQL příkazy, které byly v rámci aplikace migrace provedeny pro vytvoření tabulky v databázi.

```
CREATE TABLE "wolapp_workstation"
("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT,
"hostname" varchar(50) NOT NULL,
"ip_address" varchar(15) NOT NULL UNIQUE,
"mac_address" varchar(12) NOT NULL UNIQUE,
"user_id" varchar(128) NOT NULL);
```

Obr. 36 Příkazy SQL pro aplikování migrace

6.5 Mapování

V souboru urls.py jsou v rámci aplikace pomocí funkce *path* definovány URL cesty spolu s odpovídajícími Django pohledy (anglicky views). V tomto případě jsou importovány soubory registration_views, computer_views, modal_views a wol_views rozdělené na základě funkcí a účelu pohledů. Například cesta *'delete_pc/<int:workstation_id>'* je přiřazena k pohledu *delete_pc* z modulu *computer_views* atd. (*<int:workstation_id>* je URL parametr předáván jako argument příslušné funkci pohledu).

6.6 Formulář

Tento formulář je vytvořen v souboru forms.py a slouží pro registraci uživatelského počítače do aplikace pro možnost zaslat tomuto počítači magický paket pro probuzení. Do formuláře je potřeba zadat název počítače, jeho MAC a IP adresu. Na formulářová data je aplikována klientská (Javascript) i serverová (Django) validace.

Byla vytvořena třída formuláře s názvem *PcForm* (viz. Obr. 37), která dědí z třídy *ModelForm*, což zajišťuje vytvoření formuláře na základě databázového modelu počítače (Workstation). V definici třídy *Meta* jsou v atributu *fields* nastaveny pole databázového modelu, které jsou zobrazeny ve formuláři na straně klienta. V rámci těchto polí formuláře jsou automaticky aplikovány i serverové validace definované v modelu. [55] Třída formuláře také obsahuje nastavení popisu polí ve formuláři a jejich widgety.

```
class PcForm(ModelForm):
    mac_address = forms.CharField(max_length=17,
                                  widget=forms.TextInput(
                                      attrs={'placeholder': 'např. AA-BB-CC-DD-EE-FF'}),
                                  label='Fyzická adresa (MAC)')
    class Meta:
        model = Workstation
        fields = ['hostname', 'mac_address', 'ip_address']
        widgets = {
                'ip_address': forms.TextInput(attrs={'placeholder': 'např. 10.51.8.5'}),
                'hostname': forms.TextInput(attrs={'placeholder': 'např. muj-pracovni-pc'}),

        }
        labels = {
                'ip_address': 'Adresa IP',
                'hostname': 'Název PC'
        }
```

Obr. 37 Formulářová třída *PcForm*

6.6.1 Pole MAC adresy

Výjimkou je ve formuláři pole pro zadání MAC adresy, kdy formulář nepoužívá toto pole přímo z modelu Workstation. Je vytvořeno nové vlastní pole pro zadání MAC adresy s vlastní validací. Cílem je umožnit uživateli zadat MAC adresu ve formátu i včetně oddělovačů, tak jak by uživatel pravděpodobně očekával (může ho však zadat i bez oddělovačů). K tomuto účelu nelze použít pole přímo z modelu, protože validace pole z modelu kontroluje před uložením do databáze, zda je MAC adresa ve formátu

bez oddělovačů. Vlastní validace pole je vytvořena funkcí *clean_mac_address* (viz. Obr. 38), která provádí validaci zadané MAC adresy pomocí regulárního výrazu definovaného v proměnné *mac_addr_regex* ve funkci *validate_mac_address* (viz. Obr. 39). Po úspěšné validaci je adresa dále upravena tak, že jsou odstraněny oddělovače, aby adresa splňovala podmínky pro správnou validaci v modelu před uložením do databáze.

```
def clean_mac_address(self):
    mac_address = self.cleaned_data['mac_address']
    validate_mac_address(mac_address)
    mac_address = re.sub("[\s:-]", '', mac_address.upper())
    if len(mac_address) != 12:
        raise forms.ValidationError('Nevalidní formát MAC adresy!')
    return mac_address
```

Obr. 38 Formulářová funkce pro validaci MAC adresy

```
def validate_mac_address(value):
    mac_addr_regex = '^([A-F0-9]{2}(([:][A-F0-9]{2}){5}|([-][A-F0-9]{2}){5}))' \
                     '|([a-f0-9]{2}(([:][a-f0-9]{2}){5}|([-][a-f0-9]{2}){5}))' \
                     '$|^([A-F0-9]{12})$|^([a-f0-9]{12})$'
    if not re.match(mac_addr_regex, value):
        raise ValidationError('Nevalidní formát MAC adresy!')
```

Obr. 39 Funkce pro validaci MAC adresy

6.7 Funkce

Někdy jsou funkce v Django nazývány jako pohledy (anglicky „views“) a jsou klíčovou součástí vývoje aplikace. Definují, jak aplikace reaguje na uživatelské požadavky (requests – URL adresy) a jak generuje odpovědi (responses) a jsou tedy zodpovědné za veškerou funkcionalitu a chod aplikace (viz. kapitola 6.1.2.3). Funkce provede své specifické příkazy (vytvoření formuláře, načtení dat z databáze atd.) a pokud nedojde k chybě, vygeneruje na základě těchto informací šablonu, která uživateli zobrazí výsledek v prohlížeči. V této podkapitole jsou podrobněji popsány pouze nejdůležitější funkce aplikace pro možnost využití technologie WoL v aplikaci. To zahrnuje funkce pro

registraci, probuzení a kontrolu stavu počítače. Ostatní funkce jako zobrazení přehledu počítačů, odstranění počítače atd. jsou popsány stručně v bodech.

6.7.1 Registrace počítače

Pohled pro registraci využívá dekorátor *@csrf_protect*, který zajišťuje ochranu před útoky typu Cross-Site Request Forgery (CSRF). Pohled tedy automaticky ověřuje, zda požadavek obsahuje platný CSRF token. Pokud byl proveden požadavek pro zpracování formuláře metodou POST, je vytvořena instance formuláře třídy *PcForm* (viz kapitola 9.4), přičemž v konstruktoru jsou předávána data formuláře z požadavku typu POST. Důležitý je zde příkaz *form.is_valid*, který slouží k ověření, zda jsou data získaná z formuláře platná a splňují validační pravidla definovaná v příslušném formuláři (v tomto případě ve třídě *PcForm*) [56]. Pokud formulář obsahuje validní data, získané hodnoty (IP a MAC adresa, název počítače) jsou pomocí *form.cleaned_data* uloženy do nových proměnných. Tyto hodnoty jsou následně uloženy do nové instance třídy *Workstation* a příkazem *save* se funkce pokusí uložit instanci třídy do databáze. Pokud je uložení úspěšné, dojde k přesměrování na hlavní stránku s přehledem registrovaných počítačů (pomocí HttpResponseRedirect). V případě že není proveden požadavek metodou POST, je vytvořen prázdný formulář a vygenerován pomocí příslušné šablony do prohlížeče (viz. Obr. 40).

```
@csrf_protect
def register(request):
    if request.method == 'POST':
        form = PcForm(request.POST)
        if form.is_valid():
            ip_addr = form.cleaned_data['ip_address']
            hostname = form.cleaned_data['hostname']
            mac_addr = form.cleaned_data['mac_address']
            workstation = Workstation(mac_address=mac_addr,
                                      ip_address=ip_addr,
                                      hostname=hostname,
                                      user_id="hash_user")
            try:
                workstation.save()
                return HttpResponseRedirect(reverse('wolapp:index'))
            except IntegrityError:
                form.add_error(None, 'Došlo k chybě při registraci PC!')
    else:
        form = PcForm()
    return render(request, 'wolapp/pc_form.html', {'form': form, 'edit_mode': False})
```

Obr. 40 Funkce pro registraci počítače

6.7.2 Wake on LAN a kontrola stavu počítače

Skript pro odesílání magických paketů (viz. kapitola 8) byl implementován do pohledu *wake_pc*. Ten přebírá argument *workstation_id*, což je identifikátor počítače, kterému má být magický paket odeslán. Samotný skript pro odeslání magického paketu však přebírá jako argumenty do funkce IP a MAC adresu počítače. Pomocí API funkce *get_object_or_404* pro komunikaci z databází je získán objekt počítače na základě zadaných kritérií (název modelu – *Workstation*, identifikátor počítače – *pk*). [57] Poté je již zavolán samotný skript *wake_on_lan* s parametry *workstation.ip_address* (IP adresa cílového počítače) a *workstation.mac_address* (MAC adresa cílového počítače) pro odeslání magického paketu cílovému počítači. Pokud skript vrátí hodnotu *True*, je dále kontrolován stav počítače funkcí *get_pc_status* zasláním zprávy (Echo Request) pomocí funkce *ping* z modulu *pythonping* (viz. Obr. 41). [58] Funkce *ping* přebírá IP adresu cílového počítače jako povinný parametr. Libovolný parametr *timeout* pro přijetí odpovědi byl nastaven na 7 vteřin, parametr *count* pro počet requestů na 4. Maximální doba pro přijetí odpovědi od cílového počítače je tedy 28 vteřin. To je z důvodu, že po přijetí magického paketu může počítači delší dobu trvat, než se zapne a bude schopen po síti odeslat odpověď na *ping* požadavek.

```
def get_pc_status(ip_address, timeout):
    response = ping(ip_address, timeout=timeout, count=4)
    if response.success():
        return 1
    else:
        return 0
```

Obr. 41 Funkce pro kontrolu stavu počítače

Aby však uživatel nemusel potenciálně čekat až 28 vteřin a dále mohl interagovat s aplikací, je kontrola stavu počítače prováděna v nově vytvořeném vlákně pomocí modulu *threading* ve funkci *check_selected_background*. Proces v novém vlákně je spuštěn metodou *start* (viz. Obr. 42).

```
workstation = get_object_or_404(Workstation, pk=workstation_id)
if wake_on_lan(workstation.ip_address, workstation.mac_address):
    thread = threading.Thread(target=check_selected_background,
                              args=(pc_ids, session, results, 7))
    thread.start()
```

Obr. 42 Vytvoření nového vlákna pro kontrolu stavu počítače

6.7.3 Ostatní funkce

- *index* – výpis přehledu všech registrovaných počítačů uživatele, používá API funkci *Workstation.objects.all().filter(user_id=userId)* pro získání počítačů na základě identifikátoru uživatele
- *edit* – úprava informací o registrovaném počítači a uložení změn do databáze
- *check_pc* – pouze kontrola stavu počítače bez zaslání magického paketu
- *delete_pc* – odstranění počítače z aplikace (databáze) pomocí funkce *delete* na konkrétní instanci počítače vrácenou z databáze na základě identifikátoru počítače
- *check_selected* – kontrola stavu vícero zvolených počítačů
- *wake_selected* – odeslání magických paketů zvoleným počítačům
- *delete_selected* – odstranění zvolených počítačů pomocí funkce *delete*

6.8 Šablony

Pro zpracování a zobrazení výsledků z výše uvedených funkcí v prohlížeči byl využit systém šablon, tagů a filtrů v Djangu. Byla vytvořena základní html šablona master.html, která obsahuje základní rozvržení webové stránky. V ostatních html šablonách byl následně použit Django tag *extends* (viz. Obr. 43), což znamená že tyto šablony dědí ze základní šablony master. Důležité jsou i značky *{% block content %}* a *{% endblock %}* v základní šabloně master i v ostatních šablonách. V základní šabloně tyto

tagy znamenají, že zde bude vložen obsah zděděné šablony. Ve zděděné šabloně pak tento blok označuje obsah HTML kódu, který bude do základní šablony vložen. [59]

```
{% extends "wolapp/master.html" %}

{% block content %}
    <div class="container mt-3">
        {% if user_workstation %}
            <div class="row my-3">
```

Obr. 43 Použití tagu *extends*

Pomocí filtrů (dvojité závorky *{{}}*) byly pak zobrazovány údaje o počítačích v šabloně index.html s použitím filtru *escape* k bezpečnému zobrazení textu v šabloně, kdy filtr zabraňuje vkládání nebezpečného HTML kódu (viz. Obr. 44).

```
<h4>{{ workstation.hostname | escape}}</h4>
<h5>{{ workstation.mac_address | escape}}</h5>
<h5>{{ workstation.ip_address | escape}}</h5>
```

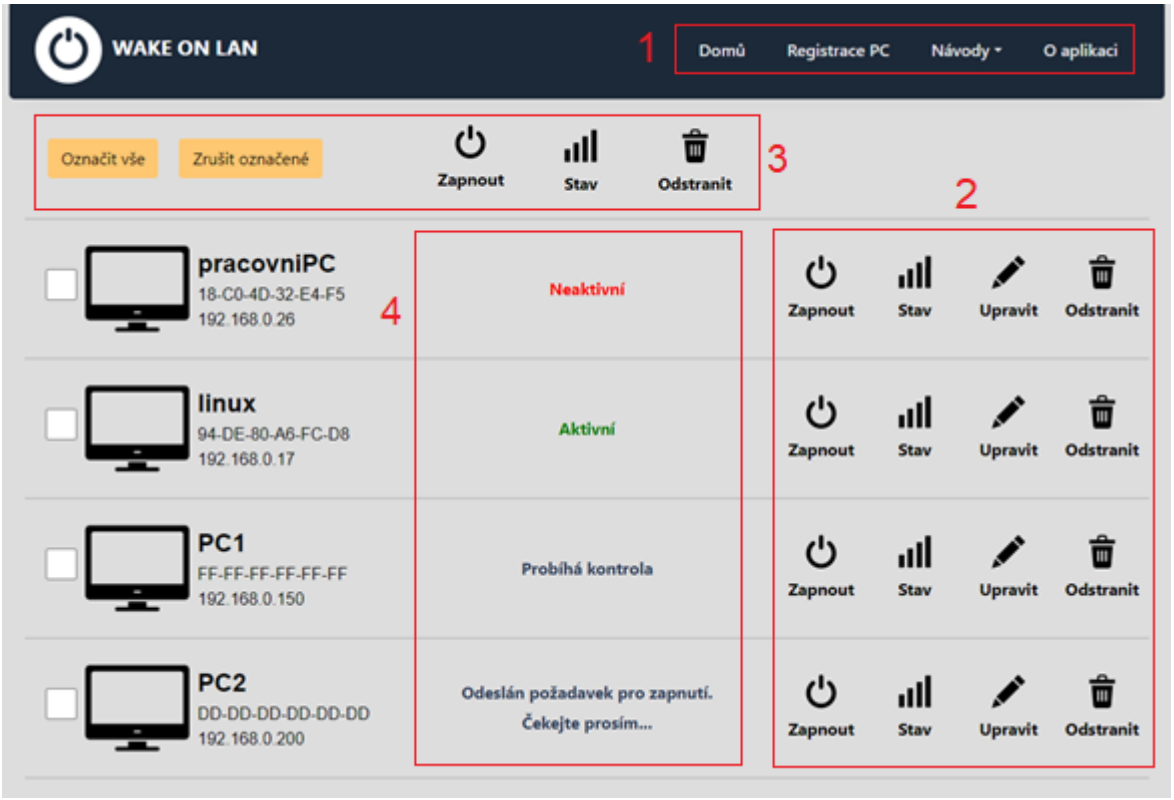
Obr. 44 Výpis údajů do šablony pomocí filtrů

6.9 Popis a vzhled aplikace

6.9.1 Popis

Rozhraní je vytvořeno jako webová aplikace ve frameworku Django na virtuálním serveru ve vnitřní síti podniku. V horní části aplikace se nachází navigace s odkazy na domovskou stránku s přehledem registrovaných počítačů uživatele – Domů, registraci počítače – Registrace PC, návody pro konfiguraci WoL na cílovém počítači – Návody a obecné informace o aplikaci – O aplikaci (viz. Obr. 45–1). Na domovské stránce je zobrazen přehled všech registrovaných počítačů, kdy u každého zvlášť je možné odeslat magický paket – Zapnout, zkontrolovat stav počítače v síti – Stav, upravit údaje o počítači – Upravit, odstranit počítač – Odstranit (viz. Obr. 45–2). Pomocí oranžových tlačítek – Označit vše a Zrušit označené a zaškrtávacích polí je možné zvolit vícero požadovaných počítačů a pomocí akčních tlačítek v horní části (Zapnout, Stav, Odstranit) provést

požadovanou akci (viz. Obr. 45-3). Ve střední části je poté zobrazen aktuální stav počítače v síti – Aktivní, Neaktivní, Kontrola stavu, a Odeslán požadavek pro probuzení (viz. Obr. 45-4).



Obr. 45 Grafické rozhraní – Domovská stránka

6.9.2 Vzhled aplikace

U vzhledu aplikace byla snaha především o jednoduchost a dostupnost s cílem alespoň částečně manipulovat s aplikací na širší škále zařízení s různými rozlišeními.

6.9.3 Boostrap

V aplikaci je pro realizaci uživatelského prostředí využitý volně dostupný framework Bootstrap 5, který využívá standartní programovací jazyky pro vytváření webových stránek (HTML, CSS, JavaScript) a obsahuje často používané grafické komponenty. [60] Důvodem výběru je jeho rychlá a pohodlná tvorba responzivních webových rozhraní. V aplikaci je využíván především systém škálovatelné mřížky, která je realizována pomocí sloupců zodpovědné za hlavní rozložení objektů na stránce (viz. Obr. 45). Tento systém je v aplikaci realizován za pomocí několika tříd [61]:

- `.container`: Kontejner obalující obsah stránky. Přizpůsobuje se šířce obrazovky a zajišťuje správné zarovnání sloupců.
- `.row`: Vytváří řádek v rámci kontejneru. Řádek může obsahovat několik sloupců.
- `.col-*`: Slouží k definici sloupce v rámci řádku. Znak * reprezentuje číselnou hodnotu od 1 do 12, která určuje šířku sloupce. Například. `col-6` vytvoří sloupec o šířce 6 sloupců (polovina šířky řádku).

V rámci vzhledu je také využito několik tříd, které ovlivňují zarovnání a uspořádání prvků v rámci mřížkového systému (např. `justify-content-start`, `justify-content-center` atd.) (viz. Obr. 46).

```
<div class="container mt-3">
    <div class="row my-3">
        <div class="col-12 col-sm-12 col-md-4 col-lg-4 d-flex
            justify-content-lg-start justify-content-center align-items-center">
        </div>
```

Obr. 46 Rozložení sloupců pro zařízení s různým rozlišením

K vytvoření vzhledu aplikace byly však využity i jiné komponenty, které framework nabízí: nadpisy, tlačítka, upozornění, modální okna. Po tvorbě responzivity byl vzhled aplikace otestován na mobilním zařízení (viz. Obr. 47).

Obr. 47 Vzhled aplikace na mobilu – iPhone 12 Pro

7 ZABEZPEČENÍ NAVRŽENÉHO ŘEŠENÍ

Zabezpečení je jedním z klíčových faktorů při návrhu a provozu systémů i webových aplikací. Bezpečnostní opatření jsou nezbytná k ochraně citlivých dat, udržení integrity systému a minimalizaci rizika útoků. Tato kapitola se zaměří na hlavní oblasti zabezpečení v rámci navrženého řešení

7.1 Virtuální stroj

Virtuální stroj poskytuje izolaci a oddělení mezi hostitelským systémem a virtuálním prostředím, kde bude server poskytující aplikaci provozován, čímž se minimalizuje riziko nežádoucího ovlivnění ostatních systémů a aplikací (viz. kapitola 3.1). Je také nastaveno jedinečné heslo pro přístup k virtuálnímu stroji. Samotná aplikace je navíc spouštěna s právy běžného uživatele bez práv administrátora (root). Na virtuálním stroji však není dostatečné nastavení firewallu v podobě omezení přístupu pouze na potřebné porty, ochrany před síťovými útoky nebo blokování nežádoucích IP adres.

7.2 Omezení přístupu – VPN služba

Navržená aplikace je dostupná pouze z vnitřní sítě podniku. K přístupu do vnitřní sítě je využívána VPN služba, který zajišťuje zabezpečené a šifrované spojení mezi zařízením a cílovým virtuálním serverem uvnitř sítě pro soukromý a bezpečný přenos dat. [62] VPN služba poskytuje také autentizaci uživatele, takže pouze oprávněné osoby mají přístup k aplikaci.

7.3 Zabezpečení webové aplikace

Zabezpečení aplikace bylo provedeno na základě OWASP Top Ten 2021, což je seznam deseti nejčastěji vyskytujících se bezpečnostních chyb ve webových aplikacích pro rok 2021. [63] Tyto chyby jsou identifikovány a sledovány komunitou Open Web Application Security Project (OWASP) a slouží jako doporučení pro vývojáře při zabezpečování aplikací. Je však třeba zdůraznit, že aplikace byla ošetřena pouze proti nejčastějším chybám ze seznamu zranitelností. Případné zabezpečení aplikace proti většímu množství (ideálně proti všem ze seznamu OWASP Top Ten) může být předmětem budoucího vylepšení aplikace.

7.3.1 SQL Injection

Jednou z nejčastějších bezpečnostních chyb je možnost injekce SQL kódu, kdy útočník může vložit nebezpečný SQL dotaz na databázi do vstupních polí a tím provést akci nad daty uloženými v databázi, což může mít za následek např. vymazání či únik dat z databáze. [64] Pro zabezpečení proti injekcím je nutné používat parametrizované dotazy databáze (prepared statements) nebo ORM (Object-Relational Mapping – viz. kapitola 6.1.2.2), který framework Django poskytuje a je používán v aplikaci pro manipulaci s databázovými daty. Byla také provedena validace a filtrace uživatelských vstupů v registračním formuláři (viz. kapitola 6.7) pro minimalizaci rizik injekčních útoků.

7.3.2 Broken Access Control

Zranitelnost řízeného přístupu (Broken Access Control) se vyskytuje v případě, že aplikace nedodržuje správné mechanismy a omezení přístupu, což umožňuje neoprávněným uživatelům získat přístup k citlivým informacím, provádět neoprávněné operace nebo získat vyšší úrovně oprávnění. [65] Je tedy nutné správně nastavit a otestovat oprávnění a kontrolu přístupu k funkcím a datům v aplikaci. Je třeba zajistit, že uživatelé mohou manipulovat pouze se svými registrovanými počítači, a že jsou zabezpečeny nepřímé přístupy k cizím počítačům, například prostřednictvím manipulace s URL adresami.

7.3.2.1 Shibboleth

Shibboleth je autentizační systém využíván v podnikové síti pro řízení přístupu k chráněným on-line zdrojům. [66] Implementace tohoto systému je pro správné fungování aplikace a pro ochranu proti Broken Access Control nezbytná, aby vícero uživatelů mohlo odděleně přistupovat k aplikaci a manipulovat pouze se svými registrovanými počítači v rámci oddělené relace (session). Principem je získat po úspěšné autentizaci údaje o přihlášeném uživateli, které následně budou využity i pro jeho autorizaci při manipulaci s daty v aplikaci. V aplikaci dosud není Shibboleth implementován. Implementaci je potřeba provést ve spolupráci s IT správci podnikové sítě.

7.3.3 Sensitive Data Exposure

Aby se zabránilo úniku citlivých dat (Sensitive Data Exposure), je důležité správně šifrovat data, jako jsou hesla a osobní údaje. Je také důležité chránit přenos dat pomocí bezpečného protokolu HTTPS a vyhnout se ukládání citlivých dat veřejně přístupným způsobem. Vytvořená aplikace však neukládá žádné citlivé údaje v podobě hesel atd. Do databáze jsou

ukládány pouze údaje běžně dostupné uživateli z vnitřní sítě podniku prostřednictvím informačních systémů nebo skenováním sítě.

7.3.4 Cross-Site Scripting (XSS)

Cross-Site Scripting (XSS) je zranitelnost, která umožňuje útočníkovi vložit nebezpečný skript (často JavaScript) do webové stránky a spustit ho ve webovém prohlížeči uživatele. [67] To může vést k úniku citlivých informací a manipulaci s webovým obsahem. Django nabízí několik mechanismů pro obranu proti XSS, včetně automatického escapování proměnných v HTML šablonách pomocí filtrů (viz. kapitola 6.9) jako nebo výchozího escapování v kontextu, který je šabloně předán. Důležitá je také validace a filtrace uživatelského vstupu v registračním formuláři.

7.3.5 Cross-Site Request Forgery (CSRF)

Cross-Site Request Forgery (CSRF) je útok, při kterém útočník využívá důvěry mezi webovou aplikací a uživatelem. Útočník přesvědčí uživatele, aby provedl nežádoucí akci v rámci aplikace, například kliknutím na odkaz nebo odesláním formuláře, které obsahují žádosti o provedení nežádoucí akce v aplikaci. [68] Django obsahuje zabudovanou ochranu proti CSRF útokům pomocí tzv. CSRF tokenů. Tyto tokeny jsou generovány a začleněny do formuláře v aplikaci vložením *{% csrf_token %}* do formulářové šablony, čímž je zajištěno, že při odesílání formuláře je CSRF token přítomen a správně ověřen, aby se zabezpečilo, že žádost byla odeslána z důvěryhodného zdroje (viz. Obr. 48).

```
<form id="myForm" method="POST">
{% csrf_token %}
```

Obr. 48 Použití CSRF tokenu v registračním formuláři

ZÁVĚR

Bakalářská práce se zaměřila na teoretický rozbor technologie Wake on LAN (WoL) a následně na praktickou implementaci v podnikovém síti.

V rámci teoretické části byla podrobněji popsána technologie WoL včetně její historie, principu fungování, hardwarových požadavků, nutné konfigurace a limitací při snaze její implementace v podnikové síti. Bylo zjištěno, že konfigurace počítače pro správné fungování technologie WoL napříč různými typy hardwaru v kombinaci s konkrétními typy a verzemi operačních systému a síťových ovladačů není zcela jednoznačná. V rámci konfigurace mohou být odlišnosti například v názvech funkcí, které je nutné aktivovat, popřípadě deaktivovat pro správné fungování WoL. Byly také probrány potenciální problémy při přenosu magického paketu pro buzení počítačů v rámci stejné sítě, napříč podsítěmi i přes Internet.

V praktické části přinesla analýza pohled na momentální stav infrastruktury podnikové sítě. Byla identifikována síťová architektura včetně fyzické i logické topologie a plánu IP adres. Pro implementaci technologie Wol byl v podnikové síti nainstalován a nakonfigurován virtuální stroj, který plní účel webového serveru pro poskytování webové aplikace. Dále byl v podnikové síti připraven a nakonfigurován testovací klientský počítač, který simuloval příjem magického paketu.

Na virtuálním serveru byl nejprve vytvořen samostatný skript pro odesílání magických paketů pro buzení počítačů, který byl následně otestován odesláním magického paketu ze serveru na testovací počítač. Bylo otestováno, že magický paket úspěšně prošel sítí od zdroje k cíli. Po testování skriptu následovalo vytvoření jednoduché webové aplikace s grafickým rozhraním. Tato aplikace byla vytvořena pomocí aplikačního frameworku Django napsaného v programovacím jazyce Python s využitím SQLite databáze. Aplikace umožňuje prostřednictvím uživatelského rozhraní registrovat počítač s následnou možností odeslat magický paket na cílový počítač v síti prostřednictvím výše zmíněného skriptu, který je do aplikace implementován, a tím počítač zapnout. Aplikace je však stále v testovacím prostředí. Chybí implementace autentizačního systému Shibboleth pro oddělení vícero uživatelských relací a pro autorizaci operací v aplikaci. Implementaci systému je nutno provést po konzultaci a ve spolupráci s IT správci sítě.

Závěrečná kapitola práce byla věnována zabezpečení navrženého řešení. Byly diskutovány možné bezpečnostní hrozby a provedená opatření pro minimalizaci rizik. Důraz byl kladen především na zabezpečení navržené webové aplikace.

V rámci budoucího vývoje, který se týká nasazení webové aplikace do produkčního prostředí podnikové sítě, je nutné implementovat autentizační systém Shibboleth a zvážit další robustnější bezpečnostní opatření jak virtuálního serveru, tak samotné webové aplikace. Dále je možné zlepšit uživatelské rozhraní aplikace a implementovat pokročilejší funkce pro správu a monitorování buzení počítačů. To může v rámci aplikace zahrnovat například možnost vytvoření úlohy, která automaticky zapne počítač uživatele v konkrétní požadovaný den a čas.

Celkově lze konstatovat, že implementace WoL v podnikové lokální síti v rámci testovacího prostředí byla úspěšně provedena a vytvořené řešení umožňuje centralizované buzení počítačů v síti prostřednictvím grafického rozhraní. Práce přinesla výhody pro podnikové prostředí, kde je důležité minimalizovat spotřebu energie a zajišťovat efektivní správu počítačů. Implementace přináší výhody i pro IT správce sítě. V mnoha podnicích je běžně využíván systém správy koncových stanic, jako je System Center Configuration Manager (SCCM), který umožňuje centralizované řízení buzení počítačů. S implementací webové aplikace s grafickým rozhraním mají sami uživatelé možnost probudit svůj počítač, bez nutnosti žádat o tuto akci přímo IT správce, což uživatelům dává větší flexibilitu a nezávislost. IT správci se naopak nemusí o buzení počítačů v případě potřeby uživatelů starat. Uživatelé s dostatečnými znalostmi a oprávněním mohou dokonce sami nakonfigurovat svůj počítač v síti pro správné fungování technologie WoL. Webová aplikace poskytuje podrobné návody, jak tuto konfiguraci provést. V případě, že sám uživatel není schopen tuto konfiguraci provést, stále existuje možnost, že IT správci provedou potřebné nastavení pomocí SCCM. Tímto tedy lze alespoň částečně snížit zátěž IT oddělení a zároveň zvýšit produktivitu uživatelů. Používání funkce WoL navíc snižuje celkovou spotřebu energie v podnicích a zmírňuje opotřebení hardwaru, protože odpadá nutnost permanentně zapnutého počítače pro možnost kdykoliv počítač vzdáleně spravovat.

SEZNAM POUŽITÉ LITERATURY

[1] Understanding Wake On LAN. Forums.ivanti [online]. ivanti Community, c2019-2022 [cit. 2023-05-23]. Dostupné z: https://forums.ivanti.com/s/article/Understanding-Wake-On-LAN?language=en_US

[2] O'MALLEY, Tim. IBM Announces Universal Management. Web.archive.org [online]. New York: IBM, 1998, 15. dubna 1998 [cit. 2023-05-23]. Dostupné z: https://web.archive.org/web/20121012155338/http://www-03.ibm.com/press/us/en/pressrelease/2705.wss

[3] DUINO, Justin. What Is Wake-on-LAN, and How Do I Enable It?. How-To Geek [online]. How-To Geek, 2021, 18. listopadu 2021 [cit. 2023-05-23]. Dostupné z: https://www.howtogeek.com/70374/how-to-geek-explains-what-is-wake-on-lan-and-how-do-i-enable-it/

[4] ESSICK, Kristi. Intel, IBM strike deal to lower PC ownership costs. Web.archive.org [online]. COMPUTERWORLD, 1996, 31. října 1996 [cit. 2023-05-23]. Dostupné z: https://web.archive.org/web/20151208132411/http://www.computerworld.co.nz/article/519210/intel_ibm_strike_deal_lower_pc_ownership_costs/

[5] Magic Packet Technology [online]. AMD, c1998 [cit. 2023-05-24]. Dostupné z: https://web.archive.org/web/20041209061029/http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/20213.pdf

[6] YASAR, Kinza. MAC address (media access control address). Techtarget [online]. TechTarget [cit. 2023-05-23]. Dostupné z: https://www.techtarget.com/searchnetworking/definition/MAC-address

[7] BOUŠKA, Petr. Wake on LAN – lokální i vzdálený subnet. Samuraj-cz [online]. Samuraj, 2008, 10. srpna 2008 [cit. 2023-05-23]. Dostupné z: https://www.samuraj-cz.com/clanek/wake-on-lan-lokalni-i-vzdaleny-subnet/

[8] LIEBERMAN, Phillip. Wake on LAN Technology. Web.archive.org [online]. LiebermanSoftware, 2002, 11. července 2002 [cit. 2023-05-24]. Dostupné z: https://web.archive.org/web/20061019012133/http://www.liebsoft.com/index.cfm/whitepapers/Wake_On_LAN

[9] WakeOnLAN. Wiki.wireshark.org [online]. Wireshark, 2008, 10. srpna 2008 [cit. 2023-05-23]. Dostupné z: https://wiki.wireshark.org/WakeOnLAN

[10] WakeOnLAN. Manpages.debian.org [online]. Debian, 2003, 31. října 2003 [cit. 2023-05-23]. Dostupné z: https://manpages.debian.org/bullseye/etherwake/etherwake.8.en.html

[11] Unicast Flooding in Switched Campus Networks. Cisco.com [online]. Cisco, 2016, 8. únor 2016 [cit. 2023-05-23]. Dostupné z: https://www.cisco.com/c/en/us/support/docs/switches/catalyst-6000-series-switches/23563-143.html

[12] Blocking Unknown Unicast Flooding. Packetlife.net [online]. PacketLife [cit. 2023-05-23]. Dostupné z: https://packetlife.net/blog/2010/jun/4/blocking-unknown-unicast-flooding/

[13] Wake-On-LAN Tools to Remotely Power Up PCs. Geekflare.com [online]. GeekFlare [cit. 2023-05-23]. Dostupné z: https://geekflare.com/wake-on-lan-tools/

[14] Wakeonlan. Manpages.debian.org [online]. debian, 1. ledna 2021 [cit. 2023-05-23]. Dostupné z: https://manpages.debian.org/bullseye/wakeonlan/wakeonlan.1.en.html

[15] Wakeonlan 3.0.0. Pypi.org [online]. debian, 2022, 5. prosince 2022 [cit. 2023-05-23]. Dostupné z: https://pypi.org/project/wakeonlan/

[16] PETERSON, Larry a Bruce DAVIE. Computer Networks: a systems approach. 5th ed. Morgan Kaufmann, 2011. ISBN 9780123850591.

[17] Packet forwarding. En.wikipedia.org [online]. Wikipedia [cit. 2023-05-23]. Dostupné z: https://en.wikipedia.org/wiki/Packet_forwarding#cite_note-1

[18] , Jason. 1. Subnet-Directed Broadcast. Home.memftw.com [online]. FTW, 24. října 2019 [cit. 2023-05-23]. Dostupné z: https://home.memftw.com/four-types-of-wake-on-lan/

[19] LEWIS, Nick a Jason FITZPATRICK. How to Port Forward on Your Router. Howtogeek [online]. FTW, 1. února 2023 [cit. 2023-05-23]. Dostupné z: https://www.howtogeek.com/66214/how-to-forward-ports-on-your-router/

[20] Prerequisites of Wake-on-LAN. Emcosoftware.com [online]. emco software [cit. 2023-05-23]. Dostupné z: https://emcosoftware.com/remote-shutdown/doc/prerequisites-of-wake-on-lan¨

[21] Using Wake-On-LAN WOL/PME to power up your computer remotely. Web.archive.org [online]. [cit. 2023-05-23]. Dostupné z: https://web.archive.org/web/20070308143030/http://xlife.zuavra.net/index.php/60/

[22] HUCULAK, Mauro. How to enable and use Wake on LAN (WoL) on Windows. Windowscentral [online]. Windows Central, 2022, 2. července 2022 [cit. 2023-05-23]. Dostupné z: https://www.windowscentral.com/how-enable-and-use-wake-lan-wol-windows-10

[23] Does it support Wake on LAN (WOL)?. Gigabyte.com [online]. GIGABYTE [cit. 2023-05-23]. Dostupné z: https://www.gigabyte.com/uk/Support/FAQ/2676

[24] Wake on LAN (WOL) behavior in Windows. Learn.microsoft.com [online]. Microsoft, 2023, 29. dubna 2023 [cit. 2023-05-23]. Dostupné z: https://learn.microsoft.com/en-us/troubleshoot/windows-client/deployment/wake-on-lan-feature

[25] Wake-on-LAN: Enable WoL on the network adapter. Wiki.archlinux.org [online]. archlinux [cit. 2023-05-23]. Dostupné z: https://wiki.archlinux.org/title/Wake-on-LAN

[26] LUBERUS, Patrick a Alfandika NYANDORO. Implementing Wake-on-LAN in Institutional Networks. Na-businesspress.com [online]. 2014 [cit. 2023-05-25]. Dostupné z: http://www.na-businesspress.com/JABE/NyandoroA_Web16_1_.pdf

[27] SCHEIBLE, Andrew. Global Information Assurance Certification Paper: Network Security for Wake-On-LAN Technology. Giac.org [online]. GIAC Certifications, c2000-2002 [cit. 2023-05-25]. Dostupné z: https://www.giac.org/paper/gsec/1838/network-security-wake-on-lan-technology/103243

[28] 5 Benefits of Virtualization. Ibm.com [online]. IMB, 2021, 8. dubna 2021 [cit. 2023-05-23]. Dostupné z: https://www.ibm.com/cloud/blog/5-benefits-of-virtualization

[29] What Is Linux?. Linux.com [online]. Linux.com, c2023 [cit. 2023-05-23]. Dostupné z: https://www.linux.com/what-is-linux/

[30] Linux Vs Windows Difference: Which Is The Best Operating System?. Softwaretestinghelp.com [online]. Software Testing Help, c2023 [cit. 2023-05-23]. Dostupné z: https://www.softwaretestinghelp.com/linux-vs-windows/

[31] Síťová instalace z minimálního CD. Debian.org [online]. Debian, c1997-2023 [cit. 2023-05-23]. Dostupné z: https://www.debian.org/CD/netinst/#netinst-stable

[32] VLAN Trunking in Hyper-V for VCL. Helpdesk.tosibox.com [online]. TOSIBOX, 2022, 14. února 2022 [cit. 2023-05-23]. Dostupné z: https://helpdesk.tosibox.com/support/solutions/articles/2100051378-vlan-trunking-in-hyper-v-for-vcl

[33] AGOSTON, Zsolt. One Interface, multiple VLAN IPs on Linux. Opentechtips.com [online]. opentechtips, 2020, 29. března 2020 [cit. 2023-05-23]. Dostupné z: https://opentechtips.com/one-interface-multiple-vlan-ips-on-linux/?utm_content=cmp-true

[34] NetworkInterfaceNames. Wiki.debian.org [online]. Debian [cit. 2023-05-25]. Dostupné z: https://wiki.debian.org/NetworkInterfaceNames

[35] Download Ubuntu Desktop. Ubuntu.com [online]. ubuntu, c2023, 29. března 2020 [cit. 2023-05-23]. Dostupné z: https://ubuntu.com/download/desktop

[36] B85M-E/CSM. Asus.com [online]. c2014 [cit. 2023-05-25]. Dostupné z: https://dlcdnets.asus.com/pub/ASUS/mb/LGA1150/B85M-E/E9915_B85M-E_Manual_v3_web.pdf

[37] TIOBE Index for May 2023. Tiobe.com [online]. TIOBE, 2023 [cit. 2023-05-23]. Dostupné z: https://www.tiobe.com/tiobe-index/

[38] CHOU, Eric. Mastering Python Networking: Your one stop solution to using Python for network automation, DevOps, and SDN. 2017. ISBN 978-1784397005.

[39] Socket — Low-level networking interface. Docs.python.org [online]. Python, c2001-2023 [cit. 2023-05-23]. Dostupné z: https://docs.python.org/3/library/socket.html

[40] Netifaces 0.11.0. Pypi.org [online]. pypi.org, c2023 [cit. 2023-05-23]. Dostupné z: https://pypi.org/project/netifaces/

[41] Binascii — Convert between binary and ASCII. Docs.python.org [online]. Python, 2023 [cit. 2023-05-23]. Dostupné z: https://docs.python.org/3/library/binascii.html

[42] Ipaddress — IPv4/IPv6 manipulation library. Docs.python.org [online]. Python, 2023 [cit. 2023-05-23]. Dostupné z: https://docs.python.org/3/library/ipaddress.html

[43] MCKAY, Dave. How to Use the chmod Command on Linux. Howtogeek.com [online]. How-To Geek, 2022, 21. října 2022 [cit. 2023-05-25]. Dostupné z: https://www.howtogeek.com/437958/how-to-use-the-chmod-command-on-linux/

[44] 11 Advantages of Django: Why You Should Use It. Pythonistaplanet.com [online]. Pythonista Planet, c2023 [cit. 2023-05-23]. Dostupné z: https://pythonistaplanet.com/advantages-of-django/

[45] BOYARKO, Elizabeth. TOP 10 WEB DEVELOPMENT FRAMEWORKS: BEST OPTIONS IN 2023. Upsilonit.com [online]. Upsilon, 2022, 12. května 2022 [cit. 2023-05-23]. Dostupné z: https://www.upsilonit.com/blog/top-10-web-development-frameworks-best-options-to-choose

[46] Understanding the MVC pattern in Django. Medium.com [online]. 2017, 19. července 2017 [cit. 2023-05-25]. Dostupné z: https://medium.com/shecodeafrica/understanding-the-mvc-pattern-in-django-edda05b9f43f

[47] Understanding the MVC pattern in Django. Docs.djangoproject.com [online]. Django Software Foundation, c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/db/models/

[48] AWATI, Rahul. Object-relational mapping (ORM). Theserverside.com [online]. TechTarget, c2002-2023 [cit. 2023-05-23]. Dostupné z: https://www.theserverside.com/definition/object-relational-mapping-ORM

[49] Writing views. Docs.djangoproject.com [online]. Django, c2005-2023 [cit. 2023-05-23]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/http/views/

[50] Templates. Docs.djangoproject.com [online]. Django, c2005-2023 [cit. 2023-05-23]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/templates/

[51] URL dispatcher. Docs.djangoproject.com [online]. Django, c2005-2023 [cit. 2023-05-23]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/http/urls/

[52] About SQLite. Sqlite.org [online]. SQLite [cit. 2023-05-23]. Dostupné z: https://www.sqlite.org/about.html

[53] Appropriate Uses For SQLite. Sqlite.org [online]. SQLite [cit. 2023-05-23]. Dostupné z: https://www.sqlite.org/whentouse.html

[54] Migrations. Docs.djangoproject.com [online]. Django Software Foundation, c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/migrations/

[55] ModelForm. Docs.djangoproject.com [online]. Django Software Foundation, c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/forms/modelforms/

[56] Form and field validation. Docs.djangoproject.com [online]. Django Software Foundation, c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/ref/forms/validation/

[57] Django shortcut functions. Docs.djangoproject.com [online]. Django Software Foundation, c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/topics/http/shortcuts/

[58] Pythonping 1.1.4. Pypi.org [online]. c2023 [cit. 2023-05-25]. Dostupné z: https://pypi.org/project/pythonping/

[59] The Django template language. Docs.djangoproject.com [online]. c2005-2023 [cit. 2023-05-25]. Dostupné z: https://docs.djangoproject.com/en/4.2/ref/templates/language/

[60] Introduction. Getbootstrap.com [online]. Bootstrap, c2023 [cit. 2023-05-23]. Dostupné z: https://getbootstrap.com/docs/5.0/getting-started/introduction/

[61] Grid system. Getbootstrap.com [online]. Bootstrap, c2023 [cit. 2023-05-23]. Dostupné z: https://getbootstrap.com/docs/5.0/layout/grid/

[62] What Is a VPN? - Virtual Private Network. Cisco.com [online]. Cisco, c2023 [cit. 2023-05-23]. Dostupné z: https://www.cisco.com/c/en/us/products/security/vpn-endpoint-security-clients/what-is-vpn.html

[63] OWASP Top Ten. Owasp.org [online]. OWASP, c2023 [cit. 2023-05-23]. Dostupné z: https://owasp.org/www-project-top-ten/

[64] A03:2021 – Injection. Owasp.org [online]. OWASP, c2023 [cit. 2023-05-23]. Dostupné z: https://owasp.org/Top10/A03_2021-Injection/

[65] A01:2021 – Broken Access Control. Owasp.org [online]. OWASP, c2023 [cit. 2023-05-23]. Dostupné z: https://owasp.org/Top10/A01_2021-Broken_Access_Control/

[66] What is Shibboleth?. Shibboleth.net [online]. Shibboleth, c2023 [cit. 2023-05-23]. Dostupné z: https://www.shibboleth.net/about-us/the-shibboleth-project/

[67] The difference between cross-site and server-side request forgery: Cross-Site request forgery. Resources.infosecinstitute.com [online]. INFOSEC, c2023 [cit. 2023-05-23]. Dostupné z: https://resources.infosecinstitute.com/topic/the-difference-between-cross-site-and-server-side-request-forgery/

[68] Cross Site Scripting (XSS). Owasp.org [online]. OWASP, c2023 [cit. 2023-05-23]. Dostupné z: https://owasp.org/www-community/attacks/xss/

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

API Application Programming Interface

ARP Address Resolution Protocol

CSRF Cross-Site Request Forgery

IP Internet Protocol

LAN Local Area Network

MAC Media Access Control

ORM Object–relational mapping

VLAN Virtual Local Area Network

WoL Wake on LAN

XSS Cross-Site Scripting

SEZNAM OBRÁZKŮ

SEZNAM PŘÍLOH