Střední průmyslová škola a Vyšší odborná škola, Liberec, příspěvková organizace

Automatizovaná instalace a správa Grafana pomocí Ansible

Maturitní práce

Autor **Jan Mizera**

Obor **Informační technologie**

Vedoucí práce **Mgr. Milan Keršláger**

Školní rok **2023/2024**

Počet stran **4**

Počet slov **1131**



Anotace

Tato práce vznikla na zadání firmy ESET Research Czech Republic s.r.o. a zabývá se použitím automatizačního nástroje Ansible. Cílem této práce je použití tohoto nástroje pro instalaci a následnou správu vizualizačního nástroje Grafana

Summary

This work was commissioned by ESET Research Czech Republic s.r.o. and deals with the use of the Ansible automation tool. The aim of this work is to use this tool for the installation and subsequent administration of the Grafana visualization tool.

Čestné prohlášení

Prohlašuji, že jsem tuto práci vypracoval/a samostatně. V práci jsem použil/a pouze zdroje uvedené v seznamu literatury a všechny převzaté části, včetně textů, obrázků, kódu a výstupů umělé inteligence, jsou řádně označeny a citovány.

V Liberci dne

Jan Mizera

Obsah

[Úvod 1](#_Toc212202066)

[Uvedení do problematiky 1](#_Toc212202067)

[Představení nástroje Ansible 1](#_Toc212202068)

[Cíl práce 2](#_Toc212202069)

[Struktura práce 2](#_Toc212202070)

[1 Teoretická východiska automatizace 3](#_Toc212202071)

[1.1 Definice a význam automatizace v IT 3](#_Toc212202072)

[1.1.1 Koncept automatizace a její role v moderním IT 3](#_Toc212202073)

[1.1.2 Idempotence 3](#_Toc212202074)

[1.2 Infrastructure as Code (IaC) 4](#_Toc212202075)

[1.2.1 Definice, historie a princip IaC 4](#_Toc212202076)

[1.3 Co všechno je možné zautomatizovat 4](#_Toc212202077)

[2 Teorie monitoringu 5](#_Toc212202078)

[1.1 Použité nástroje 5](#_Toc212202079)

[2.1.1 VirtualBox 5](#_Toc212202080)

[2.1.2 Vagrant 5](#_Toc212202081)

[2.1.3 Zabbix 5](#_Toc212202082)

[3 Druhá kapitola 6](#_Toc212202083)

[Závěr 7](#_Toc212202084)

[Seznam zkratek a odborných výrazů 8](#_Toc212202085)

[Seznam obrázků 9](#_Toc212202086)

[Použité zdroje 10](#_Toc212202087)

[A. Seznam přiložených souborů I](#_Toc212202088)

[B. Seznam Použitých nástrojů II](#_Toc212202089)

Úvod

## Uvedení do problematiky

Tato práce se zabývá automatizací IT infrastruktury, která je jednou z nejdůležitějších součástí pro fungování větších počítačových systémů, protože zjednodušuje jejich nasazení a následnou údržbu. Protože se neustále zvětšuje počet strojů a komponent v počítačových systémech, jejich samostatná správa se stává časově výrazně náročnou a s každým dalším strojem se zvyšuje šance na lidskou chybu, která může zavinit i kompletní vyřazení celého systému z provozu. Navíc při rozšiřování daného systému se oba tyto problémy stávají stále důležitější.

Proto existuje automatizace, která řeší a nahrazuje všechny předchozí problémy. Jedná se o aplikování technologie pro dosažení kýženého cíle s minimálním nasazením člověka a její hlavní výhodou je snadná škálovatelnost daného systému, protože umožňuje konfiguraci nového zařízení téměř okamžitě, bezchybně a s minimálním nasazením člověka. I proto nachází uplatnění v celé řadě průmyslových odvětví od průmyslové výroby a správu počítačových systémů až po finance a zdravotnictví. (1)

Při automatizaci IT infrastruktury, se na rozdíl od výrobního průmyslu nepoužívají stroje, ale takzvané IaC (Infrastructure as Code). Jedná se o možnost spravovat počítačovou infrastrukturu pomocí softwarových nástrojů jako jsou zdrojové kódy, scripty anebo konfigurační soubory. (2)

## Představení nástroje Ansible

Hlavním nástrojem použitým v této práci je Ansible. Jedná se o open source nástroj, pro správu konfigurace a automatizaci IT infrastruktury. (3) Pro tyto úkoly používá tzv. playbooky, které přímo implementují IaC a díky tomu, že jsou psány v YAML, jsou jednoduše čitelné a snadné na pochopení. Tyto playbooky jsou jednoduché konfigurační soubory, které je možné spustit na více zařízeních současně, a to i opakovaně. (4)

Daší z jeho hlavních výhod je, že nepotřebuje agenta pro jeho fungování, což znamená že nepotřebuje žádný další software, protože komunikuje přes nativní protokoly, zejména SSH a WinRM, a nepotřebuje žádnou další bezpečnostní infrastrukturu. (5)

## Cíl práce

Součástí této práce je praktická demonstrace pirncipů IaC. Hlavní náplní je instalace vizualizačního nástroje Grafana a následná správa pomocí automatizačního nástroje Ansible. Cílem a hlavním předmětem zadání z firmy, je vytvořit několik Ansible playbooků dle požadavků firmy, které budou spravovat výše zmíněný vizualizační nástroj Grafana. Na konci bude několik virtuálních strojů ve VirtualBoxu spravovaných pomocí Vagrantu a jejím výstupem bude několik playbooků v Ansiblu pro správu, provádění změn a konfiguraci Grafany.

## Struktura práce

Tato práce je rozdělena na teoretickou a praktickou část. Teoretické část se věnuje základním teoretickým východiskům automatizace (Kapitola 1) a detailní analýze nástroje Ansible (Kapitola 2). Praktická část se věnuje samostatné přípravě prostředí a následnému vytvoření a nasazení playbooků pro správu Grafany (Kapitola 3) a nakonec závěrečná kapitola shrne výsledky práce.

# Teoretická východiska automatizace

## Definice a význam automatizace v IT

### Koncept automatizace a její role v moderním IT

Automatizace je využití softwaru pro vykonávání opakovaných činností s omezenou nebo vůbec žádnou asistencí člověka. (6) Stává se čím dál populárnější a rozšiřuje se do nespočtu odvětví. Například dnes se používá i v průmyslové výrobě jako je výroba automobilů. (1)

Kde se ale používá softwarová automatizace? Téměř všude. Od správy podnikových sítí přes údržbu a správu výpočetních clusterů nebo serverů a cloudových služeb až po obranu a zdravotnictví. (1)

Proč je nasazována v tak široké oblasti zaměření? Protože má spoustu praktických výhod, které nejenom zajistí správné fungování, ale i snazší samotnou údržbu a správu a výrazně snižuje počet pracovníků k tomu potřebných. (1)

Hlavní tyto výhody jsou téměř úplně odstranění manuální práce (1) a redukce rizik a chyb s tím spojených. (6) Automatizace také zabraňuje takzvanému Configuration drift, to znamená postupné oddělení od zamýšleného stavu typicky způsobován manuálními úpravami nebo nekontrolovanými změnami. (7) Dále také zvyšuje rychlost nasazení služeb a aplikací, zvýšení jejich efektivity a škálovatelnost podle aktuální poptávky. Zlepšuje bezpečnost díky rychlé a kontrolované reakci na hrozby (6) a umožňuje optimalizaci správy a rozdělení cloudových zdrojů. (1)

### Idempotence

Idempotence je jednou z nejdůležitějších vlastností při automatizaci. Jedná se o vlastnost operace, která ukazuje, jestli opakované aplikování dané funkce změní výsledek oproti první aplikaci. V automatizaci je obzvláště důležitá, protože zajišťuje stabilitu a předvídatelnost systému, zvyšuje jeho odolnost proti chybám a snižuje počet nepotřebných operací, čímž také zároveň šetří čas a systémové zdroje. (8)

## Infrastructure as Code (IaC)

### Definice, historie a princip IaC

IaC je koncept, jehož historie sahá až do počátků rozmachu počítačů. S jejich přibývajícím počtem rostla potřeba najít efektivní způsob, jak spravovat spoustu strojů než jenom jejich manuální konfigurací. (9)

IaC je praxe, která automatizuje poskytování a správu služeb pomocí konfiguračních souborů na místo manuálních procesů. Tyto soubory v podstatě popisují celkový návrh dané infrastruktury, například počet serverů, nastavení sítě, verze operačních systémů a aplikací, atd. a můžou být napsány v širokém spektru jazyků, jako například YAML, JSON, atd. (10)

Mezi hlavní výhody IaC patří to, že řeší Configuration drift tak, že zdrojem, od kterého hrozí odchýlení není stroj, ale konfigurační soubor. Největší výhodou IaC ale je právě využitelnost konfiguračních souborů, jelikož se dají spustit, tak stačí spustit starší verzi konfiguračního souboru a tím dostaneme starší verzi samotného stroje, což usnadňuje zálohování a samotnou dokumentaci, např. pomocí Git. (10)

### Způsob implementace IaC

Při rozhodování, jak IaC implementovat, máme k dispozici dva přístupy, jak se píše samotný kód, z nichž většina nástrojů umí pracovat s oběma přístupy, které se dělí na: imperativní a deklarativní. (10)

#### Imperativní přístup

Tento přístup vyžaduje, aby uživatel psal přesné příkazy pro daný systém v přesně daném pořadí. (10) Má to svoji výhodu v tom, že uživatel má úplnou kontrolu nad daným systémem a sám rozhoduje o tom, jak dosáhne daného výsledku. To je užitečné např. při optimalizaci anebo při drobných úpravách, které vyřeší několik příkazů. (11)

Bohužel má spoustu nevýhod. Pro jeho efektivní nasazení je potřeba, aby uživatel měl dostatečnou znalost daného programovacího jazyka nebo daných příkazů. Také jsou často méně idempotentní oproti deklarativnímu přístupu, protože již předem definované kroky můžou vést k jiným výsledkům v závislosti na prostředí. Další velkou nevýhodou je, že je náchylný na chyby, protože pokud jeden z kroků selže, může selhat celá konfigurace. (11)

#### Deklarativní přístup

Tento přístup je nejběžnější metodou. Uživatel jenom popíše stav výsledného systému a IaC implementuje vizi uživatele, bez jeho dalšího zásahu. (10) Je velmi idempotentní, tudíž konzistentně znovu a znovu dosahuje stejných výsledků a téměř není náchylný na configuration drift. (11)

Největší jeho nevýhodou ale je, že uživatel se z vzdává většiny kontroly nad systémem. Také se nehodí pro drobné úpravy, pro které by stačilo jen pár řádků kódu, protože tento přístup to může překomplikovat a výrazně zpomalit samotný proces. (11)

### Preferovaný přístup pro správu konfigurace

Obecně je preferován přístup deklarativní, díky jeho idempotenci a nízké náchylnosti vůči configuration drift. Je výrazně lepší než přístup imperativní, který se hodí pouze na drobné jednorázové opravy nebo aktualizace, či intenzivní optimalizaci. (11) V rámci této práce je použit, nástroj, který deklarativní přístup implementuje, Ansible.

## *Přehled a srovnání automatizačních nástrojů*

### *Klasifikace nástrojů pro správu konfigurace*

Configuration management (CM), neboli správa konfigurace je způsob, jak systematicky zachovat konzistenci v počítačových systémech a jejich softwaru pro zajištění bezpečnosti, spolehlivosti a výkonu v celém prostředí. CM zajišťuje jediný zdroj pravdy nebo zdrojový Git repozitář, ze kterého vychází jednotná konfigurace. Všechny nástroje se ale dělí na tzv. agent-based a agentless, přičemž každý má svá pozitiva a negativa. (12)

#### Architektura Agent-based

Hlavní funkce agent-based systému je nasazení monitorovacího softwaru na každém zařízení v daném systému. Tito tzv. “agenti” shromažďují data na daném zařízení a následně tyto informace zpracují a výsledky obdrží hlavní monitorovací server, který je poskytne uživateli. To zajišťuje nižší síťové zatížení, protože po síti jdou pouze výsledky, ale na druhou stranu agent na samotných zařízeních musí běžet neustále a vyhodnocovat výsledky, což spotřebovává systémové zdroje na každém zařízení. Také proto že agenti jsou nasazeni na každém zařízení, zvyšuje to složitost samotné konfigurace systému a jeho údržby a může vyžadovat další konfiguraci pro zajištění bezpečnosti. (12)

#### Architektura Agentless

Agentless prostředí využívá jeden centrální řídící uzel, který spravuje celý systém. Tento řídící uzel komunikuje s jednotlivými zařízeními většinou pomocí API, SSH nebo systémových rozhraní pro určení výkonu a dostupnosti daného systému. Díky tomu je výrazně rychlejší na nasazení a má snadnou údržbu a správu než agent-base a téměř žádný dopad na systémové prostředky jednotlivých zařízení. Hrozí ale riziko, že uživatel může vytvořit zranitelnost, která nemusí být včas identifikována a způsobit problémy v budoucnu. (12)

### Rozlišení Configuration Management a Provisioning

#### Definice Provisioningu

Původní text pro případné použití později

## Co všechno je možné zautomatizovat

Zautomatizovat je možné v podstatě jakýkoliv opakující se úlohu. Díky tomu je možné dosáhnout rychlejších a efektivnějších operací napříč několika zařízeními najednou, na rozdíl od manuálního provádění daných operací.

# Teorie monitoringu

V této kapitole

## Použité nástroje

Tato kapitola se zabývá nástroji, které jsou v této práci použity a vysvětlení způsobu, jakým byly využity.

### VirtualBox

VirtualBox je virtualizační software pro správu více operačních systémů na jednom zařízení.(13) To znamená že dovoluje danému počítači zároveň spustit několik operačních systému v rámci několika virtuálních počítačů a libovolně mezi nimi přepínat. Díky tomu zvyšuje efektivitu využití hardwarových zdrojů.(14)

V této práci bude VirtualBox použit pro spuštění několika virtuálních strojů, z důvodu jednoduší správy a odstranění potřeby dalších zařízení.

### Vagrant

Vagrant je nástroj příkazové řádky pro správu jednotlivých virtuálních strojů. Usnadňuje nasazení virtuálních strojů a jejich následnou konfiguraci. Pomocí Vagrantu může uživatel snadno vytvořit konfigurační soubor a potom jen příkazem “vagrant up” se daný virtuální stroj nakonfiguruje a spustí. To zajišťuje spuštění totožných virtuálních strojů napříč různými zařízeními a zajišťuje stejné virtuální prostředí. (15)

### Zabbix

# Druhá kapitola

Závěr

Tak jsem se dostal až na konec.

Seznam zkratek a odborných výrazů

IaC – Infrastructure as Code

„je proces správy a provisioningu (vytváření) počítačových datových center pomocí strojově čitelných definičních souborů místo fyzické konfigurace hardwaru nebo interaktivních konfiguračních nástrojů“ (16)

SSH – Secure Shell

„označení pro zabezpečený komunikační protokol v počítačových sítích, které používají TCP/IP“ (17)

WinRM – Windows Remote Management

„je implementace webových služeb pro správu (WS-Management) protokolu Microsoft“ (18)

Seznam obrázků

**Nenalezena položka seznamu obrázků.**

Použité zdroje

1. **IBM.** What is automation? [Online] [Citace: 24. Říjen 2025.] https://www.ibm.com/think/topics/automation.

2. **Amazon Web Services.** What is Infrastructure as Code? [Online] [Citace: 17. Říjen 2025.] https://aws.amazon.com/what-is/iac/.

3. **Red Hat.** How Ansible works. [Online] [Citace: 17. Říjen 2025.] https://www.redhat.com/en/ansible-collaborative/how-ansible-works.

4. **Ansible Documentation.** Ansible playbooks. [Online] 8. Říjen 2025. [Citace: 17. Říjen 2025.] https://docs.ansible.com/ansible/latest/playbook\_guide/playbooks\_intro.html.

5. —. Ansible architecture. [Online] 8. Říjen 2025. [Citace: 17. Říjen 2025.] https://docs.ansible.com/ansible/latest/dev\_guide/overview\_architecture.html.

6. **Red Hat.** What is IT automation? [Online] 17. Únor 2025. [Citace: 10. Říjen 2025.] https://www.redhat.com/en/topics/automation/what-is-it-automation.

7. **Moustakis, Ioannis.** What is Configuration Drift? Tools, Causes & Risks. [Online] 15. Září 2025. [Citace: 24. Říjen 2025.] https://spacelift.io/blog/what-is-configuration-drift.

8. **Bigdan, Andrew.** Idempotency in automation. [Online] 2. Únor 2025. [Citace: 24. Říjen 2025.] https://andrewbigdan.me/idempotency-in-automation-guide/.

9. **Mike Tyson of the Cloud.** The Origins of Infrastructure as Code: A Brief History of DevOps. [Online] 23. Duben 2023. [Citace: 24. Říjen 2025.] https://medium.com/@mike\_tyson\_cloud/the-origins-of-infrastructure-as-code-a-brief-history-of-devops-a883d8877f19.

10. **Holdsworth, Jim a Badman, Annie.** What is infrastructure as code (IaC)? [Online] [Citace: 24. Říjen 2025.] https://www.ibm.com/think/topics/infrastructure-as-code.

11. **Team Copado.** Declarative vs Imperative Programming for Infrastructure as Code (IaC). [Online] 4. Červenec 2022. [Citace: 24. Říjen 2025.] https://www.copado.com/resources/blog/declarative-vs-imperative-programming-for-infrastructure-as-code-iac.

12. **Fashakin, Alexander.** Agent Vs Agentless Configuration Management. [Online] 22. Září 2025. [Citace: 28. Říjen 2025.] https://attuneops.io/agent-vs-agentless-configuration-management/#The\_Best\_Solution\_To\_Configuration\_Management.

13. **Oracle.** VirtualBox. [Online] [Citace: 10. Říjen 2025.] https://www.oracle.com/virtualization/virtualbox/.

14. **VirtualBox.** About Oracle VirtualBox. [Online] [Citace: 10. Říjen 2025.] https://www.virtualbox.org/manual/topics/Introduction.html#ct\_about-virtualbox.

15. **HashiCorp.** Introduction to Vagrant. [Online] [Citace: 10. Říjen 2025.] https://developer.hashicorp.com/vagrant/intro.

16. **Wikipedia.** Infrastructure as Code. [Online] 11. Duben 2024. [Citace: 17. Říjen 2025.] https://cs.wikipedia.org/wiki/Infrastructure\_as\_Code.

17. —. Secure Shell. [Online] 4. Říjen 2025. [Citace: 17. Říjen 2025.] https://cs.wikipedia.org/wiki/Secure\_Shell.

18. **Microsoft.** Důležité informace o zabezpečení vzdálené komunikace PowerShellu s využitím WinRM. [Online] 21. Květen 2025. [Citace: 17. Říjen 2025.] https://learn.microsoft.com/cs-cz/powershell/scripting/security/remoting/winrm-security?view=powershell-7.5.

1. Seznam přiložených souborů
2. Seznam Použitých nástrojů

* Gemini AI (dostupné online na: <https://gemini.google.com/app>)
  + Prompt: “Napiš mi strukturu pro maturitní práci o automatizaci s pomocí Ansiblu s nižší hranicí 4000 slov“