VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

KEY MANAGEMENT SERVER V PROSTŘEDÍ VSPEHERE 7.0

KEY MANAGEMENT SERVER FOR VSPHERE 7.0 ENVIRONMENT

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. DAVID DEJMAL**
AUTHOR

VEDOUCÍ PRÁCE — **Mgr. KAMIL MALINKA, Ph.D.**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Dejmal David, Bc.**
Program: Informační technologie
Obor: Kybernetická bezpečnost
Název: **Server pro správu klíčů v prostředí vSphere 7.0**
Key Management Server for vSphere 7.0 Environment
Kategorie: Bezpečnost

Zadání:

1. Seznamte se s problematikou správy klíčů v prostředí virtualizace a prostudujte používané přístupy správy klíčů v již existujících implementacích.
2. Navrhněte architekturu serveru pro správy klíčů (KMS) v prostředí VMware vSphere 7.0 a definujte množinu funkcí, které by měl podporovat.
3. Implementujte server pro správu klíčů v prostředí VMware vSphere 7.0 se základní funkcionalitou. Zaměřte se na podporu šifrování jednotlivých VM.
4. Proveďte testování, zaměřte se na bezpečnostní a výkonové parametry.
5. Diskutujte rozšiřitelnost řešení pro podporu dalších virtualizačních platforem.

Literatura:

- Key Manager for VMware vSphere from HyTrust (dostupnné online https://www.hytrust.com/keycontrol-for-vsphere-encryption/)
- Encryption & Key Management for VMware The Definitive Guide

Při obhajobě semestrální části projektu je požadováno:

- Body 1 a 2.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Malinka Kamil, Mgr., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 11. listopadu 2020

Abstrakt

Cílem této práce je vytvořit funkční server pro správu klíčů (Key Managment server – KMS) se základní funkcionalitou pro platformu vSphere 7.0. Ten by měl komunikovat s vCenter a dohromady poskytovat funkci na šifrování jednotlivých virtuální strojů.

Komerční řešení v této oblasti jsou velmi drahá, a proto vyvstala otázka, zda lze celý server implementovat pomocí volně dostupných nástrojů. Jelikož vCenter používá ke komunikaci s KMS veřejně dostupný protokol KMIP, ukázalo se, že to lze.

Výsledná implementace je založena na operačním systému Ubuntu 20.04. Pro aplikační logiku byla použita knihovna PyKMIP pro python 3.9 a jako úložiště bylo použito ETCD. Pro spojení aplikace a úložiště bylo potřeba vytvořit vlastní modul. Pro veškerou potřebnou konfiguraci včetně instalace byly vytvořeny Bash skripty.

Celkový výsledek je plně funkční a během testování nebyly nalezeny žádné nedostatky. Na práci probíhala spolupráce se společností Master Internet, s.r.o.

Abstract

The purpose of this work is to create a functional Key Management Server (KMS) with basic functionality for the vSphere 7.0 platform. It should communicate with vCenter and together provide the functionality to encrypt individual virtual machines.

Commercial solutions in this area are very expensive and therefore the question arose whether the entire server can be implemented using freely available tools. Since vCenter uses the publicly available KMIP protocol to communicate with KMS, it turns out to be possible.

The resulting implementation is based on the Ubuntu 20.04 operating system. The PyKMIP library for python 3.9 was used for the application logic and ETCD as storage. To connect the application and storage, a custom module was created. Bash scripts were created for whole installation and all of the necessary configuration.

The overall result is fully functional and no flaws were found during testing. This work was done in cooperation with Master Internet, s.r.o.

Klíčová slova

Server pro správu klíčů, KMS, vSphere, VMware, KMIP, PyKMIP, ETCD

Keywords

Key Managment Server, KMS, vSphere, VMware, KMIP, PyKMIP, ETCD

Citace

DEJMAL, David. *Key Management Server v prostředí vSpehere 7.0.* Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Mgr. Kamil Malinka, Ph.D.

Key Management Server v prostředí vSpehere 7.0

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana doktora Kamila Malinky. Další informace, včetně testovacího prostřední, mi poskytli Martin Žídek a František Bednář (pracovníci Master Internet, s.r.o.). Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
David Dejmal
16. května 2021

Poděkování

Chtěl bych poděkovat všem výše zmíněným za pomoc při tvorbě práce. Také chci poděkovat své rodině a přítelkyni za podporu v průběhu studia.

Obsah

Kapitola 1

Úvod

S rozmachem informačních technologií je kladen stále větší důraz na její bezpečnost. Je to dáno tím, že digitalizace prosakuje stále více jak do osobního, tak do pracovního života. Lidé vytvářejí velké množství dat, která mají nemalou hodnotu pro potencionální útočníky. Jedním z nejlepších způsobů, jak data ochránit, je jejich šifrování.

Dalším velkým pojmem v IT je virtualizace. Jedná se o praktiku, která mimo jiné významně šetří systémové zdroje. V posledních letech je tento koncept stále více používán a některé produkty jsou postavené již přímo na něm. I zde je ale nutné stále myslet na bezpečnost dat.

Tato práce spojuje daná dvě odvětví a řeší šifrování virtualizovaného prostředí. Cílem je dokázat, že lze pro tento úkol implementovat funkční server pro správu klíčů (Key Managment Server - KMS). Jako vizualizační platforma byla zvolena VMware vSphere ve verzi 7.0. KMS tedy bude muset komunikovat pomocí standardizovaného protokolu KMIP (Key Management Interoperability Protocol) ve verzi 1.1 či vyšší. Podporována bude základní množina funkcí pro korektní zašifrování virtuálních strojů. Server bude řídit celý životní cyklus klíčů.

Diplomová práce byla vytvářena ve spolupráci s českou společností Master Internet s.r.o., která se zaměřuje na řízení a správu datacenter. Společnost pro vývoj poskytla testovací prostředí. Také s ní byly konzultovány všechny podstatné části práce, zejména souhrn požadavků a testování.

Pro pochopení celé problematiky je nejprve nutné vysvětlení základních pojmů, definic a použitých konceptů. Dále je potřebné stanovení cílů a s tím spojená motivace celé práce. Všechny tyto informace jsou v druhé kapitole. Následující třetí kapitola popisuje použité testovací prostředí. Také obsahuje specifikaci požadavků a samotný návrh systému. Ve čtvrté kapitole se nalézá podrobný popis vytvořeného řešení. Dále postup testování a celkové vyhodnocení. V poslední, páté kapitole, jsou shrnuty všechny dosažené výsledky práce. Zmíněny jsou také podněty k vylepšení a rozšíření pro budoucí práci. K textu jsou přiloženy dvě přílohy s popisem odevzdávaných souborů a návodem na instalaci.

Kapitola 2

Teoretický základ

Před samotný popisem vypracovaného řešení práce je nutné jej zasadit do patřičného teoretického kontextu. Ten poskytuje celá tato kapitola.

2.1 Motivace a cíl práce

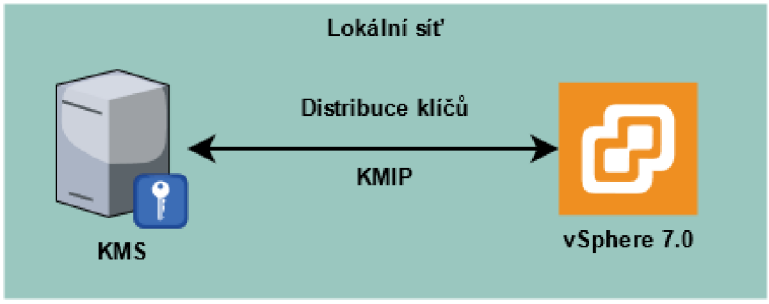
Jak již bylo napsáno v úvodu a v samotném zadání, práce se zabývá implementací serveru pro správu klíčů, určenou pro virtualizované prostředí. Čtenáře může napadnout otázka, proč je potřeba tento problém zpracovávat. Rozeberme si tedy tuto otázku podrobněji.

Proč tedy šifrovat? Protože pro citlivá data je nutné mít nasazené v určité formě řízení přístupu[1]. Šifrování je jedním z jeho nejsilnějších nástrojů. Pokud je implementováno správně, můžeme si být jisti, že bez patřičných klíčů se data v čitelné podobě do nesprávných rukou nedostanou. Stejný požadavek je i pro virtuální stroje a datové prostory. Pokud by například došlo k odcizení celého virtuálního disku, je nutné jej zabezpečit tak, aby jej útočník nemohl přečíst. Šifrování na úrovni hostujícího operačního systému (např. pomocí BitLocker[2] či VeraCrypt[3]) není vhodné na globální administraci a také nemusí zašifrovat celý virtuální disk. Proto, pokud je to podporováno, je doporučené šifrování na úrovni hypervisoru. Právě vSphere tuto funkcionalitu poskytuje hned pomocí třech způsobů (Standart, Thusted a Native Key Provider, viz 2.6.2). Tato práce řeší využití Standart Key Provider, tedy zapojení, kde se využívá externí Key Managment Server. S tím probíhá komunikace skrz protokol KMIP (podkapitola 2.7).

Další otázka, která může čtenáře napadnout, je, proč nepoužít již existující řešení. Zde je odpovědí bezpečnost a cena. Ačkoli výrobci KMS budou tvrdit, že jejich produkty jsou nejbezpečnější na trhu, nemusí to být vždy úplně pravda. Všechny oficiálně podporované KMS (viz Tabulka 2.2) nejsou open source, tedy nemají otevřený zdrojový kód. Cena stávajících řešení také není zanedbatelná. Protokol KMIP má na druhou stranu volně dostupný celý standard. Proto vznikla motivace k vytvoření vlastního KMS.

Cílem práce je tedy vyvinout Key Managment Server pouze za pomoci volně dostupných technologií. Fungovat musí s prostředím VMware vSphere ve verzi 7.0, tedy komunikovat skrz protokol KMIP, jak ukazuje Obrázek 2.1. Poskytovat bude základní operace potřebné pro korektní zašifrování zadaného virtuálního stroje. Podrobná specifikace požadavků je v podkapitole 3.2.

[1] Řízení přístupu je bezpečnostní funkce, která chrání sdílené prostředky před neoprávněnými přístupy [8].
[2] https://docs.microsoft.com/windows/security/information-protection/bitlocker
[3] https://www.veracrypt.fr



Obrázek 2.1: Schéma komunikace KMS a vSphere

2.2 Základní pojmy

V textu práce se vyskytují určité zavedené termíny či zkratky z oblasti virtualizace a šifrování. Pro případ neznalosti nebo neúplnosti některých údajů je nutné je zde definovat.

Hypervisor

Hypervisor je software, který vytváří a provozuje virtuální stroje (VM). Hypervisor, někdy nazývaný monitor virtuálního počítače (Virtual Machine Monitor – VMM), izoluje operační systém a prostředky hypervisoru od virtuálních strojů a umožňuje vytváření a správu těchto virtuálních počítačů.[23]

Virtuální stroj (VM)

Virtuální stroj je izolované virtuální prostředí v hostitelském OS. Veškerý hardware virtuálních strojů, jako je procesor, paměť či pevný disk je emulován a spravován prostřednictvím aplikace pro virtualizaci (hypervisoru). V tomto prostředí z pravidla běží hostující operační systém, který neví o tom, že je virtualizován. [1]

Virtuální disk

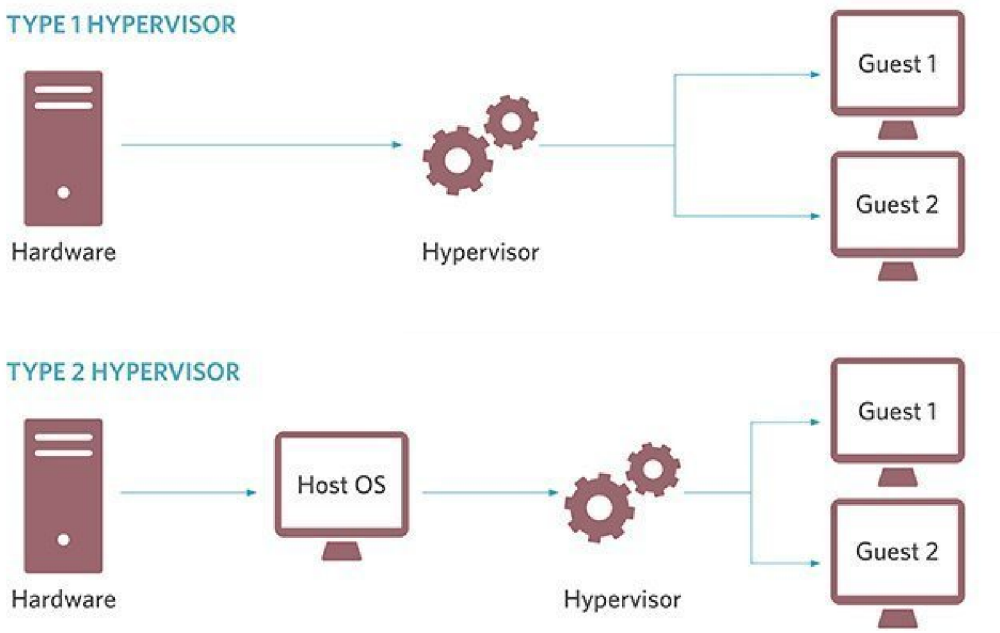
Jedná se o emulovanou paměť, která simuluje pevný disk počítače. Vůči hostujícímu OS (Guest OS) se tváří jako fyzický pevný disk, ale vůči hostitelskému systému (Host OS) se může jednat pouze o soubor. Může být také rozprostřen skrz více úložišť nebo simulovat větší velikost, než jakou má k dispozici. [25]

Typy hypervisorů

Hypervisory dělíme na dva typy. Toto rozdělení vychází z jejich architektury a způsobu přístupu k podřazené vrstvě. Typ jedna (označován taktéž jako bare-metal nebo hardwarová virtualizace) komunikuje přímo s hardwarem. Nepoužívá tedy jako prostředníka žádný operační systém. Vyřazením operačního systému dosahuje lepšího výkonu, vyšší stability a rozdělení zdrojů.

Typ dva, někdy označovaný jako softwarová virtualizace, komunikuje s operačním systémem a nemá přístup přímo k hardwaru. Jeho výhodou je lehčí správa a instalace. Vhodný je pro testovací účely, vývoj nebo pro prostředí, kde se používají pouze jednotky virtuálních strojů.

Toto rozdělení ovšem není vždy jednoznačné a existují nástroje u kterých je zařazení sporné kvůli různým akceleracím a podobně. Rozdělení schématicky zobrazuje Obrázek 2.2. Tato sekce je převzata z [26].



Obrázek 2.2: Rozdělení hypervisorů, převzato z [26]

Hostitelský systém (Host)

Tímto pojmem se označuje ten počítač a systém, v němž se bude virtualizovat. Pojmem hostitelský operační systém (Host OS) se pak míní pouze operační systém hostitelského stroje. Může jít pouze o systém, který podporuje virtualizaci. [32]

Pokud se nehovoří o virtualizaci, může Host také znamenat mateřský operační systém, tedy ten, na který je daná aplikace instalována.

Hostující systém (Guest)

Je to virtuální stroj a případně operační systém (Guest OS), běžící ve virtualizovaném prostředí. Každý Guest musí mít svého Hosta. Host může mít více Guestů, v závislosti na typu virtualizace. Může jít o libovolný operační systém.[32]

Key Managment Server (KMS)

Server pro správu klíčů. Implementace je cílem této práce. Podrobné informace jsou v podkapitole 2.6

DEK – Data Encryption Key

Jedná se o šifrovací klíč, jehož funkcí je šifrování a dešifrování samotných dat [2].

KEK – Key Encryption Key

Jde o šifrovací klíč, jehož funkcí je šifrování a dešifrování DEK [2]. Tento klíč neslouží k šifrování dat, slouží pouze k šifrování dalších klíčů.

TLS – Transport Layer Security

Jedná se o komunikační protokol zakládající se na certifikátech. Zajišťuje zabezpečenou komunikaci typu klient-server pomocí kryptografie. Jde o nástupce zastaralého protokolu SSL. Podporuje širokou škálu šifer. [30]

FIPS 140

Je standart o bezpečnostních požadavcích pro kryptografické moduly. Vydávává jej NIST (National Institute of Standards and Technology) a aktuálně je platná verze 140-2 a 140-3. Definuje pravidla například pro: porty a rozhraní, správu klíčů, fyzickou bezpečnost nebo stavový model. Definuje 4 bezpečnostní úrovně, kde každá je nadřazená té předešlé. Jednotlivé úrovně také korespondují s učitou úrovní v CC (Common Criteria) EAL (Evaluation Assurance Level). [20]

Hardware Security Module (HSM)

Jde o samostatné hardwarové zařízení, které poskytuje kryptografické funkce pro zabezpečené transakce. Je tamper-resistant, tzn. odolné proti neoprávněné manipulaci. Mezi jeho funkce patří například ukládání šifrovacích klíčů, generování náhodných čísel a celých klíčů nebo samotné šifrování zpráv. Může se jednat o USB tokeny, PCI karty nebo o celé samostatné zařízení. [27]

Trusted Platform Module (TPM)

Má obdobnou funkci jako HSM, ale jedná se pouze o čip umístěný v nějakém zařízení. Logicky je tedy levnější, protože má menší náklady, ale poskytuje menší bezpečnost. Tyto čipy jsou často používané a nacházejí se ve velkém množství dnešních zařízení. [29]

2.3 Virtualizace

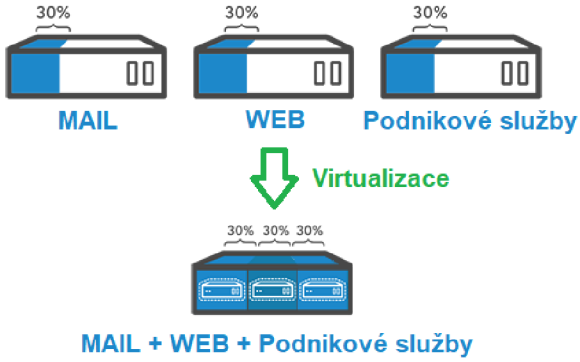
Jelikož se tato práce celá zaobírá problematikou ve virtualizovaném prostředí, je vhodné tuto oblast alespoň obecně popsat a vysvětlit základní principy.

2.3.1 Popis virtualizace

Obecně lze říci, že virtualizace je emulace jedné či více pracovních stanic pomocí jednoho fyzického zařízení. Jedná se tedy o emulaci hardwarových prostředků na softwarové. V praxi to znamená, že na jednom fyzickém zařízení může současně fungovat více počítačů. Virtualizaci schématicky porovnává Obrázek 2.3. [1]

Virtualizace je v posledních letech velký trend a stále roste množství virtualizovaných zařízení[4]. V praxi se virtualizace velmi ujala pro své nesporné výhody, jenž jsou popsány

[4] https://www.spiceworks.com/marketing/reports/state-of-virtualization/



Obrázek 2.3: Výhoda Virtualizace oproti tradičnímu přístupu, převzato a upraveno z [24]

níže. Vzhledem k poptávce probíhá v této oblasti i živelný vývoj. Vznikají různé alternativy ke klasickému přístupu jako například kontejnerizace, která získala velkou popularitu v poslední dekádě. Také se stále více aplikací přesouvá do cloudového prostředí.

Motivace a cíle virtualizace

Virtualizace přináší hned několik výhod, a proto je tak oblíbená. Prvním je zvýšení využití hardwarových zdrojů. Využití zdrojů u standardních serverů, kde není využita virtualizace, se pohybuje pouze v malých desítkách procent[5]. Využitím virtualizace se využití razantně zvýší. Se sníženým počtem fyzických strojů klesá pořizovací cena a klesají hlavně nemalé provozní náklady.

Dalším faktorem je zvýšení flexibility. Virtuální stroj je lehce konfigurovatelný a přenositelný. Vytvoření nového virtuálního serveru trvá pouze jednotky minut. Také zapínání virtuálního stroje je výrazně rychlejší, což má za následek zvýšení dostupnosti. Hypervisory také podporují veliké množství operačních systémů a tím zajišťují nezávislost na použitém hardwaru.

2.3.2 Bezpečnost virtualizace

Vzhledem k tomu, že virtualizace přidává mezivrstvu do celé architektury, roste tím pádem plocha možného útoku. Ačkoliv se virtualizace obecně považuje za bezpečnou, existují v ní nové bezpečnostní hrozby, například tyto:

- Absence ochrany hypervisoru. Neoprávněné získání přístupu k řízení systému virtuálních serverů.
- Při slabé izolaci VM může VM útočníka proniknout přes hypervisor a získat řízení fyzického serveru (VM Escape).
- Neoprávněné interakce mezi VM útočníka a fyzickým serverem obvykle přes sdílené paměťové zdroje (např. přes sdílenou paměť (shared clipboard) apod.).

[5] https://www.energystar.gov/products/low_carbon_it_campaign/12_ways_save_energy_data_center/server_virtualization

- Neoprávněný síťový přístup VM útočníka dovnitř virtuální infrastruktury a monitorování síťového provozu mezi VM.
- Útoky na jiné VM prostřednictvím VM útočníka (VM Hopping)
- Monitorování činnosti VM ze strany fyzického serveru.
- Neoprávněný přístup uživatelů k virtuálním strojům a instalace škodlivého SW. Chybějící kontrola činností uživatelů a jimi prováděných změn konfigurace virtuálního prostředí.

Výčet je převzat z [16]. Mohl by být ale podstatně větší. Tato práce se zaobírá šifrováním VM. Tedy případy, kdy útočník získá přístup na úložiště virtuálních strojů. Po zkopírování celých nešifrovaných VM do vlastního prostředí není problémem z ní získat jakékoliv data.

Pro ochranu virtualizovaného prostředí je nutné dodržovat některá pravidla jako například: pravidelně aktualizovat hypervisor, mít hypervisor korektně nakonfigurován, spouštět VM pouze z věrohodného zdroje a zabezpečit jejich OS nebo monitorovat komunikaci VM. Je jich ovšem podstatně více a liší se v závislosti na dané platformně. Postupů a článků na toto téma je nepřeberné množství, lze využít například tyto[6][7][8].

2.3.3 Nástroje pro virtualizaci

Pro přehled je vhodné uvést několik nejpoužívanějších vizualizačních nástrojů. Často se liší přístupem a celkovou architekturou. Nástroje od společnosti VMware zde nejsou uvedeny, podrobně jsou rozepsány v následující podkapitole 2.4.

Hyper-V

Jedná se o hypervisor typu jedna od společnosti Microsoft. Vychází ve třech verzích. Jako nástroj desktopové verze Windows 10 (určené pro vývoj a testovaní), ve Windows server od verze 2008 a také samostatně jako Windows Hyper-V Server. Jednotlivá vydání se od sebe liší počtem nabízených nástrojů. Hyper-V Server se v praxi moc nepoužívá, přestože je dostupný zdarma. Hyper-V má omezenou množinu podporovaných Guest OS. Čistě ve Windows infrastrukturách se jedná o velmi oblíbenou a výkonou platformu pro virtualizaci. [7]

Amazon Web Services (AWS)

AWS je platforma od firmy Amazon pro cloudové služby. Je široce dostupná a využívají ji miliony uživatelů po celém světě. AWS je distribuovaná jako služba skrz internet a nelze si vytvořit vlastní instanci. Lze si objednat velké množství funkcí, nástrojů a aplikací (IaaS, Paas)[9]. Podporována je i tvorba vlastního clusteru z jednotlivých virtuálních strojů. Platby jsou založeny na modelu Pay-as-you-go, tedy platí se za přímé využívání služeb dle stanovených parametrů. [5]

[6] https://cyberexperts.com/virtualization-security/
[7] https://blog.netwrix.com/2020/01/09/virtualization-security/
[8] IEEE – Virtualization and security: Examination of a virtualization platform structure: https://doi.org/10.1109/UBMK.2017.8093379
[9] Infrastructure as a service (IaaS), Platform as a service (PaaS) https://www.bmc.com/blogs/saas-vs-paas-vs-iaas-whats-the-difference-and-how-to-choose/

Kernel-based Virtual Machine (KVM)

V českém doslovném překladu: virtuální stroj založený na jádře. Dostupný je pro operační systémy s linuxovým jádrem. Vyvíjí jej společností Red Hat a distribuován je pod licencí GNU GPL licencí[10] . Není přesně specifikováno, do jaké třídy hypervisorů patří. Podporuje přímou hardwarovou virtualizaci, ale nelze spustit bez hostujícího operačního systému (Host OS). V dnešní době již podporuje velké množství typů hostujících OS. Spravuje se skrz CLI, ale existuje i nespočetné množství grafických nadstaveb.

Oracle VM VirtualBox

Jedná se o multiplatformní hypervisor druhého typu. Nicméně v posledních verzích již také podporuje hardwarovou akceleraci. Tedy dokáže komunikovat s vybraným hardwarem bez komunikace s hostujícím operačním systémem (Host OS). Obecně není doporučován pro produkční nasazení kvůli způsobu virtualizace. Umí všechny základní funkce hypervisoru. Dostupný je pro Windows, Mac OS X, Linuxové distribuce a Solaris. Od roku 2009 jej vyvíjí společnost Oracle. Má také otevřený zdrojový kód. V komunitě je velmi oblíbený pro svou jednoduchost. [22]

2.4 VMware vSphere

Jedná se o komplexní virtualizační platformu výpočetních prostředků pro hybridní cloud. Skládá se z několika aplikací a z velkého množství nástrojů. Popis těch nejpodstatnějších aplikací je rozepsán níže. Celá tato podkapitola je až na uvedené výjimky převzata z oficiální dokumentace VMware [31].

vSphere vychází ve více edicích:

- **Essentials Kit**

  Limitace na 3 servery s až dvěma procesory. Obsahuje pouze základní funkce pro virtualizaci (ESXi a vCenter Server Essentials). Určený je pro malé podniky (600 $).

- **Essentials Plus kit**

  Opět limitace na 3 servery s až dvěma procesory. Poskytuje skoro všechny funkce. Vhodné pro kritické a náročné aplikace pro menší podniky. Stále má limitaci na velikost (6 000 $).

- **Standard**

  Již bez omezení na velikost. Zahrnuje základní množinu funkcí pro každodenní provoz. Vhodné pro většinu aplikací (1 300 $).

- **Enterprise Plus**

  Komplexní řešení obsahující všechny nástroje a rozšíření. Bez omezení na velikost. Určený pro největší podniky a nejnáročnější zákazníky (4 350 $).

Uvedené ceny jsou za roční licenci a jsou pouze orientační. Části odpovídají stavu v březnu roku 2021. Souhrnný přehled rozdílu mezi licencemi je dostupný opět v oficiální dokumentaci nebo zde[11].

[10] https://www.gnu.org/licenses/gpl-3.0.html
[11] https://www.mustbegeek.com/understanding-vmware-vsphere-7-editions-and-features/

Cena licence se u edice Standart a Enterprise platí dle počtu fyzických procesorů (CPU). Pro CPU s více jak 32 fyzickými jádry je potřeba dvou licencí. Velikost RAM nebo počet virtuálních strojů nemá vliv na licenci. Zdarma dostupný je pouze ESXi hypervisor v základní konfiguraci s omezenými nástroji. I tak je velmi oblíbený mezi uživateli.

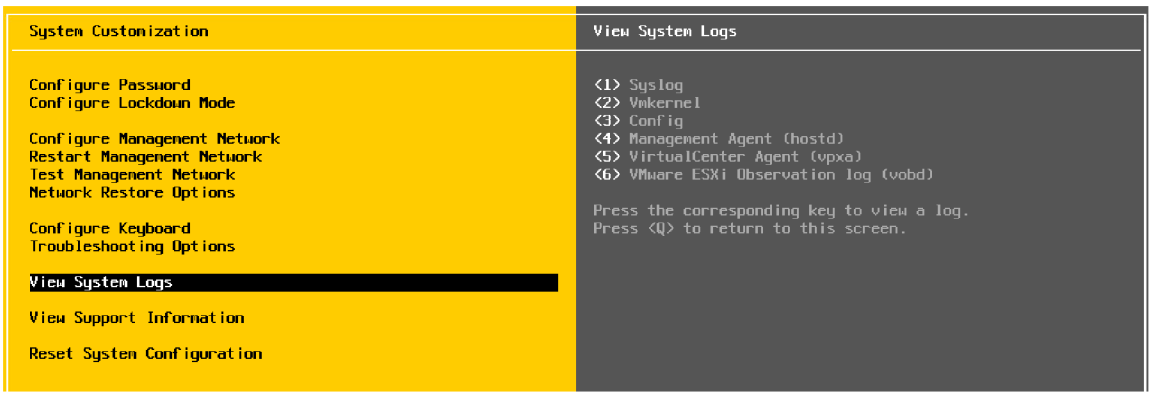
2.4.1 Společnost VMware

Je vhodné uvést alespoň pár slov o společnosti VMware. Jedná se o americkou firmu se sídlem v Kalifornii. Zaměřuje se na cloudové služby a virtualizaci. Má globální dosah a její produkty jsou jedny z nejrozšířenějších na světě. Založena byla roku 1998 a nyní má přes 31 000 zaměstnanců. V roce 2016 společnost Dell odkoupila majoritní podíl za 67 miliard dolarů.

2.4.2 VMware ESXi

Jedná se o základní prvek z vSphere platformy. Je to bare-metal hypervisor, viz 2.2. Původní projekt byl Elastic Sky X (ESX), později byl nahrazen Elastic Sky X Integrated (ESXi). Někdy se také používá název vSphere Hypervisor.

Má speciální ultra tenký kernel. Celý systém je navržen tak, aby šel spravovat pomocí vlastního rozhraní. Ovládání systému je tedy možné pomocí příkazové řádky nebo webové aplikace. Ukázky jsou na Obrázcích 2.4 a 2.5.

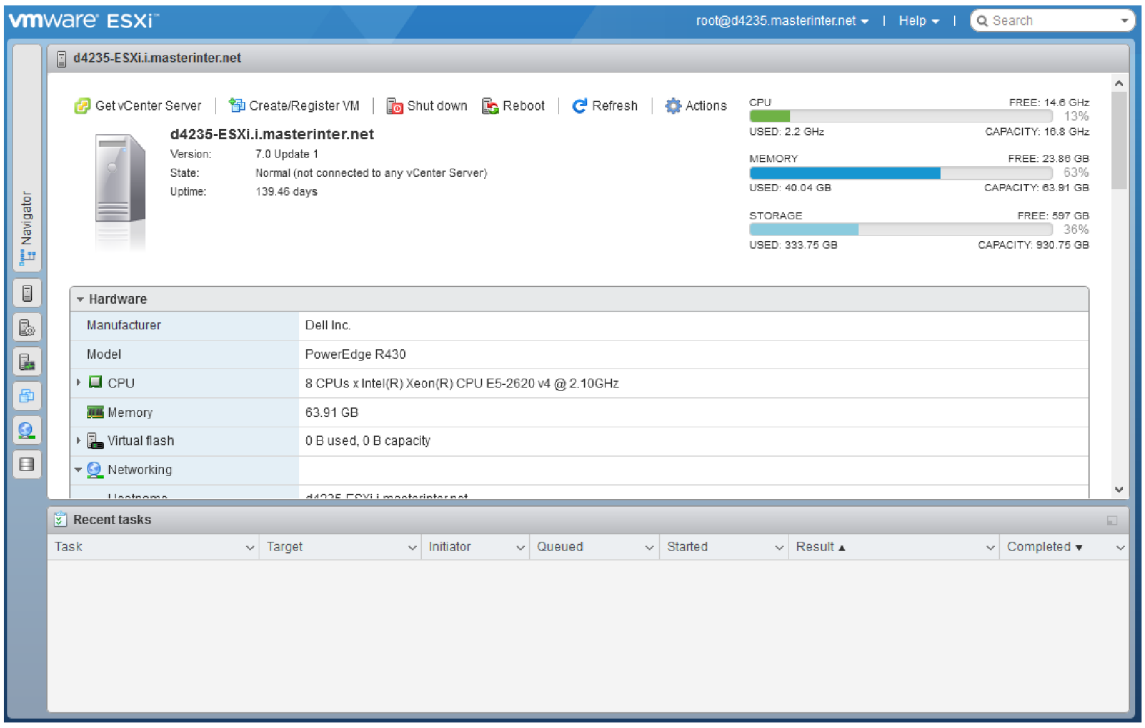


Obrázek 2.4: Ukázka CLI rozhraní ESXi z testovacího prostředí

Zkrácené minimální hardwarové požadavky jsou:

- CPU – 2 fyzická jádra,
- RAM – 4 GB,
- Boot disk – 8 GB SD karta nebo USB disk.

ESXi má tedy samo o sobě velice malé hardwarové nároky. Prostředky se ale musí optimalizovat vzhledem k množství a parametrům plánovaných virtuálních strojů. Před samotnou instalací je vhodné zkontrolovat, zda je daný server pro vSphere podporován. Pro některé série serverů existují speciálně optimalizované instalační balíčky.



Obrázek 2.5: Ukázka webového rozhraní ESXi z testovacího prostředí

2.4.3 VMware vCenter Server

Jedná se o aplikaci pro centrální správu platformy vSphere. Skrze ni je možné spravovat všechny prvky a funkce v celé síti. Je velice dobře škálovatelná, umožňuje efektivní správu i pro tisíce virtuálních strojů. Dělá podrobný monitoring všech zapojených ESXi a jejich prvků. Poskytuje pro administrátory kompletní nástroj k řízení celé infrastruktury. Schéma architektury je v Obrázku 2.6.

Samotná správa probíhá opět skrze webové rozhraní. Ukázka je na Obrázku 2.7. Instaluje se opět přímo na server jako operační systém. Skládá se z Photon OS[12], Postgres databáze a webové aplikace pro správu.

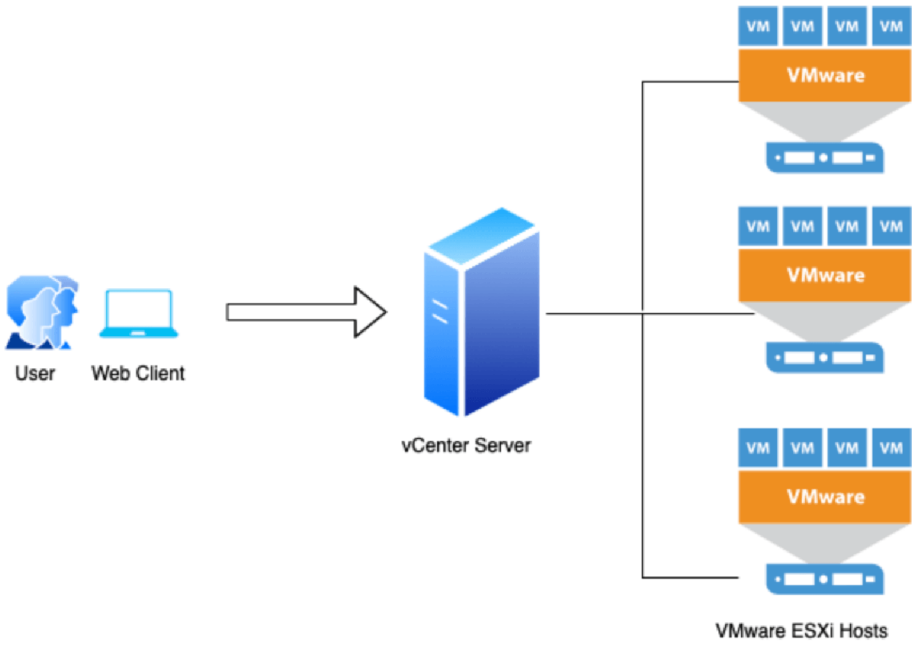
VMware vCenter sdružuje jednotlivé ESXi do vSphere clusterů. V rámci jednoho clusteru spolu servery sdílejí prostředky a je možné používat určité pokročilé funkce. Takto sdružovat servery je velmi výhodné při nastavování politik a monitoringu.

2.4.4 Dostupné funkce

vSphere má krom samotné virtualizace i velké množství dalších funkcí. Zde je jen velice zkrácený popis těch nejdůležitějších:

- **High Availability (HA)** – po chybě fyzického serveru automaticky obnoví virtuální stroje.
- **Fault Tolerance** – nepřetržitá dostupnost při hardwarové chybě.

[12]Open source minimalistický Linux od společnosti VMware https://github.com/vmware/photon



Obrázek 2.6: Schéma vCenter architektury, převzato z [6]

- **vMotion** – poskytuje tzv. live migration, tedy přesun VM mezi hosty bez ztráty dostupnosti.
- **Storage vMotion** – umožňuje přesun celých diskových polí při zachování dostupnosti během potřebné odstávky hardwaru.
- **Policy-Based Management** – řízení clusterů dle automatických politik.

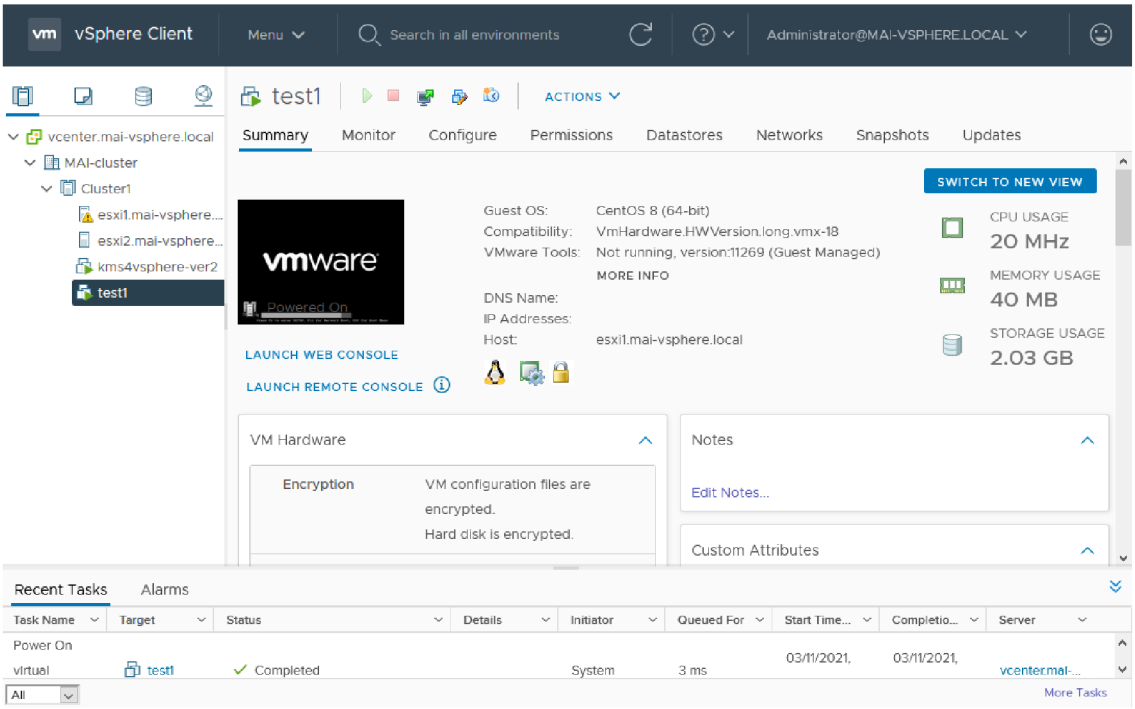
Verze Enterprise má navíc ještě mimo jiné tyto nástroje:

- **Distributed Resource Scheduler (DRS)** – vyvažování zátěže (Load balancing) v rámci vSphere clusteru.
- **NVIDIA GRID vGPU** – podpora virtualizace serverových grafických karet.
- **I/O Controls** – priorizace sítě a úložiště podle jednotlivých VM.

Nutné je ještě zmínit **VMware vSAN**. Je to nástroj, který poskytuje vysoce výkonné datové úložiště na softwarové bázi. V případě All Flash konfigurace má masivní, distribuovaný výkon, kterému jen těžko monolitická disková pole konkurují. Obrovskou výhodou je také jeho škálovatelnost. Celé úložiště lze jednoduše za běhu rozšiřovat a tím navyšovat jeho výkon i kapacitu. [18]

2.4.5 vSphere verze 7.0

Aktuální verze vSphere je 7.0 Update 1d z 4.2.2021 (k 7.3.2021). Vývoj jednotlivých verzí shrnuje Tabulka 2.1. Před verzí 4 bylo pojmenování VMware Infrastructure a jednalo se o trochu jiný produkt. Verze 7 přináší několik novinek:



Obrázek 2.7: Ukázka webového rozhraní vCenter z testovacího prostředí

- **VMware Tanzu** – přímá podpora kontejnerových (Kubernetes) aplikací a podpora jejich centralizované správy.
- **vSphere Lifecycle Manager (VLCM)** – automatizované nasazovaní aktualizací a patchů na VM.
- **Extension of elements** – navýšení maximálních limitů na systém – 768 CPU, 24TB RAM, 96 hostů na cluster, 2,500 ESXi, 45,000 VM a v linked modu 15,000 ESXi a 150,000 zapnutých VM.

Verze vSphere	Datum vydání	Konec podpory*
4.0	21.5.2009	21.3.2014
4.1	13.7.2010	21.3.2014
5.0	24.8.2011	24.8.2016
5.1	10.9.2012	24.8.2016
5.5	22.9.2013	19.9.2018
6.0	12.3.2015	12.3.2020
6.5	15.11.2016	15.11.2021
6.7	17.4.2018	15.10.2022
7.0	2.4.2020	–

* End of General Support (EoGS), End of Technical Guidance (EoTG) končí ještě později

Tabulka 2.1: Tabulka shrnující verze platformy vSphere, převzato z [31]

2.4.6 Bezpečnost

Obecně se vSphere považuje za velmi bezpečnou platformu a věří mu i velké společnosti. To ale nemusí znamenat, že jde o bezpečný software. Z pohledu zranitelnosti se ESXi pohybuje v žebříčcích National Vulnerability Database[13] velice vysoko. V seznamu nejzranitelnějších produktů roku 2020 zaujímá čtvrté místo[14]. Nutno ale podotknout, že umístění v tomto seznamu nemusí být přímo na škodu. Znamená to, že ESXi je velmi podrobně kontrolováno a chyby jsou reportovány. Každopádně se nejedná o tak bezchybný software, jak uvádí VMware v propagačních materiálech.

Mimo základní bezpečnost z pohledu virtualizace (striktní oddělení paměti jednotlivých virtuálních strojů, ...), poskytuje také několik dalších nástrojů pro zvýšení bezpečnosti, například:

- **Podpora Trusted Platform Module (TPM)** – podpora jak hardwarových TPM 2.0, tak virtualizací těchto zařízení (vTPMh).
- **FIPS 140-2, TLS 1.2** – soulad s bezpečnostním standardem a podpora šifrovaných komunikačních protokolů.
- **Virtualization Based Security** – podpora nových bezpečnostních funkcí pro Windows jako Credential Guard.

Dále také podporuje šifrování celých virtuálních strojů a disků. Při zcizení virtuálních úložišť se můžou útočníci jednoduše dostat k citlivým informacím. Zvlášť podstatné je šifrování i během vMotion, kdy může VM putovat mezi servery. Jedná se o velmi podstatnou funkci a celá tato práce se zaobírá právně tímto šifrováním. Podrobněji bude popsána níže v textu.

Ve verzi 7 přibyly dvě nové bezpečnostní funkce. vSphere Trust Authority (vTA) slouží jako autorita pro ověření a zajištění bezpečnosti. Podrobněji je popsána dále v této práci. Dále podpora Active Directory Federation Services (AD FS) pro zajištění řízení přístupu uživatelů.

Pro zvýšení bezpečnosti z pohledu ESXi jako operačního systému je vhodné si projít VMware ESXi 7.0 Benchmark. Jedná se o bezpečnostní zprávu renomované společnosti CIS (Center for Internet Security)[15]. Obsahuje popis pro správnou konfiguraci za účelem snížení množství zranitelností. V dokumentu se nacházejí dvě úrovně zabezpečení:

- **Level 1 (L1) – komerční sektor (obecné použití)** – měl by být nasazen jako výchozí konfigurace pro komerční použití, nastavení mají jasný bezpečnostní přínos a neomezují žádnou funkcionalitu.
- **Level 2 (L2) – vyšší bezpečnost pro prostředí s citlivými daty (limitovaná funkcí)** – obsahuje všechny prvky L1 a dále přidává další prvky, které mohou omezovat funkce a vzdálenou správu ESXi na úkor zvýšení zabezpečení.

Pokud budou v ESXi tak citlivá data že bude nutné VM šifrovat, doporučuje se zvážení nasazení úrovně L2.[10]

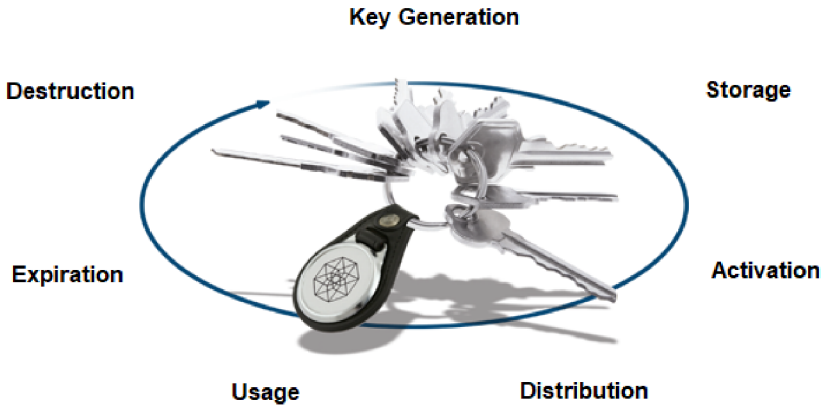
[13] https://nvd.nist.gov/vuln/data-feeds

[14] https://www.zive.cz/clanky/nejderavejsi-firmy-a-produkty-roku-2020-hlavne-windows-a-je-to-dobra-zprava/sc-3-a-207853/default.aspx

[15] https://www.cisecurity.org/

2.5 Nástroje pro správu klíčů

Dále je popsána teorie k části práce ohledně šifrování. Při šifrování je velmi podstatná samotná práce s klíči. Zkráceně se jedná o jejich vytvoření, distribuci, uložení a způsob použití. Důležité je také jejich případné zničení či jiné znehodnocení při vypršení platnosti nebo odcizení. Souhrnně lze tyto části pojmenovat jako životní cyklus jednotlivých klíčů (Key Lifecycle), který znázorňuje Obrázek 2.8. Je nutné podotknout, že všechny zmíněné funkce nespadají pouze do digitálního prostředí, ale používají se také ve fyzickém světě s reálnými klíči a zámky.



Obrázek 2.8: Životní cyklus klíčů, převzato z [11]

Touto problematikou se obecně zabývá Key Managment (správa klíčů). Každý šifrovací protokol, modul či aplikace tuto správu v určité podobě musí obsahovat. V závislosti na robustnosti řešení a požadavcích na bezpečnost je někdy vhodné použít pro správu klíčů samostatně oddělenou entitu. Tento prvek se někdy zkráceně pojmenovává KMS a nemusí být výlučně dodáván vývojářem šifrovací části. Tato zkratka má několik velice podobných významů a nemusí být vždy jasné, o který typ se právě jedná:

- **Key Management System**

  Představují komplexní řešení pro správu klíčů. Protože se může jednat také o zařízení pro fyzické klíče, jak lze vidět na Obrázku 2.9, bývají někdy přesněji označovány jako CKMS (Cryptographic Key Management System). Zde se zahrnují všechna zařízení nebo podsystémy, které mají přístup k nezašifrovanému klíči [19] . Tato definice je velmi obecná. Většinou bývají takto označovány komplexní produkty skládající se z vícero částí, například samostatného, někdy i specializovaného hardwaru, celého clusteru Key Management Serverů či uživatelských procedur a kryptografických protokolů.

- **Key Management Server**

  Samostatné servery (virtuální či fyzické). Komunikace s okolím je dosažena pomocí definovaných síťových rozhraní. Hlavním cílem je oddělit část pro správu klíčů od té, která je využívá. Vhodné pro střední až velké řešení. Podrobný popis je níže v textu.

Obrázek 2.9: Fyzický systém pro správu klíčů, převzato z [17]

- **Key Management Service**

  Jak už název napovídá, jedná se pouze o službu, která je hostována na serveru, který má ještě další účel, ne pouze správu klíčů. Výhodou tedy je ušetření výpočetních prostředků. To je ovšem na úkor bezpečnosti, protože další aplikace zvyšují plochu možného útoku. Takovéto řešení bývá poměrně časté a využívá se pro menší až střední infrastruktury. Funkcionalita těchto služeb může být už oproti výše zmíněným serverům značně menší. Například operační systémy Windows (desktopové i serverové verze) poskytují službu pro distribuci aktivačních klíčů[16].

- **Key Management Software**

  Do této kategorie lze v podstatě zařadit všechny softwarové části z výše zmíněných skupin. V praxi se ale většinou tímto termínem míní jednoúčelové uživatelské aplikace, které slouží pro uložení přihlašovacích údajů. Ty mohou pocházet z jedné aplikace (typicky webový prohlížeč), nebo mohou pracovat napříč celým OS. Před použitím přihlašovacích údajů bývá nutné zadat hlavní heslo. Přínosem těchto programů bývá uživatelská konformnost, kdy si nemusí pamatovat vícero hesel do různých aplikací. Na druhou stranu nesou velké riziko v podobě kompromitace jednoho hlavního hesla a s tím potenciálního zneužití všech uložených. Úroveň kvality jednotlivých řešení může být velmi různá (od nativních programů, které si nešifrují databázi klíčů, až po robustně zabezpečená úložiště pro citlivá data uživatelů).

Správa klíčů je při zabezpečení velice podstatná a nelze říct, že by v této oblasti bylo nějaké ideální řešení. Právě proto i zde panuje celkem živelný vývoj nových implementací. Pro ověření úrovně bezpečnosti slouží také různé certifikáty od nezávislých společností. Například NIST FIPS-140, která je popsána výše v 2.2. KMS nemusí striktně pracovat pouze s klíči. Může spravovat i další data citlivého charakteru, jako jsou certifikáty, tokeny či hesla.

[16] https://docs.microsoft.com/en-us/windows/deployment/volume-activation/activate-using-key-management-service-vamt

2.6 Key Management Server

Jak už z názvu vyplývá, jedná se o servery, které řeší správu klíčů. Měly by mít vždy pouze jeden účel, a to správu klíčů. Další aplikace by zmenšovaly bezpečnost serveru jako celku. Může se jednat jak o hardwarové servery, tak i servery softwarové. Fyzické servery jsou obecně bezpečnější volba. Měly by obsahovat patřičné bezpečnostní součástky a moduly. To ovšem nemusí být vždy pravda, a proto je vhodné ověřovat zařízení auditory. V praxi se výrobci snaží získat certifikaci pro bezpečnostní hardware. Ukázkou fyzických serverů jsou například KMES Series 3 (na Obrázku 2.12) nebo HPE Enterprise Secure Key Manager[17].

Obrázek 2.10: Fyzický KMS Futurex KMES Series 3, převzato z [14]

Software KMS lze rozdělit na dvě kategorie podle distribuce. Buďto je aplikace KMS dostupná jako program k instalaci, nebo jako virtual appliance (celé přednastavené VM). Při virtual appliance je výhodou to, že systém je celý nakonfigurován se zaměřením na bezpečnost. Většinou jde o opensource (unix-like) operační systém a samotnou aplikaci. Při distribuci ve formě programu je výhodou zase možnost si vše sám nastavit a zabezpečit dle typu nasazení. Pokud se nainstaluje na server, jenž by měl i jinou službu (např. DNS, DHCP, ...) spadá podle výše zmíněného rozdělení pod Key Management Service. Obecné schéma zapojení KMS je na Obrázku 2.11.

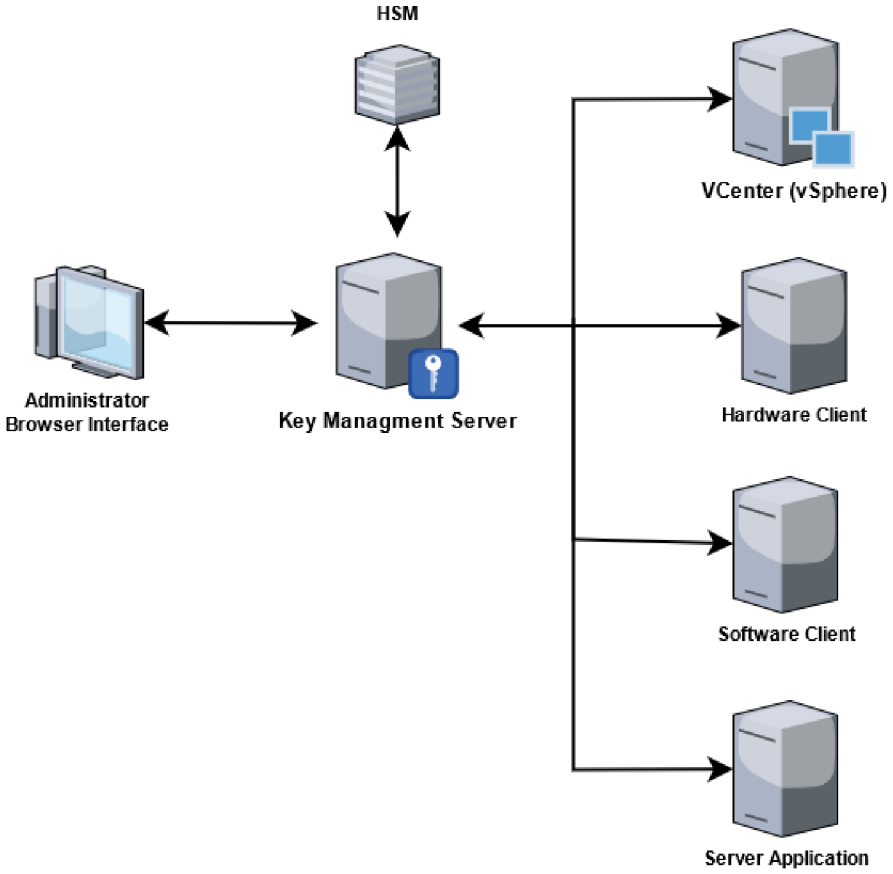
2.6.1 Motivace nasazení KMS

Cílem je oddělit použití klíčů (šifrování a dešifrování) od zbytku jejich správy. Aplikace, jenž používá klíče, implementuje pouze standardní šifrovací algoritmy. Tyto části jsou takřka bezchybové, protože jsou pevně definovány. Možné zranitelnosti převážně pramení ze špatné práce s klíči. Generování náhodných čísel pro klíče, autentizace přístupu či zajištění persistence klíčů, to jsou všechno funkce, které nemusí aplikace podchycovat a může se od nich oprostit.

Dále může být velice výhodné přesunout KMS do jiného segmentu sítě. Ten může mít jinou bezpečnostní politiku než samotná aplikace. Výhoda je hlavně v robustnosti a škálovatelnosti. Design KMS je určen ke komunikaci s více aplikacemi pomocí různých kanálů. Při zapojení do clusteru může komunikovat s velkým množstvím aplikací, což pak může být veliká výhoda centrální administrace.

Na druhou stranu musí být řešena zabezpečená komunikace mezi klientem a KMS. Při narušení komunikace útočníkem může dojít k odcizení či kompromitaci klíčů. Dále je nutné počítat s určitým vytížením zdrojů z pohledu hardwaru, jelikož se má jednat o jednoúčelový

[17] https://support.hpe.com/hpesc/public/docDisplay?docId=emr_na-c02907295



Obrázek 2.11: Obecné zapojení Key Managment Serveru do infrastruktury

server. Také musí být zajištěna vysoká dostupnost. Odstavení KMS povede k zablokování všech připojených aplikací.

Celkově se KMS hodí do středních až velkých architektur. Vyžadován může být pro extrémně citlivá data, u kterých je nutné získání certifikace. Jako koncept je velice perspektivní a podporuje ho i protokol KMIP, který je popsán níže v textu.

2.6.2 KMS a VMware

Šifrování VM je v prostředí vSphere dostupné až od verze 6.5 skrze Standard Key Provider. Verze 7.0 přináší další dva koncepty: Trusted Key Provider a od update 2 také Native Key Provider. V poslední verzi jsou tedy pro šifrování dostupné hned tři nástroje. Celá tato podkapitola je převzata z [31]

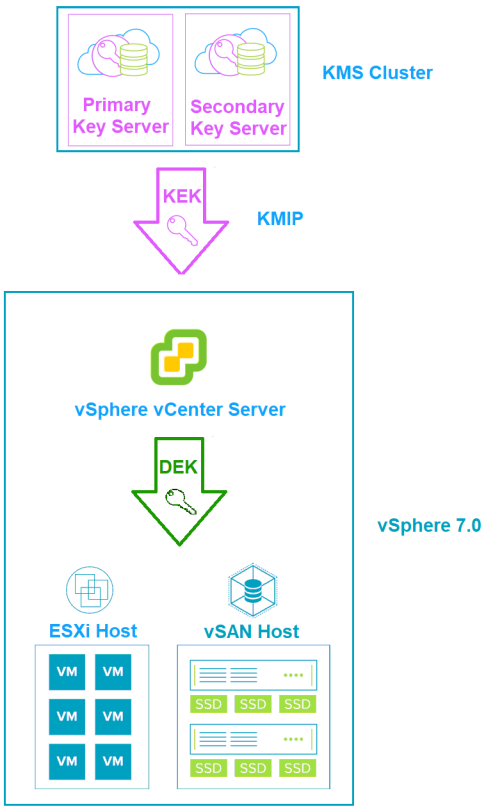
Standard Key Provider

Cíl této práce je implementace serveru, který by pracoval s vSphere v tomto módu. Vyžaduje tedy samostatný KMS. Komunikace probíhá pomocí protokolu KMIP verze 1.1 a vyšší. Při inicializaci spojení je nutné ustálit spojení (Establish Trust) mezi KMS a vCenter. V praxi to nejčastěji znamená nahrát TLS certifikáty KMS do vCenter. Protože se pracuje v lokální

síti, nemusí být dostupná věrohodná certifikační autorita. Samotný proces práce u Standard Key Provideru se skládá z několika kroků:

1. Ve vCenter vznikne šifrovací úloha, například zašifrování VM. Je vyslán požadavek do KMS pro nový klíč.
2. KMS vygeneruje nový asymetrický klíč AES-256, který bude sloužit jako KEK (Key Encryption Key).
3. vCenter si uloží ID klíče a přepošle jej dál na daný ESXi (host daného VM). Samotný klíč se neukládá, zaznamenává se pouze jeho ID.
4. ESXi vygeneruje vnitřní šifrovací klíče DEK (Data Encryption Key) pro všechny části VM (XTS-AES-256). Tyto klíče následně zašifruje pomocí přijatého KEK. DEK se ukládají pouze v zašifrované podobě. Při každém použití musí vždy poslat požadavek o KEK, aby se dostal k odšifrování DEK.
5. ESXi zašifruje samotné části VM pomocí DEK. Při přesunu se můžou přesouvat i DEK klíče, protože nejsou v čitelné podobě.

Popis obecně znázorňuje Obrázek 2.12. Obecně lze říci, že tento režim je vhodný pro ochranu dat ve středně velké infrastruktuře.

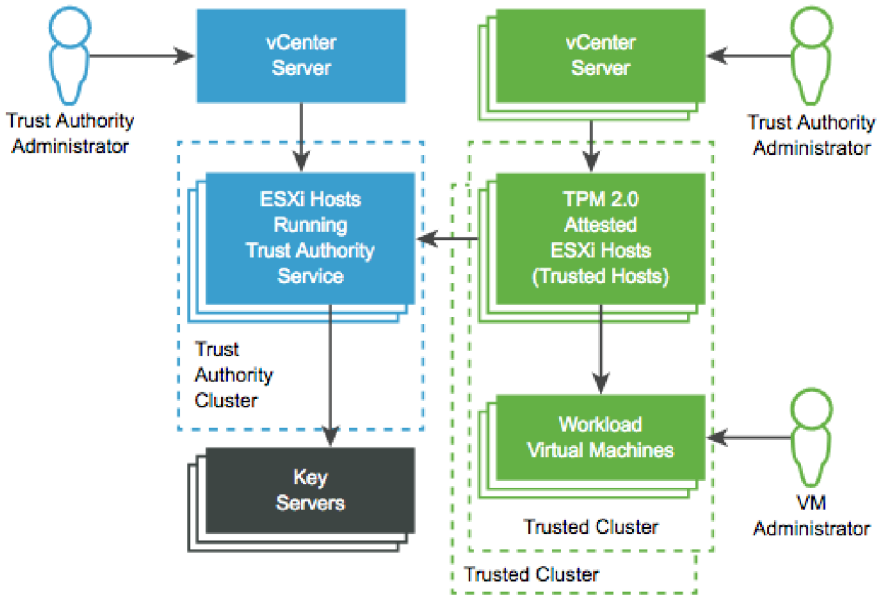


Obrázek 2.12: Proces zasílání klíčů z KMS pro prostředí vSphere, převzato a upraveno z [3]

Thrusted Key Provider

Nejprve je nutné vysvětlit, co znamená VMware Trust Authority (vTA). Jedná se o bezpečnostní nástroj pro ověření, zda daný ESXi host pracuje správně na svém hardwaru. Například při kompromitaci bootu ESXi a následném přenesení zašifrovaného VM na něj bude při spuštění VM dešifrován a poté dojde ke kompromitaci i tohoto stroje samotného. Proto se zavádí vTA, která ověřuje (attestation), zda je běžící software přímo od společnosti VMware nebo jejích partnerů.

Komunikace s KMS přechází skrze vTA a pouze pokud je ověřena bezpečnost. Schéma architektury znázorňuje Obrázek 2.13. Proces práce při Thrusted Key Provideru se skládá opět z několika kroků:



Obrázek 2.13: Schéma komunikace pro Thrusted Key Provider, převzato z [31]

1. vCenter z Trusted Clusteru zkontroluje, zda výchozí Thrusted Key Provider je dostupný pro daného ESXi hosta, který vytváří novou šifrovanou VM.
2. Pokud je dostupný, přidá vCenter Thrusted Key Provider do konfigurace VM.
3. Požadavek na tvorbu VM je zaslán ESXi.
4. Pokud ověřovací token (attestation token) není dostupný, tak je o něj požádáno v Attestation Service.
5. Key Provider Service zkontroluje ověřovací token a vytvoří KEK pomocí KMS. KEK je zašifrován pomocí primárního klíče servisy a poslán příslušnému ESXi.
6. ESXi vytvoří DEK pro danou VM, pomocí KEK a primárního klíče.
7. Pomocí DEK se zašifruje daná VM a všechny její části.

Obecně lze říci, že takovýto způsob ochrany je nutný až při opravdu velkých infrastrukturách (desítky serverů a clusterů, tisíce VM), kde se nacházejí extrémně citlivá data.

Native Key Provider

Tento nástroj poskytuje pro uživatele větší komfortnost, ale za cenu menší bezpečnosti. Není při něm nutné externí KMS tak jako u výše zmíněných. Pracuje pouze při podpoře TPM, které využívá pro práci s klíči. V podstatě vCenter sám vygeneruje globální klíč pro všechny VM, tedy KDK (Key Derivation Key). Tento globální klíč slouží k zašifrování všech DEK.

Tato možnost byla přidána ve verzi 7 update 2. Slouží hlavně pro zašifrování malého počtu vybraných VM. Využití samostatného KMS pouze pro tuto funkcionalitu by bylo zbytečně finančně náročné. Vhodné je tedy pro malé infrastruktury.

2.6.3 Existující KMS pro vSphere

Existující a podporované řešení shrnuje Tabulka 2.2. Produkty jsou uvedeny k březnu 2021. Aktualizovaný seznam se nachází zde[18]. Vybrané produkty jsou ještě detailně popsány níže. Pro všechny platí, že se jedná o placené produkty a není známo jak interně pracují.

Název společnosti	Produkt
Dell EMC	CloudLink
EntIT	ESKM
Fornetix LLC	Fornetix VauilCore
Fortanix, Inc	SDKMS
Fujitsu	Eternus SF KM
HashiCorp	HashiCorp Vaul
Hy Trust, Inc	HyTrust KeyKontrol
IBM	IBM Security Key Lifecycle Manager
IBM	KMIP for VMware on IBM Cloud
QuintessenceLabs	qCrypt v200
Thales eSecurity	G350v KeySecure for Government
Thales eSecurity	Vormetric Data Security Manager
Thales eSecurity	CipherTrust Manager
Townsend Security, Inc	Alliance Key Manager
Unbound Tech	Unbound Key control
Utimaco Inc	ESKM
Zettaset	Zettaset Key Manager

Tabulka 2.2: Tabulka shrnující podporované KMS

HyTrust KeyControl

Tento produkt vyvíjí stejnojmenná firma HyTrust, Inc. Zmíněná společnost byla založena roku 2009 a mezi hlavní investory patří VMware, Cisco či Intel. Zaměřuje se na bezpečnost v oblasti virtualizace a cloudových služeb. Jedním z předních produktů je právě KMS KeyControl.[15]

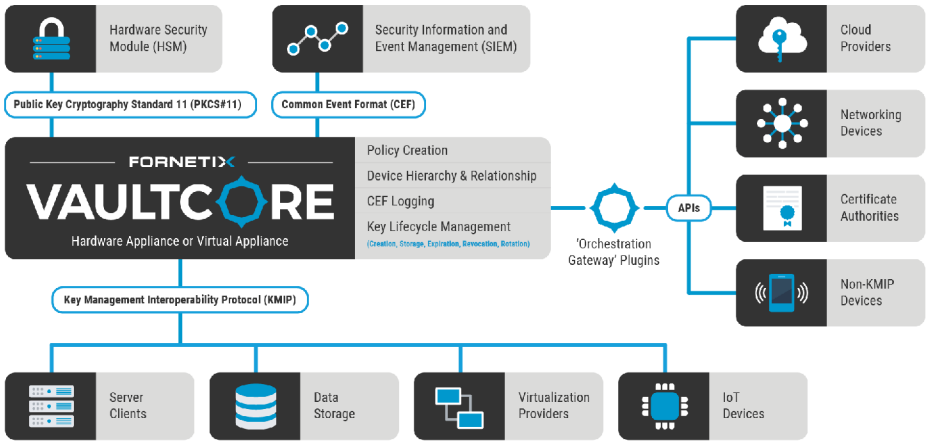
Na rozdíl od většiny výše zmíněných v Tabulce 2.2 je KeyControl dostupný v šedesátidenní trial verzi[19]. Jedná se o robustní řešení, které obsahuje mnoho funkcí. Mezi základní

[18] https://www.vmware.com/resources/compatibility/pdf/vi_kms_guide.pdf
[19] https://dl.hytrust.com/Publishing/Default.aspx?CollectionID=1IX9P3W2D0E7H

patří plná podpora standartu KMIP ve všech verzích a certifikace FIPS 140-2. Proto byl v této práci použit jako referenční řešení. Cena tohoto produktu není veřejná, k jejímu zjištění je nutné kontaktovat prodejce. Tento produkt byl na níže uvedeném testovacím prostředí úspěšně otestován.

Fornetix VaultCore

Fornetix je americká společnost zaměřená na vývoj Key Managment produktů. VaultCore je jejím hlavním produktem. Pyšní se opravdu velkým množstvím integrací. Krom KMIP protokolu umí komunikovat i s HSM či aplikacemi v kontejnerech. Integrační schéma je v Obrázku 2.14. Pro otestování byla společnost kontaktována za účelem získání trial verze, ovšem bez odpovědi. [13]



Obrázek 2.14: Schéma integrací produktu Fornetix VaultCore, převzato z [13]

2.6.4 Šifrování v jiných virtualizačních platformách

Pro úplnost je vhodné ještě stručně popsat, jak probíhá šifrování na jiných virtualizačních platformách, které již byly popsány výše.

Hyper-V

V tomto prostředí je šifrování řešeno přímo uvnitř virtuálních strojů. Hyper-V podporuje TPM a dokáže vytvořit vTPM pro každou VM. Šifrovací klíče tedy nejsou součástí VM. Guest může využít jakýkoliv šifrovacích programů, pro Windows je doporučován Bitlocker[20].

AWS

V cloudové infrastruktuře AWS je možné použít dva nástroje pro podporu šifrování. AWS Key Management Service[21] je služba podporující kompletní správu klíčů. Cena je 1$ měsíčně za každý uložený objekt (klíč, certifikát, ...). AWS CloudHSM[22], jak už z názvu napovídá, podporuje práci s HSM.

[20] https://community.windows.com/en-us/stories/what-is-bitlocker-windows-10
[21] https://aws.amazon.com/kms/
[22] https://aws.amazon.com/cloudhsm/

KVM

Podpora pro šifrování je zde obdobná jako pro výše zmíněný Hyper-V. Doporučuje se šifrovat na úrovni operačního systému hosta i guesta. Podpora je i virtualizace TPM. Nástroje vychází z typu samotného OS.

VirtualBox

Pro šifrovací funkci je nutný Extension Pack. Před šifrováním je nutné zadat uživatelské heslo, které slouží jako KEK. Na pozadí se vytvoří DEK (AES-256 nebo AES-128) a tím se zašifruje daná VM. DEK je uložen v konfiguračních souborech. Šifrované VM nelze exportovat, je nutné je nejprve dešifrovat. Z pohledu bezpečnosti se jedná o nejslabší zabezpečení ve srovnání s níže uvedenými postupy.

2.7 Key Management Interoperability Protocol (KMIP)

KMIP je komplexní protokol pro komunikaci mezi klienty, kteří požadují šifrovací klíče a servery, které tyto klíče spravují. Cílem KMIP je nahrazení redundantních, proprietárních a nekompatibilních protokolů pro správu klíčů. Principem je relativně jednoduchý nízkoúrovňový protokol, který lze použít k vyžádání a doručení klíčů mezi jakýmkoli KMS a jakýmkoli šifrovacím systémem (klientem).

Má volně dostupný standart a cílí na použití jednoho KMS pro vícero aplikací. Protokol vyvíjí Organization for the Advancement of Structured Information Standards (OASIS)[23], což je neziskové konsorcium s globálním rozsahem. Obsah celé této podkapitoly je až na vybrané části převzat z oficiální dokumentace dostupné na webu zde [21].

Protokol má již několik verzí, jak zobrazuje Tabulka 2.3. Zpětná kompatibilita je podporována pouze pro stejnou hlavní verzi. Zpětná podpora mezi hlavními verzemi není zaručena a záleží na tom, jak bude klient či server implementován. V praxi se používá zatím převážně verze 1, jelikož verze 2 byla zveřejněna relativně nedávno. Jelikož se protokol stále živelně vyvíjí, změny ve verzích bývají značné. V blízké budoucnosti se plánuje zahájení procesu na ISO certifikaci. VMware podporuje práci ve verzi 1.1 a vyšší. KMIP podporuje koncept profilů. Je tedy možné pro určité použití implementovat pouze zadanou část protokolu.

Verze protokolu	Rok vydání
1.0	2010
1.1	2013
1.2	2015
1.3	2017
1.4	2018
2.0	2019
2.1	2020
3.0	(*Draft 2021*)

Tabulka 2.3: Tabulka shrnující verze KMIP

[23] https://www.oasis-open.org

2.7.1 Objekty

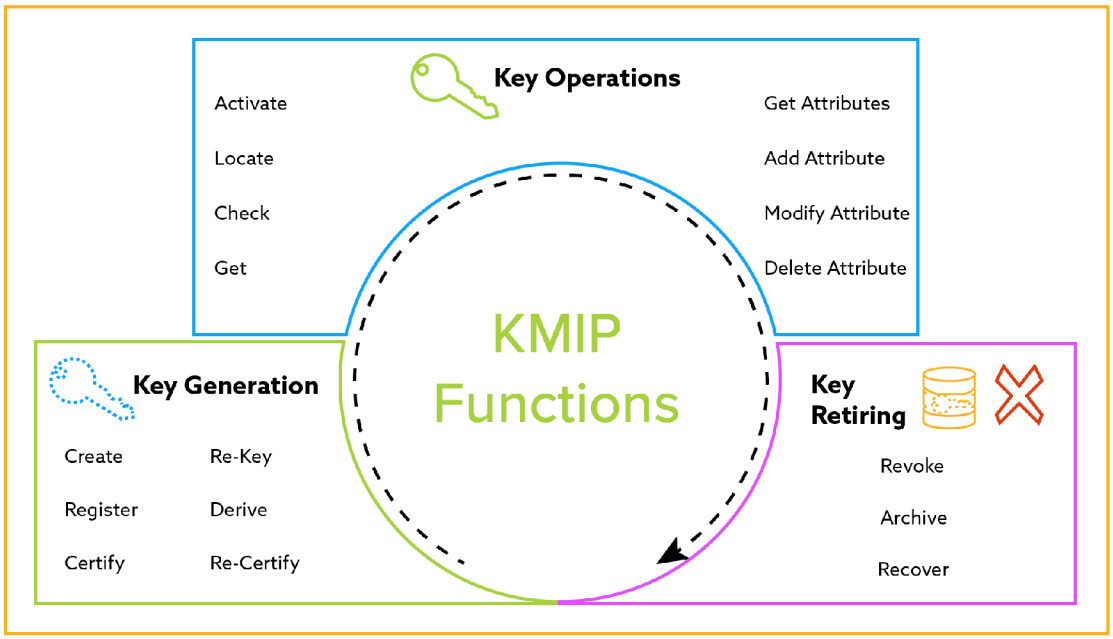
Podstatou komunikace je vždy nějaký serverem spravovaný objekt (Managed Object), nad kterým se vykonává některá z operací. Mezi typy takových objektů patří například:

- Symetrické klíče,
- Privátní a veřejné asymetrické klíče,
- Certifikáty,
- Pretty Good Privacy (PGP) klíče.

Typů objektů je ještě více a jsou podrobně popsány ve standartu. K objektům se přiřazují určité doprovodné atributy (Object Attributes). Pro šifrovací klíče to může být například použitý šifrovací algoritmus, délka klíče či datum aktivace. Opět jich je ale velké množství. Jedná se o metadata daného objektu. Jsou určeny množiny atributů, které jsou pro určité objekty povinné.

2.7.2 Operace

Samotný protokol podporuje širokou sadu operací nad objekty. Od těch nejpřímočařejších, které zajišťují životní cyklus klíče (Create, Get, Destroy, ...), přes sofistikovanější, jako RNG Seed (inicializace generátoru náhodných čísel), Hash (hashování zaslaných dat) nebo Import a Export. Zjednodušené schéma lze vidět na Obrázku 2.15.



Obrázek 2.15: Schéma KMIP operací, převzato z [2]

2.7.3 Komunikace

Komunikace probíhá skrze TLS. Zajištění důvěryhodnosti certifikátů pro komunikaci je nutné zajistit ručně. Podporovaný formát zpráv je TTLV (Tag, Type, Length, Value), HTTPS, JSON a XML. Záleží ovšem na zvoleném profilu. Podstata komunikace je taková, že klient pouze zadává požadavky na server a ten je zodpovídá. Komunikaci zahajuje tedy vždy klient.

Kapitola 3

Použité prostředí a návrh řešení

V této kapitole je podrobně popsáno testovací prostředí (hardware i software). Poté je soupis všech požadavků na práci. Závěr kapitoly obsahuje návrh celého řešení s podrobným popisem pro každou část sytému.

3.1 Popis testovacího prostředí

Testovací prostředí bylo poskytnuto společností Master Internet. V jejich infrastruktuře byl vyčleněný celý jeden fyzický server pouze pro tuto DP. Ke stroji se přistupovalo vzdáleně. Veškerá instalace VMware clusteru, jakožto i přidružených částí, byla již pro tuto práci předpřipravena. Nicméně vzhledem ke komplexnosti prostředí a samotné platformy vSphere trvalo nemalou dobu se s každou částí seznámit a zdokumentovat ji.

3.1.1 Použitý hardware

Pro testování byl přidělen server Dell PowerEdge R430. Přesněji v konfiguraci:

- CPU – 8x Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz,
- RAM – 64GB – 4x16GB DDR-4 2133 MHz,
- Řadič RAID – PERC H730 Mini,
- HDD – RAID 1 – 2x160GB SSD (operační systém), 2x1TB SATA (virtuální data),
- OS – ESXi 7.0 Update 1 – DellEMC Customized version.

Jedná se o základní rackový server určený pro datacentra. Je poměrně velmi škálovatelný a všestranný. Podstatné pro tuto práci je, že krom jiných funkcí je také certifikovaný pro virtualizaci pomocí VMware. [12]

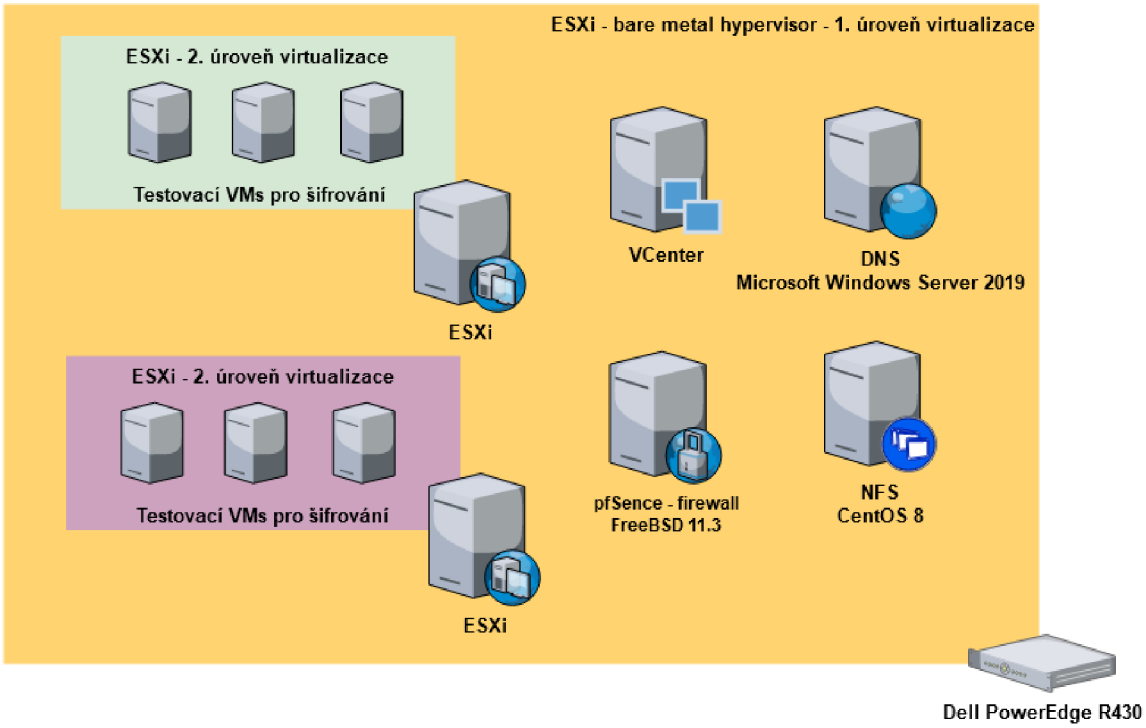
3.1.2 Použitý software

Na serveru byl předinstalován tento software, blíže je popsán níže:

- VMware ESXi 7.0.1 – 1x bare-metal hypervisor, 2x virtualizovaný pro vSphere cluster,
- VMware vCenter Server 7.0.1 – pro správu ESXi clusteru,

- DNS server – Microsoft Windows Server 2019,
- NFS – CentOS 8,
- Firewall – pfSence – FreeBSD 11.3.

Na Obrázku 3.1 je schéma popisující instalovaný software v jednotlivých vrstvách. Pro přehlednost jsou vytvořeny jednotlivé abstraktní úrovně, kde nultá odpovídá OS fyzického serveru. První je virtualizovaná přímo pod ni, tedy na hlavním ESXi a konečně druhou představují VM, které jsou pod podružnými ESXi v vSphere clusteru. VM v třetí úrovni budou sloužit pouze pro otestování procesu šifrování. Jelikož se může jednat o jakékoliv systémy, nejsou více popsány, ale pro přehlednost jsou na schématu znázorněny. Vytvářené KMS a jeho testovací verze byly instalovány na první úrovni (po boku vCenter).



Obrázek 3.1: Schéma instalovaného softwaru

Jednotlivé parametry virtuálních strojů nejsou podstatné a nikterak neovlivňují celkovou funkčnost. Všechny prvky v první úrovni jsou propojeny virtuální lokální sítí a je tedy zajištěna jejich vzájemná konektivita. Krom hlavního ESXi jsou všechny části virtualizované a níže popsané se nachází v první úrovni virtualizace.

ESXi

Na nejnižší úrovni (jako operační systém serveru) je instalován VMware ESXi 7.0 Update 1 – DellEMC Customized version[1] . Pod ním je virtualizován celý vSphere cluster, včetně dalších komponent, tedy první úroveň. V praxi by tato úroveň nebyla virtualizovaná. Zde je

[1] https://www.dell.com/support/manuals/cs-cz/vmware-esxi-7.x/vmware_7.0_icg_pub/introduction?guid=guid-3c867498-b577-4929-b7ee-fb4528e80fb2

pouze za cílem ušetření zdrojů. Tedy podřazené ESXi a vCenter by v produkčním prostředí byly instalovány přímo na hardware.

Dvě podřízené ESXi (1. úroveň) jsou zapojeny do vSphere clusteru. Nacházejí se v základní konfiguraci, není od nich potřeba žádná specializovaná funkce. Jejich počet může být různý, zde jsou použity dva. Umožňuje to simulaci přenosu VM mezi servery (vMotion).

vCenter

Představuje souřadný prvek pro celý vSphere cluster. Skrze vCenter se řídí a celý konfiguruje. Mimo mnoho jiného spravuje distribuci klíčů pro samotné šifrování. Jedná se tedy o nezbytný prvek pro tuto práci a nelze jej nahradit.

DNS server

Pro fungování všech komponent v clusteru je nezbytný funkční lokální překlad adres. Zde byl použit Microsoft Windows Server 2019, ale šel by nahradit jiným operačním systémem. Vybrán byl pro jednoduchou konfiguraci. V některých situacích může být také výhodné mít ve vnitřní síti stroj s grafickým rozhraním (všechny prvky VMware se spravují přes webové rozhraní).

Network File System (NFS)

Pro jednoduchost byl použit CentOS 8. Poskytuje úložiště pro oba virtuální ESXi. Ty zde sdílejí jeden Datastore. Zde se tedy budou nacházet testovací VM, které budou určeny k šifrování. Tato část by mohla být jakkoliv nahrazena jiným řešením.

Firewall

Pro zabezpečeni přístupu k celému prostředí je nasazen pfSence[2] jako virtuální firewall pod FreeBSD 13.1. Tato část je v infrastruktuře nedůležitá z pohledu KMS a šifrování VM.

3.2 Specifikace požadavků

Cílem práce je vytvořit softwarový produkt pro správu klíčů ve spolupráci s externí firmou. Je tedy nezbytné využít některý model pro jeho vývoj. Vzhledem k tomu, že celá práce je individuální, byl vybrán klasický vodopádový model. Jednotlivé kroky tedy jdou konkrétně v tomto pořadí:

- Návrh – specifikace požadavků, architektonický návrh,
- Implementace – samotný vývoj KMS ,
- Testování – jak systémové tak akceptační (ze strany Master Internet),
- Předání – odevzdání diplomové práce a nasazení produktu.

Minimálně dokončení každé fáze se konzultovalo s vedoucím práce doktorem Kamilem Malinkou a s firmou Master Internet, s.r.o.

[2] https://www.pfsense.org/

3.2.1 Popis požadavků

Tato sekce obsahuje specifikaci požadavků na systém, který je produktem této práce. Vznikla ve spolupráci se společností Master Internet, s.r.o. Skládá se pouze ze slovního popisu a shrnutí faktů v Tabulce 3.1 níže. Požadavky jsou pouze obecné, jelikož podrobnější procesy jsou přesně dané a vycházejí z textu výše (KMS, vSphere, KMIP).

Funkční požadavky

Cílem je implementovat funkční Key Managment Server pro prostředí vSphere 7.0 se základní funkcionalitou. Konkrétně tedy bude možné jej připojit k vCenter a distribuovat klíče pro šifrování jednotlivých VM či vSAN. Pro ustálení bezpečného spojení (establish trust relationship) bude nutné do vCenter nahrát certifikáty KMS (obdobně jako u HyTrust KeyControl). KMS bude zprostředkovávat veškeré funkce pro správu klíčů (KEK). Systém bude jednoúčelový, nebude poskytovat žádné další funkce. Je kladen velký důraz na bezpečnost celého řešení, protože systém bude zpravovat citlivé údaje (šifrovací klíče).

Systémové požadavky

KMS bude předáváno ve formě virtual appliance ve formátu OVA či OVF. Bude tedy připraven pro spuštění ve virtuálním prostředí. Minimální hardwarové požadavky představují 2 virtuální procesory, 8 GB RAM a 50 GB paměti. Tyto požadavky vycházejí hlavně z minimálních požadavků na samotný hostitelský operační systém, který bude z rodiny GNU/Linux. Server nebude pracovat s Trusted Platform Module (TPM) žádného druhu. Server bude dostupný pouze z vnitřní sítě, tím pádem nebude dostupný z internetu, a to z důvodů bezpečnosti.

Požadavky na rozhraní

Server bude komunikovat s vCenter pomocí KMIP a to ve verzi 1.1 či vyšší. Port pro tuto službu bude konfigurovatelný (výchozí nastavení 5696). KMS bude podporovat komunikaci pouze s jedním vCenter serverem. Také není plánovaná žádná komunikace s dalším KMS serverem pro sdílení klíčů, případně zapojení do KMS clusteru.

Přístup přímo na server bude pomocí standardního SSH. Uživatelské rozhraní aplikace bude pomocí CLI. Krom konektivity s vCenter serverem bude také dostupný repozitář daného OS pro zajištění aktualizací systému. Server má být dostupný pro co nejmenší část sítě pro zvýšení bezpečnosti. Znázornění je na Obrázku 3.2.

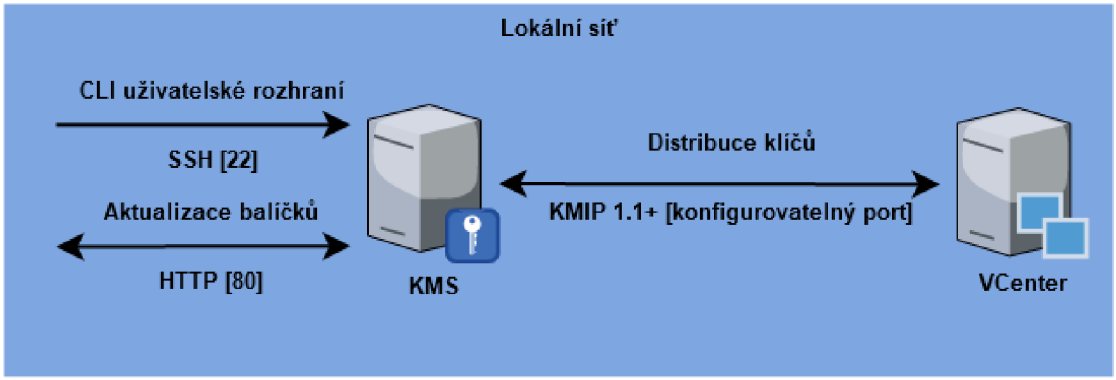
Aplikační požadavky

Samotná serverová služba pro správu klíčů bude psána v jazyku Python. Pro komunikaci pomocí KMIP bude využita knihovna PyKMIP[3]. Pro generování šifrovacích klíčů bude použita doporučená kryptografická knihovna daného jazyka. Samotné klíče se budou ukládat do zabezpečené databáze.

Další požadavky

Pro zprovoznění služby bude nutné stroj importovat do virtuálního prostředí. Dále na jednom místě nakonfigurovat síťové rozhraní (IP adresu, masku sítě, ...) a případně zapnout

[3] https://github.com/OpenKMIP/PyKMIP



Obrázek 3.2: Schéma požadavků na rozhraní KMS

inicializační skript. Po těchto úkonech bude již služba dostupná. Aplikace bude dále umožňovat další úpravy chování pomocí konfiguračních souborů.

Celý operační systém bude nastavený tak, aby byl co možná nejvíce bezúdržbový. Minimálně bude mít nastaveny automatické aktualizace systému, včetně plánovaných restartů či nastavení úrovně ukládání systémových a aplikačních logů. Operační systém také bude obsahovat pouze minimální možnou množinu aplikačních balíčků potřebnou pro provoz samotné aplikace. Nástroj pro zálohování nebude implementován, pouze navrhnut.

Pro zvýšení zabezpečeni bude server také mít nastavený firewall, který bude povolovat pouze minimální nutnou komunikaci pro správný chod serveru. Připojení pomocí SSH bude také ošetřeno tak, aby poskytovalo větší bezpečí. Součástí řešení bude krom textu této diplomové práce také stručná instalační dokumentace, jenž je v příloze B.

Požadavky pro budoucí rozšíření

Při návrhu a implementaci se musí myslet také na budoucí rozvoj projektu. Konkrétně na možnost zapojení do KMS clusteru, což představuje společné sdílení klíčů s dalšími servery. Další částí je podpora služby pro více aplikací, a to nejen ty od firmy VMware. KMS bude komunikovat pomocí KMIP a ten podporuje široká škála produktů. Výhledově také nebude komfortní používat CLI jako uživatelského rozhraní. Ideálním řešením by bylo webové rozhraní. A poslední, ne méně důležitou oblastí je podpora vTPM, případně dalších speciálních modulů přínosných pro správu klíčů.

3.2.2 Celkový souhrn požadavků

Vlastnost systému	Zvolené řešení
Účel	KMS pro vCenter, šifrování jednotlivých VM a vSAN
Připojené aplikace	1 server vCenter*
Formát systému	Virtual appliance (OVA/OVF)
vCPU	2
RAM	8 GB
HDD	50 GB
OS	GNU/Linux
Podpora TPM/vTPM	ne*
Distribuce klíčů	KMIP verze 1.1+
Konektivita	pouze LAN, WAN ne
KMS cluster	ne*
Uživatelské rozhraní	CLI*
Serverová implementace	python, knihovna PyKMIP
Ukládání klíčů	zabezpečená databáze
Generování klíčů	Doporučená kryptografická knihovna pro python
Údržba a konfigurace	minimální, vše připravit
Zálohování	ne**
Firewall	ano

* Při návrhu se předpokládá, že se požadavek při budoucím rozšiřování bude měnit.

** Nebude implementováno, ale bude popsán návrh možného řešení.

Tabulka 3.1: Tabulka shrnující požadavky na KMS

3.3 Architektonický návrh

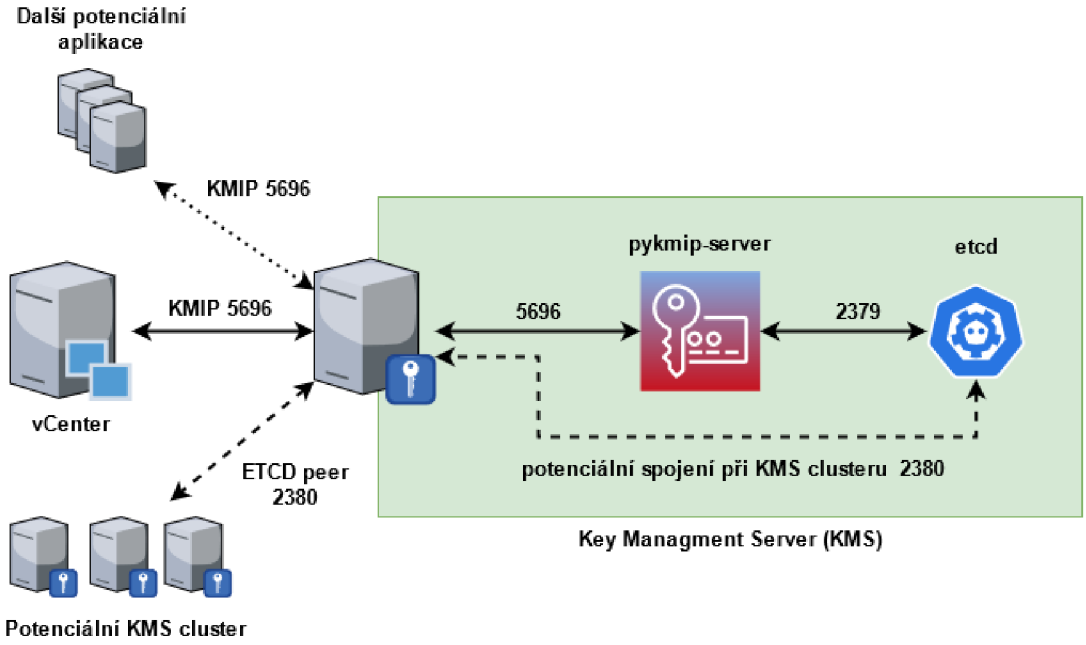
V této podkapitole je nejprve obecněji a poté podrobně popsán návrh jednotlivých částí tvořeného KMS. Vychází z předchozího textu, kde byly stanoveny požadavky. Zde budou uvedeny přímo použité technologie, včetně způsobu jejich propojení a další potřebné detaily.

Jako retenční řešení pro vytvářené KMS byl produkt HyTrust KeyControl, viz 2.6.3. Při návrhu bylo cíleno na jednoduchost a jednoúčelovost řešení. Vzhledem ke komplexnosti problematiky se cílí hlavně na využití volně dostupných nástrojů než na implementaci vlastních.

3.3.1 Souhrnný přehled

V této sekci je obecně popsán návrh KMS. Každá část je podrobněji rozepsána níže. Systém se bude skládat ze tří hlavních částí a to samotného operačního systému (Ubuntu 20.04), PyKMIP aplikace a ETCD úložiště klíčů. Operační systém bude nakonfigurovaný tak aby poskytoval zvýšenou bezpečnost a pouze jednu funkci a to správu klíčů. Tu bude zajišťovat PyKMIP aplikace, která bude poslouchat na portu 5696 (výchozí pro KMIP). Lokálně bude komunikovat s ETCD úložištěm na portu 2379. Schématický popis je v Obrázku 3.3.

Dále jsou také znázorněny budoucí možné komunikace. Server bude moct na portu 5696 obsluhovat více aplikací než jen jeden vCenter. Dále je navrženo zapojení do KMS clusteru



Obrázek 3.3: Schéma KMS a jeho komunikace

což by představovalo komunikaci na portu 2380 kde by si ETCD úložiště sdílely všechny klíče. Tyto rozšíření budou jednoduše nasaditelná ale tato práce je nebude obsahovat. Nejprve je nutné vytvořit stabilní a bezpečnou práci v poměru 1:1 (klient:KMS) a vše řádně otestovat.

3.3.2 Operační systém

Je nutné zvolit vhodný operační systém. GNU/Linux serverových distribucí je velice mnoho, je tedy z čeho vybírat. Podstatným kritériem je bezpečnost a jednoduchá udržitelnost. Na základě aktuálních článků[4][5][6] byl seznam zúžen na Ubuntu, CentOS a Debian. Z nich bylo vybráno Ubuntu kvůli široké podpoře, rychlým aktualizacím a hlavně již výše zmíněné certifikaci FIPS 140-2.

Bohužel certifikované moduly Ubuntu jsou placené a také jsou pouze pro verzi 18.04. Pro práci bude použita momentálně nejaktuálnější LTS (Long-term support) verze tedy 20.04.1 v serverové instalaci. Samotná instalace proběhne z oficiálního obrazu ze stránek Ubuntu[7]. Instalace bude čistá, nebude obsahovat žádné další než základní balíčky vyjma SSH. Hardwarové požadavky budou odpovídat těm uvedených v systémových požadavcích výše.

Po instalaci bude proveden tzv. hardening serveru. Konkrétně budou nastaveny automatické aktualizace systému včetně pravidelných restartů. Následně odstranění nežádoucích balíčků nepotřebných pro základní běh systému. Poté nastavení logování celého OS včetně

[4] https://www.ubuntupit.com/best-linux-server-distro-top-10-compared-recommendation/
[5] https://www.techradar.com/best/best-linux-server-distro
[6] https://www.tecmint.com/10-best-linux-server-distributions/
[7] https://ubuntu.com/download/server

pomocných skriptů na jejich archivaci či mazání. Všechny tyto nastavení budou precizně zdokumentovány a půjde je jednoduše upravovat, případně vypnout.

3.3.3 Síťové rozhraní

Součástí hardeningu bude také konfigurace firewall pravidel a úpravy přihlašování pomocí SSH. Firewall pravidla budou povolovat pouze základní komunikaci, přesněji:

- INPUT – povolen loopback, ustálené TCP a UDP, ICMP, SSH, HTTP a port KMS,
- FORWARD – vše zakázáno,
- OUTPUT – vše povoleno.

Připojení pomocí SSH bude povoleno pouze pro určitého (ne root) uživatele. Také bude vyžadováno přihlašování pomocí klíče. Síťové rozhraní bude statické, tedy ne pomocí DHCP. Po spuštění systému jej bude nutné nastavit podle lokální sítě.

3.3.4 PyKMIP KMS

Samotný KMIP server bude implementován v jazyce python, pomocí knihovny PyKMIP. Jedná se o knihovnu zveřejněnou pod Apache Licence 2.0[8]. Převážně je vyvíjena Peterem Hamiltonem[9], softwarovým inženýrem na Johns Hopkins University. Jedná se o jedinou komplexní a udržovanou implementaci KMIP protokolu. Právě proto byla vybrána do této práce.

Obsahuje částečné řešení serverové a klientské části protokolu. Využita bude pouze serverová část, klientská bude použita při vývoji a testování. Pro nasazení bude muset být upravena, a to především v oblasti ukládání klíčů. Podrobně prozkoumána a případně i upravena bude část ohledně generování klíčů a celková práce s kryptografickým modulem.

3.3.5 Úložiště klíčů

Pro ukládání klíčů je nutné připravit zabezpečené úložiště. Z požadavků na budoucí rozšíření je také nutné počítat s budoucím sdílením na jiné stroje. Přesněji jsou tedy kladeny extrémní nároky na:

- Bezpečnost – klíče jsou velice citlivé data, musí být chráněny.
- Dostupnost – klíče musí být dostupné v co největší možné míře.
- Konzistenci – není přípustné ztratit ani jeden klíč.
- Replikaci – při zapojení do clusteru musí všechny KMS vracet stejné klíče.

Každý klíč bude opatřen svým ID, takže budou v databázi tvořit pár. Samotný klíč bude ve formátu sekvence bitů o větší délce (stovky až jednotky tisíc). Tradiční SQL databáze není pro takováto data vhodná. Lepší pro použití bude NoSQL databáze. Z povahy dat bude výhodné použít úložiště typu klíč-hodnota, konkrétně MongoDB nebo ETCD. Druhé

[8] https://www.apache.org/licenses/LICENSE-2.0
[9] https://github.com/PeterHamilton

zmíněné ETCD má lepší potenciál pro replikaci a zapojení do clusteru, proto bylo vybráno pro tuto práci.

Jedná se o rychlé a spolehlivé úložiště. Ukládá hodnoty ve formátu klíč-hodnota. Interně tvoří b+stromy. Velká výhoda je v zapojení do clusteru, kde k odpovědi využívá hlasování o odpovědi (Quorum). Využívá se především pro ukládání konfiguračních souborů pro kontejnerové aplikace (Kubernetes). V třetí verzi `etcd v3` přichází z přepracovaným a bezpečnějším způsobem autentizace.[28]

Úložiště bude na serveru předem připravené. Nebude nutné jej před spuštěním nikterak konfigurovat. Integrace bude přímo ve zdrojovém kódu, nebude určená pro uživatelský přístup. Přihlašování bude realizované na základě certifikátu, který bude uložen přímo na serveru. Jednoznačný identifikátor bude ukládán jako klíč (key) a šifrovací klíč jako jeho hodnota (value). Získat více informací o ETCD lze v oficiální dokumentaci nebo také skrze interaktivní prostředí[10]

[10] http://play.etcd.io/home

Kapitola 4

Implementace a vyhodnocení

V této kapitole je popsána celá praktická část práce. Od samotné implementace, přes testování až k shrnutí výsledků.

4.1 Popis implementace

V této podkapitole jsou podrobně rozepsány jednotlivé kroky, které byly prováděny při vývoji nového KMS. Jsou seřazeny podle logické návaznosti. Uvedeny jsou všechny úpravy, které byly provedeny.

4.1.1 Operační systém

Pro zajištění bezpečnosti aplikace je nezbytné mít správně zabezpečený samotný operační systém. Pro instalaci bylo použito standardní ISO ze stránek Ubuntu[1]. Celý postup instalace je zdokumentován a přiložen k práci. Hardwarové nároky na VM se oproti návrhu zmenšily na 2 CPU, 4GB RAM a 10 GB pevný disk.

Pro validaci konfigurace a všech úprav serveru je potřeba udělat celý proces změn transparentní. Jelikož je možné, že bude výsledek práce použit v produkčním prostředí, je opravdu nutné tuto podmínku splnit. Proto jsou všechny úpravy provedeny ve formě Bash skriptů. Zde je jejich zkrácený popis. Jejich použití je blíže popsáno v příloze B. Skripty nejsou komplikované a dobře okomentované. Pro zjištění podrobných informaci o úpravách systému je vhodné si je projít.

Součástí práce měl být i hardening operačního systému. K němu došlo jen z části z časových důvodů. Ošetřen byl jen firewall a připojení pomocí SSH (podrobněji níže). Pro další činnost byla připravena publikace zabývající se touto problematikou [9]. Jedná se o renomovanou zprávu od společnosti CIS[2] obsahující i postup ohledně jednotlivých úprav. Před produkčním nasazením by bylo vhodné nasadit alespoň úroveň L1-server.

00-network.sh

Slouží pro nastavení síťového rozhraní. Podle zadaných hodnot vyplní konfigurační soubor `/etc/netplan/00-installer-config.yaml`. Je možné vybrat DHCP, případně vyplnit všechny potřebné údaje (IP adresu, gateway, DNS).

[1] http://cz.releases.ubuntu.com/releases/20.04/ubuntu-20.04.2-live-server-amd64.iso
SHA256 `d1f2bf834bbe9bb43faf16f9be992a6f3935e65be0edece1dee2aa6eb1767423`

[2] https://www.cisecurity.org/

01-system.sh

Nakonfiguruje celý operační systém. Nejprve systém aktualizuje a nainstaluje potřebné systémové balíčky. Dále zakáže IPv6 a opraví systémové chyby. Vyžádá si po uživateli SSH klíče a nakonfiguruje podmínky připojení (povolení pouze SSH2, povolení připojení pouze uživateli `kms`,...)[3]. Poté nastaví firewall tak, jak bylo popsáno v návrhu. Přidá také povolovací pravidlo na port 5696 (aplikace KMS), na který povolí pouze jednu uživatelem zadanou IP adresu (adresa vCenter).

Následně nastaví automatické aktualizace. Lze nastavit hned několik proměnných. Například čas vyhledávání aktualizací a možnou odchylku od tohoto času. Dále zda má docházet k automatickému restartování a případně v jaký čas. Na závěr celého skriptu je vytvořen nový uživatel `pykmip-server-user`, pod kterým se bude spouštět aplikace serveru. Pro projevení některých těchto změn je nutný restart stroje.

02-aplikace.sh

Jedná se o poslední z inicializačních skriptů. Slouží k instalaci všech potřebných komponent pro chod KMS. Nejprve nainstaluje ETCD a upraví jeho výchozí konfiguraci. Pokračuje vygenerováním přihlašovacích údajů do ETCD. Uloží je do souborů v adresářích `/root/` a `/home/pykmip-server-user/.secrets/`. Jelikož se za uživatele `pykmip-server-user` nelze přihlásit, pro zjištění těchto hesel by bylo nutné získat `root` oprávnění.

V dalším kroku je instalován python ve verzi 3.9. Veškeré balíčky potřebné pro běh PyKMIP serveru se nejprve stáhnou (pro případ porušení závislostí a opravy) a následně se nainstalují. Poté je celý PyKMIP nakonfigurován včetně vygenerování TLS certifikátů. Na závěr vytvoří pro aplikaci vlastní službu (service) a vše spustí. Po dokončení tohoto skriptu je KMS připraveno k práci.

4.1.2 PyKMIP server

Další část úprav se týká již samotné aplikace. Je psaná v jazyce python ve verzi 3.9.0+. PyKMIP byl instalován ve verzi 0.10, etcd3 ve verzi 0.12. Verze ostatních závislých balíčků lze vypsat pomocí příkazu `pip3 freeze`.

Kryptografický modul

Při práci s klíči je nutné využívat kvalitní a ověřený kryptografický modul. Prostředí python pro tento účel nabízí velké množství možností[4]. Na druhou stranu se při velkém výběru ukazuje problém, kterou zvolit. Konkrétně PyKMIP využívá knihovnu cryptography.io[5]. Z analýzy této problematiky vzešlo, že nebude nutné tuto knihovnu vyměnit za jinou.

Na otázku, která kryptografická knihovna nabízí nejbezpečnější funkcionalitu, lze nalézt mnoho odpovědí. Problematická se ukazuje ovšem jejich věrohodnost. Jako ověřený zdroj se ukázal komplexní článek na toto téma [4]. Na škodu je pouze stáří dokumentu (2016). Z jeho výsledků vychází cryptography.io velice dobře. Také z jiných zdrojů (fóra, blogy či diskuze) nebyl nalezen žádný důkaz, že by cryptography.io mělo s bezpečností problémy. Vývoj je stále aktivní a v přehledu využívání se pohybuje v předních pozicích[6]. Pro vše údaje výše zmíněné bylo rozhodnuto tuto knihovnu ponechat v implementaci.

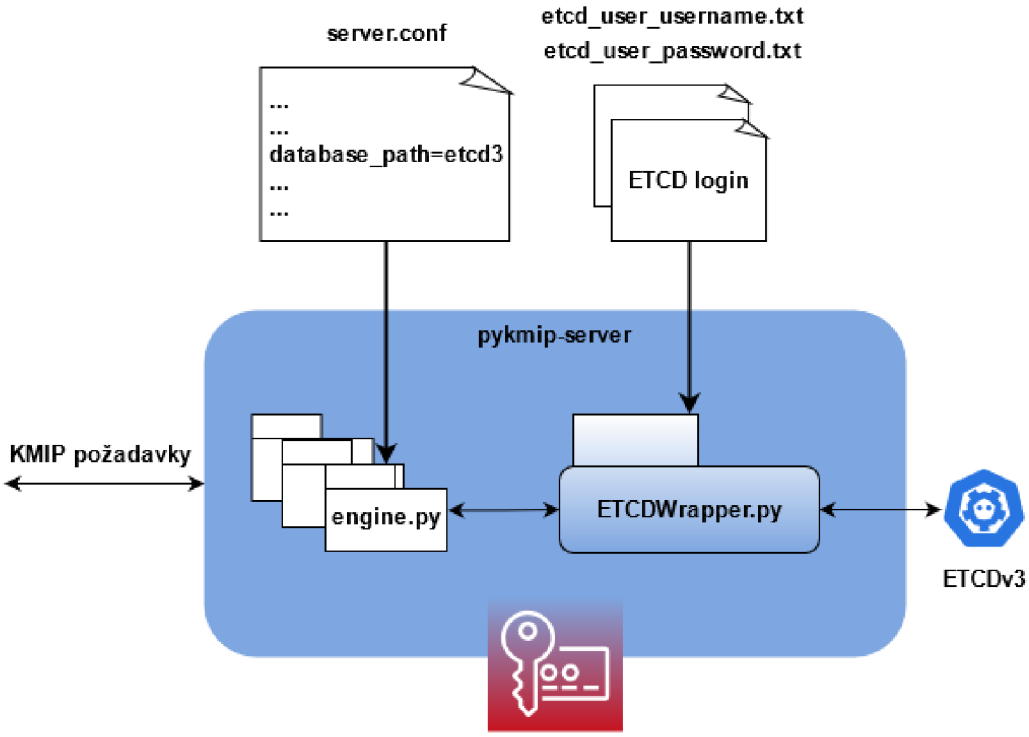
[3] https://www.sshaudit.com/hardening_guides.html#ubuntu_20_04_lts
[4] https://www.hackingloops.com/python-libraries-for-cybersecurity-development/
[5] https://cryptography.io/en/latest/
[6] https://hugovk.github.io/top-pypi-packages/, https://pythonwheels.com/

Úložiště klíčů

V předchozí kapitole byly stanoveny určité požadavky na úložiště klíčů. PyKMIP používá pro ukládání databázi SQLite[7], která jim ovšem nevyhovuje. Jedná se odlehčenou, opensource a multiplatformní SQL databázi. Nevyhovující je ale z pohledu zabezpečení a případné replikace při zapojení do clusteru. Jako vhodná platforma byla vybrána ETCD.

Bylo tedy nutné upravit práci s daty přímo v knihovně PyKMIP. Při bližším pohledu se ukázalo, že se neukládají pouze klíče, jak bylo popsáno v kapitole 3. Ve standartu KMIP má každý objekt ještě jasně definovanou množinu metadat. Ty se také ukládají a tvoří se tedy určité datové struktury. K nim přidává PyKMIP ještě vybrané systémové proměnné.

Veškerou práci s ukládáním do databáze obstarává jeden modul `engine.py`. Ten byl upraven tak, aby dokázal pracovat i s ETCD pomocí přesměrování požadavků na datový wrapper `ETCDwrapper.py`. Byla ovšem zachována i možnost využití SQLite. V konfiguračním souboru PyKMIP serveru je parametr pro nastavení cesty k SQLite databázi. ETCD bude použito, pokud se zadá jako hodnota `etcd3`. Schéma práce aplikační vrstvy je vidět v Obrázku 4.1. Jedná se o detailní pohled na část Obrázku 3.3.



Obrázek 4.1: Schéma ukládání aplikace do ETCD

Kvůli zachování kompatibility a komplexnosti dat se data ukládají v serializované podobě. Pro přímou práci s ETCD byl napsán nový modul `ETCDwrapper.py`. Ten využívá volně dostupný balíček `etcd3`[8]. Pro serializaci byl použit balíček `pickle`[9]. Wrapper obsahuje šest funkcí (connect, add, delete, find, find_all, update), které jsou dostatečné pro veškerou práci s novým úložištěm.

[7] https://www.sqlite.org/index.html
[8] https://pypi.org/project/etcd3/
[9] https://docs.python.org/3/library/pickle.html#module-pickle

ETCD je spouštěna jako služba a spouští se pod uživatelem `etcd`. Konfigurační soubor je `/etc/etcd/etcd.conf`[10]. Nastaveno je přihlašování pomocí uživatelského jména a hesla. Heslo se generuje v inicializačním skriptu a oba údaje jsou uloženy v souborech v adresáři `/home/pykmip-server-user/.secrets`. Samotná data se ukládají do adresáře `/var/lib/etcd/kms_data/`.

Komunikace by mohla probíhat šifrovaně pomocí TLS. Tento přístup by ale nepřispěl ke zvýšení bezpečnosti. K přihlašovacím údajům má přístup uživatel `pykmip-server-user`, případně `root`. K TLS certifikátům by bylo nastaveno stejné zabezpečení. Při prolomení do jednoho z těchto dvou uživatelů by došlo k prozrazení jak certifikátů, tak hesla. Oba případy by vedly k plnému získání kontroly nad úložištěm. Jelikož komunikace probíhá pouze lokálně, pro odposlechnutí by také bylo nutné získat práva uživatele `root`.

Logy KMS

Pro tvorbu logů je použita standardní knihovna `logger`[11]. Aplikace automaticky vytváří soubor s logem v adresáři `/var/log/pykmip/`. Velikost tohoto souboru je limitována na 1 MB. Při překročení se stávající log přejmenuje a vytvoří se nový soubor. Takto je uchováváno nejaktuálnějších 5 souborů. Starší soubory jsou mazány. Ve výchozím stavu je nastavena úroveň logu na hodnotu `INFO`.

Logy operačního systému jsou ponechány ve výchozí konfiguraci. Tvoří je služba `rsyslog` a konfigurační soubor je `/etc/rsyslog.d/50-default.conf`. Systémové logy se ukládají do adresáře `/var/log/`.

Problémy v komunikaci s vCenter

Při komunikaci s vCenter je v určitém bodě při vytváření nového klíče vyvolána výjimka. PyKMIP neimplementuje operaci `create_add_attribute_payload`. Nicméně na funkčnosti celého systému to nic nemění. Je pravděpodobné, že vCenter si chce ke klíči v KMS nahrát vlastní metadata. Při případném rozšiřovaní této práce by bylo vhodné tento nedostatek blíže prozkoumat a dodělat.

4.1.3 Zálohování

Ztráta klíčů by měla na KMS katastrofální následky. Proto je nutné obsah ETCD úložiště v produkčním nasazení patřičně zabezpečit proti ztrátě. Postupy jsou v podstatě dva. Zapojení ETCD do clusteru a zálohování obsahu. Při zapojení N strojů do clusteru je úložiště odolné proti selhání až $(N-1)/2$ členů. Pokud dojde k výpadu více prvků, dojde k výpadku hlasovacího kvora a odstavení celého clusteru.

Druhou možností je tvorba snapshotů úložiště. Jedná se o export celého úložiště do datového souboru. Tento soubor lze uchovávat na zařízeních, které nejsou v ETCD clusteru. Tvorba je možná jak na vyžádání, tak periodicky (podle počtu záznamů nebo časově). V praxi by mohlo být vhodné použít oba nástroje, pokud to infrastruktura dovoluje a je to smysluplné. Tato podkapitola je převzata z oficiální dokumentace [28]. Podrobné informace a postupy jsou zde[12].

[10] https://etcd.io/docs/v3.4/op-guide/configuration/
[11] https://docs.python.org/3/library/logging.html
[12] https://etcd.io/docs/v3.4/op-guide/recovery/

4.1.4 Budoucí rozšíření

Je také vhodné podrobněji popsat postup pro budoucí rozšíření. Před jejich tvorbou je ale nutné ještě celkově otestovat a prověřit bezpečnost vytvořeného řešení nad rámec této práce.

Prvním možným rozšířením je tvorba KMS clusteru. To představuje úpravu konfigurace ETCD v souboru `/etc/etcd/etcd.conf` tak, aby sdílela úložiště s dalšími servery. Jedná se o výchozí funkcionalitu, takže by neměla být nikterak složitá. Také bude potřeba synchronizace konfiguračních souborů PyKMIP aplikace, hlavně `/etc/pykmip/policy/policy.json`.

Další částí je připojení více klientu pomocí KMIP. Zde je nutné upravit stávající firewall, ve kterém je na port 5696 povolena pouze jedna adresa. Poté otestovat, zda server zvládá obsluhovat požadavky další aplikace. Taktéž se nejedná o komplikovanou záležitost.

Při používaní KMS se může ukázat práce s ním jako nekomfortní. Uživatelské rozhraní CLI nemusí být dostačující a proto se může vytvořit webové rozhraní. Webová aplikace by musela umět upravovat konfigurační soubory jak PyKMIP, tak ETCD a restarty těchto služeb. Pro získávání informací by bylo vhodné zpracování jednotlivých logů. Momentální implementace obsahuje pouze wrapper na data, která serializuje a ukládá do ETCD. Pro webovou aplikaci by bylo vhodné udělat kompletní transformaci těchto dat do formátu JSON a ty ukládat do ETCD v čitelné podobě. Výpis informací o klíčích by pak byl již jednoduchý.

Oblastí, ve kterých lze dále práci rozšiřovat, je více. Může jít o lepší zabezpečení samotného OS. Revizi kódu celého PyKMIP nebo něco úplně jiného. Každý nový produkt tohoto typu lze rozšiřovat z mnoha stran.

4.2 Testování

Patřičné otestování je vždy potřebné a při tvorbě bezpečnostních komponent zvláště. Hned na úvod této podkapitoly je ale nutno podotknout, že celá práce byla vyvíjena pouze jedním člověkem. I přes konzultace s vedoucím práce a zaměstnanci Master Internet nebylo u většiny práce ani dodrženo pravidlo čtyř očí[13]. Také nebyly použity žádné nástroje na statickou ani dynamickou analýzu. Již jen na základě těchto informací není vhodné použít výsledný produkt k přímému produkčnímu nasazení. Je nutné celou implementaci nezávisle prověřit jak z pohledu celého systému, tak po jednotlivých částech (OS, pykmip-server, ETCD).

Základní ruční testování funkcionality probíhalo během vývoje a není zde blíže popsáno. Pro otestování byly použity příklady z knihovny PyKMIP z adresáře `/kmip/demos/pie/`, který se při standardní instalaci nachází v `/usr/local/lib/python3.9/dist-packages/`. Příklady testují jednotlivé funkce KMS a ten všechny korektně zpracovává. Více informací je v příloze B.2 Níže popsáno je pouze přímé testování systému jako celku. Pro všechny testy platí, že obsahují návrh s předpokládaným chováním. Poté je popsáno reálné chování systému s komentářem.

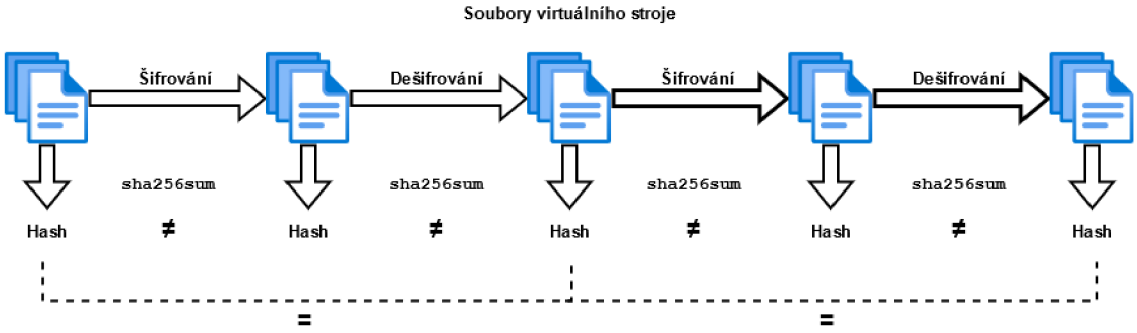
4.2.1 Ověření šifrování

Jedním z nejpřímočařejších testů je to, zda vCenter vůbec zašifroval dané soubory. Při zašifrování musí vCenter požádat KMS o symetrický klíč (při novém požadavku klíč vytvořit). Poté bude probíhat samotné šifrování. Po šifrování by se měla v detailu VM zobrazit ve

[13] Zdrojový kód viděli alespoň dva lidé.

webovém rozhraní vCenter u seznamu virtuálních pevných disků poznámka `Encrypted`. Po šifrování by se měla změnit struktura dat při zachování stejné velikosti.

Šifrovat a dešifrovat se bude dvakrát, jak popisuje Obrázek 4.2. Po každé operaci se spočítá hash všech souborů, který by se při dešifrovaném stavu měl shodovat. Oproti šifrovanému by se měl ale lišit. Pro výpočtu hash ze souborů VM je nutné přístup na Datastore. Zde se provede příkaz `sha256sum` nad všemi soubory z daného adresáře. Také se předpokládá, že vCenter vygeneruje pro druhé šifrování nový klíč a stávající zneplatní.



Obrázek 4.2: Postup šifrování a výpočtů hash

Výsledek

Pro test byla vytvořen nový virtuální stroj. Všechny operace se prováděly bez jeho zapnutí. Cílem bylo, aby se změny v souborech virtuálního disku projevily pouze na základě šifrování/dešifrování. Pro výpočet hash funkce je potřebný přístup do Datastore, který se v testovacím prostředí nachází na NFS (CentOS 8), viz sekce Použitý software 3.1.2.

Vypočtené hodnoty hash se z šifrovaných stavů lišily oproti těm nešifrovaným. Nešifrované hodnoty se shodovaly tak, jak bylo navrženo. Soubory `*.hlog` a `*.vmx` zůstaly v čitelné podobě, ale šlo vidět zmínku o tom, že ostatní data jsou zašifrovaná. Soubory s vypočtenými hodnotami jsou přiloženy k této práci.

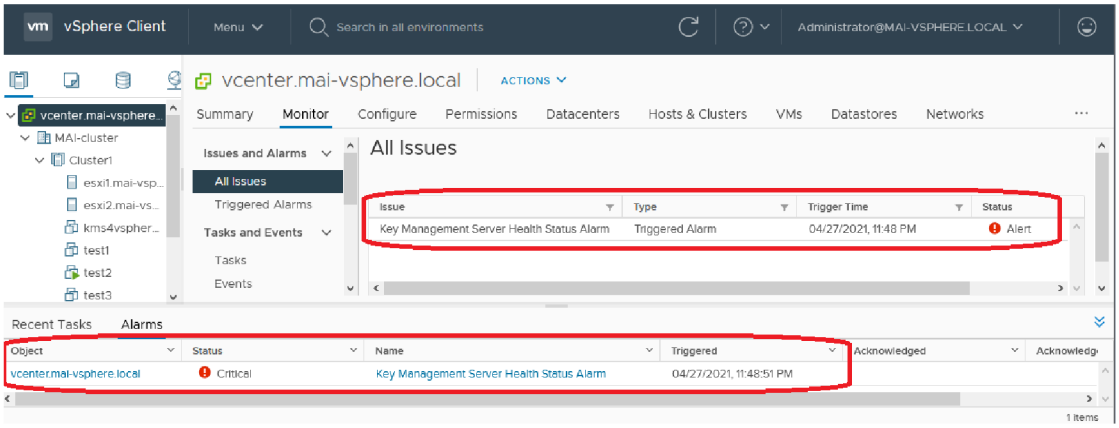
Při prvním pokusu o zašifrování byly zálohovány 4 operace: `Create` (AES-256), `Get`, `GetAttributes` a na závěr `Active`. Po dokončení šifrování šlo v grafickém prostředí vidět informaci o tom, že jsou data šifrována. Při dešifrování nedošlo k žádné komunikaci vůči KMS. DEK byl nejspíše stále v dešifrovaném stavu v paměti vCenter. Při následném šifrování (čtvrtý stav z Obrázku 4.2) došlo opět k vytvoření nového klíče. Původní klíč v KMS zůstal ve stavu `Active`, což vypadá nestandardně. Nedošlo ke zneplatnění původního klíče.

4.2.2 Výpadek KMS

Při výpadku KMS by nemělo být možné pracovat se zašifrovanými VM. Pro otestování se připraví patřičné VM. Zkoušet se bude nově zašifrovat novou VM, spustit zašifrovanou VM a dešifrovat VM. Předpokládané chování je, že všechny tyto tři operace bez připojení ke KMS skončí s chybou. O odpojení KMS bude ve vCenter také patřičná notifikace.

Výsledek

Z analýzy komunikace bylo zjištěno, že vCenter navazuje každých 5 minut spojení s KMS i při neaktivitě. Při výpadku KMS tedy vCenter nejpozději do 5 minut pozná, že došlo k přerušení spojení. Samozřejmě pouze v případě, že nebyla komunikace dříve vyžadovaná. Pro notifikaci má také vCenter dostupný `Alarm`, viz Obrázek 4.3.



Obrázek 4.3: Notifikace při výpadku spojení s KMS

Zašifrování VM se nezdařilo, vCenter nemohl vygenerovat šifrovací klíč. Ovšem spuštění zašifrovaného VM i úplně dešifrovaného se zdařilo oproti očekávání. vCenter si nechává DEK v dešifrované podobě po určitou dobu a při použití pokaždé nevyžaduje KEK. V oficiální dokumentaci nebyla o tomto chování nalezena zmínka. Z testování se také nepodařilo zjistit podrobnější informace o tomto chování. Zajímavé by určitě bylo zjistit například jak dlouho a kolik dešifrovaných DEK si vCenter ukládá.

4.2.3 Množství virtuálních strojů

KMS musí také zvládat šifrování většího množství virtuálních strojů. Pro ty si musí ukládat řádově stejné množství šifrovacích klíčů. Pro otestování bude nastavena šifrovací politika pro vícero virtuálních strojů a sledování chování systému. Testovat se bude 1, 5 a 20 virtuálních strojů současně určených k šifrování. KMS by mělo zvládnout zpracování všech požadavků bez rozdílů na počet požadavků. vCenter by mělo zvládnout všechny operace také standardně.

Výsledek

Všechny tři případy dopadly dle očekávání a šifrování se zdařilo. KMS nemělo žádné potíže při zpracovávání požadavků. vCenter také nezaznamenalo žádné potíže.

Limity ETCD úložiště široce převyšují maximální množství virtuálních strojů v jednom clusteru. Jeden vCenter, pro který je momentálně KMS navrhován, nemůže překročit velikost úložiště. Pro jistotu by měla být v pravidelné údržbě i kontrola využití pevného disku KMS. Mohlo by dojít k využití veškerého místa na disku pomocí souborů samotného OS.

4.2.4 Rychlost vyřízení požadavku

K předešlému testu se váže i tento. Při stejném scénáři se ovšem ale bude sledovat délka vykonávání. KMS musí zvládnout vyřídit všechny požadavky v relativně krátkém čase, který nebude zdržovat práci vCenter.

Výsledek

Ukázalo se, že délka zpracování není závislá na KMS, ale na vCenter. Pro 20 VM byly všechny požadavky na KMS obslouženy za 19 vteřin. Na druhou stranu ale vCenter šifrovalo déle než 10 minut. Délka šifrování je totiž odvozená z velikosti VM a výpočetního výkonu clusteru. Rychlost KMS není tedy podstatná a stávající rychlost je plně dostačující.

4.2.5 Ověření stability

KMS musí být stabilní a poskytovat vysokou dostupnost. Ruční údržba se plánuje pouze jedenkrát měsíčně a představovat bude kontrolu, že všechny části správně pracují. Z časových údajů bude testován kratší časový úsek. KMS bude spuštěno a připojeno po jeden týden a bude kontrolována instalace aktualizací, komunikace s vCenter a celkový chod systému. Během té doby se předpokládá, že vše bude fungovat dle návrhu.

Výsledek

KMS bylo zkontrolováno po týdnu chodu. Instalace aktualizací proběhla úspěšně. Úspěšně proběhl také plánovaný restart a start systému. Spojení s vCenter bylo zachováno a nebyla hlášena ani nedostupnost KMS. Na to, že nebylo z pohledu vCenter hlášeno přerušení spojení, má vliv několik faktorů. Hlavním je krátký downtime (restart netrvá déle než jednu minutu) a plánovaný restart v brzkých ranních hodinách (3:00), tedy neaktivita spojení během restartu KMS.

Kontrola spojení se provádí každých 5 minut ze strany vCenter a v tento okamžik bylo KMS opět dostupné. Všechny tyto informace jsou dostupné z logu služby `pykmip-server` v souborech adresáře `/var/log/pykmip/`. Z delšího běhu KMS lze vidět, že logy jsou zahlceny testovacím spojením. Pro produkční nasazení by bylo vhodné toto upravit. Například zkrátit výpisy u operací, které používá vCenter pro kontrolu spojení nebo je nezapisovat vůbec při určité úrovni logování.

4.2.6 Pokus o nelegitimní získání klíče

Dále bude testována bezpečnost celého řešení. Jak už bylo výše psáno, jelikož je celá práce psána jedním člověkem, který zároveň navrhuje testy a zajišťuje bezpečnost, je problematické navrhnout relevantní test. Tento test tedy bude ověřovat hlavně zda fungují bezpečnostní mechanizmy.

Testováno bude zabezpečení KMS před prozrazením klíče nelegitimnímu klientovi. Konkrétně bude testováno připojení z jiného stroje než je vCenter. V první řadě by měl komunikaci hned zastavit firewall.

Při nastavování stroje se zadává IP adresa vCenter a veškerá ostatní komunikace na potřebný port je blokována. Pro testovací účely bude toto pravidlo odstraněno. V reálném světě to bude představovat stav, kdy se útočníkovi podaří podvrhnout svou IP adresu za tu od vCenter.

V dalším kroku by měl útočníka zastavit sám server při pokusu o připojení. Nebude mít k dispozici TLS certifikáty KMS. Ty se ovšem nahrávají ručně do vCenter při ustalování spojení. Předpokládejme, že tuto komunikaci útočník odposlechl a certifikáty získal.

Do třetice by měla zafungovat interní logika aplikace a žádný symetrický klíč nevyzradit. Pravidla se konfigurují v souboru `/etc/pykmip/policy/policy.json`, který pro objekt `SYMMETRIC_KEY` nastavuje práva pro všechny operace na `ALLOW_OWNER`. Jelikož útočník nebude vlastníkem klíče, nebude mu klíč poskytnut. KMS totiž rozlišuje uživatele na základě jejich TLS certifikátu. Nicméně vCenter nepodporuje dodatečné přihlašování pomocí uživatelského jména a hesla. Při získání i těchto certifikátů by byly již utajené klíče vystaveny.

Výsledný útok by tedy představoval získání všech TLS certifikátů, podvrhnutí IP adresy vCenter a také znalost, který objekt chceme od KMS získat. Schématicky je toto znázorněno na Obrázku 4.4.

Obrázek 4.4: Schéma útoku na KMS

Výsledek

Pro otestování byl využit druhý virtuální stroj ve stejné síti jako KMS. Stroj obsahoval také PyKMIP server a byla využita jeho klientská část. Při prvním pokusu o připojení zafungoval firewall. Komunikace byla odepřena.

Po povolení připojení na firewallu byla komunikace opět otestována. Nyní se již klient připojil ke KMS, ale nedošlo k validnímu předání TLS certifikátu (TLS handshake). Zde byla komunikace opět zastavena.

Následně byly TLS certifikáty KMS překopírovány na útočící stroj. Poté již šlo s KMS pracovat (podařilo se uložit vlastní klíč). K prozrazení klíče ale stejně nedošlo. Zafungovaly interní polity KMS tak, jak byly navrženy a přístup k objektům vCenter zamítnut. Tímto bylo dokázáno, že implementované bezpečnostní mechanizmy zafungovaly.

4.3 Dosažené výsledky

Tato podkapitola obsahuje celkové shrnutí výsledků práce. Nejprve je rozepsáno, co ze specifikace požadavků bylo splněno. Poté následuje shrnutí z výše popsaného testování.

Podařilo se navrhnout, implementovat a otestovat Key Managment Server komunikující pomocí KMIP. Poskytuje základní funkce pro správné fungování v prostředí vSphere 7.0. Pracuje pouze pro jeden vCenter server, který s jeho pomocí korektně šifruje virtuální stroje.

Oproti návrhu server funguje se sníženými hardwarovými nároky. Plně mu postačuje 2 CPU, 4GB RAM a 10 GB pevný disk. Operačním systémem je Ubuntu 20.04 a celé řešení je ve formě Virtual appliance. Server je určen pouze pro správu klíčů a obsahuje minimum nadbytečných nástrojů.

Celá instalace a konfigurace je v Bash skriptech, které jsou interaktivní. Žádná úprava serveru se neprováděla ručně. Po inicializaci server již vyžaduje pouze minimální údržbu. Konfigurace je zaměřena na jednoduchost a bezpečnost. K serveru se přistupuje pomocí SSH. Pro komunikaci s vCenter je nutné do něj nahrát vygenerované certifikáty. V konfiguračních skriptech se mimo jiné nachází i nastavení firewallu a automatické aktualizace.

Ve skriptech je začleněna i instalace aplikace PyKMIP včetně celkové konfigurace. Stejně tomu je i pro ETCD. Nakonec je nahrán vytvořený wrapper pro data a je nastaveno spojení obou komponent. Obě části fungují jako služba a jsou spouštěny automaticky. Celá aplikace funguje správně a byla i zachována zpětná kompatibilita s využitím SQLite databáze.

Celá aplikace se dá upravovat pomocí několika konfiguračních souborů. Postup zálohování byl navrhnut a popsán. Taktéž postup pro budoucí rozvoj v KMS cluster či správu více aplikací. Požadavky na systém byly tedy splněny všechny. Shrnutí je v Tabulce 4.1, která rozšiřuje Tabulku 3.1 ze specifikace požadavků .

Ze samotného testování vyšlo najevo, že výsledné řešení je stabilní a kompletní. Finální řešení bylo také testováno ze strany společnosti Master Internet. Testována byla spolupráce s vCenter a proces šifrování v dalších funkcích, jako je vMotion či úpravy VM. Vše fungovalo správně a nebyly nalezeny žádné problémy. Vyjma vypracování postupu tvorby a obnovy záloh není známa žádná překážka pro produkční nasazení. Nicméně vzhledem k citlivosti zpracovávaných dat bude vhodné ještě podrobnější otestování celého vytvořeného KMS.

4.3.1 Shrnutí dosažených výsledků

Vlastnost systému	Zvolené řešení	Implementace	Splněno
Účel	KMS pro vCenter, šifrování jednotlivých VM a vSAN	Funkční	✓
Připojené aplikace	1 server vCenter	Připojeno	✓
Formát systému	Virtual appliance (OVA/OVF)	OVF	✓
vCPU	2	2	✓
RAM	8 GB	4 GB	✓
HDD	50 GB	10 GB	✓
OS	GNU/Linux	Ubuntu 20.04	✓
Podpora TPM/vTPM	ne*	ne	✓
Distribuce klíčů	KMIP verze 1.1+	1.1+	✓
Konektivita	pouze LAN, WAN ne	LAN	✓
KMS cluster	ne	ne	✓
Uživatelské rozhraní	CLI	Bash skripty	✓
Serverová implementace	python, knihovna PyKMIP	python 3.9.0+ PyKMIP 0.10	✓
Ukládání klíčů	zabezpečená databáze	ETCD 3.2.26	✓
Generování klíčů	Doporučená kryptografická knihovna pro python	cryptography.io	✓
Údržba a konfigurace	minimální, vše připravit	Bash skripty	✓
Zálohování	ne	Navrženo	✓
Firewall	ano	Nastaveno	✓

Tabulka 4.1: Porovnání požadavků s výslednou implementací

Kapitola 5

Závěr

Cílem bylo vytvořit funkční Key Management Server (KMS) pro prostředí vSphere 7.0. To se podařilo a vytvořený server poskytuje plnou funkcionalitu pro šifrování virtuálních strojů. Všechny zadané požadavky na práci byly splněny. Na práci se spolupracovalo s firmou Master Internet s.r.o, která se zabývá cloudovými službami.

Pro tuto práci bylo vytvořeno komplexní testovací prostředí v infrastruktuře Master Internet, pro které byl vyhrazen jeden fyzický server. Firma také spolupracovala na tvorbě požadavků. Dále byl vytvořen návrh řešení a následovala samotná implementace. Práce je cílena spíše na integraci stávajících volně dostupných komponent než vytváření nových.

Výsledné KMS je založeno na operačním systému Ubuntu 20.04. Aplikační vrstva se skládá z python knihovny PyKMIP, které využívá úložiště ETCD. Celkový výsledek prošel všemi navrženými testy a nebyly nalezeny žádné chyby. Testování probíhalo i ze strany Master Internet.

Pro integraci PyKMIP a ETCD byl vyvinut vlastní modul pro ukládání dat. Pro konfiguraci operačního systému (firewall, SSH, ...) byly vytvořeny Bash skripty. Ty obsahují také instalaci celé aplikační vrstvy. Spuštění KMS se tedy skládá z importu virtuálního stroje ve formátu OVF a poté spuštění těchto skriptů. Ty interagují s uživatelem a pomůžou mu nastavit systém dle jeho požadavků.

Práce je velice vhodná pro rozšíření. Oblastí pro rozvoj je hned několik. Asi nejpotřebnější je rozšíření na KMS cluster, pro zajištění vysoké dostupnosti. Návrh celého systému je tomu také uzpůsoben. Dále je také možné dodělat webové rozhraní pro správu klíčů, zvýšit zabezpečení serveru (jak OS, tak aplikační vrstvy) nebo otestovat nasazení pro jinou aplikaci než vSphere. KMS totiž využívá standardizovaný komunikační protokol KMIP.

Jedinou známou překážkou pro nasazení v produkčním prostředí je absence plánu obnovy záloh a jeho důkladné otestovaní. Tento proces je v práci pouze navržen. V blízké době se plánuje dokončení této části a poté nasazení v infrastruktuře Master Intenet.

Literatura

[1] *A complete guide on virtualization.* Australia: Emereo Pty Ltd, 2008. ISBN 978-1-921523-91-5.

[2] *The Definitive Guide to Encryption Key Management Fundamentals.* Townsend Security, 2016.

[3] *Encryption Key Management for VMware - The Definitive Guide.* Townsend Security, 2020.

[4] Acar, Y., Backes, M., Fahl, S., Garfinkel, S., Kim, D. et al. Comparing the Usability of Cryptographic APIs. In: *2017 IEEE Symposium on Security and Privacy (SP).* 2017, s. 154–171. DOI: 10.1109/SP.2017.52.

[5] Amazon Web Services, Inc.. *What is AWS* [online]. Navštíveno 17.3.2021. Dostupné z: https://aws.amazon.com/what-is-aws/.

[6] Bipin Giri. *What is VMware vCenter Server?* [online]. Navštíveno 6.3.2021. Dostupné z: https://www.mustbegeek.com/what-is-vmware-vcenter-server/.

[7] Brien Posey. *Top 11 Microsoft Hyper-V Terminologies you need to know* [online]. Navštíveno 17.3.2021. Dostupné z: https://www.vembu.com/blog/top-11-microsoft-hyper-v-terminologies-you-need-to-know/.

[8] Brose, Gerald. Access Control. In: *Encyclopedia of Cryptography and Security.* Boston, MA: Springer US, 2011, s. 2–7. DOI: 10.1007/978-1-4419-5906-5-179. ISBN 978-1-4419-5906-5.

[9] CIS (Center for Internet Security). *CIS Ubuntu Linux 20.04 LTS Benchmark* [online]. Navštíveno 30.4.2021. Dostupné z: https://www.cisecurity.org/benchmark/ubuntu_linux/.

[10] CIS (Center for Internet Security). *CIS VMware ESXi 7.0 Benchmark* [online]. Navštíveno 30.4.2021. Dostupné z: https://www.cisecurity.org/benchmark/vmware/.

[11] Cryptomathic, Terry Anton. *The link between HSMs and a Centralized Key Management System.* [online]. navštíveno 8.11.2020. Dostupné z: https://www.cryptomathic.com/news-events/blog/the-link-between-hsms-and-a-centralized-key-management-system.

[12] Dell Inc. *Rackový server PowerEdge R430* [online]. Navštíveno 6.11.2020. Dostupné z: https://www.dell.com/cz/domacnosti/p/poweredge-r430/pd.

[13] Fornetix, LLC. *Fornetix VaultCore - The Key to Data Security* [online]. Navštíveno 16.3.2021. Dostupné z: https://www.fornetix.com/solutions/vaultcore/.

[14] Futurex. *Key Management Enterprise Server (KMES) Series 3* [online]. Navštíveno 12.3.2021. Dostupné z: https://www.futurex.com/products/kmes-series-3.

[15] Hytrust, Inc.. *About our company* [online]. Navštíveno 24.10.2020. Dostupné z: https://www.hytrust.com/company/.

[16] Karel Šiman, Ministerstvo Vnitra ČR. *Cloud pro utajované informace* [online]. Navštíveno 4.5.2021. Dostupné z: https://cybersecurity.cz/data/Siman1.pdf.

[17] KeyGuard. *Key managment system* [online]. Navštíveno 10.1.2021. Dostupné z: https://key-guard.eu.

[18] Martin Sztula. *vSAN – co to je a proč jí chtít?* [online]. Navštíveno 8.3.2021. Dostupné z: https://virtualguru.cz/2018/11/27/vsan-cast-1-co-to-je-a-proc-ji-chtit/.

[19] National Institute of Standards and Technology - NIST. *A Framework for Designing Cryptographic Key Management Systems - SP 800-130*. 2013. DOI: 10.6028/NIST.SP.800-130.

[20] National Institute of Standards and Technology - NIST. *Security Requirements for Cryptographic Modules - FIPS 140-3*. 2019. DOI: 10.6028/NIST.FIPS.140-3.

[21] OASIS. *OASIS Key Management Interoperability Protocol (KMIP)* [online]. Navštíveno 1.3.2021. Dostupné z: https://www.oasis-open.org/committees/kmip.

[22] Oracle Corporation. *VM VirtualBox - User Manual* [online]. Navštíveno 18.3.2021. Dostupné z: https://www.virtualbox.org/manual/.

[23] Red Hat, Inc. *What is a hypervisor?* [online]. Navštíveno 1.3.2021. Dostupné z: https://www.redhat.com/en/topics/virtualization/what-is-a-hypervisor.

[24] Red Hat, Inc. *What is virtualization?* [online]. Navštíveno 4.5.2021. Dostupné z: https://www.redhat.com/en/topics/virtualization/what-is-virtualization.

[25] Ruest, D. *Virtualizace : podrobný průvodce*. Vyd. 1. Brno: Computer Press, 2010. ISBN 978-80-251-2676-9.

[26] Stephen J. Bigelow. *What's the difference between Type 1 vs. Type 2 hypervisor?* [online]. Navštíveno 3.5.2021. Dostupné z: https://searchservervirtualization.techtarget.com/feature/Whats-the-difference-between-Type-1-and-Type-2-hypervisors.

[27] Sustek, Laurent. Hardware Security Module. In: *Encyclopedia of Cryptography and Security*. Boston, MA: Springer US, 2011, s. 535–538. DOI: 10.1007/978-1-4419-5906-5-179. ISBN 978-1-4419-5906-5.

[28] The Linux Foundation. *ETCD* [online]. Navštíveno 6.4.2021. Dostupné z: https://etcd.io/.

[29] Trusted Computing Group (TCG). *Trusted Platform Module (TPM) Summary* [online]. Navštíveno 485.2021. Dostupné z: https://trustedcomputinggroup.org/resource/trusted-platform-module-tpm-summary/.

[30] van Tilborg, Henk C. A. and Jajodia, Sushil. Transport Layer Security (TLS). In: *Encyclopedia of Cryptography and Security*. Boston, MA: Springer US, 2011, s. 1305–1305. DOI: 10.1007/978-1-4419-5906-5_1043. ISBN 978-1-4419-5906-5.

[31] VMware. *Stránky společnosti VMware* [online]. Navštíveno 3.3.2021. Dostupné z: https://www.vmware.com.

[32] Ward, Brian. *VMWARE : provozujeme více operačních systémů na jednom počítači*. Vydání první. Brno: Computer Press, 2004. ISBN 80-251-0129-0.

Příloha A

Obsah přiloženého DVD

- Konfigurační soubory:
  - etcd.conf - konfigurační soubor pro ETCD úložiště,
  - server.conf - konfigurační soubor pro PyKMIP KMS,
  - pykmip.conf - ukázka konfigurace pro testování serveru,
  - policy.json - nastavení uživatelských práv pro objekty PyKMIP,
  - pykmip-server.service - nastavení PyKMIP jako služby,
  - sshd_config - konfigurace SSH.
- Postup instalace - snímky obrazovky z instalace operačního systému.
- APP skripty:
  - ETCDwrapper.py - wrapper dat pro ukládání do ETCD,
  - ___init___.py - soubor pro zabalení ETCDwrapper.py do modulu ,
  - engine.py - upravená komponenta PyKMIP.
- Virtualni stroj - Virtual Appliance včetně přihlašovacích údajů KMS.
- VM skripty:
  - 00-network.sh - nastavení sítě,
  - 01-system.sh - nastavení operačního systému,
  - 02-aplikace.sh - instalace a konfigurace aplikace pro obsluhu KMS,
  - create_certificates.py - generování TLS certifikátů (použito v 02-aplikace.sh).
- Testování hash - soubory s vypočtenými hash hodnotami z testování šifrování.

Příloha B

Instalace a provoz KMS

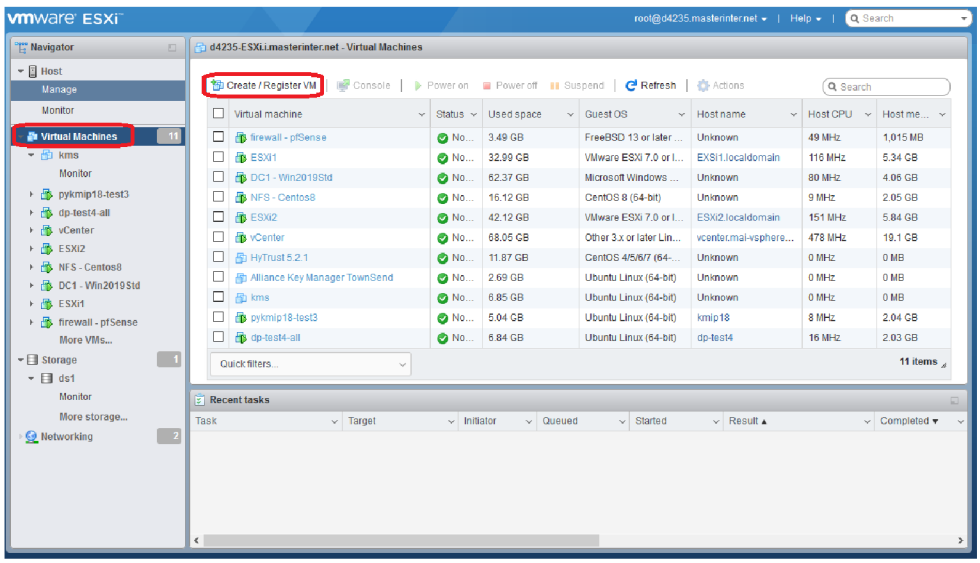
Tato příloha obsahuje podrobný popis pro instalaci celého KMS a dalších potřebných informací o jeho provozu.

B.1 Instalace a inicializace systému

Instalace KMS se skládá z importu virtuálního stroje, jeho inicializace a konfigurace ve vCenter.

B.1.1 Import virtual appliance

Prvním krokem ke zprovoznění KMS je jeho import do virtualizovaného prostředí. Formát OVF je podporován širokou řadou hypervizorů. Zde je krátce popsán postup pro ESXi 7.0. Po přihlášení klikněte na záložku `Virtual Machines` a poté zvolte možnost `Create / Register`, jak je zobrazeno na Obrázku B.1. Následně vyberte `Deploy a virtual machine from an OVF or OVA file` a nahrajte oba soubory virtual appliance. Poté pokračujte za pomoci průvodce.



Obrázek B.1: Obrazovka zobrazující správu poskytovatelů klíčů

B.1.2 Inicializace operačního systému

Po zdařeném importu je možné virtuální stroj spustit. Přihlašovací údaje jsou přiloženy k této práci. Prvotní přihlášení je nutné provést přímo na stroj skrze hypervisora. Prvním krokem je totiž konfigurace sítě. Výchozí nastavení je statická IP adresa 192.168.12.234, gateway 192.168.12.1 a DNS na 1.1.1.1.

V domovském adresáři `\home\kms\VM_skripty` se nacházejí konfigurační skripty. Všechny vyžadují pro spuštění právo `sudo`. Spusťte skript `00-network.sh`. Budete vyzváni k zadání informací ohledně sítě (IP adresa, gateway a DNS servery). Lze také nastavit adresu DHCP serveru. IPv6 není podporováno.

Po dokončení skriptu lze zkontrolovat konfiguraci skrze příkaz `ip a`. Nyní se můžete připojit pomocí SSH na nově nakonfigurované adrese. Dalším konfiguračním skriptem je `01-system.sh`. Nastaví všechny úpravy systému. Budete vyzváni k zadání SSH klíčů, IP adresy vCenter a nastavení aktualizací systému. Na konci se stroj restartuje, aby se všechny úpravy a aktualizace projevily. V případě problému či pádu skriptu jej spusťte opětovně.

Po restartování stroje se můžete připojit pomocí SSH certifikátu. Posledním skriptem je `02-aplikace.sh`. Zde se již nezadávají žádné údaje. Skript nainstaluje a inicializuje ETCD a PyKMIP včetně všech úprav. Během inicializace se vygenerují TLS certifikáty a hesla pro všechny uživatele. Zařazena je i kontrola připojení do ETCD. Po dokončení skriptu je již KMS plně funkční a připraven obsluhovat KMIP dotazy na výchozím portu 5696.

Kontrola instalace

Kontrola správné inicializace jednotlivých částí je možná pomocí následujících příkazů:

- `cat /etc/netplan/00-installer-config.yaml` - nastavení sítě,
- `ip a` - kontrola IP adresy,
- `iptables -L` - nastavení firewallu,
- `systemctl status etcd` - stav ETCD úložiště,
- `systemctl status pykmip-server` - stav služby pykmip-server,
- `ps -fu pykmip-server-user` - výpis procesů uživatele pykmip-server-user.

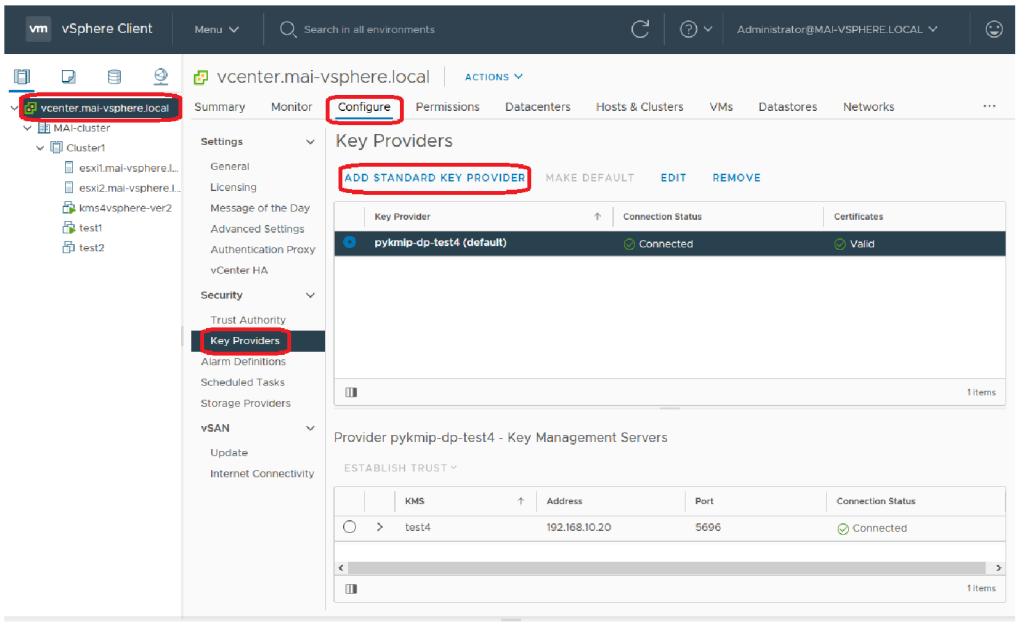
B.1.3 Propojení s vCenter

Další část vychází z oficiální dokumentace VMware[1]. Popisuje, jak připojit KMS jako Standard Key Provider, viz sekce 2.6.2. Před dalšími kroky zkontrolujte vzájemnou konektivitu obou serverů.

Přihlaste se do vCenter a z nástěnky se přepněte do položky `Hosts and Clusters`. Dále z levé postranní lišty vyberte jméno vybraného vCenter a vyberte možnost `Configure`. Ze sekce `Security` zvolte `Key Providers`. V tomto pohledu lze konfigurovat všechny poskytovatele klíčů. Pro přidání nového klikněte na `Add Standard Key Provider`, jak zobrazuje Obrázek B.2.

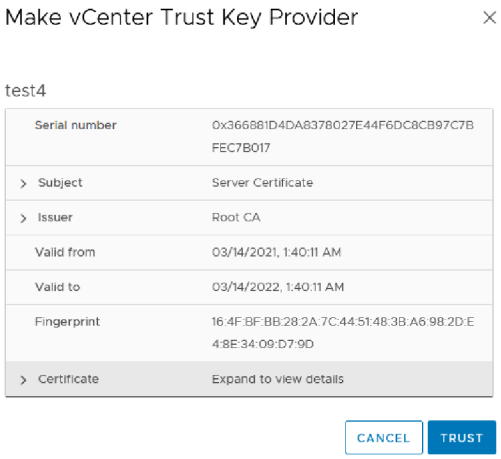
Ve vyskakovacím okně zadejte patřičné údaje pro KMS. Názvy jsou pouze informativní pro uživatelské rozhraní, nevycházejí z žádné konfigurace. Výchozí port KMS je 5696. Volby

[1] https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.security.doc/GUID-78DD547A-6FFC-49F1-A5F2-ECD7507EE835.html



Obrázek B.2: Obrazovka zobrazující správu poskytovatelů klíčů

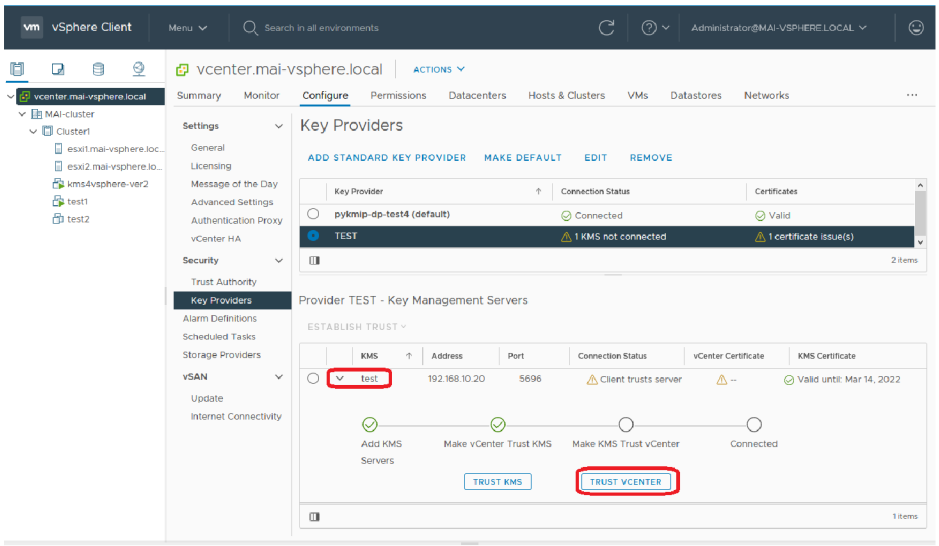
proxy nechte nevyplněnou stejně jako přihlašovací údaje. Služba `pykmip-server` musí být na KMS spuštěna. Po kliknutí na `ADD KEY PROVIDER` se v případě správné konfigurace zobrazí certifikát ke schválení, viz Obrázek B.3.



Obrázek B.3: Okno pro schválení certifikátu KMS

V opačném případě je nutné odstranit všechny překážky v připojení. Pokud vCenter blíže nespecifikuje chybu, zkontrolujte log KMS. Pokud nevykazuje žádné známky po připojení, je problém v konektivitě obou serverů a zvoleného portu. Zkontrolujte pravidla firewallu. Pokud nebude z logu jasná příčina selhání, je možné změnit úroveň logování z výchozího `INFO` na `DEBUG` v konfiguračním souboru `server.conf`.

Dále je nutné nahrát certifikát KMS přímo do vCenter. Do té doby nebude možné vytvořit zabezpečenou komunikaci. Při kliknutí na nově přidaného poskytovatele klíčů se zobrazí jeho detail. Zde vyberte možnost `TRUST VCENTER` podle Obrázku B.4.



Obrázek B.4: Obrazovka s konfigurací poskytovatele klíčů

Poté vyberte možnost `KMS certificate and private key`. V dalším kroku nahrajte certifikát a jeho privátní klíč. Zadejte zde soubory z KMS `/etc/pykmip/ssl/server_cert.pem` a `/etc/pykmip/ssl/server_key.pem`. Při úspěšném nahrání je konfigurace KMS ve vCenter hotová. Šifrování bude ve vCenter dostupné. Další postup je dostupný v oficiální dokumentaci[2]

B.2 Testování jednotlivých funkci

Pro testování jednotlivých částí aplikace KMS jsou připraveny ukázkové programy. Ty se nacházejí v adresáři `/usr/local/lib/python3.9/dist-packages/kmip/demos/pie`. Společně s konfiguračním souborem `/etc/pykmip/pykmip.conf` simulují lokálně práci klienta KMS. Už z názvu programu bývá často jasná jeho funkce. Níže jsou blíže popsány včetně ukázky spuštění:

- `create.py -a AES -l 256 # vytváří symetrický klíč`
- `create_key_pair.py -a RSA -l 2048 # vytvoří key pair`
- `destroy.py -i 8 # smaže zadaný objekt`
- `revoke.py -i 9 # anuluje zadaný objekt`
- `get.py -i 7 # vrátí vybraný objekt`
- `get_attribute_list.py -i 9 # vrátí seznam definic atributů`

[2] https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.security.doc/GUID-5E2C3F74-38C1-44C3-ABC5-C2C9353B9DC4.html

- `get_attributes.py -i 10 # vrátí hodnoty atributů`
- `mac.py -i 10 -a AES # Message Authentication Codes`
- `encrypt.py -m message # vytvoří nový symetrický klič a zašifruje zprávu`
- `decrypt.py -i 12 -m b'ee46d93288285a5f0b006b8238b7d29e' # dešifrování`
- `sign.py # zaregistruje privátní klíč a podepíše zadaná data`
- `signature_verify.py # zaregistruje privátní klič a zkontroluje podpis`
- `devire_key.py # odvodí tajemství na základě jednoho klíče`
- `locate.py # vyhledávání, bez parametrů vrátí všechny objekty`

B.3 Údržba systému

Při nasazení KMS je doporučena pravidelná měsíční údržba. Skládat by se měla z celkové kontroly systému, přesněji je potřeba zkontrolovat například: instalaci aktualizací, systémové logy či aplikační logy. KMS nevyžaduje žádné speciální akce při údržbě. Stačí ty standardní pro danou verzi operačního systému (Ubuntu 20.04).