UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

CENTRÁLNA SPRÁVA POČÍTAČOV

2018 **Eduard DVORNÝ**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

CENTRÁLNA SPRÁVA POČÍTAČOV

BAKALÁRSKA PRÁCA

Študijný program:	Aplikovaná informatika
Pracovisko (katedra/ústav):	Ústav informatiky
Vedúci bakalárskej práce:	RNDr. JUDr. Pavol Sokol, PhD.

Košice 2018

Eduard Dvorný

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Eduard Dvorný
Študijný program: Aplikovaná informatika (Jednoodborové štúdium, bakalársky I. st., externá forma)
Študijný odbor: 9.2.9. aplikovaná informatika
Typ záverečnej práce: Bakalárska práca
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Centrálna správa počítačov

Názov EN: Central administration of computers

Cieľ: (1) Analyzovať možnosti a prístupy k centrálnej správe počítačov.
(2) Porovnať a analyzovať aktuálne riešenia k centrálnej správe počítačov.
(3) Navrhnúť a implementovať centrálnu správu v prostredí vysokej školy.

Literatúra:
[1] S. Monk: Raspberry Pi Cookbook, O´Reilly Media, 2014.
[2] S.B. Dissanayake:Tiny Linux Distributions and Their Utility Value: Small is Beautiful!, CreateSpace 2012.
[3] S. Nelson: Pro Data Backup and Recovery, Apress 2011.
[4] B. Desmond: Active Directory: Designing, Deploying, and Running Active Directory, O´Reilly Media, 2013.

Vedúci: RNDr. JUDr. Pavol Sokol, PhD.
Ústav : ÚINF - Ústav informatiky
Riaditeľ ústavu: prof. RNDr. Viliam Geffert, DrSc.

Dátum schválenia: 16.08.2018

Abstrakt v štátnom jazyku

Hlavnou motiváciou práce je zefektívniť spôsob poskytovania služieb v oblasti IT podpory. Jedným spôsobom ako dosiahnuť tento cieľ je použiť centralizáciu a automatizáciu správy pracovných staníc, centralizovať správu a vyriešiť problematiku zálohovania a obnovy údajov. Dôležitou časťou je spraviť prehľad súčasne dostupných riešení, porovnať ich vzhľadom na akademické podmienky a vybrať to najvhodnejšie. V praktickej časti sa práca venuje návrhu a implementácii konkrétneho riešenia, ktoré pristupuje k infraštruktúre ako ku kódu (IaC). Za použitia nástrojov s otvoreným zdrojovým kódom je možné stále vylepšovať funkcionalitu a bezpečnosť IT správy, dynamicky ju meniť podľa požiadaviek klientov a potrieb správcov. Výstupom má byť plne funkčný systém s dokumentáciou.

Abstrakt v cudzom jazyku

The main motivation of this thesis is to streamline the way IT service delivery is provided. One approach how to achieve this goal is to centralize and automate the workstation management and solve the issue of data backup and restore. It begins with the important part of reviewing all current solutions, comparing them according to the academic needs, and deciding for the competent one. In the practical part it is important to apply the accumulated knowledge by designing and implementing the optimal solution which is accessing to the infrastructure as code (IaC). With the use of open source based tools it is possible to improve the security and functionality of the IT support, dynamically change according to the end-user requests and system administrators´ needs. The output of this thesis should be a completely functional system with documentation.

Obsah

Zoznam ilustrácií

Zoznam tabuliek

Zoznam skratiek a značiek

AES	Advanced Encryption Standard, pokročilý šifrovací štandard
BYOD	Bring your own device, politika nosenia vlastného zariadenia
CIFS	Common Internet File System, všeobecný internetový súborový systém
DHCP	Dynamic Host Configuration Protocol, protokol na dynamickú konfiguráciu
DSL	Domain-specific language, jazyk špecifický pre doménu
DNS	Domain Name Systém, systém názvov domén
IaC	Infrastructure as code, infraštruktúra ako kód
IPAM	IP address management, systém spravovania IP adries
ISP	Internet service provider, poskytovateľ internetového pripojenia
FTP	File transfer protocol, protokol na prenos súborov
NAT	Network address transation, preklad sieťových adries
NFS	Network file system, sieťový súborový systém
NTP	Network time protocol, protokol pre synchronizáciu času cez internet
PXE	Preboot eXecution Environment, spôsob bootovania po sieti
SSH	Secure shell, zabezpečený komunikačný protokol
TCP	Transmission Control Protocol, protokol pre kontrolu prenosu
UDP	User Datagram Protocol, protokol užívateľských datagramov
WinRM	Windows Remote Management, vzdialená správa Windows-u
YAML	Yet Another Markup Language, ďalší značkovací jazyk

Úvod

Správa klientskych staníc, výučbových priestorov a počítačových učební predstavuje pre vzdelávaciu inštitúciu, vrátane univerzity, dôležitú a kľúčovú úlohu. Jedným z hlavných cieľov informačnej bezpečnosti je dosiahnuť dôvernosť, integritu a dostupnosť základných služieb organizácie. Medzi základné služby univerzity zaraďujeme aj poskytovanie vzdelávacej činnosti. Inými slovami, je potrebné zabezpečiť, aby počítačové učebne boli funkčné v čase výučby (zabezpečenie dostupnosti), aby uložené údaje, programy a používateľské údaje boli modifikované len oprávnenými osobami (zabezpečenie integrity), a aby k údajom, ktoré majú byť dostupné napr. učiteľom, boli dostupné len pre túto skupinu (zabezpečenie dôvernosti).

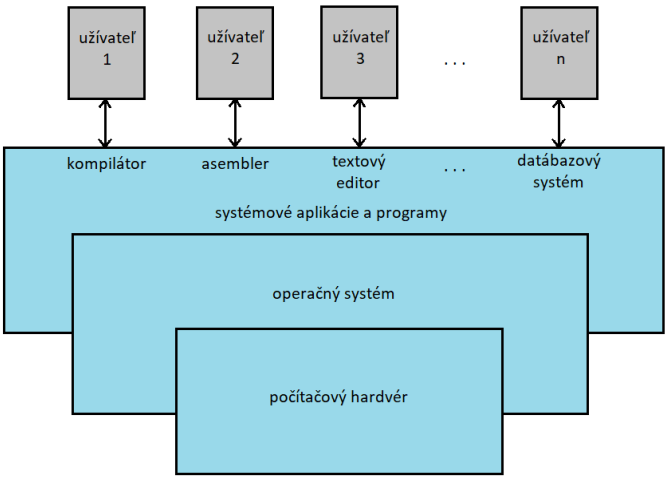
Aby bolo možné zabezpečiť vyššie uvedené základné prvky informačnej bezpečnosti, je nutné zabezpečiť centrálnu správu zariadení. V rámci tejto centrálnej správy je možné zaviesť opatrenia, ktorými dochádza k odstráneniu alebo minimalizácii bezpečnostných hrozieb.

Táto práca si stanovuje tri ciele. Prvým cieľom je analyzovať možnosti a prístupy k centrálnej správe počítačov. V tejto záverečnej práci sa zameriavame na pracovné stanice. Druhým cieľom práce je porovnať a analyzovať aktuálne riešenia k centrálnej správe pracovných staníc. Posledným cieľom je navrhnúť a implementovať centrálnu správu v prostredí vysokej školy, teda v akademickom prostredí.

Práca je rozdelená, vzhľadom na vyššie uvedené ciele, na tri časti. V rámci prvej kapitoly sa venujeme teoretickým základom správy pracovných staníc s prihliadnutím na špecifiká centrálnej správy. Druhá kapitola sa venuje prístupom k centrálnej správe pracovných staníc. V rámci kapitoly sa venujeme trom základným prvkom centralizovanej správy. Postupne porovnávame nástroje k vzdialenej správe pracovných staníc, nástroje k zálohovaniu a obnove údajov, a nástroje k vytvoreniu a obnove obrazov disku. Posledná kapitola predstavuje praktickú časť tejto záverečnej práce. V rámci nej sa zameriavame na návrh a implementáciu centralizovaného systému pre správu pracovných staníc. V rámci kapitoly sa venujeme čiastkovým problémom, ktoré je potrebné riešiť pri tejto správe.

1.1 Stratégie správy pracovných staníc

Jednou z najdôležitejších funkcií IT oddelenia v rámci organizácie je zvládnuť správu rôznych skupín počítačových systémov. **Počítačový systém** je zobrazený na Obrázku č. 1 a možno ho rozdeliť na štyri komponenty, a to hardvér, operačný systém, aplikačné programy a používateľov [1]. Ak sa tieto počítačové systémy používajú ľuďmi na zvládnutie ich pracovných úloh, môžeme ich označiť ako **pracovné stanice** (**workstations**) [2]. V tomto prípade nezáleží na tom, či je to počítačový systém používaný v kancelárii alebo premiestňovaný z miesta na miesto. Naopak od pracovných staníc rozoznávame servery, ktoré sú ukryté z dohľadu v oddelených miestnostiach (serverovniach). Servery poskytujú služby a vykonávajú výpočty vzdialene.



Obr. 1 Abstraktný pohľad na komponenty počítačového systému [1].

Z pohľadu používateľov je dôležité, aby pracovné stanice zabezpečili nasledujúce dôležité vlastnosti [2]:

- **lokalita** - pracovná stanica nie je užitočná, ak nie je tam, kde ju používateľ potrebuje,
- **spoľahlivosť** - začína dobrým hardvérom, ale závisí aj od softvéru a aktualizácií firmvéru,
- **produktivita** – znamená, že používateľ môže pracovať s minimálnymi prekážkami. Jeho prístup k službám a funkciám je rýchly a pohodlný (nie zdĺhavý a frustrujúci). Dôležité z pohľadu produktivity je si uvedomiť pomer, koľko času používateľ denne strávi produktívnou prácou, a aký čas venuje udržaniu funkčného systému.

- **používateľská agenda** – v tomto smere je dôležité si odpovedať na to, či majú používatelia kontrolu nad svojím prostredím, či môžu využiť svoj kreatívny potenciál alebo sú obmedzení predvolenou sadou funkcionalít. Tiež je dôležité si stanoviť, či používatelia majú kontrolu nad inštaláciou programov.
- **aktuálnosť** – predstavuje stanovenie časového rozdielu medzi dopravením novej funkcie a jej inštaláciou na konkrétne pracovnej stanici.

Definícia služieb poskytovaných pri centrálnej správe by sa mala začínať **požiadavkami.** Požiadavky sa postupne pretransformujú na technické špecifikácie. Požiadavka môže byť napríklad to, aby platforma podporovala určité aplikácie pri určitom výkone. Tieto preložíme do technických špecifikácií - aký hardvér, operačný systém a príslušenstvo potrebujeme obstarať, prípadne nakonfigurovať.

Za ideálnych okolností sú klienti zapojení do formulácií požiadaviek už na úplnom začiatku. Každá aplikácia má určený a dohodnutý rozsah podpory očakávaný od správcov. Nové verzie operačných systémov sú sledované a schvaľované.

Pri správe pracovných staníc je potrebné sa venovať nasledujúcim otázkam, ktoré si bližšie priblížime v nasledujúcich kapitolách:

- správe hardvéru pracovných staníc,
- správe softvéru pracovných staníc,
- správe operačného systému a
- správe používateľských údajov.

1.1.1 Odstupňovaná podpora

Niektoré organizácie vyžadujú od svojich správcov schopnosť ovládať všetky operačné systémy. S týmto sa najčastejšie stretávame na univerzitách a vo výskumných prostrediach, kde je bežné experimentovanie s novou technológiou. Pokúšať sa podporovať priveľa rôznych platform môže rozdeliť celý tím. Žiadna platforma nemusí byť podporovaná dostatočne. Je efektívnejšie ponúkať úplnú podporu pre pár špecifických platform a mať prispôsobené pravidlá pre ostatné. Toto nazývame **odstupňovaná podpora [2,3]**. Väčšina používateľov dostane lepšie podporovanú platformu. Tí, ktorí vyžadujú špeciálne, na mieru upravené platformy, sú prevažne dosť schopní, aby si pomohli aj sami. Odstupňovanú podporu môžeme rozdeliť do nasledujúcich stupňov [4]:

- **prvý stupeň** – pracovné stanice v tomto stupni sú plne podporované. Je k dispozícii podpora hardvéru a ovládačov, jednoduchá reinštalácia operačného systému, automatické aktualizácie alebo testovanie nových verzií operačných systémov pred nasadením.
- **druhý stupeň** – v rámci tohto stupňa je vynaložená vysoká snaha spravovať pracovné stanice. Tieto stanice môžu byť pripojené na centrálne služby. To, že niečo funguje v určitú dobu, nemusí fungovať nasledujúci deň (napr. emailoví klienti).
- **ad hoc podpora** – pracovné stanice v rámci tejto podpory nie sú priamo podporované. Akákoľvek podpora je svojpomocná. Pracovné stanice môžu byť izolované na osobitnej virtuálnej sieti, ktorá je oddelená od zvyšku počítačovej siete.
- **zakázaná podpora** - všetky ostatné pracovné stanice nemajú povolené pripájať sa na pracovnú počítačovú sieť.

V rámci akademického prostredia využívame všetky vyššie spomenuté stupne. Prvý stupeň zahŕňa väčšinu spravovaných pracovných staníc, medzi ktoré patria pracovné stanice v učebniach u administratívy a u väčšiny klientov. Druhý stupeň podpory je určený pre staršie pracovné stanice, na ktorých by najnovší softvér mal negatívny vplyv na výkonnosť a plynulosť práce. V kategórii Ad Hoc podpory sú špeciálne a vedecké pracovné stanice vzhľadom na komplexnosť hardvéru a proprietárny softvér. Poslednú kategóriu podpory tvoria pracovné stanice s nepodporovaným operačným systémom, ktoré by znamenali bezpečnostné riziko pre počítačovú sieť.

1.1.2 Pracovné stanice ako služba (outsourcing)

Protikladom vyššie uvedených princípov a definície správy pracovných staníc je postup, keď má organizácia pracovné stanice ako veľký balík rozdelený medzi používateľov za fixnú cenu. Táto služba sa nazýva aj **outsourcing** [2]. Balíček služieb obsahuje všetky časti životného cyklu pracovnej stanice: počítač a jeho náhrada za určitý časový cyklus, predinštalovaný operačný systém s automatizovanými reinštaláciami, pravidelné aktualizácie, centrálne úložisko, správcovskú podporu, zabezpečenie, opravy na mieste a podobne. Oddelenia sú automaticky fakturované na mesačnej báze za fixnú cenu, čo zjednodušuje odhad na finančnú záťaž.

Výhodou outsourcingu je, že oddelenia organizácie dostanú lepšie služby, ako keby si ich zabezpečovali samé. Pracovné stanice ako služba zvyknú byť nazývané aj ako prenajaté pracovné stanice na základe ich financovania. Takéto služby môžu vyjsť lacnejšie, pretože pre poskytovateľa sú jednotlivé organizácie len klienti s malými variáciami na požiadavky a veľa z týchto služieb sa opakuje.

Na druhej strane existujú aj nevýhody. Služba sa môže zdať drahšia pre zákazníkov, ktorí porovnávajú mesačnú cenu s cenou nového počítača, ignorujúc všetky ostatné služby, ako napríklad diskové úložiská, počítačová sieť alebo tlačové služby.

Keďže na počítačový systém, a teda aj na pracovné stanice, môžeme nazerať ako na systém pozostávajúci z hardvéru, operačného systému softvéru a údajov [1], v nasledujúcich kapitolách sa budeme venovať správe pracovných staníc po jednotlivých oblastiach.

1.1.3 Infraštruktúra ako kód

Infraštruktúra ako kód (IaC) predstavuje stratégiu správy systému, v ktorej je infraštruktúra zabezpečovaná a riadená prostredníctvom strojovo definovateľných súborov definícií radšej, než manuálnou prácou [2]. Pomocou tohto prístupu je možné riadiť infraštruktúru technikami softvérového inžinierstva, ako je riadenie zdrojového kódu, jednotkové testy a nepretržitá integrácia a dodávka (CI / CD) [5]. IaC je relatívne nový termín, ale spája množstvo myšlienok, ktoré sa už desaťročia vyskytujú v správe systému.

V prostredí IAC nedochádza ku komunikácii priamo s infraštruktúrou (napr. pracovnými stanicami). Namiesto toho sa aktualizuje kód a údaje, ktoré sa používajú na vytvorenie prostredia. Napríklad, namiesto konfigurácie mnohých pracovných staníc, pri ktorých sa musí inštalovať jednotný softvér a nastavovať bezpečnostné politiky, stačí napísať program, ktorý načíta súbor s konfiguráciou a aktualizuje tieto stanice.

IaC nie je konkrétny programovací jazyk alebo systém. Je to stratégia, ktorá je implementovaná nástrojmi, ktoré si bližšie priblížime v druhej kapitole. Výhodami IaC sú nižšie náklady, lepšia rýchlosť a nižšie riziko. Náklady sa znižujú, pretože manuálna práca je znížená alebo eliminovaná. Nielenže sa úlohy môžu robiť rýchlejšie, ale môžu byť vykonávané aj paralelne. Použitím IaC sa napríklad znižuje bezpečnostné riziko,

pretože je možné dosiahnuť súlad v uplatňovaní bezpečnostnej politiky organizácie (napr. jednotné bezpečnostné nastavenia pracovných staníc).

1.2 Správa hardvéru klientskych staníc

Prvou dôležitou otázkou pri správe pracovných staníc predstavuje zvolenie si **vhodného hardvéru**. **Hardvér** môžeme definovať ako centrálnu procesorovú jednotku (CPU), pamäť a vstupno-výstupné (I/O) zariadenia, ktoré spoločne poskytujú základné výpočtové zdroje pre počítačový systém [1].

Na výber sú stolové počítače, notebooky, tablety a mobilné zariadenia. Všetky tieto zariadenia je možné obstarať v rôznych konfiguráciách. Výber vhodného hardvéru závisí aj od rozhodnutia medzi fyzickými a virtuálnymi strojmi, a od použitia počítača poskytnutého organizáciou alebo politikou BYOD (prines si vlastné zariadenie). Bližšie sa aspektami fyzických zariadení a virtuálnych zariadení budeme zaoberať v nasledujúcich podkapitolách.

1.2.1 Fyzické zariadenia

Pri výbere fyzického zariadenia je dôležitá otázka, či ísť cestou notebooku, alebo stolového počítača (desktopu). **Desktopy** sú štandardne viac rozšíriteľné. Majú sloty na prídavné karty, na dodatočnú operačnú pamäť, resp. prídavné disky. Na druhej strane notebooky sú obvykle drahšie ako desktopy s rovnakou konfiguráciou. Pri notebookoch sa počíta s mobilitou, ktorá vyžaduje odolnosť voči nárazom, pádom a podobným nehodám. Z tohto dôvodu je potrebné dôsledné testovanie zariadení a premyslenejšia konštrukcia. To sa môže prejaviť na výslednej cene. Servisné zásahy bývajú drahšie kvôli cene súčiastok a komplexnosti šasi. Ak má notebook proprietárne periférne sloty alebo konektory, tak cena príslušenstva bude vyššia kvôli menšej konkurencii. Mnohokrát sa pri notebookoch uprednostňuje úspornejší procesor s menšou tepelnou stratou kvôli výdrži a chladeniu. Výhodou notebookov je jednoduchší prenos v prípade potreby servisu. Ďalšou výhodou je ich použitie na pracovisku s dynamickým usadením.

Dôležitý aspekt pri výbere hardvéru predstavuje **výkon** [2]. Produktová línia, ktorá si zakladá na výkone, disponuje špecifikáciou, ktorá zvláda procesorovo a graficky náročné úlohy, ako sú virtualizácie, rôzne simulácie, kompilácie kódu, výpočty a podobne. Takéto modely majú najsilnejšie procesory, rozšíriteľné o veľkú operačnú

pamäť a taktiež viac možností pevných diskov. Výrobcovia ponúkajú najsilnejšie procesory za najvyššiu cenu, pretože zákazníci vedia oceniť ich pridaný výkon.

1.2.2 Virtuálne zariadenia

Oproti fyzickým zariadeniam je možné využiť virtuálne zariadenia. **Virtuálny stroj (Virtual Machine, VM)** môžeme definovať ako abstrakciu hardvéru jedného počítača (CPU, pamäť, diskové jednotky, sieťové karty atď.) do viacerých rôznych prostredí, čím sa vytvára ilúzia, že každé jednotlivé prostredie predstavuje vlastný počítač [1].

Príkladom použitia virtuálnych strojov sú situácie, kedy je potrebné mať na fyzickom zariadení niekoľko rôznych operačných systémov. V rámci univerzity virtuálne stroje využívame najmä v počítačových učebniach, pri výučbe operačných systémov odlišných od operačného systému Windows, alebo z dôvodu nutnosti využívania aplikačného vybavenia využívajúceho operačný systém Linux.

1.2.3 Cyklus obmeny hardvéru

Dôležitým prvkom pri správe hardvéru pracovných staníc je **cyklus obmeny hardvéru**. V rámci organizácie je potrebný systematický spôsob identifikácie staršieho hardvéru a jeho výmena. Hardvér časom zostarne a novšie operačné systémy ho prestanú podporovať. Častokrát nové aplikácie potrebujú výkonnejší hardvér pre ich plynulý chod. **Obmeny** [2] sú plánované upgrady starého hardvéru. Príkladom môže byť upgrade komponentov počítača (pamäť, disk) každé 2 roky a upgrade celého počítača (výmena počítača) každých päť rokov. Bez správne vypracovaného plánu zvyknú organizácie meniť hardvér na základe najvyššieho počtu sťažností alebo osobných preferencií.

1.3 Správa operačného systému pracovných staníc

Operačný systém [1] je program, ktorý spravuje hardvér počítača, poskytuje základ pre aplikačné programy a pôsobí ako sprostredkovateľ medzi používateľom počítača a hardvérom počítača. Ako súčasť architektúry pracovných staníc môžeme prevádzkovať jeden alebo viac operačných systémov. Poznáme hlavné skupiny operačných systémov s malými variáciami v rámci skupiny. Medzi hlavné skupiny patria

v súvislosti s dodávateľmi: Microsoft Windows, Apple OS X a Linux. V každej z nich sú malé variácie, Microsoft a Apple vytvorili rôzne generácie operačných systémov počas rokov svojej existencie. Linux má aj rôznych pôvodcov a verzie. Napríklad Ubuntu, Debian, RedHat, Fedora a pod. Každá distribúcia vypúšťa pravidelne nové verzie, s malými aktualizáciami medzi nimi. Podobne Microsoft a Apple majú serverové a užívateľské variácie každého operačného systému.

Diskusia, koľko operačných systémov podporovať je podobná otázke o podpore rôznych hardvérových konfigurácií. Minimalizovaním rôznorodosti dostaneme ľahšie podporovateľný systém. Organizácia, ktorá podporovala len jeden operačný systém, pridaním ďalšieho operačného systému spraví veľký krok. Zdvojí sa podpora infraštruktúry, systémy pre automatické nahrávanie operačného systému, aktualizácie a podobne. Zdvojnásobí sa aj potrebná expertíza, ktorú musí organizácia dodržiavať. Jedna stratégia je zamestnať ľudí, ktorí majú znalosti oboch operačných systémov, čo môže byť ťažké a drahé. Druhá stratégia je zväčšiť tím a nabrať špecialistov na každý operačný systém. Aj tento krok prináša zvýšené náklady a ďalšie výzvy.

Nezávisle od počtu podporovaných operačných systémov, v každom vydaní sú rozdiely. Je lepšie podporovať všetky verzie, najaktuálnejšiu alebo nejakú podmnožinu. Dobre zadefinovaná podmnožina operačných systémov je najlepšie riešenie. Napríklad organizácia môže mať oficiálne podporované dva operačné systémy, a to aktuálny a predchádzajúci. Napríklad Windows 10 a Windows 7. Požívatelia môžu naďalej požívať Windows 7, ale očakáva sa, že do určitého času prejdú na Windows 10. Lokálna správa určila plán, aby zabezpečila, že budú identifikované všetky klientske stanice s Windows 7, ktoré budú neskôr konvertované na Windows 10.

1.3.1 Inštalácia operačného systému

Inštalácia operačného systému je proces, kedy je vymazaný akýkoľvek predošlý systém a je nahradený novým a teda je v stave „čistý“ [2]. Inštalácia je najlepšie dosiahnutá pomocou automatizácie. Tá nám zabezpečí správne nastavenie každej klientskej stanice a vyhneme sa tak zbytočným chybám.

Kritérium každej stratégie inštalácie operačného systému je, aby výsledné pracovné stanice boli nastavené konzistentne a správne. Pokiaľ inštalácia nie je dokončená správne, používatelia a ich pracovná činnosť tým môžu trpieť. Ak má používateľ viac pracovných staníc, každý rozdiel medzi nimi ho môže zmiasť. Z tohto

dôvodu je jednotnosť dôležitejšia. Dokonalosť je skoro nemožná, najmä keď je veľa preferencií subjektívnych. Aj keď sa dokonalosť nedá dosiahnuť, je možné sa k nej priblížiť inkrementálnymi úpravami.

Dokonalým výsledkom inštalácie operačného systému by bola automatizácia, kde by proces prebiehal samoobslužne. Investícia času do automatizácie sa neskôr vráti viacnásobne. Inštalácia operačného systému a prvotných aplikácií môže byť dosiahnutá nižšie uvedenými spôsobmi, ktoré si bližšie predstavíme v nasledujúcich podkapitolách [2]:

- manuálna inštalácia,
- automatizovaná inštalácia a
- klonovanie operačného systému.

1.3.2 Manuálna inštalácia operačného systému

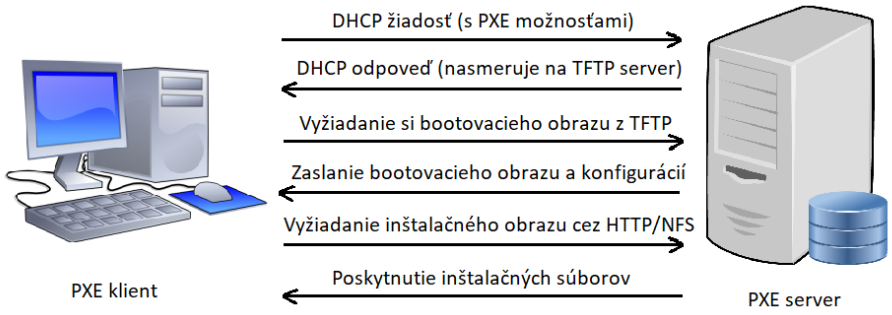
Najzakladanejším a najmenej preferovaným spôsobom inštalácie operačného systému je **manuálna inštalácia**. Postup manuálnej inštalácie je pomerne jednoduchý a pozostáva z vloženia inštalačného média (napr. DVD disku), vymazania pevného disku pracovnej stanice, nainštalovaní operačného systému, jeho aktualizácii a napokon základných nastavení. Aby bola pracovná stanica použiteľná, je potrebné ešte doinštalovať programy, nastaviť počítačovú sieť, povoliť, resp. zakázať niektoré funkcionality systému. Tieto úkony sú ale zdĺhavé. Najčastejšie k manuálnej inštalácii operačného systému dochádza, keď sa nové pracovné stanice inštalujú tak zriedkavo, že nie je vôbec efektívne automatizovať inštaláciu operačného systému.

1.3.3 Automatizácia inštalácie operačného systému

Druhým prístupom **je automatizovaná inštalácia operačného systému**. Všetky moderné operačné systémy ponúkajú prostriedky k automatickej inštalácii. Microsoft Windows má Microsoft Deployment Toolkit (MDT), RedHat Linux má KickStart, Debian má Debconf a pod. Väčšina riešení zdieľa rovnaký postup. Najprv sa načíta systém z inštalačného servera po počítačovej sieti. Následne sa pripraví kompaktný operačný systém, pripravia sa partície a nainštaluje operačný systém. Napokon sa pripravia údaje, aplikácie, aktuálne nastavenia. To všetko z počítačovej siete. Po dokončení sa počítač reštartuje a spustí sa novo nainštalovaný operačný systém.

Pri automatizácii inštalácie je dôležitá dokumentácia tohto procesu. Pri udržiavaní automatizovanej inštalácie operačného systému alebo pri administrácii obrazov diskov, je potrebné dodržiavanie metodiky zhromažďovania a zapisovania vykonaných zmien.

Základ automatizácie čohokoľvek je začať to vykonávať. Ak to nevieme spraviť manuálne, automatizovať to bude omnoho ťažšie. Je potrebné začať po malých krokoch. Prvý cieľ pri automatizácii inštalácie operačného systému by malo byť bootovanie po počítačovej sieti pomocou **Preboot eXecution Environment (PXE) [6]**. Použitie PXE v inštalačnom procese je znázornené na obrázku č. 2. Ďalším krokom je eliminácia manuálnych príkazov vedúcich k tomu. Následne je potrebné riešiť nastavenia Biosu, ovládačov, inštalačných balíčkov a pod. Netreba odkladať nasadenie systému, kým bude dokončený. Čím skôr sa nasadí, tým skôr sa prejavia nedostatky, ktoré môžeme opraviť. Časom bude vyvíjaný postup automatizácie vylepšovaný, odolnejší voči chybám a rýchlejší. Tento postup nazývame **inkrementálny vývoj** [2].



Obr. 2 PXE boot a inštalačný proces.

Zaujímavou otázkou je, v ktorých prípadoch **nepoužiť automatizáciu**. Neprítomnosť automatizácie sa dá ospravedlniť len v prípade, kedy je zástupcom každej hardvérovej konfigurácie iba jeden stroj, ak náklady na automatizáciu presiahnu ušetrený čas, alebo ak je to skutočne nemožné vplyvom vývojára softvéru a podobne.

Najzásadnejší krok je mať dobre zdokumentované postupy, aby bolo možné ich inštaláciu opakovať znovu a znovu. Dokumentácia môže byť vo forme poznámok. Každý uvedený krok, či poznámka, neskôr poslúži ako základ pre skript automatizácie. Keď sa všetky skripty spoja, výsledkom bude plná automatizácia.

1.3.4 Klonovanie operačného systému

Pri **klonovaní operačného systému** sa najprv nainštaluje prvý počítač podľa požiadaviek a potom sa z neho spraví **bitová kópia,** ktorá sa uloží a neskôr použije pri klonovaní. Takáto bitová kópia sa nazýva **„Golden Image" (zlatý obraz)** [2]. Nástrojom na vytváranie obrazov diskov sa venujeme v ďalšej časti tejto práce. Pri správe pracovných staníc v učebniach sa v prostredí univerzity robí Golden Image každý semester. Vychádza sa z poslednej verzie, ktorá sa doplní o nový softvér a existujúci softvér sa aktualizuje. Súčasne sa aktualizuje operačný systém a zmenia sa nastavenia podľa aktuálnej politiky. Z takto otestovaného systému sa vyrobí bitová kópia Golden Image, ktorá bude slúžiť pre aktuálny semester akademického roka.

Automatizácia inštalácie operačného systému je vo väčšine prípadov výhodnejšia ako klonovanie pre niekoľko dôvodov. Prvým je, že ak máme viac hardvérových konfigurácií, tak je potrebné pre každú vytvoriť nový „Golden Image", čo môže dospieť do situácie, keď máme veľa separátnych bitových kópii a urobiť jednu zmenu na všetkých obrazoch diskov je náročné. Pri klonovaní je tiež ťažké odhaliť rozdiely medzi jednotlivými iteráciami, pokiaľ nie sú niekde písomne uvedené. Ak je vytvorená nová kópia, ktorá má chyby, je ťažké ich odstrániť inak, ako vrátiť sa na predošlú kópiu.

Vyššie uvedené nevýhody tohto prístupu môžu byť eliminované automatizáciou výroby obrazov disku. Packer [7] a Vagrant [8] sú príklady dvoch nástrojov, ktoré to dokážu. Ak sú potrebné zmeny v obraze disku, pridajú sa do konfiguračného súboru a vytvorí sa úplne nový obraz disku.

1.3.5 Aktualizácia operačného systému

Vždy musí existovať spôsob, ako aktualizovať operačný systém a softvér. Softvér nikdy nie je „dokončený", vždy môže byť vylepšený [2]. Kým je softvér podporovaný výrobcom, majú byť k dispozícii nové aktualizácie, záplaty, resp. opravy. Očakávať, že operačný systém alebo softvér nebude nikdy potrebovať aktualizácie, je iracionálne. Z tohto dôvodu je v organizácii potrebné mať stratégiu, podľa ktorej budú aktualizácie distribuované.

Aktualizácie môžeme inštalovať manuálne alebo automaticky. Pod **manuálnou aktualizáciou** sa rozumie aktualizácia operačného systému každej pracovnej stanice

zvlášť. To je možné osobne, alebo cez vzdialenú plochu (Remote Desktop). Od konca roka 2017 aj pomocou klienta SSH. Na druhej strane, **automatickou aktualizáciou** sa rozumie aktualizácia bez zásahu správcu. Môže byť vykonávaná pravidelne, napríklad týždenne alebo na príkaz správcu.

Vysoko odporúčaným trendom sú automatické aktualizácie. K tomuto záveru existuje niekoľko dôvodov. Prvým dôvodom je **škála**. Po čase pribudne viac pracovných staníc a udržiavať ich manuálne aktualizované bude stále viac a viac časovo náročné. Druhým dôvodom je **úplnosť**. Ak je aktualizácia dôležitá, je potrebné, aby bola stiahnutá a nainštalovaná bez ohľadu na okolnosti.

1.3.6 Konfigurácia sieťových nastavení

Pracovné stanice môžu byť pripojené do počítačovej siete káblom alebo bezdrôtovo. Infraštruktúra a postupy organizácie určujú, či sieťová konfigurácia bude uložená v zariadení alebo dynamicky (dodaná cez počítačovú sieť). Tieto parametre obsahujú IP adresu zariadenia, masku podsiete, predvolenú bránu, DNS servery a ďalšie parametre. V rámci operačného systému môžeme rozlišovať nasledujúce sieťové konfigurácie [9]:

- dynamickú konfiguráciu
- statickú konfiguráciu a
- hybridnú konfiguráciu.

Pri **dynamickej sieťovej konfigurácii** si pracovná stanica vypýta potrebné parametre konfigurácie. Dynamická konfigurácia umožňuje nastavenie sieťovej konfigurácie ovládať centrálne. Schopnosť centrálne ovládať konfigurácie je kľúčom k efektívnemu spravovaniu veľkého počtu zariadení (pracovných staníc). V inom prípade by musel administrátor navštíviť každé zariadenie osobitne aj pri najmenšej zmene.

Pri **statickej** (pevnej) **konfigurácii** sú parametre sieťovej konfigurácie uložené v samotnom zariadení. Výhody takejto konfigurácie sú, že fungujú, aj keď nie je dostupný DHCP server. Servery obvykle používajú statické konfigurácie, aby sa znížila ich závislosť od externých zdrojov. Napríklad adresa zálohovacieho servera sa nemení. DHCP server potrebuje statickú konfiguráciu, nakoľko sa sám nedokáže nastaviť.

Pracovné stanice môžu mať aj **hybridnú konfiguráciu**, kde sú sieťové parametre uložené v lokálnych súboroch. Server ale pravidelne kontroluje sieťové parametre cez

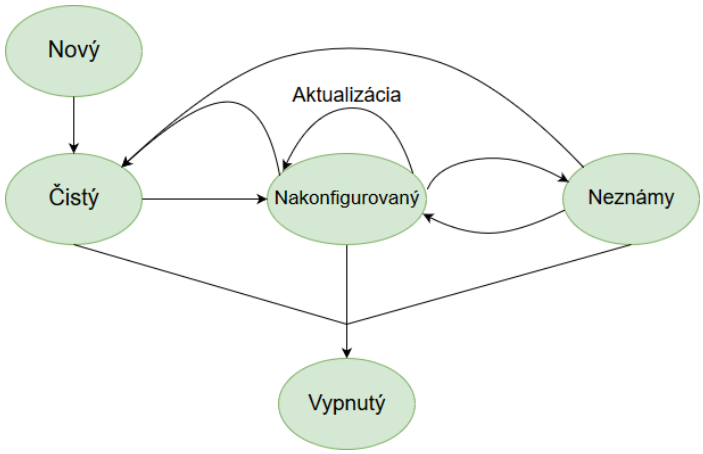
protokol DHCP. Táto technika používa DHCP INFORM požiadavku na získanie parametrov ako DNS servery, NTP servery a podobne. Ak sú detegované zmeny, tak sú súbory konfigurácie postupne zmenené. Tento prístup kombinuje jednoduchosť udržiavania rozsiahlych inštalácií s benefitom odstránenia zdĺhavého bootovania na DHCP pre kritické zariadenia.

1.4 Správa softvéru pracovných staníc

Druhým nemenej dôležitým aspektom pri správe pracovných staníc je správa softvéru. **Softvérom** rozumieme programy a ďalšie operačné informácie používané počítačovým systémom [10].

1.4.1 Životný cyklus softvéru klientskych staníc

Pred inštaláciou softvéru musí byť nainštalovaný a nastavený operačný systém. Časom je potrebné aktualizovať operačný systém aj aplikácie, aby boli zistené chyby odstránené, pridané nové vlastnosti, resp. zaplátané odhalené zraniteľnosti. Nastavenia systému by mali byť prispôsobené požiadavkám používateľa a politikám organizácie. Súbory a nastavenia sa môžu časom poškodiť, a teda môžu vyžadovať odborný zásah.



Obr. 3 Životný cyklus pracovnej stanice a jej operačného systému a softvéru.

Diagram na obrázku č. 3 zobrazuje päť stavov, v ktorých sa môže nachádzať pracovná stanica so svojim operačným systémom a softvérom. Prvým stavom je **nový (new)**, čo znamená úplne novú, nekonfigurovanú pracovnú stanicu. Druhý stav je **čistý**

(clean), ktorý predstavuje pracovnú stanicu, na ktorú bol nainštalovaný operačný systém a softvér, ale neboli vykonané žiadne nastavenia. Ďalším stavom je **nakonfigurovaný (configured)**, čo predstavuje pracovnú stanicu so správne nakonfigurovaným a prevádzkovým prostredím. **Neznámy (uknown)** predstavuje ďalší stav, v ktorom pracovná stanica bola nesprávne nakonfigurovaná alebo operačný systém a softvér sa stali zastaralými. Posledným stavom je **vypnutý (off)**, čo znamená, že pracovná stanica je vypnutá alebo vyradená.

1.4.2 Výber vhodného softvéru

Jedným z prvých krokov pri správe pracovných staníc je zvolenie si softvéru, ktorý bude na týchto staniciach nainštalovaný. Toto sa neskôr stane základom **softvérového štandardu.** Všetky ďalšie zmeny by mali byť zadefinované v zmysle tohto štandardu. Tento štandard prevažne obsahuje určenie konkrétnych operačných systémov a softvérov, používaných v rámci organizácie. Napríklad ide o používanie schváleného emailového klienta dodržujúceho určitý štandard. K výberu softvéru bývajú zriedkavo formálne zasadnutia. V niektorých organizáciách je postup formálnejší a každá skupina používateľov si určí, aký softvér k práci potrebuje. V rámci organizácie funguje pracovná komisia zložená z používateľov, správcov a manažérov, ktorá schvaľuje dané požiadavky.

1.4.3 Aktualizácia a inštalácia softvéru

Práca správcu nekončí inštaláciou operačného systému. Ten je časom potrebné udržiavať v aktuálnom stave, chrániť pred zraniteľnosťami, sledovať dostupnosť nových verzií softvéru a podobne. Všetky tieto úlohy vyžadujú aktualizácie softvéru a niekoho, kto sa o to postará. Podobne ako inštalácie softvérov, aj aktualizácie by mali byť automatizované. Systémy pre softvérové aktualizácie by mali byť schopné inštalovať nové aktualizácie, aktualizovať existujúce a aktualizovať operačný systém. Napríklad **Microsoft WSUS** (Windows Server Update Services) zvládne všetky tri spomenuté akcie. Linux je založený na skupinách mnohých balíčkov (napr. Debian používa deb balíčky), pričom linuxové aplikácie používajú rovnaký formát. V tomto smere existuje viacero balíčkovacích systémov aj pre pracovné stanice s operačným systémom

Windows. Možnosť automatizovane pretlačiť softvérové balíčky je kľúčom k dobre zabehanému prostrediu [2].

1.4.4 Prístupy k aktualizácii

V rámci aktualizácií existuje niekoľko postupov, ako pristúpiť k aktualizácii pracovných staníc [2]:

- prístup bez aktualizácií,
- používateľom prevzaté aktualizácie a
- automatické aktualizácie s užívateľovým súhlasom.

Prvým prístupom je **neaktualizovať vôbec**. Je to najjednoduchší prístup, ale zároveň najťažší. Kým spočiatku nie je potrebná žiadna aktivita, ku koncu môže vďaka nezabezpečenému a neaktuálnemu systému pribudnúť viac práce, ako sa na začiatku ušetrilo.

Druhým prístupom sú **aktualizácie,** pri ktorých je potrebné, aby ich spustil jeho **používateľ**. Väčšina používateľov na to nemyslí, kým ich niekto nevyzve. Je to veľmi častý prístup, ktorý sa ale neodporúča. Používatelia by nemali byť zodpovední za aktualizácie. Aktualizácie softvéru pracovných staníc by mali zabezpečovať správcovia.

Najlepšia voľba je používať prístup **aktualizácií s používateľským súhlasom**. Automatické aktualizácie by mali byť koordinované, aby nenarušili pracovnú činnosť. Pri pracovných staniciach by aktualizácia počas práce mohla spôsobiť stratu aktuálne vykonaných zmien alebo rozpracovaných súborov. Aktualizácie so súhlasom by mali zabrániť takýmto situáciám.

1.5 Správa používateľských údajov

Používateľ potrebuje na pracovnej stanici ukladať údaje alebo stav. Tie sa ukladajú obvykle vo forme súborov alebo adresárov. Môžu ale obsahovať taktiež nastavenia a prispôsobenia, ktoré používatelia nastavajú v prostredí operačného systému. Rozoznávame nasledujúce spôsoby správy používateľských údajov [2]:

- lokálne,
- bezstavové a
- bezdiskové.

Prvým prístupom je, keď sú všetky **údaje uložené lokálne**. Systém je nastavený tak, že užívateľské údaje sú uložené na lokálnom disku, ktorý je použitý aj pre operačný systém a všetky s ním spojené údaje. Lokálne disky sa často nazývajú **Direct attached storage (DAS).** Na tomto princípe funguje väčšina pracovných staníc v našom akademickom prostredí.

Druhý prístup je bezstavový, čo znamená **žiadne lokálne unikátne údaje**. Požívateľské údaje sú uložené vzdialene na sieťovom disku. Akákoľvek informácia uložená lokálne je kópia údajov, ktoré sú uložené na inom mieste. Lokálny disk je využitý len pre operačný systém, dočasné súbory a pod. Tento prístup je našom akademickom prostredí použitý pre pracovné stanice administratívy.

Posledný prístup nazývame **bezdiskový**, čo vyjadruje použitie bez lokálneho disku. Operačný systém a používateľské údaje sú uložené na vzdialenom sieťovom disku za použitia protokolov ako **iSCSI alebo NFS**. Pokiaľ sa zariadenie poškodí alebo zlyhá, tak údaje naviazané naň môžu byť pripojené na náhradné zariadenie. Tento systém sa najčastejšie používa pri virtuálnych systémoch, tenkých klientoch alebo bladových systémoch.

Pri vytváraní generických a zastupiteľných pracovných staníc je cieľom spraviť ich bezstavovými. To sa dá niekoľkými spôsobmi, najmä však vytvorením [9]:

- **vzdialeného dátového úložiska** - Klient pristupuje k údajom zo vzdialeného servera ako zo sieťového disku, ktorý zobrazuje údaje, akoby boli lokálne. Napríklad NFS pri Unix-och alebo SMB pri Windows-och.
- **sieťovo-synchronizovaného alebo cloudového úložiska** - Užívateľove údaje sú uložené lokálne, ale zároveň kopírované na sieťovú službu. Napríklad Dropbox, OneDrive, iCloud, Google Drive. So súbormi je manipulované lokálne, ale akékoľvek zmeny sú synchronizované s kópiou na sieťovej službe.

1.6 Centralizovaný model správy

V rámci organizácií sa odporúča **centralizovaný model správy pracovných staníc**. Tento model rieši otázky, ako majú jednotlivé systémy medzi sebou komunikovať, ako má byť navrhnutá počítačová sieť, podporné služby a pod [9]. S centralizovaným prístupom má organizácia jednotnú architektúru a všetky služby sú

postavené rovnakým spôsobom vo všetkých organizačných zložkách. Každá služba môže byť v rámci architektúry organizácie postavená ako celok a prihliada sa aj na interakcie medzi jednotlivými službami a systémami. Jednotná infraštruktúra sa ľahšie prevádzkuje, a dokumentuje. Hlavná výhoda je efektívnejšia automatizácia nakoľko sa dá nasadiť cez celú organizáciu bez väčších zmien.

1.6.1 Infraštruktúra

Dôležitým prvkom centralizovanej správy je infraštruktúra. Tá poskytuje služby celej organizácii. Nie je cielená konkrétnej skupine používateľov. Služby infraštruktúry by mali byť unifikované a poskytovať rovnakú úroveň spoľahlivosti a služieb v celej organizácii. Centralizácia infraštruktúry zaručuje kompatibilitu, zjednodušuje dokumentáciu, znižuje náklady a zlepšuje služby. Medzi centrálnu infraštruktúru môžeme zaradiť:

- správu rozsahu IP adries,
- správu menného priestoru (doménových mien),
- komunikáciu,
- správu údajov,
- monitorovanie a
- zaznamenávanie (logovanie).

Systém spravovania IP adries (IPAM) je naviazaný na správu počítačovej siete. Vo veľkých organizáciách sú veľké rozsahy IP adries pridelené lokálnym tímom (správcom), ktorí prideľujú jednotlivé podsiete pre konkrétne kancelárie, datacentrá a iné služby. Centrálna správa IP adries zaručuje, že siete nebudú použité na viacerých miestach súčasne. Umožňuje to zavádzať štandardy a politiky v rámci organizácie.

Centralizácia DNS služieb a **správy doménových mien** (menného priestoru) umožňuje jednoduchšiu podporu a lepšiu automatizáciu. Znižuje zložitosť, ktorá by sťažovala hľadanie chýb, najmä pri problémoch s protokolmi DNS alebo DHCP. Pri absencii centralizácie DNS sú tieto systémy spravované rôznymi tímami administrátorov, čo môže viesť k rôznym rizikám.

Komunikačné systémy obsahujú správu emailov, chatov, hlasových služieb a video konferencií. Pri centralizovanej platforme môžu rôzne oddelenia a jednotlivci pristupovať k službám. Klienti majú v súčasnej dobe vysoké očakávania od

komunikačných systémov. Medzi takéto očakávania patrí možnosť komunikovať s kýmkoľvek, kedykoľvek. Inými slovami fungovanie za všetkých okolností. Je oveľa ľahšie a lacnejšie spravovať tieto služby, ak sa o nich stará jeden tím administrátorov. Pri používaní rovnakých produktov, verzií a nastavení na všetkých miestach je zaručená vysoká kompatibilita. Uľahčuje to aj testovanie a správu.

Správa údajov obsahuje úložný priestor a zálohy. Tieto služby sú využívané mnohými globálnymi službami (napr. uloženie údajov pre zálohovaciu službu). Centralizácia tejto platformy umožňuje klientom vybrať množstvo, typ, rýchlosť a miesto ukladania údajov. Ukladanie údajov a zálohy sú zložité technické služby. Očakávať od každého tímu vysoké znalosti v tejto oblasti je nereálne. Pri zálohovaní je dôležité aj vedieť obnoviť údaje. Skúšať pravidelne funkčnosť zálohy a obnovy je jednoduchšie pri centralizovanom prístupe.

Centralizované monitorovanie má veľa výhod. Monitorovanie je ale náročné a je výhodnejšie, ak sa tým zaoberá jeden tím. V opačnom prípade to bude každý tím robiť na rôznej úrovni. Monitorovanie umožňuje nájsť spojitosť medzi udalosťami a odhaliť tak pôvod problémov rýchlejšie. Poskytnutie monitorovania ako centralizovanej platformy znamená, že ktokoľvek má možnosť prihlásiť sa k službe.

Centralizované zaznamenávanie (logovanie) umožňuje korelácie medzi udalosťami a možnosť jednoduchšie tak odhaliť pôvod problému. Centralizované služby by mali mať prístup k rýchlym dátovým úložiskám s veľkou kapacitou a automatickou archiváciou. Musia obsahovať rozhranie, ktoré umožní správcom pristupovať k denníkom a logom ich systémov, aby odhalili problémy. Centralizované logovacie zriadenie musí mať bezpečnosť schopnú prideľovať prístup k záznamom (logom) podľa poverení.

1.6.2 Centrálna podpora

Centralizácia podpory služieb je tak dôležitá ako samotná infraštruktúra. Centralizácia podpory neznamená, že celý tím správcov musí sídliť v jednej budove. Klienti očakávajú, nezávisle od úrovne centralizácie, rovnakú úroveň starostlivosti. Pre všetkých by mali platiť rovnaké pravidlá na rovnakej úrovni a všetci by mali mať prístup k rovnakej podpore. Pokiaľ bude mať jedno oddelenie k dispozícii lepšiu podporu , alebo prístup, bude to vnímane negatívne.

Všetky pracoviská by mali mať jeden **centrálny helpdesk**, na ktorý sa môžu používatelia obrátiť pokiaľ potrebujú podporu. Centrálny helpdesk je jediným bodom prístupu nezávisle od toho, či zamestnanec cestuje, je na pracovisku, alebo doma [9]. Zároveň centrálny helpdesk môže byť nástroj na kontaktovanie iných správcov pri uvoľnení nových verzií operačného systému, aktualizácií, alebo služieb. V centrálnom helpdesku sa tiež buduje jednotná databáza znalostí, ktorá je obsiahlejšia ako podobné systémy spravované jednotlivo.

Od centrálnej správy je potrebné rozlíšiť **lokálnu správu**, resp. podporu koncového užívateľa (End-User Support). Lokálna správa je dôležitá pre vnímanie kvality služieb u klientov. Od lokálnej podpory sa očakáva odpoveď na jednoduché otázky bez potreby zadávať všetko do helpdesku. Nevýhodou podpory na každom pracovisku sú zvýšené náklady. Lokálna podpora by mala patriť pod centrálnych správcov kvôli informovanosti, zmenám, postupom a jednotnému poskytovaniu služieb.

1.6.3 Bezpečnosť

Pre zabezpečenie bezpečnosti v rámci organizácie je odporúčaný centralizovaný model. Bezpečnosť na pracovisku zahŕňa definovanie globálnych pravidiel dovoľujúcich ovládanie aktív a predstavenie technologických riešení dovoľujúcich zaviesť tieto pravidlá. Nekonzistentná bezpečnosť a nedostatočné mechanizmy na prístup k citlivým údajom môžu spôsobiť narušenie bezpečnosti. Aby bola efektívna , musia byť nastavené celostne a dobre koordinované celou organizáciou. Tabuľka č. 1 zobrazuje zoznam bezpečnostných služieb, ktoré je vhodné centralizovať.

Tab. 1 Bezpečnostné komponenty centrálnej správy [2].

Bezpečnostný komponent	Doporučený prístup
Autentifikácia	Výlučne centralizovaná služba
Autorizácia	Centralizovaná platforma
Prístup na internet	Výlučne centralizovaná služba
Vzdialený prístup	Výlučne centralizovaná služba
Externé pripojenia	Centralizovaná platforma
Nove akvizície	Výlučne centralizovaná služba
Zistenie škodlivého kódu	Výlučne centralizovaná služba

Ochranu úniku údajov	Výlučne centralizovaná služba alebo centralizovaná s decentralizovanou delegáciou

1.6.4 Repozitáre operačných systémov a softvéru

Softvérový repozitár [2] je systém, ktorý uchováva softvérové balíčky a sprístupňuje ich pre inštaláciu klientom. **Softvérové balíčky** obsahujú všetky súbory súvisiace s aplikáciami, ich inštaláciou a ich odstránením. Toto zahŕňa spustiteľné súbory, skripty súvisiace s inštaláciou, údaje, konfiguračné súbory, dokumentáciu a všetko ostatné, čo je súčasťou aplikácie. Balíčky sú prevažne pripravené na spustenie, predkompilované. Výhodou repozitárov je, že ponúkajú širokú paletu nástrojov, čím používateľovi šetria čas pri hľadaní konkrétneho softvéru. Softvérové balíčky obsahujú aj popisy a znalosti, ktoré používatelia nemusia poznať, napríklad softvérové pre rekvizity. Softvérový repozitár sa skladá z troch častí, a to zo servera, klienta a softvérových balíčkov [2].

Softvérové balíčky sú uložené na serveri a ten ich ďalej sprístupňuje klientom. Klienti sú zariadenia (napr. pracovné stanice), ktoré využívajú službu. Balíčky sú základnou jednotkou výmeny. Softvérové balíčky môžu mať rôzne formáty, ktoré sú na seba úzko naviazané, napríklad **Yellowdog Updater Modified (Yum)** repozitáre ponúkajú len RPM balíčky. Na druhej strane **Debian's Advanced Package Tool (APT)** repozitáre ponúkajú len DEB balíčky. Dôležité je na tomto mieste spomenúť, že operačné systémy sú úzko previazané s používaním konkrétnych repozitárov a softvérových balíčkov. YUM je používaný na operačnom systéme SUSE Linux alebo RedHat-e. APT sa používa na operačnom systéme Debian alebo Ubuntu. V rámci operačných systémov Windows sa používajú napríklad repozitáre Chocolatey [11].

Repozitár nie je len miesto, kde sú uložené softvérové balíčky, ale **aj služba**. Ako by každá služba mala byť naplánovaná,, monitorovaná a mala by mať záložný postup v prípade poruchy. V jednoduchosti sa dajú repozitáre predstaviť ako adresáre, ktoré obsahujú balíčky, indexy a iné metadáta, aby prebehli prenosy rýchlejšie. K repozitárom sa obvykle pristupuje cez protokoly HTTP alebo FTP. Môžu byť prístupné aj cez sieťový súborový systém ako NFS alebo CIFS.

2 Prístupy k centrálnej správe pracovných staníc

Cieľom tejto kapitoly je predstaviť si niekoľko existujúcich dostupných prístupov, resp. nástrojov. V rámci kapitoly si tieto systémy porovnáme z pohľadu troch kľúčových prvkov centrálnej správy pracovných staníc, a to vzdialenej správy, zálohovania a vytvárania a obnovy obrazov.

2.1 Porovnanie nástrojov k vzdialenej správe

Neodmysliteľnou súčasťou centrálnej správy pracovných staníc je vzdialená správa, resp. vykonávanie príkazov vzdialene. Vďaka nástrojom na centrálnu správu pracovných staníc, je možné automatizovať množstvo úloh, ktoré administrátori robia manuálne. V rámci porovnania sme sa zamerali na štyri najpoužívanejšie nástroje, a to Ansible [12], Chef [14], Puppet [15] a SaltStack [16].

2.1.1 Ansible

Ansible [12] je jednoduchý nástroj, ktorý automatizuje manažment konfigurácií zariadení, rozvoj aplikácií, orchestráciu a ďalšie. Ansible modeluje infraštruktúru popisovaním, aký je vzájomný vzťah medzi systémami, než by v danom čase len manažoval jeden systém. Tento nástroj nepoužíva žiadnych agentov a taktiež žiadnu prídavnú infraštruktúru. Z tohto dôvodu je ľahký na rozvíjanie. Navyše používa jednoduchý jazyk (YAML, vo forme Ansible Playbooks), ktorý umožňuje popisovať automatizovanú prácu.

Ansible bol vyvinutý, aby zjednodušil komplexnú orchestráciu, a tiež aj centrálnu správu. Platforma je písaná v programovacom jazyku Python. Ponúka veľa modelov, ako poslať príkazové moduly do zariadení (uzlov) cez protokoly SSH, HTTP, HTTPS. Príkazy sú vykonané sekvenčne. Tento nástroj sa snaží integrovať politiku bezpečnosti a dodržiavania pravidiel do automatizovaných procesov.

Výhodou nástroja Ansible je jeho jednoduchosť, ktorá sa prejavuje jednoduchou a rýchlou správou zariadení. Prostredníctvom tohto nástroja je možné zdieľať informácie medzi viacerými zariadeniami [13]. Inštalácia a počiatočné nastavenia sú jednoduché. Pochopiť syntax a pracovný postup tohto nástroja je pre nových používateľov pomerne triviálne. Tento nástroj využíva postupné vykonávanie procesov. Podporuje tiež „push“

aj „pull“ modely. Vďaka tomu, že Ansible funguje cez protokoly SSH a HTTPS, je zabezpečená vysoká bezpečnosť. Vďaka vývoju bez agenta je komunikácia omnoho rýchlejšia ako s agentom.

Na druhej strane, nevýhodou nástroja Ansible je zvýšený dôraz na orchestráciu, ktorý ide na úkor konfigurácie správy. SSH komunikácia je pomalšia v niektorých prispôsobených prostrediach. Navyše Ansible vyžaduje super-používateľa pri SSH prístupe a administrátorské oprávnenia pri prístupe na operačný systém Windows. Naviac grafické rozhranie nie je dostatočne rozvinuté a má len limitované funkcie.

2.1.2 Chef

Chef [14] je nástroj s otvoreným zdrojovým kódom, ktorý je založený na programovacom jazyku Ruby. Tento nástroj, podobne ako Ansible, je možné použiť na centralizovanú správu pracovných staníc. Tento nástroj používa klient-server architektúru a ponúka konfiguráciu v Ruby DSL. Architektúra používa prístup založený na „pull“ konfigurácii. Agenti získavajú informácie z hlavných serverov, ktoré komunikujú prostredníctvom protokolu SSH. Chef ponúka množstvo funkcií, napríklad automatizáciu cloudových riešení, automatizáciu vývoja aplikácií a pod. Navyše je možné tento nástroj použiť na správu bezpečnosti systému.

Výhodou tohto nástroja je flexibilita vo výbere operačných systémov. Na webovom sídle tohto nástroja je k dispozícii veľmi dobre spracovaná dokumentácia, podpora a mnoho príspevkov od aktívnej komunity. Prostredie je dizajnované pre programátorov. Navyše je tento nástroj stabilný a spoľahlivý, a to hlavne pre veľké projekty vo verejnom aj súkromnom prostredí. Podobne ako Ansible využíva postupné vykonávanie procesov [5].

Nevýhodou tohto nástroja sú inicializačné nastavenia. Inštalácia nástroja je pomerne komplikovaná. Okrem toho vyžaduje rýchle učenie sa a obťažnosť narastá každým krokom. Chýba „push“ model, čo znamená, že tento nástroj nepozná okamžité zmeny. Dokumentácia je príliš rozšírená a môže vzniknúť problém orientovať sa v nej.

2.1.3 Puppet

Puppet [15] je ďalším zástupcom nástrojov, ktorých úlohou je automatizácia konfigurácie a vývoj orchestrácie pre distribuované aplikácie a infraštruktúru (napr.

pracovné stanice). Produkt pôvodne vyvinul Luke Kanies za účelom automatizovania úloh pre systémových administrátorov, aby nestrávili roky konfigurovaním, poskytovaním služieb, riešením problémov a udržiavaním operácií serverov.

Puppet predstavuje riešenie s otvoreným zdrojovým kódom na konfiguráciu správy. Je založený na programovacom jazyku Ruby a ponúka DSL a Embedded Ruby šablóny na vytvorenie vlastných Puppet jazykových súborov. Ponúka tiež deklaratívny spôsob písania kódu. Puppet používa klient-server (agent-master) architektúru. Agenti manažujú uzly a požadujú relevantné informácie od mastrov, ktorí kontrolujú konfiguračné informácie. Používatelia si tiež môžu nastaviť „master-less“ mód a decentralizovanú Puppet inštaláciu.

Rozšírením tohto nástroja je Puppet Enterprise nástroj, ktorý ponúka napríklad orchestráciu, automatické poskytovanie služieb, automatizovanú konfiguráciu, vizualizáciu a nahlasovanie. Ďalšou možnosťou je spravovať kód a uzly. Prístup k tomuto nástroju je založený na roliach.

Výhodou nástroja Puppet sú nástroje na nahlasovanie. Na podporu okolo vývojových nástrojov slúži aktívna komunita. Poskytuje intuitívne webové rozhranie, ktorého úlohou je správa uzlov v reálnom čase. Tento nástroj má schopnosť pracovať s konštrukciami na úrovni príkazového riadku. Prvotné nastavenia sa dajú nastaviť plynulo a podporujú rôzne operačné systémy. Puppet je vhodný na riadenie heterogénnej infraštruktúry (napr. pracovné stanice s rôznymi operačnými systémami).

Nevýhodou nástroja Puppet je jeho počiatočná obťažnosť pre nových používateľov. Puppet DSL alebo Ruby môžu predstavovať prekážku pre prvotné nasadenie. Navyše pokročilé úlohy vyžadujú zvyčajne vstup z príkazového riadka. Inštalačným procesom chýba schopnosť adekvátne nahlasovať chyby. DSL kód môže narásť do veľkých rozmerov a môže sa stať zbytočne komplikovaným. Používaním viacerých masterov sa môže skomplikovať proces správy, a teda sa vzdialené vykonávanie príkazov môže stať výzvou.

2.1.4 SaltStack

SaltStack [16] predstavuje silný nástroj na správu vzdialeného vykonávania príkazov, ktorý môže byť efektívne a rýchlo použitý na administráciu serverov alebo pracovných staníc. SaltStack bol navrhnutý tak, aby povolil nízku latenciu a vysokú

rýchlosť komunikácie na zber údajov a vzdialené vykonávanie v administrátorskom prostredí. Platforma je napísaná v Pythone a používa „push“ model na vykonávanie príkazov cez SSH protokol. Salt povoľuje paralelné vykonávanie viacerých príkazov šifrovaných cez kryptografický protokol AES a ponúka vertikálne aj horizontálne škálovanie. Jediný master môže spravovať viac mastrov a peer rozhranie povoľuje používateľom kontrolovať agentov (minions). Táto platforma podporuje master-agent, decentralizáciu a taktiež aj non-master modely. Podobne ako pri nástroji Ansible, používatelia môžu písať skripty používaním YAML šablón. Vstavaný systém na vzdialené vykonávanie spracúva úlohy sekvenčne.

Výhodou tohto systému je efektivita vďaka vysokej škálovateľnosti a odolnému prostrediu. Po úvodnej inštalácii je používanie SaltStack veľmi jednoduché. Má aktívnu komunitu a podporu, konzistentnú YAML syntax pri všetkých skriptovacích úlohách. Navyše programovací jazyk Python je na naučenie pre vývojárov jednoduchý.

Nevýhodou tohto systému je nie príliš triviálna úvodná inštalácia pre nových používateľov. Dokumentácia nie je veľmi dobre spracovaná a je pomerne ťažké sa v nej orientovať. Webové rozhranie ponúka len limitované množstvo funkcií. Salt je vhodnou voľbou hlavne pre operačný systém Linux, pre ostatné operačné systémy nie je celkom vyvinutý.

2.1.5 Zhrnutie

V predchádzajúcej kapitole sme sa venovali popisu, výhodám a nevýhodám niektorých nástrojov, pomocou ktorých môžeme centrálne spravovať pracovné stanice. Pred ich samotným porovnaním je potrebné si určiť kritéria, na základe ktorých ich budeme porovnávať a ktoré určia, ktoré riešenie je najvhodnejšie pre našu infraštruktúru a prostredie. Vzhľadom na špecifiká akademického prostredia sme si určili nasledujúce vlastnosti, podľa ktorých porovnáme vyššie uvedené nástroje: náročnosť inštalácie, náročnosť samotnej správy, programovací jazyk, v ktorom je nástroj napísaný, spôsob písania kódu pre správu, podporované operačné systémy klienta a architektúra celého systému s daným nástrojom. Pri porovnaní vychádzame z porovnaní [5], ktoré dopĺňame o svoje postrehy a skúsenosti s týmito nástrojmi. Porovnanie nástrojov je zobrazené v Tabuľke č. 2.

Tab. 2 Porovnanie nástrojov k vzdialenej správe.

	Ansible	**Chef**	**Puppet**	**SaltStack**
Open Source/ proprietárny	Open Source	Open Source	Open Source	Open Source
Náročnosť inštalácie	Jednoduchá	Zložitá	Zložitá	Priemerná
Náročnosť správy	Jednoduchá	Jednoduchá	Zložitá	Zložitá
Jazyk konfigurácie	Python	Python	Puppet DSL	Ruby DSL
Typ konfigurácie	Push	pull	pull	push
Spôsob písania kódu	Procedurálny	Procedurálny	Deklaratívny	Deklaratívny
OS klienta	Win/Lin/Mac	Win/Lin/Mac	Win/Lin/Mac	Win/Lin/Mac
Architektúra	Klient	Klient/server	Klient/server	Klient/server

Všetky štyri porovnávané nástroje sú s **otvoreným zdrojovým kódom**, čo znamená žiadne vstupné náklady pre nasadenie týchto nástrojov, možnosť úpravy zdrojového kódu pre potreby organizácie a možnosť kontroly nástroja a jeho činnosti.

Náročnosť inštalácie a náročnosť správy predstavujú pomerne subjektívne prvky. Z tohto dôvodu sme urobili prieskum v rámci názorov na rôznych fórach a porovnali to so svojimi skúsenosťami. Najjednoduchším nástrojom z pohľadu inštalácie a správy je Ansible, čo je výhodné najmä pre nových používateľov. Na druhej strane, najzložitejším nástrojom z pohľadu inštalácie a správy je Puppet. Zvyšné nástroje sa nachádzajú z pohľadu obťažnosti medzi týmito nástrojmi.

Z pohľadu **jazyka konfigurácie**, nástroj Chef používa na manažovanie konfigurácií Ruby DSL. To si vyžaduje programátorské zručnosti. Klient získava konfigurácie zo Servera. Puppet používa vlastný programovací jazyk nazývaný Puppet DSL. V prípade týchto nástrojov nie je veľmi jednoduché spravovať takéto konfigurácie. Na druhej strane, Ansible používa YAML (Yet Another Markup Language), ktorý sa podobá na angličtinu, preto je veľmi jednoduchý na spravovanie. Server posiela konfigurácie všetkým uzlom, čo je výhodné pre aplikácie v reálnom čase alebo pre

používané pracovné stanice. Vzdialené vykonávanie príkazov, resp. činností, sa uskutoční okamžite. SaltStack je veľmi podobný Ansible, keďže tiež používa YAML.

Ďalšou vlastnosťou týchto nástrojov je **typ konfigurácie**. Nástroje Puppet a Chef využívajú „pull“ konfiguráciu a Ansible a SaltStack využívajú „push“ konfiguráciu. „Push“ konfigurácia je taká, kde sú všetky konfigurácie z centrálneho servera posielané do uzlov, zatiaľ čo pri „pull“ konfigurácii uzly automaticky získavajú konfigurácie z centrálneho servera bez akýchkoľvek príkazov.

Nástroje Chef a Ansible podporujú procedurálny **štýl kódu**, v rámci ktorého sa krok po kroku špecifikuje, ako dosiahnuť požadovaný koncový stav. Na druhej strane, SaltStack a Puppet podporujú deklaratívny štýl kódu, v rámci ktorého sa naopak špecifikuje koncový stav. Nástroj je zodpovedný za to, ako sa tento stav dosiahne, čo môže byť nevýhodou. Výhodou procedurálneho kódu je, že správca má plnú kontrolu nad písaním a prispôsobovaním kódu. Zvolením postupnosti krokov vieme ovplyvniť aj konečný výsledok.

Dôležitou vlastnosťou nástroja na centrálnu správu pracovných staníc je **podpora operačného systému** pracovnej stanice. Z tohto pohľadu všetky nástroje podporujú hlavné vetvy operačných systémov, a to Windows, Linux a Mac OS.

Z pohľadu **architektúry** nástroje Chef, Puppet a SaltStack štandardne používajú klient/server architektúru. Klient v tejto architektúre môže byť webové rozhranie alebo príkazový riadok, ktorý vydáva rozkazy. Tieto príkazy smerujú na server, ktorý je zodpovedný za ich vykonanie a za uloženie stavu systému. Pri tejto architektúre je potrebné nainštalovať špeciálny softvér na každom serveri, udržiavať ho, aktualizovať, vytvárať zálohy a pod. Na druhej strane, Ansible používa len klient architektúru. Ansible klient je priamo pripojený na zariadenie, kde sa zadávajú príkazy na vykonanie. V rámci použitia sa využíva protokol SSH, HTTP alebo HTTPS. Klient architektúra predstavuje najlepšie riešenie pre naše prostredie, keďže nie je potrebné na každú pracovnú stanicu inštalovať klienta.

Vzhľadom na to, že akademické prostredie vyžaduje nízke vstupné a prevádzkové náklady, jednoduchosť inštalácie a správy, podporu najrozšírenejších typov operačných systémov pre pracovné stanice, ako aj rýchlosť vykonaných zmien a architektúru bez nutnosti inštalácie klienta, sme sa rozhodli pre nástroj Ansible.

2.2 Porovnanie nástrojov k zálohovaniu a obnove údajov

Ďalšou súčasťou centrálnej správy pracovných staníc je zálohovanie údajov a ich obnova. Zálohovanie údajov patrí súčasne k základným prvkom prevádzkovej bezpečnosti organizácie. V rámci porovnania sme sa zamerali na päť najpoužívanejších nástrojov, a to Amanda [17], BackupPC [18], Bacula [19], Bareos [20] a Rsync [21].

Amanda (Advanced Maryland Automatic Network Disk Archiver) [17] je prvým príkladom nástroja na zálohovanie a obnovenie údajov. Tento nástroj je dostupný s otvoreným zdrojovým kódom. Je schopný bežať na zariadeniach s rôznymi operačnými systémami, ako napr. Linux, UNIX, BSD, Mac OS-X a Microsoft Windows. Rozoznávame tri edície tohto nástroja – Amanda Community Edition, ktorá je zadarmo, Amanda Enterprise Edition, ktorá podporuje zálohovanie živých (live) aplikácií a databáz a posledná je Zmanda Backup Appliance, ktorá zálohuje celú sieť systémov a aplikácií.

BackupPC [18] je nástroj s otvoreným zdrojovým kódom, ktorý poskytuje vysokovýkonný systém na zálohovanie zariadení s operačným systémom Linux, Windows a Mac. Používa kompresiu a deduplikáciu záloh na minimalizácii miesta na disku.

Bacula [19] poskytuje kompletné riešenie zálohovania. Je to súbor nástrojov s otvoreným zdrojovým kódom, ktoré nám umožňujú spravovať zálohovanie, zotavenie, komprimáciu, šifrovanie a verifikáciu údajov, ktoré prechádzajú sieťou počítačov rôzneho druhu. Tento nástroj nie je relatívne jednoduchý na použitie, ale poskytuje možnosti, ktoré po prvotnej náročnejšej konfigurácii a odladení vyvážili cenu náročnejšej inštalácie. Bacula ponúka aj pokročilé funkcie, ktoré uľahčujú nájdenie a zotavenie stratených alebo poškodených súborov. Je ponúkaný v dvoch verziách, a to enterprise a community.

Bareos (Backup Archiving Recovery Open Sourced) [20] je nástroj s otvoreným kódom, ktorý je odvodený od nástroja Baculy. Beží na zariadení a je schopný odzálohovať rôzne médiá zahŕňajúc pásky a disky. Bareos povoľuje IT administrátorom spravovať zálohy, obnovenia záloh a verifikácie, podobne ako Bacula.

Rsync [21] je nástroj primárne určený pre operačný systém Unix/Linux, ktorý synchronizuje súbory a adresáre z jednej lokality do inej. Je schopný kopírovať alebo zobraziť obsah adresárov a kopírovať súbory. Pri týchto činnostiach je možné používať kompresiu a rekurziu. V režime démona, rsync počúva štandardne na TCP porte 873, ktorý slúži na preberanie súborov v natívnom protokole rsync alebo cez vzdialený shell

(RSH alebo SSH). V inom prípade musí byť spustiteľný súbor klienta nainštalovaný na lokálnom aj vzdialenom zariadení.

Vyššie sme si stručne uviedli a popísali nástroje, ktoré sa používajú na zálohu a obnovenie údajov. Na to, aby sme vedeli určiť, ktorý z nich je najvhodnejší pre nás, navrhli sme niekoľko triediacich parametrov, a to, či ide o proprietárny nástroj alebo nástroj s otvoreným zdrojovým kódom, náročnosť inštalácie, podpora najčastejšie používaných súborových systémov, podpora operačného systému pracovnej stanice a architektúra. Tabuľka č. 3 zobrazuje výsledky porovnania.

Tab. 3 Porovnanie nástrojov k zálohovaniu a obnove údajov

	Bacula	**Amanda**	**Bareos**	**BackupPC**	**Rsync**
Open Source/ proprietárny	Open Source + platená verzia	Open Source	Open Source	Open Source	Open Source
Náročnosť inštalácie a	Náročná	Priemerná	Náročná	Priemerná	Jednoduchá
Typ súborového systému	Ext3/Ext4/ Fat32/NTFS/ APFS	Ext3/Ext4/ Fat32/NTFS/ APFS	Ext3/Ext4/ Fat32/NTFS/ APFS	Ext3/Ext4/ Fat32/NTFS/ APFS	Ext3/Ext4/ Fat32/NTFS/ APFS
Možnosti automatizácie	Rozsiahle, zabudované		Rozsiahle, zabudované		Nutné si všetok kód napísať
OS klienta	Win/Lin/Mac	Win/Lin/Mac	Win/Lin/Mac	Win/Lin/Mac	Win/Lin/ Mac
Zálohovacie médium	Disky, Pásky, Optické médiá	Disky, Pásky, Optické médiá	Disky, Pásky, Optické médiá	Disky, Pásky, Optické médiá	Disk
Architektúra	Klient, riadiaci server a storage server	Klient/ server	Klient, riadiaci server a storage server	Klient/ server	Klient/ server

Z pohľadu finančnej dostupnosti a možnosti úpravy zdrojového kódu, preferujeme nástroje s **otvoreným kódom** (open source) pred proprietárnym softvérom. Všetky vyššie spomínané nástroje túto podmienku spĺňajú.

Z pohľadu **náročnosti inštalácie**, samotná inštalácia Baculy zahŕňa inštaláciu balíčkov a databázy na serverovej časti, inštaláciu klienta a konfiguráciu na strane servera a klienta. Štandardne tento nástroj používa databázu sqlite, ale pre vyšší výkon sa odporúča používať databázu mysql. Prvotné nastavenie Baculy nie je veľmi jednoduché, nakoľko je potrebné inštalovať ju na viacerých serveroch a vykonať prvotné nastavenia. Nástroj Amanda je dostupný v repozitároch väčších distribúcií. Balíčky je potrebné nainštalovať na oboch stranách, teda na strane servera aj klienta. Pred inštaláciou BackupPC je odporúčané najprv nainštalovať niekoľko modulov, ktoré môžu uľahčiť prácu. Samotná inštalácia tohto nástroja nie je veľmi zložitá. Použitie nástrojov Bareos a Rsync si nevyžaduje špeciálne vedomosti a skúsenosti.

Podpora súborového systému je úzko spätá s **podporou operačného systému** pracovnej stanice. Všetky nástroje podporujú najrozšírenejšie operačné systémy pre pracovné stanice, a to Windows, Linux, Mac a tiež súborové systémy Ext3, Ext4, NTFS, FAT32 a Apple File System (APFS).

Ďalšou sledovanou vlastnosťou nástrojov bola **architektúra**. Takmer všetky nástroje používajú minimálne klient/server architektúru. Bacula a Bareos majú dokonca trojstupňovú architektúru klient/server. Výnimkou je nástroj Rsync, ktorý môže mať klient/server architektúru, ale je schopný pracovať aj samostatne.

Pre navrhovaný centralizovaný systém sme si nakoniec vybrali nástroj Bacula, pre ktorý sme sa rozhodli až po otestovaní iných nástrojov, kvôli poslednému parametru - podpore páskovej knižnice, ktorou fakulta disponuje. Po otestovaní ďalších vybraných nástrojov sa Bacula ukázala ako jediná, ktorá vedela ovládať našu páskovú knižnicu, viesť evidenciu pások a spoľahlivo ovládať robotické rameno zodpovedné za výmenu pások.

2.3 Porovnanie nástrojov k vytvoreniu a obnove obrazov disku

Poslednou súčasťou centrálnej správy pracovných staníc, ktorá je nevyhnutná pre rýchle nasadenie prípravu klientskych staníc, je vytvorenie a obnova obrazov disku. V rámci porovnania sme sa zamerali na štyri najpoužívanejšie nástroje, a to Acronis True Image [22], Clonezilla [23], Fog [24] a Ghost [25].

Acronis True Image [22] je nástroj, ktorý poskytuje ochranu dát, zálohu, archivovanie, prístup a obnovu dát pre Microsoft, OSX, iOS a Android operačné systémy. True Image vie obnoviť predtým vytvorený obraz na nový disk, zreplikuje štruktúru a obsah na nový disk. Taktiež povoľuje klonovanie disku a zmenu veľkostí partícií, aj napriek tomu, že disk môže mať inú kapacitu. Zálohy sú vo vlastnom formáte „.tib“.

Clonezilla [23] je zálohovací a obnovovací softvér s otvorenou licenciou. Poznáme dve verzie Clonezilly – Clonezilla Live a Clonezilla Server Edition. Clonezilla Live je vhodná na zálohu a obnovu údajov z jedného zariadenia. Na druhej strane Clonezilla Server Edition môže klonovať 40 počítačov naraz. Clonezilla ukladá a obnovuje len použité bloky na pevnom disku, čo zvyšuje efektivitu klonovania. Obe verzie sú dostupné zadarmo a s otvoreným kódom. Clonezilla je písaná primárne v Perl programovacom jazyku a zdrojový kód sa dá nájsť na domovskej stránke projektu. Podporuje všetky súčasné súborové systémy a beží na Linuxových jadrách. Nezanedbateľnou výhodou je vytváranie gz archívov.

Fog (Free Opensource Ghost) [24] poskytuje webovo založený nástroj na klonovanie diskov využívajúci rôzne nástroje s otvoreným kódom. Používa PXE (Preboot eXecution Environment) štartovanie zariadenia na stiahnutie malého Linuxového klienta a vytvára obrazy na diaľku. Zdrojový kód tohto nástroja je napísaný v programovacom jazyku PHP a je dostupný na webovom sídle projektu.

Ghost [25] je komerčný nástroj od spoločnosti Symantec, ktorý je schopný klonovať obsah celého pevného disku na iný pevný disk alebo na pamäťové médium tak, že automaticky sformátuje a rozdelí nový disk. Tento produkt je užitočný, keď chceme jeden systém replikovať na množstvo počítačov, alebo keď niekto chce zálohovať všetko na svojom počítači. Ghost podporuje všetky súčasné súborové systémy.

Vyššie sme si stručne uviedli a popísali nástroje, ktoré sa používajú na vytvorenie a obnovenie údajov. Na to, aby sme vedeli určiť, ktorý z nich je najvhodnejší pre nás, navrhli sme niekoľko triediacich parametrov, a to či ide o proprietárny nástroj alebo nástroj s otvoreným zdrojovým kódom, podpora súborových systémov, kompresia, rýchlosť a napokon nároky na výkon infraštruktúry. Tabuľka č. 4 zobrazuje výsledky porovnania.

Tab. 4 Porovnanie nástrojov k vytvoreniu a obnove obrazov disku.

	Ghost	**Clonezilla**	**FOG**	**Acronis True Image**
Open Source / proprietárny	Platené	Open Source	Open Source	Platené
Podpora súborových systémov	Všetky	Všetky	Všetky	FAT, NTFS, EXT2/3/4, ReiserFS3/4, XFS, JFS
Kompresia	30%	35%	30%	30%
Rýchlosť	1,4 GB/s	3,1 GB/s	1,4 GB/s	4 GB/s
Nároky na výkon	nízke	nízke	nízke	Vysoké

Podobne, ako pri predchádzajúcom porovnaní, z pohľadu finančnej dostupnosti a možnosti úpravy zdrojového kódu, preferujeme nástroje s **otvoreným kódom** (open source) pred proprietárnym softvérom. Nástroje Ghost a Acronis True Image sú proprietárne nástroje. Na druhej strane, FOG a Clonezilla sú nástroje s otvoreným zdrojovým kódom. Výhodou tak je napríklad priebežne aktualizovaná podpora ovládačov nových sieťových kariet.

Z pohľadu **podpory súborových systémov**, väčšina nástrojov podporuje všetky súborové systémy s výnimkou Acronis True Image. Ten sa nezameral najmä na súborové systémy operačného systému Mac OS X.

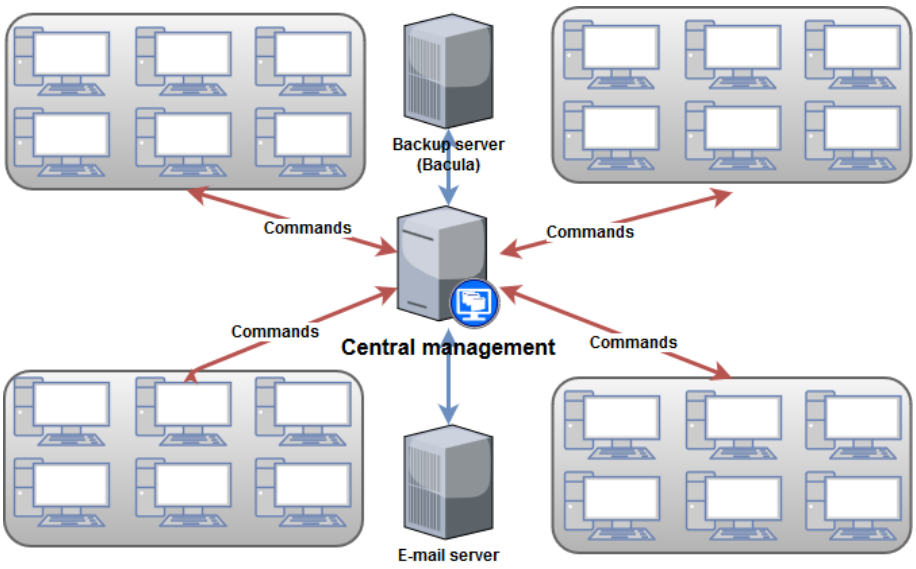
Čo sa týka kompresie zálohovaných údajov, sledované nástroje majú kompresný pomer cca 30% okrem nástroja Clonezilla, ktorý má kompresný pomer približne 35%. Najväčšiu prenosovú rýchlosť dokáže dosiahnúť Acronis True Image (4 GB/s), kým druhá v poradí Clonezilla má len cca 4 GB/s. Na druhej strane práve Acronis True Image má vysoké nároky na výkon. Z vyššie uvedených zistení sme si pre naše riešenie zvolili Clonezillu, keďže je to nástroj s otvoreným zdrojovým kódom, má nízke nároky na výkon a pomerne vysokú rýchlosť prenosu.

3 Návrh a implementácia systému pre centrálnu správu pracovných staníc

V rámci tejto kapitoly sa venujeme návrhu a implementácii parciálnych častí systému na centrálnu správu pracovných staníc. Kapitola nadväzuje na predchádzajúce dve kapitoly tejto záverečnej práce. V rámci tejto kapitoly sa venujeme všeobecnému návrhu systému a implementácii jednotlivých častí tohto systému.

3.1 Návrh systému

Nami navrhovaný systém pre centrálnu správu pracovných staníc je znázornený na obrázku č. 4. Systém pozostáva z troch hlavných častí, a to centrálnej správy, zálohovacieho servera a emailového servera. Úlohou centrálnej správy je manažment a vzdialené vykonávanie činností na pracovných staniciach. Na druhej strane, účelom zálohovacieho servera je poskytnutie centralizovanej služby, zálohy a obnovy údajov pracovných staníc. Súčasťou návrhu je už aj existujúci emailový server, ktorý je využívaný v prípade notifikácie administrátorovi.



Obr. 4 Schéma systému pre centrálnu správu pracovných staníc.

Pre návrh a implementáciu systému sme ako testovacie prostredie zvolili počítačové učebne Prírodovedeckej fakulty UPJŠ v Košiciach. Súčasťou testovacieho prostredia boli aj virtuálne servery s operačným systémom Ubuntu 18.04, na ktorých

bežia nástroje Ansible a Bacula, ktorým sme sa čiastočne venovali v druhej kapitole tejto práce, a ktorých činnosť bližšie predstavíme v nasledujúcich podkapitolách. Na vykonávanie činností v rámci pracovných staníc s operačným systémom Windows využívame Powershell vo verzii 5.1. Na niektoré činnosti potrebné v rámci nástrojov Ansible a Bacula používame programovací jazyk Python vo verzii 2.7.

3.2 Vzdialená správa pracovných staníc

Na vzdialenú správu pracovných staníc používame nástroj Ansible, ktorému sme sa bližšie venovali v časti venovanej porovnaniu nástrojov na centralizovanú správu. V rámci tejto kapitoly si priblížime použitie tohto nástroja pri správe pracovných staníc s operačným systémom. Tento nástroj natívne využíva Windows PowerShell a vďaka nemu je možné:

- zbierať fakty Windows host-a,
- inštalovať a odinštalovať MSI,
- povoliť a zakázať Windows funkcie,
- začať, ukončiť a manažovať služby operačného systému Windows,
- vytvoriť a manažovať lokálnych používateľov a skupiny.

3.2.1 Princíp playbookov

Playbooky sú v nástroji Ansible jazykom pre konfiguráciu, vývoj a orchestráciu. Popisujú politiku, ktorú chceme vynútiť vo vzdialených systémoch alebo množinu krokov v rámci procesov. Playbooky sú používané na spravovanie vývoja a nastavení vzdialených strojov. Na pokročilejšej úrovni môžu postupne spravovať aktualizácie, presúvať akcie na iné stroje, komunikovať s monitorovacími servermi a podobne. Sú čitateľné a vyjadrené v YAML formáte – snažia sa byť modelom konfigurácie alebo procesu (nie programovacím jazykom alebo skriptom).

Playbooky sú zložené zo scenárov (plays), a teda je možné spustiť isté kroky na všetkých pracovných staniciach v určitej skupine, a iné kroky napríklad pre pracovné stanice používané v učebniach. Môžeme mať zadefinovaných mnoho scenárov, ktoré ovplyvňujú systém a taktiež môžu byť spustené rôzne scenáre v rôznom čase.

3.2.2 Moduly pre Ansible

Moduly sú preddefinované samostatné časti systému určené na vykonávanie niektorých akcií. Z tohto dôvodu nie je potrebné ich opäť manuálne vytvárať. V rámci nášho centralizovaného systému používame tieto moduly:

- **win_chocolatey** – spravuje balíčky pomocou Chocolatey.
  Ak Chocolatey chýba, tento modul ho nainštaluje.
- **win_command** – vezme meno príkazu s nasledujúcim listom argumentov a vykoná ho na zvolených uzloch.
- **win_get_url** – sťahuje súbory z HTTP, HTTPS alebo FTP na vzdialený server. Vzdialený server musí mať priamy prístup na vzdialený zdroj.
- **win_firewall_rule** – povoľuje vytvoriť/odstrániť/aktualizovať pravidlá firewallu.
- **win_msg** – posiela správu prihláseným užívateľom na Windows host-och.
- **win_package** – nainštaluje alebo odinštaluje balíček MSI alebo EXE formátu. Balíčky môžu byť z lokálneho súborového systému, ale aj zo siete.
- **win_ping** – Windows verzia klasického „ping“ modulu. Kontroluje pripojenie host-u.
- **win_regedit** – pridáva, modifikuje alebo maže kľúče a hodnoty z registra.
- **win_service** – spravuje a dopytuje Windows služby.
- **win_shell** – vezme meno príkazu s nasledujúcim listom argumentov a vykoná ho na zvolených uzloch. Je to podobný modul ako win_command s tým rozdielom, že príkazy vykonáva cez „shell“.
- **win_stat** – získava a vracia informácie o Windows súboroch.
- **win_updates** – vyhľadáva, sťahuje a inštaluje Windows aktualizácie.
- **win_wakeonlan** – posiela „magic“ Wake-on-LAN broadcastové pakety.
- **win_whoami** – získava informácie o aktuálnom užívateľovi a procese.

3.2.3 WinRM služba

WinRM služba počúva požiadavky na jednom alebo viacerých portoch (štandardne port 5985 pre protokol HTTP a 5986 pre protokol HTTPS). Každý z týchto portov musí mať vytvoreného a nakonfigurovaného listener-a. Na pozretie aktuálnych listener-ov,

ktorí práve bežia na WinRM službe, je potrebné spustiť nasledujúci príkaz: `winrm enumerate winrm/config/Listener`. Príklad výstupu

```
Listener
    Address = *
    Transport = HTTPS
    Port = 5986
    Hostname = WORKSTATION
    Enabled = true
    URLPrefix = wsman
    CertificateThumbprint = 6BDD004030E0277EA011281E7F15C190513F1B85
    ListeningOn = 127.0.0.1, 169.25.31.19
```

3.2.4 Nastavenie pracovnej stanice s operačným systémom Windows 10

Aby Ansible mohol komunikovať s pracovnou stanicou s operačným systémom Windows a použiť Windows moduly, pracovná stanica musí spĺňať nasledujúce požiadavky:

- operačný systém Windows 7, 8.1 a 10,
- PowerShell 3.0 a novší,
- .NET 4.0.

Vyššie uvedené požiadavky tvoria základom k tomu, aby prebehlo pripojenia na pracovnú stanicu. Niektoré Ansible moduly vyžadujú dodatkové požiadavky, napríklad ako novší operačný systém alebo novšiu verziu PowerShell-u. K tomu, aby sme vedeli používať Ansible, nepostačuje len splnenie vyššie uvedených požiadaviek, nutné sú aj nasledujúce kroky:

- vytvorenie lokálneho používateľa na pracovnej stanici s operačným systémom Windows 10,
- priradenie administrátorských oprávnení pre daného lokálneho používateľa a
- vytvorenie a zapnutie WinRM služby.

Prvým krokom je na pracovnej stanici **vytvorenie používateľa** s administrátorskými oprávneniami. Na testovacie účely použijeme používateľa ansible s heslom ansible. Ak neexistuje lokálny používateľ s menom „ansible“, vytvoríme nové

konto s menom „ansible“ a zadáme aj heslo. Zároveň tohto užívateľa pridáme do skupiny administrátorov, aby mal všetky práva.

```
$Password = ConvertTo-SecureString "*******" -AsPlainText -Force
$op = Get-LocalUser | Where-Object {$_.Name -eq "ansible"}
if ( -not $op)
 {
      New-LocalUser "ansible" -Password $Password
      Add-LocalGroupMember -Group "Administrators" -Member "ansible"
 }
```

V rámci nášho testovacieho riešenia používame predvolený Powershell skript ConfigureRemotingForAnsible.ps1, ktorý postačí na inštaláciu základov. Tento skript nainštaluje HTTP a HTTPS služby s podpísaným certifikátom, a povolí základnú možnosť autentifikácie v službe. Na použitie tohto skriptu je potrebné použiť tieto príkazy v PowerShell-i:

```
$url                                                                    =
"https://raw.githubusercontent.com/ansible/ansible/devel/examples/scri
pts/ConfigureRemotingForAnsible.ps1"
$file = "$env:temp\ConfigureRemotingForAnsible.ps1"

(New-Object -TypeName System.Net.WebClient).DownloadFile($url, $file)

powershell.exe -ExecutionPolicy ByPass -File $file
```

Pri niektorých pracovných staniciach bude potrebné po skončení práce zrušiť používateľa a zakázať WinRM službu. Na kompletné zrušenie tejto služby je potrebné vymazať listener-ov, zastaviť službu WinRM a obnoviť hodnotu LocalAccountTokenFilterPolicy na 0. Nasledujúci skript vykonaná vyššie uvedené činnosti:

```
Disable-PSRemoting -Force
winrm delete winrm/config/listener?address=*+transport=HTTP
Stop-Service winrm
Set-Service -Name winrm -StartupType Disabled
```

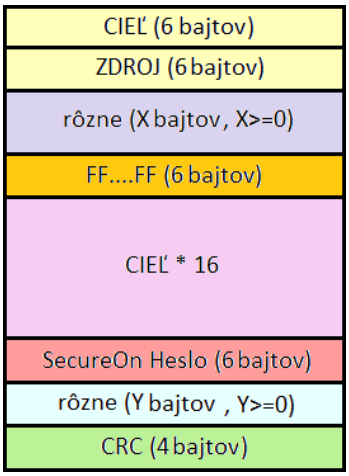
```
Set-ItemProperty -Path
HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System -Name
LocalAccountTokenFilterPolicy -Value 0 -Type DWord
```

Na to, aby sme vedeli použiť prvotné nastavenie pracovných staníc, použijeme repozitár Chocolatey [11]. Najprv je potrebné nainštalovať a aktualizovať Chocolatey. To urobíme pomocou modulu **win_chocolatey**. Potom prostredníctvom neho nainštalujeme základné softvérové vybavenie pre novú pracovnú stanicu, a to Adobe Reader, webové prehliadače – Google Chrome a Mozilla Firefox, VLC Media Player a 7Zip. Nižšie je uvedený skript pre prvotnú inštaláciu softvéru pracovných staníc:

```
Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object
System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))
choco upgrade chocolatey
choco install adobereader
choco install googlechrome
choco install firefox
choco install vlc
choco install 7zip
```

3.3 Vzdialené zapnutie pracovných staníc

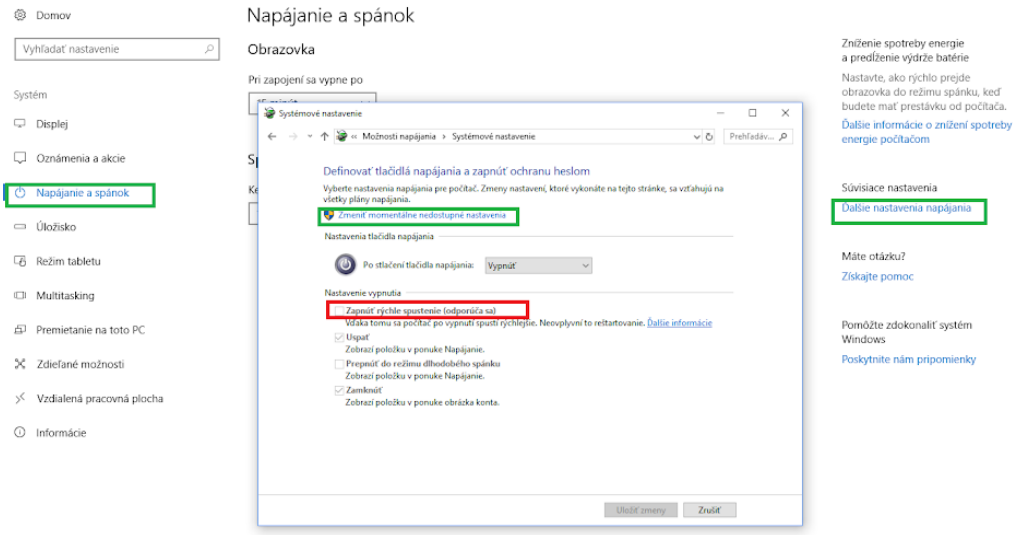
Veľmi dôležitým aspektom pre centralizovanú a automatizovanú správu je vzdialené zapnutie pracovných staníc. Vo väčšine prípadov je potrebné, aby sa pracovné stanice „zobudili“ v presne definovaný čas. Následne sa na týchto pracovných staniciach môže vykonať plánovaná činnosť (napr. update softvéru a operačného systému). **Prebúdzanie po počítačovej sieti** (wake on lan) je spôsob, ako špeciálnym paketom (tzv „Magic packet“) prebudiť pracovnú stanicu, ktorá je vypnutá. **Magický paket** je zobrazený na obrázku č. 5. Ide o špeciálny broadcastový paket typu UDP. Pri vypnutom počítači nie je k dispozícii IP adresa, a špeciálny paket musí byť typu broadcast, aby zastihol cieľový počítač v počítačovej sieti. Vypnutý počítač nemôže odpovedať na TCP paket, preto sa používa UDP paket.



Obr. 5 Magický paket.

Ak sieťová karta deteguje takýto paket a daný paket obsahuje jej MAC adresu, tak sa pracovná stanica zapne. Pre správne fungovanie vzdialeného zapnutia, musí túto technológiu podporovať sieťová karta, základná doska pracovnej stanice s biosom. Pri prebúdzaní pracovnej stanice dochádza k jej zobudeniu z úsporného stavu. Toto správanie sa menilo od Windows 7 až po Windows 10. Od Windows 8 je podporované hybridné vypnutie, ktoré zastaví reláciu používateľa, ale povolí obsah relácie jadra operačného systému zapísať na disk, čo umožňuje rýchlejšie zapínanie.

Problémom, ktorý môže nastať pri nových verziách operačného systému Windows 10 je, že majú zapnutú funkciu rýchleho spustenia. Túto funkciu je potrebné vypnúť. V opačnom prípade nie je možné zapnúť pracovnú stanicu. Ukážka nastavení pracovnej stanice s operačným systémom Windows 10 je zobrazená na obrázku č. 6.



Obr. 6 Sieťové nastavenia Windows 10 potrebné k prebudeniu pracovnej stanice po počítačovej sieti

K vzdialenému zapnutiu pracovných staníc môžeme použiť niekoľko dostupných nástrojov. Keďže v našom riešení využívame nástroj Ansible, budeme na vzdialené zapnutie využívať tento nástroj a jeho modul **win_wakeonlan**. Playbook pre zapnutie pracovných staníc s MAC adresami `ac:16:2d:00:6a:00` a `ac:16:2d:00:6a:01` v rámci počítačovej siete s broadcastovou adresou `192.168.0.255`, by mohol vyzerať nasledovne:

```
- name: wake on lan
  hosts: win
  tasks:
  - name: Send a magic Wake-on-LAN packet to ac:16:2d:00:6a:00
    win_wakeonlan:
      mac: ac:16:2d:00:6a:00
      broadcast: 192.168.0.255
  - name: Send a magic Wake-on-LAN packet to ac:16:2d:00:6a:01
    win_wakeonlan:
      mac: ac:16:2d:00:6a:01
      broadcast: 192.168.0.255
```

3.4 Automatizácia niektorých činností

Pre centrálnu správu pracovných staníc je potrebné vykonať niekoľko jednoduchých krokov, bez ktorých by nebolo možné dokončiť prvotnú inštaláciu pracovných staníc. Týmito jednoduchými činnosťami sú:

- Aktivácia Windows 10 a Office 2016
- Zistenie MAC adresy a IP adresy pracovnej stanice
- Aktualizácia softvérového vybavenia od spoločnosti Microsoft

3.4.1 Aktivácia Windows 10 a Office 2016

Dôležitým krokom pri vzdialenej správe pracovných staníc je aktivácia licencií Windows 10 a Office 2016. Pomocou nástroja (slmgr.vbs) môžeme spravovať softvérové licencie, a teda aktivovať Windows zadaním jedného príkazu. Zároveň pomocou nástroja

na zabezpečenie balíka Office (cscript) vložíme licenčný kľúč a následne Office aktivujeme.

```
slmgr.vbs /ipk *****-*****-*****-*****-*****
cd "C:\Program Files\Microsoft Office\Office16"
cscript ospp.vbs /inpkey: *****-*****-*****-*****-*****
cscript ospp.vbs /act
```

3.4.2 Zistenie MAC adresy a IP adresy pracovnej stanice

Dôležitým nástrojom pri centrálnej správe je zistenie MAC adresy a IP adresy pracovnej stanice. Tieto údaje získame pomocou príkazu Get-NetAdapter, ktorý nám poskytne základné informácie o sieťovej karte a vypíšeme MAC adresu. Ďalší príkaz (Win32_NetworkAdapterConfiguration) slúži na zisk IP adresy. Celý skript je zobrazený nižšie:

```
Get-NetAdapter |  Where-Object {$_.Name -eq "Ethernet"} | Select-
Object -ExpandProperty MacAddress
$ip=get-WmiObject Win32_NetworkAdapterConfiguration|Where
{$_.Ipaddress.length -gt 1}
$ip.ipaddress[0]
```

3.4.3 Aktualizácia softvérového vybavenia od spoločnosti Microsoft

Ďalším dôležitým prvkom pri centrálnej správe je aktualizácia softvérového vybavenia od spoločnosti Microsoft. Zväčša to budú aktualizácie samotného operačného systému, .Net prostredia, kancelárskeho balíka Office a pod. Pred inštaláciou samotného modulu `PSWindowsUpdate` najprv nainštalujeme poskytovateľa balíkov `NuGet`. Po spustení `PSWindowsUpdate` následne zaregistrujeme nového Windows Update manažéra. Aktualizujeme všetky produkty od Microsoft-u a vypneme počítač. Ukážka tohto skriptu je uvedená nižšie:

```
Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
Install-Module PSWindowsUpdate -Force
Get-Command -module PSWindowsUpdate
```

```
Add-WUServiceManager -ServiceID 7971f918-a847-4430-9279-4a52d1efe18d -
Confirm:$false
Get-WUInstall –MicrosoftUpdate –AcceptAll –IgnoreReboot -Install
Stop-Computer
```

3.4.4 Získavanie informácii o diskovej kapacite pracovnej stanice

V prílohe D sa nachádza PowerShellovský skript, ktorý získa v cykle potrebné údaje o pracovnej stanici (písmeno disku, voľné miesto na disku, celkovú kapacitu disku) a následne IP adresu klientskej stanice. Tieto údaje neskôr budeme chcieť zapísať do databázy.

Pomocou Powershell skriptu z prílohy E je možné jednoducho získať údaje z databázy. Jeho úpravou bude možné zapísať údaje o pracovnej stanici získané predchádzajúcim skriptom do databázy.

Python skript v prílohe F kontroluje databázu, v ktorej sa nachádzajú dáta o voľnom priestore na diskoch pracovných staníc. Skript zráta koľko percent voľného priestoru sa nachádza na disku a ak voľný priestor klesne pod 15% tak odošle email na požadovanú adresu s informáciou o pracovných staniciach s nedostatkom voľného priestoru na disku.

Spojením predchádzajúcich troch skriptov získavame jednoduchý a účinný spôsob získavania informácií o stave diskového úložiska na pracovných staniciach.

3.5 Zálohovanie a obnova údajov z pracovných staníc

Na základe porovnania riešení zálohovania, ktoré sme opísali v kapitole 2.2. Porovnanie nástrojov k zálohovaniu a obnove údajov“ sme sa rozhodli, že náš systém bude využívať Baculu. Ako bolo vyššie spomenuté, Bacula je open source zálohovacie riešenie, ktoré umožňuje spravovať zálohy, obnovovať a verifikovať dáta po sieti. Bacula podporuje nasledujúce typy zálohovania:

- **plná záloha** – je bezpečná úplná záloha definovaných súborov, najnáročnejšia vzhľadom na výkon, no najjednoduchšia a najrýchlejšia na obnovu.
- **inkrementálna záloha** – je záloha súborov, v ktorých nastali zmeny od poslednej zálohy. Je menej náročná na výkon, ale po čase môže byť problém manažovať obnovenie z inkrementálnych záloh.

- **diferenčná záloha** – obsahuje všetky kombinované zmeny súborov od poslednej plnej zálohy. Diferenčné zálohy môžu značne znížiť dobu obnovy zálohy, pretože nebudú potrebné všetky inkrementálne a plná záloha, ale len posledná plná záloha, jedna diferenčná a možno niektoré inkrementálne zálohy.

V našom systéme máme vytvorené dva plány na zálohovanie. Prvý je taký, kde zálohovanie nemôže byť vykonávané v pracovný deň v pracovnom čase. Plná záloha sa teda vykonáva prvú nedeľu v mesiaci o 23:05. Diferenčná sa vykonáva druhú až piatu nedeľu v mesiaci, tiež o 23:05, a inkrementálna sa vykonáva každý pondelok až sobotu o 23:05.

Druhý plán na zálohovanie je taký, ktorý môže bežať kedykoľvek. Nie každý server má veľa dát, a teda zálohovanie môže byť hotové za pár minút. Plná záloha sa vykonáva prvý deň v mesiaci, diferenčná druhú až piatu nedeľu a inkrementálna každý piatok až sobotu – všetky o 23:05.

3.5.1 Návrh zálohovania

Pre náš systém na univerzite sme sa rozhodli zaviesť takéto vlastnosti pre zálohovanie. V prvom rade, Bacula je nainštalovaná na serveri na virtuálnom počítači s operačným systémom Debian s antivírusovou ochranou a SAMBA serverom.

Zálohujú sa vopred určené priečinky ako pracovná plocha, moje dokumenty, stiahnuté súbory a podobne. Zálohy z pracovných staníc sa vytvárajú každý deň v čase obeda, zatiaľ čo kópie sa robia každý deň po polnoci. Sú chránené proti vymazaniu alebo napadnutiu ransomewarom, vymazanie z klientskych počítačov sa nesynchronizuje. Každý užívateľ má prístup len k svojim zálohám za posledných sedem dní, a zároveň k nim vie pristúpiť len z vyhradenej školskej podsiete kvôli bezpečnosti. Ak užívateľ potrebuje staršie zálohy, je potrebné kontaktovať systémového administrátora. Všetky zálohy na serveri sa ukladajú na páskové úložisko, ktoré je dnes jedno z najbezpečnejších riešení. Plná záloha sa robí raz do mesiaca, inkrementálna sa robí každý deň. Po šesťdesiatich dňoch nastáva expirácia týchto záloh, je ich teda možné prepísať, keďže sú na páskach. Krátkodobé zálohy sú komprimované, používa sa dedikovaný server s ôsmimi jadrami, 24GB operačnej pamäte a súborovým systémom ZFS. Aktuálny

používaný kompresný pomer je 1:1,14. Virtuálne disky je možné dynamicky škálovať podľa vlastných potrieb.

3.5.2 Nastavenie servera pre zálohovanie

Serverová strana (bacula director) plní úlohu spravovania všetkých zálohovacích procesov. Potreba škálovateľnosti systému si vynútila rozdelenie serverovej strany na riadiaci server (director) a úložné zariadenie. Podľa potreby je potom možné prevádzkovať rôzny počet riadiacich serverov a úložných zariadení, a tým zabezpečiť odolnosť voči výpadku riadiaceho servera.

Riadiaci server udržiava informácie o všetkých klientoch, ich mená, heslá a IP adresy a konfiguračné parametre, ako napríklad štandardné úložisko, dobu udržiavania záloh a podobne.

Pre každého klienta je možné definovať čo, kedy a kam sa ma zálohovať. Pri definovaní požadovaných súborov na zálohovanie je možné definovať cesty v súborovom systéme, v ktorých má hľadať súbory na zálohovanie, prípony súborov, prípadne vylúčiť zo zálohovania špecifické adresáre a typy súborov (napríklad nezálohovať *.avi súbory). Ak je to potrebné, špecifickejší výber súborov na zálohovanie je možné dodať pomocou skriptu, v ktorom bude zoznam súborov. Tiež je možné definovať plánovače jednotlivých zálohovacích úloh, napríklad takto:

```
Run = Level=Full on 1 at 2:05
Run = Level=Differential 1st-5th sun at 3:05
Run = Level=Incremental mon-sat at 3:05
```

Pred zálohovaním je potrebné pripraviť konfiguráciu na strane servera aj klienta. Každá záloha má vypočítaný kontrolný súčet MD5 na overenie integrity. Ďalej je potrebné zadefinovať, čo sa má zálohovať. V ďalšom kroku sa dedí konfigurácia, zadá sa názov, typ úlohy, ktorá sa bude vykonávať, typ zálohovania. Zvolíme si klienta, ktorého dáta chceme zálohovať a nastavíme parametre. Príklad konfigurácie na strane servera s vysvetlivkami je možné vidieť v prílohe A, a v prílohe B môžeme vidieť príklad plánu zálohovania.

3.5.3 Nastavenie klienta pre zálohovanie

Na strane klienta je potrebné mať nainštalovanú aplikáciu zo stránky https://blog.bacula.org/binary-download-center/. Nie je potrebné inštalovať konzolu ani administračné nástroje, ale je treba pozmeniť nastavenie firewallu. V konfiguračnom súbore, ktorý sa nachádza v „C:\ProgramFiles\Bacula", je potrebné nastaviť meno a heslo v časti „Director" na „Name = srv-backup.science.upjs.sk-dir". Konfiguračný súbor môže byť pozmenený pomocou Ansible na diaľku alebo môže priamo obsahovať obraz disku, ktorý bol nakopírovaný na počítač.

Na strane servera sa vytvorí konfigurácia pre každú klientsku stanicu. Po inštalácii na strane klienta je potrebné upraviť konfiguračný súbor „bacula-fd.conf", nachádzajúci sa v adresári „/etc/bacula/" na director serveri, pridaním názvu director-a a hesla.

Príklad konfigurácie na strane klienta s popisom jednotlivých parametrov je možné vidieť v prílohe C.

Záver

Zabezpečenie informačnej bezpečnosti, zjednodušenie a zrýchlenie podpory a správy pracovných staníc si vyžaduje zavedenie centrálnej správy týchto staníc. Táto práca sa podrobnejšie venuje práve tejto problematike. Ako sme ukázali v rámci práce, existuje niekoľko prístupov, resp. stratégií, ako postupovať pri správe pracovných staníc. Štandardne používaným postupom správy pracovných staníc je odstupňovaná podpora, v rámci ktorej sa určia určité skupiny pracovných staníc, a v rámci nich sa poskytuje podpora.

Keďže naším cieľom bola centralizácia a automatizácia procesov pri správe týchto staníc, zamerali sme sa na iný prístup k správe pracovných staníc, a to k prístupu infraštruktúra ako kód (IaC). Tento prístup nepredpokladá priamy prístup k infraštruktúre (v našom prípade pracovným staniciam), ale pracovné stanice predstavujú objekty, s ktorými je možné v rámci kódu pracovať a určovať činnosti, ktoré sa s nimi majú vykonať. Týmto spôsobom vieme veľmi rýchlo a účinne zabezpečiť centrálnu a automatizovanú správu pracovných staníc.

Táto záverečná práca mala stanové tri ciele. Prvým cieľom bolo analyzovať možnosti a prístupy k centrálnej správe počítačov. Ako sme už uviedli vyššie, bližšie sme sa zamerali na najčastejšie sa vyskytujúce prístupy, ktoré sme doplnili novým prístupom, a to infraštruktúrou ako kódom (IaC). Následne sme v rámci prvej kapitoly tejto práce rozobrali ďalšie aspekty správy, resp. centrálnej správy pracovných staníc. Postupne sme sa venovali správe hardvéru, operačnému systému, softvéru a údajom. Túto analýzu sme doplnili o centralizovaný model správy a jeho špecifiká.

Druhým cieľom práce bolo porovnať a analyzovať aktuálne riešenia k centrálnej správe pracovných staníc. Tento cieľ sme si rozdelili do troch častí. Primárne sme sa venovali porovnaniu nástrojov na vzdialenú správu infraštruktúry (v našom prípade pracovných staníc). Tu sa najviac prejavil prístup IaC. Na základe porovnania sme dospeli k názoru, že najvhodnejším nástrojom pre našu centralizovanú správu bude nástroj Ansible. Keďže navrhovaná centralizovaná správa sa venuje aj problematike zálohovania a obnove údajov, v druhom porovnaní sme sa zamerali na porovnanie nástrojov pre vytvorenie zálohy údajov pracovnej stanice, a ich následnú obnovu. V rámci porovnania sme dospeli k vyhodnoteniu, že najvhodnejším nástrojom pre navrhovanú centralizovanú správu bude nástroj Bacula. A to aj vzhľadom na jej obťažnejšiu prvotnú inštaláciu. Následne sme porovnali nástroje pre jednu z dôležitých činností správy pracovných

staníc, a to vytváranie a obnova obrazov disku pracovných staníc. Z porovnávania nám ako najlepší vyšiel nástroj Clonezilla.

Posledným cieľom bolo navrhnúť a implementovať centrálnu správu v prostredí vysokej školy, teda v akademickom prostredí. Nami navrhovaný systém obsahuje vzdialené vykonávanie činností na pracovných staniciach, zálohovanie a obnovu ich údajov, ako aj vzdialené zapnutie pracovných staníc. Na vzdialenú správu pracovných staníc využívame nástroj Ansible. V rámci tretej kapitoly tejto práce sa venujeme jeho princípom a možnostiam. Ukazujeme, ako je možné tento nástroj použiť v rámci centralizovanej správy. Súčasťou tejto časti sú aj ukážky jednotlivých automatizačných skriptov a konfigurácií. Súčasne sa v rámci tohto cieľa venujeme aj zálohovaniu a obnove údajov. Tretia kapitola tejto práce obsahuje základné údaje o zálohovaní v rámci použitého systému Bacula, základné konfiguračné nastavenia serverovej aj klientskej časti tohto systému.

V rámci budúcej práce by bolo možné nadviazať na niektoré časti navrhovaného systému. V dôsledku rozsahu práce, a aj časových možností, sme sa nemohli venovať centrálnemu a automatizovanému vytváraniu a obnove obrazov diskov pracovných staníc. Pre tento účel sme urobili analýzu dostupných riešení. Súčasne by bolo možné doplniť ďalšie čiastkové skripty a playbooky pre vykonávanie ďalších vzdialených činností.

Zoznam použitej literatúry

[1] SILBERSCHATZ, Abraham, et al. Operating System Concepts Essentials: Binder Ready Version. John Wiley, 2011.

[2] LIMONCELLI, Tom; HOGAN, Christina J.; CHALUP, Strata R. The Practice of System and Network Administration: Volume 1: DevOps and other Best Practices for Enterprise IT. Addison-Wesley Professional; 3 edition, 2016.

[3] SANCHEZ, Andrew; SLEETH, Karen. Technical Support Essentials: Advice to Succeed in Technical Support, 2009.

[4] WALKER, Gary S. IT problem management. Prentice Hall Professional, 2001.

[5] WHALEY, Ben; SNYDER, Garth; NEMETH, Evi; MACKIN, Dan; HEIN, Trent R. UNIX and Linux System Administration Handbook. Addison-Wesley Professional; 5 edition, 2017.

[6] Projekt RFC [online]. [cit. 2018-08-08]. Dostupné z: https://tools.ietf.org/html/draft-henry-remote-boot-protocol-00/

[7] Projekt Packer [online]. [cit. 2018-08-08]. Dostupné z: https://www.packer.io/intro/

[8] Projekt Vagrant [online]. [cit. 2018-08-08]. Dostupné z: https://www.vagrantup.com/

[9] LIMONCELLI, Tom; HOGAN, Christina J.; CHALUP, Strata R. The practice of system and network administration. Pearson Education, 2007.

[10] Softvér [online]. [cit. 2018-08-08]. Dostupné z: https://en.oxforddictionaries.com/definition/software

[11] Projekt Chocolatey [online]. [cit. 2018-08-08]. Dostupné z: https://chocolatey.org/

[12] Projekt Ansible [online]. [cit. 2018-08-08]. Dostupné z: https://www.ansible.com/

[13] MCKENDRICK, Russ. Learn Ansible. Packt Publishing, 2018.

[14] Projekt Chef [online]. [cit. 2018-08-08]. Dostupné z: https://www.chef.io/

[15] Projekt Puppet [online]. [cit. 2018-08-08]. Dostupné z: https://puppet.com/

[16] Projekt SaltStack [online]. [cit. 2018-08-08]. Dostupné z: https://saltstack.com/

[17] Projekt Amanda [online]. [cit. 2018-08-08]. Dostupné z: https://www.zmanda.com/

[18] Projekt BackupPC [online]. [cit. 2018-08-08]. Dostupné z: http://backuppc.sourceforge.net/

[19] Projekt Bacula [online]. [cit. 2018-08-08]. Dostupné z: http://blog.bacula.org/

[20] Projekt Bareos [online]. [cit. 2018-08-08]. Dostupné z: http://www.bareos.com/en/

[21] Projekt Rsync [online]. [cit. 2018-08-08]. Dostupné z: https://rsync.samba.org/

[22] Projekt Acronis True Image [online]. [cit. 2018-08-08]. Dostupné z: https://www.acronis.com/

[23] Projekt Clonezilla [online]. [cit. 2018-08-08]. Dostupné z: https://clonezilla.org/

[24] Projekt FOG [online]. [cit. 2018-08-08]. Dostupné z: https://fogproject.org/

[25] Projekt Ghost [online]. [cit. 2018-08-08]. Dostupné z: https://www.symantec.com/products/ghost-solutions-suite/

Prílohy

Príloha A: Príklad konfigurácie servera pre zálohovanie nástrojom Bacula

Príloha B: Príklad plánu zálohovania na strane servera nástrojom Bacula

Príloha C: Príklad konfigurácie klienta pre zálohovanie nástrojom Bacula

Príloha D: Powershell skript na získanie údajov o diskovej kapacite

Príloha E: Powershell skript na pripojenie k databáze

Príloha F: Python skript na poslanie informačnej emailovej správy

Príloha G: DVD médium a disketa

Príloha A

```
FileSet {
        Name = "Windows Ucebna Set"
        Include {
                Options {

                        signature = MD5       # každá záloha má vypočítanú kontrolnú
sumu na overenie integrity
                }
        File = D:
        File = C:/Users                       # ľubovoľný počet direktív “File = cesta“

        }
}

Job {
        JobDefs = "DefaultJob"                # zdedenie konfigurácie z DefaultJob
definície
        Name = "WindowsBackup"                # nazov
        Type = Backup                         # typ ulohy - zaloha/obnova/migracia ...
        Level = Incremental                   # level zalohy - ak je zvolený incremental a
neexistuje full zaloha, systém automaticky najprv vytvorí plnú zálohu a následne ďalšie
zálohy budú incrementalne voči tejto full zálohe
        Client = desktop-mq6h2p7-fd
        FileSet = "Windows Ucebna Set"
        Pool = TapePool
}

Client {
        Name = desktop-mq6h2p7-fd             # názov zálohovanej klientskej stanice
        Address = 158.197.36.129              # IP adresa zálohovanej klientskej stanice
        FDPort = 9102
        Catalog = MyCatalog                   # definícia katalógu záloh, v prípade, že ich
je viacero
        Password = "**********"               # heslo zálohovanej klientskej stanice
        File Retention = 2 months             # ako dlho udrziavat obsah zalohy
        Job Retention = 2 months              # ako dlho udržiavať informáciu o
konkrétnom spustení procesu zálohy
        AutoPrune = yes                       # automaticky premazať po prekročení
}
```

Príloha B

```
Schedule {
        Name = "WeeklyCycle"                    # mesačný cyklus podľa týždňov - nastevené tak, aby nezačalo plnú zálohu v pracovný deň
        Run = Full 1st sun at 23:05             #  plná záloha každú prvú nedeľu
        Run = Differential 2nd-5th sun at 23:05 # diferenciálna záloha
        Run = Incremental mon-sat at 23:05      # inkrementálna záloha
}

Schedule {
        Name = "MonthlyCycle"                   # mesačný cyklus podľa dní
        Run = Level=Full on 1 at 2:05           # plná záloha sa bude spúšťať vždy prvého, čo môže byť aj pracovný deň
        Run = Level=Differential 1st-5th sun at 3:05 # diferenciálna záloha
        Run = Level=Incremental mon-sat at 3:05 # inkrementálna záloha
}

# This schedule does the catalog. It starts after the WeeklyCycle
Schedule {
        Name = "WeeklyCycleAfterBackup"
        Run = Full sun-sat at 23:10
}
```

Príloha C

```
Director {
        Name = srv-backup.science.upjs.sk-dir        # riadiaci server musí mať toto
meno
        Password = "********"                        # riadiaci server musí poznať toto
heslo
}
FileDaemon {                                         # definícia samotného file daemona
        Name = desktop-mq6h2p7-fd                    # riadiaci server musí poznať meno
zálohovanej klientskej stanice
        FDport = 9102                                # port
        WorkingDirectory = "C:\\Program Files\\Bacula\\working"
        Pid Directory = "C:\\Program Files\\Bacula\\working"
        Plugin Directory = "C:\\Program Files\\Bacula\\plugins"
        Maximum Concurrent Jobs = 10
}
```

Príloha D

```
$dp = Get-WmiObject win32_logicaldisk | Where-Object {$_.drivetype -eq 3}
foreach ($item in $dp) {
$deviceID = $item.DeviceID
$freeSpace = $item.FreeSpace / 1gb -as [int]
$size = $item.Size / 1gb -as [int]
}
```

$ip=get-WmiObject Win32_NetworkAdapterConfiguration|Where {$_.Ipaddress.length -gt 1} a ku konkretnej ip adrese pristupis ako $ip.ipaddress[0]

Príloha E

```
Function Run-MySQLQuery {
    Param(
        [Parameter(
                Mandatory = $true,
                ParameterSetName = '',
            ValueFromPipeline = $true)]
                [string]$query,
        [Parameter(
                Mandatory = $true,
            ParameterSetName = '',
                ValueFromPipeline = $true)]
                [string]$connectionString
    )

    Process {
        try {

    [void][System.Reflection.Assembly]::LoadWithPartialName("MySql.Data")
    $connection = New-Object MySql.Data.MySqlClient.MySqlConnection
    $connection.ConnectionString = $ConnectionString
    $connection.Open()
    $command = New-Object MySql.Data.MySqlClient.MySqlCommand($query, $connection)
    $dataAdapter = New-Object MySql.Data.MySqlClient.MySqlDataAdapter($command)
    $dataSet = New-Object System.Data.DataSet
    $recordCount = $dataAdapter.Fill($dataSet, "data")
    $dataSet.Tables["data"] | Format-Table
        }
```

```
                catch {
                        Write-Host "Could not run MySQL Query" $Error[0]
                }
                Finally {
                        $connection.Close()
                }
    }
        End {
                Write-Verbose "Starting End Section"
        }
}

$DB = run-MySQLQuery -ConnectionString Server=mysqlserver.science.upjs.sk;Uid=ansible;Pwd=GhNrXqUQHFlnRTE5;database=ansible;SslMode=none;' -Query 'SELECT * from Storage'

$DB
```

Príloha F

```
import smtplib
import sys

import MySQLdb as mdb

con = mdb.connect('dbadmin.science.upjs.sk', 'ansible', '*********', 'ansible');

data = ""

mail_user = 'user@server.com'
mail_password = '*******'
mail_server = 'mail.server.com'
mail_port = 25

sent_from = "ansible@email.com"
to = ['user1@mail1.com', 'user2@mail2.com']

subject = 'CMS report'

space_limit=0.85

with con:
    cur = con.cursor()
    cur.execute("select IP,deviceID,freeSpace,size from Storage")

for i in range(cur.rowcount):
    row = cur.fetchone()
    IP,deviceID,free,size = row

    used_space = 1-free/size
    if used_space>space_limit:
        data += """IP: {}
- celkový diskový priestor C: - {} GB
- aktuálny diskový priestor C: - {} GB
""".format(IP,size,free)

body = """
Vážený administrátor,

CMS systém zaznamenal prekročenie hodnôt pre nasledujúce pracovné stanice:
{data}
""".format(data=data)

email_text = """From: %s
To: %s
```

```
Subject: %s

%s
""" % (sent_from, ", ".join(to), subject, body)

def send_mail(text,recipient):
    """ Simple mail sending function"""

    try:
        server = smtplib.SMTP(mail_server,mail_port)
        server.ehlo()
        server.starttls()
        server.login(mail_user, mail_password)
        server.sendmail(sent_from, recipient, text.encode("utf8"))
        server.close()

        print('Email sent!')
    except:
        print('Error, email not sent')

send_mail(email_text,to)
```