Univerzita Pardubice

Fakulta elektrotechniky a informatiky

Monitoring síťové infrastruktury v GNS3

Vladislav Ermolaev

Bakalářská práce

2022

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Vladislav Ermolaev**
Osobní číslo:	**I18118**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Téma práce:	**Monitoring síťové infrastruktury v GNS3**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování

Cílem bakalářské práce je monitoring síťové infrastruktury (servery a jejich služby, aktivní síťové prvky, síťová komunikace) fiktivní firmy v prostředí GNS3. Výběr vhodného nástroje pro monitoring bude vztažen na potřeby fiktivní firmy a jeho výběr bude odůvodněn. Práce bude obsahovat popis všech klíčových aktivit (GNS3, technologie pro monitoring, SNMP).

Rozsah pracovní zprávy: **30**
Rozsah grafických prací:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

JULIAN, Mike. Practical Monitoring: Effective Strategies for the Real World. O'Reilly Media, 2017. ISBN 978-1491957356.
C. NEUMANN, Jason. The Book of GNS3: Build Virtual Network Labs Using Cisco, Juniper, and More. 1. San Francisco: William Pollock, 2014. ISBN 978-1-59327-554-9.
KOCJAN, Wojciech a Piotr BELTOWSKI. Learning Nagios. 3. Birmingham, UK: Packt Publishing, 2016. ISBN 978-1-78588-595-2.

Vedoucí bakalářské práce: **Ing. Soňa Neradová, Ph.D.**
Katedra informačních technologií

Datum zadání bakalářské práce: **31. října 2020**
Termín odevzdání bakalářské práce: **14. května 2021**

L.S.

Ing. Zdeněk Němec, Ph.D. v.r.
děkan

Ing. Jan Panuš, Ph.D. v.r.
vedoucí katedry

V Pardubicích dne 26. února 2021

Prohlašuji:

Tuto práci jsem vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Beru na vědomí, že v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů, a směrnicí Univerzity Pardubice č. 9/2012, bude práce zveřejněna v Univerzitní knihovně a prostřednictvím Digitální knihovny Univerzity Pardubice.

V Pardubicích dne 10. 5. 2022

Vladislav Ermolaev

Poděkování

Rád bych tímto poděkoval vedoucí bakalářské práce Ing. Soně Neradové, Ph.D. za odborné vedení, veškerou pomoc a cenné rady při zpracování této bakalářské práce.

ANOTACE

Předmětem této bakalářské práce je monitoring síťových zařízení. V teoretické části budou uvedeny typy monitoringu, jejich účely a způsoby provádění. Také bude představen nástroj GNS3, sloužící k emulaci počítačových sítí. Praktická část se zabývá emulací podnikové síti, v rámci které budou implementovány vhodné strategie monitoringu. Proto, aby monitoring probíhal nepřetržitě, bude použít nástroj Nagios Core, který umožňuje automatizaci zasílaní dotazu o současném stavu síťového zařízení.

KLÍČOVÁ SLOVA

Monitoring, GNS3, Nagios, SNMP, Zabbix, topologie průmyslové síti

TITLE

Network infrastructure monitoring in GNS3

ANNOTATION

The subject of this bachelor's thesis is network device monitoring. Theoretical part will include description of several monitoring types, theirs purpose and ways to conduct them. It also will include a presentation of GNS3, a tool for computer networks emulation. Practical part looks into a emulation of an enterprise network, where some appropriate monitoring strategies would be implemented. To make sure that monitoring would run continuously, Nagios Core software will be used. This tool allows to send requests to devices automatically.

KEYWORDS

Monitoring, GNS3, Nagios, SNMP, Zabbix, enterprise network topology

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM ZKRATEK

GNS3	Graphical Network Simulator-3
SNMP	Simple Network Management Protocol
OSI	Open Systems Interconnection
ISO	International Organization for Standardization
TCP	Transmisson Control Protocol
IP	Internet Protocol
IPV4	Internet Protocol Version 4
NAT	Network Address Translation
IPV6	Internet Protocol Version 6
TTL	Time To Live
BGP	Border Gateway Protocol
AS	Autonomous System
RIP	Routing Information Protocol
RIPv2	Routing Information Protocol Version 2
EIGRP	Enhanced Interior Gateway Routing Protocol
DUAL	Diffusing Update Algorithm
OSPF	Open Shortest Path First
IS-IS	Intermediate System – Intermediate System
UDP	User Datagram Protocol
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
FTP	File Transfer Protocol
SSH	Secure Shell Protocol
ICMP	Internet Control Message Protocol
IOS	Internetwork Operating System
MIB	Management Information Base
IPFIX	IP Flow Information Export
SCTP	Stream Control Transmission Protocol
RDD	Round-Robin Database

ÚVOD

Dnešní míra rozvoje informačních technologií a počítačových sítí vznáší velké požadavky na síťovou infrastrukturu. Běžný uživatel dnes očekává, že síťová služba je nejen nepřetržitě připravena obsloužit kterýkoliv z jeho požadavků, ale i to, že odpověď na jeho požadavek zaručeně dorazí v řadě několika milisekund. Služba, která tyto požadavky nesplňuje, nemůže být efektivně použita pro komerční účely.

Hlavní roli při plnění úkolu zajištění rychlého a nepřetržitého provozu služby hrají zejména aplikační a databázové servery, a také směrovače, které řídí tok dat napříč datovou síti. Podnik, který chce na trhu uspět, měl by být schopen rychle identifikovat a odstraňovat vzniklé závady a mít možnost se rychle vrátit do provozuschopného stavu v případě selhání některého ze zařízení. Ten podnik by také mělo zajímat, zda infrastruktura je využívána efektivně a zda nedochází k zahlcovaní na některém z uzlů, který zpomaluje provoz celého systému. Řešením výše uvedených problémů je monitoring síťové infrastruktury.

Cílem bakalářské práce je monitoring síťové infrastruktury (servery a jejich služby, aktivní síťové prvky, síťová komunikace) fiktivní firmy v prostředí GNS3.

1 MODELY OSI A TCP/IP

Pro vysvětlení principů síťové komunikace se nejčastěji používá referenční model OSI. Tento model vznikl v roce 1983 jako výsledek snahy ISO standardizovat výměnu dat mezi rozlišnými počítačovými systémy. Základní myšlenkou modelu je rozdělení zodpovědnosti na sedm rozdílných vrstev, kde každá úroveň plný svůj úkol a využívá svých abstrakci pro představení komunikace.

Model prezentuje několik úrovni, nespecifikuje však jakým konkretně způsobem by měla komunikace probíhat. Jeho účelem je vymezit rozsah odpovědností jednotlivých vrstev. Každá úroveň poskytuje své služby výše ležící úrovni a komunikuje s ní pomocí rozhraní. Protokoly jsou vnitřní záležitostí každé vrstvy a můžou být změněny tak, že nedojde ke změně ostatních vrstev. [1]

Vrstvy a jejich funkce jsou představené na obrázku 1.1.

Aplikační vrstva	• Poskytuje standardní služby, jako je virtuální terminál, přenos souborů a úloh atd.
Prezentační vrstva	• Skrývá rozdíly reprezentace dat mezi neshodnými systémy • Řeší kódování, šifrování a kompresi dat
Relační vrstva	• Spravuje uživatelská sezení • Navazuje a ruší jednotlivá sezení
Transportní vrstva	• Doručuje zprávy mezi dvěma koncovými body • Poskytuje spolehlivý způsob doručení zpráv
Síťová vrstva	• Směruje pakety podle unikátních síťových adres • Poskytuje prostředky k řízení toku a zahlcení linky
Linková vrstva	• Zapouzdřuje pakety do rámců • Detekuje a opravuje chyby, které se můžou vyskytnout během přenosu
Fyzická vrstva	• Slouží rozhraním mezi zařízeními a přenosovým médiem • Definuje optické, elektrické a mechanické vlastnosti

Obrázek 1: Vrstvy modelu OSI[4]

Pokud je vrstvový model vytvořený pro určitou technologii, poté se označuje jako síťová architektura nebo protokolový síťový model příslušné technologie. Příkladem je asi nejznámější architektura TCP/IP.

Relace mezi modely ISO a TCP/IP jsou zobrazený v tabulce 1.

Tabulka 1: Vrstvy podle modelů OSI a TCP/IP

Model OSI	Model TCP/IP	Protokoly
Aplikační vrstva Prezentační vrstva Relační vrstva	Aplikační vrstva	HTTP, FTP, SNMP atd.
Transportní vrstva	Transportní vrstva	TCP, UDP
Síťová vrstva	Síťová vrstva	IP
Linkova vrstva Fyzická vrstva	Linkova vrstva	Ethernet, IEEE 802.11

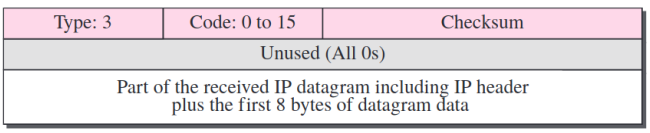
2 PROSTŘEDKY PRO SPRÁVU A MONITORING SÍTĚ

Většina činností spojených s monitoringem je obvykle prováděná pomocí speciálního softwaru. Nabídka konkretních prostředků je velmi rozmanitá a zahrnuje jak poměrně jednoduché utility pro příkazový řádek, tak i sofistikované systémy, které poskytují kompletní řešení v oblasti správy a monitoringu. I přes veškeré odlišnosti a různorodost konečných nástrojů existuje něco, co je pojí dohromady. Tím jsou níže ležící protokoly, které jsou použity drtivou většinou softwarových prostředků pro plnění svých činnosti. Právě tyto protokoly jsou popsány v této kapitole.

2.1 ICMP

Základním prostředkem pro zjišťování dostupnosti uzlů je protokol ICMP. Tento protokol poskytuje základ pro širokou škálu softwarových nástrojů a je integrální části technologií sady protokolů TCP/IP. Podpora ICMP je nezbytná pro každý IP prvek v síti.

Důvodem existence ICMP je neschopnost protokolu IP garantovat doručení paketu. Pokud by navíc neexistovala možnost zjistit, jaký úsek sítě je zodpovědný za zahození paketu, mělo by to za následek znesnadnění lokalizace chyb a výpadků. Pomocí ICMP probíhá komunikace aktivních síťových prvků a konečných hostů mezi sebou. Hlavním úkolem protokolu je oznamování nemožnosti doručit paket jeho adresátovi v důsledku různých příčin. ICMP je protokolem síťové úrovně, ICMP zpráva je ale vždy zapouzdřena do IP paketu.



Obrázek 2: Formát paketu ICMP[7]

Důvod zasláni ICMP se určuje pomocí její typu a kódu. Typ uvádí o jaký druh zprávy se jedná. Kód popisuje podrobnosti situace. Tyto hodnoty jsou uvedeny v záhlaví zprávy. ICMP zpráva může být zaslána z následujících důvodů:

- Destinace není dosažitelná.

  Typ zprávy 3, kód 0 znamená nedosažitelnost sítě, 1 – nedosažitelnost hostu, 2 – byl použit nepodporovaný protokol, 3 – byl použit neočekávaný port, 4 – pro doručení paketu je zapotřebí ho fragmentovat a zároveň fragmentace paketu je zakázána odesilatelem, 5 – byla použita špatná zdrojová adresa. Zprávy s kódy 0, 1, 4 a 5 můžou být obdrženy od aktivních síťových prvků, kódy 2 a 3 znamenají, že zpráva pochází od hostu.
- Vypršení času.

  Každý IP paket má konečnou životnost a táto zpráva se zasílá, pokud došlo k zahození paketu z důvodu hodnoty 0 v poli TTL. Také může nastat situace, že příjemce nemůže zreplikovat paket kvůli absence některého z fragmentů, který nebyl doručen. Typ zprávy 11, kód 1 znamená, že důvodem je nulová hodnota TTL, 1 – některý z fragmentů nebyl včas doručen.
- Výskyt neočekávaného parametru v záhlaví IP paketu.

  Typ zprávy 12, kód je vždy 0. Pro sdělení polohy parametru se používá speciální pole v záhlavi.
- Zahlcení směrovače.

  Paket je zahozen kvůli tomu, že směrovač nestíhá odesílat pakety a jeho paměť je již přeplněná. Typ zprávy 4, kód je vždy 0.
- Přesměrování.

  Pokud směrovač rozhodne, že existuje lepší možnost směrování paketu (například v okamžik, kdy směrovač musí odeslat paket zpět do stejné sítě, z níž paket dorazil) může byt tato informace zaslána zpět odesilatelovi pomocí tohoto druhu zprávu. Typ zprávy 5, kód nula znamená přesměrovaní pro síť, 1 – pro konečný uzel, 2 – přesměrování pro síť se změnou hodnoty v poli TOS, 3 – přesměrování pro konečný uzel se změnou hodnoty v poli TOS.
- Echo zprávy.

  Slouží za účelem testování konektivity mezi prvky v síti. Kód je vždy 0, typ vyjadřuje druh zprávy. Hodnota 8 znamená zprávu, 0 se používá u odpovědí.
- Časová razítka.

  Pomocí výměny razítek můžou být směrovače synchronizovány. Kód je vždy 0, typ 13 se používá u výchozí zprávy, 14 – u odpovědi.

- Informační požadavky a odpovědi.

  Zastaralý druh zpráv, dříve se používal pro automatickou konfiguraci hostů. Kód je vždy 0, typ 15 u požadavků, 16 – u odpovědí.[8]

ICMP zprávy jsou široko využity různými monitorovacími nástroji. Utilita ping například zasílá Echo zprávy pro testování konektivity mezi dvěma uzly. Dalším příkladem může být utilita traceroute, která využívá toho, že každý z dotázaných aktivních síťových prvků může odpovědět pomocí ICMP zprávy s typem 11. Oznamuje tím, že paket byl zahozen v důsledku nulové hodnoty v poli TTL. Pokud dotazující bude záměrně posílat dotazy se zvyšující se hodnotou TTL (jedná a více), může získat informace o všech směrovačích, které spojují jeho a cílovou sítě.

2.2 SNMP

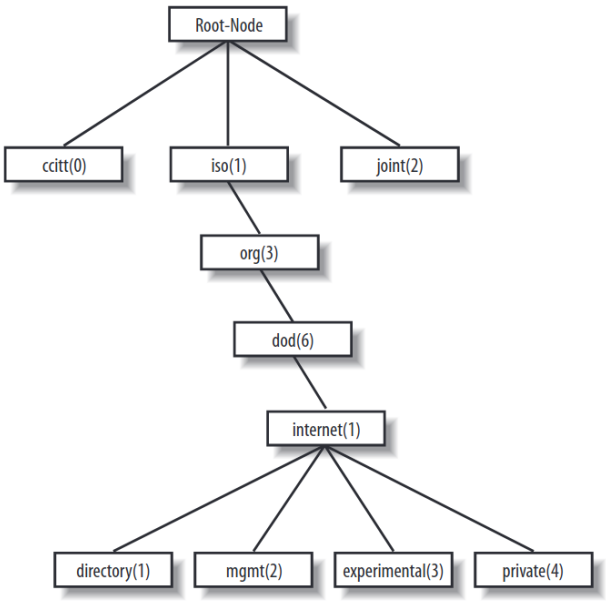
Protokol SNMP byl vyvinut za účelem usnadnění správy sítě. Zařízení, které tento protokol podporují, můžou sdělovat informace o svém stavu správci sítě. Na rozdíl od ICMP však není SNMP podporován na všech zařízeních, obyčejné domácí směrovače podporu SNMP obvykle postrádají. Drtivá většina průmyslových zařízení SNMP nicméně podporují.

SNMP definuje dva druhy entit. Prvním z nich je manažer – server, kde běží software pro správu sítě. Tento software komunikuje s druhým typem entit – agentem. Agent je software, který běží na aktivním síťovém prvku nebo na hostě. Může se jednat o samostatný program, občas můžou být SNMP agenty implementovány jako součást operačního systému zařízení (například Cisco IOS).

Výměna dat mezi uzly také může probíhat podle jednoho ze dvou scénářů. Manažer má možnost zaslat dotaz na agenta, aby zjistil jeho aktuální ukazatele. Takové dotazování může probíhat jednou za určitý čas a díky tomu lze shromažďovat statistiky, které pak můžou být použity k řešení širokého spektra problémů. Tento způsob ale není příliš vhodný k detekci a lokalizace selhání. Kdyby došlo k poruše, údaje o selhání některého ze zařízení by byly získány jen po příštím kole dotazování.[9]

Druhá možnost komunikace spočívá v tom, že agent může sám provádět kontrolu svých parametrů, a v okamžik, kdy některá z hodnot přesáhne předem nastavenou kritickou hranici, bude zaslána speciální zpráva zpět manažerovi. Manažer pak tuto zprávu zpracuje a rozhodně, jakým způsobem bude na ni reagovat. [9]

Aby mohla probíhat komunikace mezi manažerem a agenty, každý ze kterých může byt vyroben jinou společností, musí mít všechna zařízení nějaký společný „slovník“. Tuto roli plní speciální databáze – MIB. MIB je v podstatě jen textový soubor, ve kterém jsou popsány vlastnosti objektů a vazby mezi nimi. Každý z agentů může poskytovat přístup ke svým unikátním druhům údajů, většina zařízení však podporuje objekty z databáze MIB-II. V této databázi jsou definovány základní údaje síťových zařízení. Druhou důležitou databází je Host Resources MIB, která rozšiřuje seznám objektů o položky pro kontrolu konečného hosta v sítě, kam patří například volné místo na disku, využití paměti atd. Podpora Host Resources MIB je dost široká, nejedná se však o část standardu SNMP a tím pádem nelze spoléhat na to, že zařízení, které podporuje SNMP, bude podporovat i tuto rozšiřující databázi. Manažer obvykle podporuje všechny široce dostupné databáze, navíc umožňuje přidávat i vlastní definice.[9]



Obrázek 3: Struktura MIB[9]

Objekty jsou seřazený v rámci stromové hierarchii. Hierarchie obsahuje podstromy (složky) a listy (samotný údaj) Unikátní identifikátor objektu se skládá z celých čísel a teček mezi nimi. Každému číslu je přiřazen alternativní textový zápis.

Existuje několik verzí protokolů SNMP. Verze 1 je poněkud omezenější než verze 2, je ale podporována u největšího počtu zařízení. Verze 2 přidává některou funkcionalitu navíc, trpí ale stejným kritickém nedostatkem v oblasti bezpečnosti. Obě verze 1 a 2 umožňují

Tabulka 2: Některé významné podstromy databáze MIB-II

Jméno	OID	Popis
system	*1.3.6.1.2.1.1*	Definuje seznam objektu spojených se systémovými údaji jako jméno systému, čas uběhlý od startu systému atd.
interfaces	*1.3.6.1.2.1.2*	Sleduje stav všech rozhraní zařízení. Eviduje se počet přijatých a odeslaných paketů, chybných a zahozených paketů atd.
at	*1.3.6.1.2.1.3*	Již zastaralý podstrom, který je v MIB-II jen s cílem zaručit zpětnou kompatibilitu.
ip	*1.3.6.1.2.1.4*	Zahrnuje objekty spojené s provozem síťové úrovni včetně směrování.
icmp	*1.3.6.1.2.1.5*	Sleduje údaje spojené s ICMP.
tcp	*1.3.6.1.2.1.6*	Tady je evidován stav TCP spojení.
udp	*1.3.6.1.2.1.7*	UDP statistiky, počet přijatých a odeslaných datagramů atd.

ukládat heslo pouze jako nešifrovaný řetězec. To nutí k tomu, aby monitorovací činnost probíhala vždy pouze v rámci privátní zabezpečené sítě.

Verze protokolu 3 nenabízí oproti předchozím verzím žádnou dodatečnou funkcionalitu, dovoluje ale šifrovat heslo. Tím se odstraňuje bezpečnostní problém předchozích verzí. I když verze 3 nenabízí nové funkce, došlo ke změně základních definicí protokolu. Tak například místo manažeru a agentu verze 3 zavádí pojem SNMP entity. Každá entita podporuje základní sadu operací, mezi které patří například šifrování a zasílaní zpráv správci. Dál můžou agenty podporovat dodatečné aplikace, které slouží ke zjištění konkretních údajů agenta.[9]

Činnost SNMP probíhá v aplikační vrstvě. Protokol definuje následující typy zprav:

- GET. Zasílán manažerem pro dotazování jedné proměnné.
- GETNEXT. Zasílán manažerem pro zjištění hodnoty té proměnné, která následuje proměnnou uvedenou v dotaze. Pomocí této zprávy lze projít celou stromovou strukturu proměnných zařízení.
- GETBULK. Podporován od verze 2. Umožňuje získávat hodnoty více proměnných najednou a navíc umožňuje nevracet všechny požadované údaje.
- SET. Zasílán manažerem s cílem změnit hodnotu jedné nebo více proměnných.
- GETRESPONSE. Zasílán agentem jako odpověď na dotazy GETREQUEST, GETNEXTREQUEST a SETREQUEST.
- TRAP. Zasílán agentem s cílem nahlásit manažerovi změnu svého stavu, která vyžaduje pozornost.

- NOTIFICATION. Podporován od verze 2. Tento typ vznikl s cílem standardizace formátu zpráv typu TRAP. Důvodem bylo to, že formát TRAP zprávy ve verzi 1 nebyl shodný s formátem zpráv GET a SET.
- INFORM. Podporován od verze 2. Slouží k komunikaci manažerů mezi sebou, pokud je jich v síti víc než jeden.
- REPORT. Typ byl definován ve verzi 2, ale nebyl prakticky implementován. Ve verzi 3 je tento typ vyhrazen pro komunikaci entit mezi sebou s cílem nahlásit problém zpracování SNMP zprávy.

SNMP je obvykle provozován přes protokol transportní úrovně UDP. SNMP zařízení používají port 161 pro zasílaní dotazů a příjem odpovědí a port 162 pro zachycení zpráv iniciovaných agentem. SNMP může být implementován i pomocí TCP, tato varianta je ale vhodná jen za některých okolností (nejčastěji se jedná o některou z proprietárních implementací).

Pro interakci s protokolem SNMP je nejčastěji zapotřebí použít speciální software. Vzhledem k rozšířenosti protokolu existuje spousta nástrojů, které ho podporují. Jako příklady lze uvést například nástroje Zabbix a Nagios, které jsou často používány pro monitoring síťových zařízení.

3 NÁSTROJE PRO MONITORING

Uvedené v předchozí kapitole protokoly jsou zcela nepostradatelné pro zajištění monitoringu. Ale samotné protokoly jsou jen prostředky pro popis interakce dvou nebo víc uzlů. Pro jejich zužitkování je zapotřebí použit softwarové nástroje, které výše uvedené protokoly podporují. V této kapitole je uvedeno několik nástrojů, které můžou být přímo použité za účelem sledování stavu sítě.

Je nutné říct, že monitoring je vždy těsně vázán na podmínky každého konkretního podniku. Tím pádem nelze jednoznačně říct, že nějaký software je vždy lepší volbou než jiný. Výběr nástroje pak záleží jednak na poskytované funkcionalitě, jednak na zkušenostech a zvyklostech technického personálu konkretní firmy.

V této kapitole jsou zohledněny tři populární nastroje, které můžou být využity k monitorovací činnosti. Jedná se o Cacti, Zabbix a Nagios. Důvodem zařazení pravě těchto třech nástrojů do této kapitoly je to, že každý z nich je volně šiřitelný software, který navíc poskytuje dostatečnou míru funkcionality k tomu, aby byl jediným potřebným nástrojem.

Alternativou k tomu může sloužit kompozice různých dílčích nástrojů v rámci jednoho systému „na míru“ podniku. Obvykle je ten přistup používán ve větších společnostech, neboť je složitější na vývoj, nasazení a údržbu.

3.1 Cacti

Nástroj Cacti slouží k automatickému dotazování síťových prvků a vytváření grafů na základě nasbíraných údajů. Dotazování probíhá s využitím sady nástrojů Net-SNMP, která implementuje protokol SNMP. Získané údaje jsou pak uloženy pomocí databázového enginu RRDTool. První vývojář Cacti, Ian Berry, zjistil, že nástroj RRDTool je dost flexibilní na to, abych mohl byl použit pro generaci grafů a reportů, chybí mu vsak přivětivé uživatelské rozhraní. Tato myšlenka stala za zrodem Cacti.[10]

Operační chod Cacti lze rozdělit na následující části:

- Sběr dat. Shromažďování údajů probíhá pomocí pravidelného dotazování síťových uzlů prostřednictvím protokolu SNMP. Tím pádem, každé zařízení, které podporuje protokol SNMP, může být monitorováno pomocí Cacti. Za dotazování odpovídá speciální softwarový prvek – tzv. poller. Cacti nabízí uživatelovi dva pollery. První

z nich je součásti standardní instalace Cacti a je napsán v jazyce PHP. Druhou možností je využití polleru Spine – tento prostředek pro sběr dat je napsán v jazyce C a podporuje multivlaknové zpracování dat. Spine je doporučen pro použití ve větších podnikových sítích.

- Skladování dat. Cacti ukládá údaje pomocí nástroje RRDTool. Databáze RRD slouží k ukládaní časových řad údajů. Počet záznamů v databázi RRD nikdy nepřesahuje stanovený limit, starší data jsou konsolidována do větších shluků.
- Zobrazení údajů. Jednou z nejdůležitějších vlastností nástroje RRDTool je jeho schopnost zobrazovat nasbíraná data ve formě grafů. Cacti využívá tuto funkcionalitu a navíc poskytuje možnost současného zobrazení hned několika různých charakteristik.[10]

Základní instalace Cacti dovoluje jen dotazování uzlů, které bylo inicializováno pollerem. Zařízení nemají možnost oznámit sběrači dat, že během provozu došlo k selhání nějaké komponenty. Tento nedostatek lze odstranit pomocí pluginů. Pluginy rozšiřují funkcionalitu Cacti a dovolují přizpůsobit Cacti požadavkům podniku.

3.2 Zabbix

Dálším příkladem monitorovacího nástroje je Zabbix. Ten poskytuje přibližně stejnou funkcionalitu jako Nagios, některé jeho vlastnosti se ale liší.

Tak například pro ukládání konfiguraci používá Zabbix databázi, když Nagios uchovává svá nastavení v podobě konfiguračních souborů. Kvůli tomu je Zabbix o něco méně šetrnější na systémové prostředky, nabízí však jednodušší a centralizovány přístup ke konfiguraci.

Silnou stránkou Zabbix je sofistikovanější webové rozhraní. Nagios (bez použití pluginů) zobrazuje jen základní údaje objektů, Zabbix umožňuje správci přizpůsobit zobrazované údaje podle jeho potřeb. Kromě toho i základní instalace Zabbix podporuje zobrazování grafů, když Nagios potřebuje nejdříve nainstalovat odpovídající plugin (např. NagVis).[12]

Lze obecně říct, že konfigurace Zabbix je jednodušší. Kromě centralizovaného ukládaní nastavení přispívá k tomu i podpora šablon. Šablona je souborem předem definovaných nastavení pro monitorovaní činnosti služby nebo protokolu. Díky tomu lze rychleji uvést systém monitoringu do provozu.[12]

Hlavní nevýhodou Zabbix je méně široká nabídka pluginů. Modulární přístup Nagios umožňuje přizpůsobit ho potřebám více uživatelů. Nagios také poskytuje více druhu agentů, které běží na monitorovaných zařízeních.[12]

3.3 Nagios

Posledním uvedeným v této kapitole monitorovacím prostředkem je nástroj Nagios. Tento nástroj vznikl v roce 2002 pod názvem NetSaint pro operační systém Linux. Nagios podporuje jak aktivní (přímé dotazování uzlů), tak i pasivní (zaslání zprav z uzlu na server Nagios) kontroly.

Nagios umožňuje monitorovat stav nejen zařízení (servery, pracovní stanice a síťové prvky) ale také i běžících na nich služeb (například webového serveru jako instance procesu httpd). Každá služba je asociována s zařízením, na němž běží. Zařízeni a servisy lze zařadit do skupin s cílem snadnější orientace.[11]

Důležitou vlastnosti Nagios je to, že kontrola prvků probíhá výlučně pomocí využití pluginů. Samotný Nagios jen koordinuje jejich činnost. Výsledkem je flexibilnější systém, který ale potřebuje víc času na konfiguraci než například výše zmíněný Cacti.

Výsledky provedených kontrol nejsou zobrazeny jako číselné hodnoty (např. procento zatížení procesoru). Místo toho každý zkoumaný objekt může se nacházet v jednom ze čtyř předem definovaných stavů. Přechod mezi stavy probíhá po dosažení některé kritické přednastavené hodnoty. Tento přístup dovoluje správcům abstrahovat výsledky jednotlivých kontrol a operovat na vyšší úrovni. Toto je žádoucí zejména ve větších systémech, kdy není potřeba vědět přesné hodnoty indikátorů, ale je důležité rychle ocenit nastalou situaci. Přípustnými stavy jsou „OK“, „WARNING“, „CRITICAL“ a „UNKNOWN“. Vytvořené reporty umí zobrazit počet objektu ve stavech „WARNING“ a „CRITICAL“, což pomáhá pochopit, které části systému potřebují pozornost.[11]

Další výhodou Nagios je vyspělý systém závislosti. Například, pokud by došlo k výpadku na směrovači, všechny zdroje za ním by se také mohly jevit jako nepřístupné. Výsledkem by byla spousta chybových hlášení a správcům by trvalo nějaký čas, předtím než by problémový prvek mohl být lokalizován. Nagios umí tuto nepříjemnost odstranit. Označením prvku jako závislého na jiném prvku lze dosáhnout toho, že při selhání některé komponenty další závislé na ní uzly se již nebudou zkoumat. Stejným způsobem lze označit i závislost jedné služby na jiné.[11]

Během provozu mohou nastat okamžiky krátkodobého výpadku některé komponenty např. kvůli zahlceni sítě nebo restartu počítače. Tyto situace není zpravidla zapotřebí řešit, mohou se však stát zdrojem zbytečné informace. To má za důsledek rozptýlení pozornosti obslužného personálu a spuštění falešných poplachu. Aby se tomu předešlo, Nagios rozlišuje mezi měkkými a tvrdými stavy monitorovaných objektů. Objekt přechází do měkkého stavu hned poté, co Nagios objeví změnu jeho indikátorů. V tomto stavu objekt setrvává jen dočasně. Pokud další kontroly budou vracet stejný výsledek během několika pokusů (konkrétní číslo pokusu se zadává v konfiguračních souborech), bude měkký stav změněn na stav tvrdý, který se považuje za stabilní. Teprve teď bude systém na výpadek nějakým způsobem reagovat.[11]

Lze také nastavit časové úseky, kdy budou některé objekty během provádění kontrol ignorovány. To se hodí zejména v těch situacích, kdy je nutně dočasně vypnout některé komponenty sítě např. kvůli údržbě.[11]

Nagios existuje ve dvou podobách: Nagios Core a Nagios XI. Nagios Core je volně šiřitelný software, když Nagios XI je proprietární verze, která nabízí větší míru funkcionality a má propracovanější uživatelské prostředí.

4 METODOLOGIE MONITORINGU

Každý podnik provozuje svou činnost ve svém zvláštním prostředí a má své specifické požadavky i na systém, který bude jeho činnost monitorovat. To ale neznamená, že neexistuje společný metodologický základ, který může být výchozím bodem v otázce zajištění sledovatelnosti stavu služeb a síťových prvku.

Tato kapitola uvádí metodiku FCAPS, která byla vyvinuta s cílem usnadnění správy síťových zařízení. Popisuje také charakteristiky některých zapojených do komunikace prvků, které je obecně vhodné sledovat.

4.1 Metodika FCAPS

Metodika FCAPS byla vytvořena organizací ISO a její cílem je popsat nezbytnou funkcionalitu systému pro správu sítě. Metodika definuje pět různých okruhů, každý z nich je zodpovědný za určitou část správy systému. Samotná zkrátka FCAPS je akronym, kde každé písmeno odkazuje na příslušný okruh. Těmito okruhy jsou:

- Fault Management – systém reakce na poruchy a selhání,
- Configuration Management – správa konfigurace,
- Accounting Management – správa účtů,
- Performance Management – systém řízení výkonnosti,
- Security Management – systém řízení bezpečnosti.

Cílem zavedeni této metodiky je stimulace proaktivního přístupu k monitoringu. Výhodou tohoto přístupu je možnost všimnout si nedostatků a omezení systému ještě předtím, než dojde k poruše a tím pádem i k finančně újmě.

4.1.1 Fault Management

Hlavním úkolem tohoto systému je reagovat na nastalé výpadky v síti. Každý výpadek by měl byt detekován, lokalizován a správci by měly dostat příslušné upozornění. V podnikových systémech nejsou žádné výpadky akceptovatelné a proto jejich odstranění by mělo probíhat co nejrychleji.

Standardní postup při výskytu selhání obvykle vypadá následovně:

- Lokalizovat vznikly problém pomocí speciálních nástrojů,

- Odstranit problém,
- Zadokumentovat vznikly problém a kroky k jeho odstranění.[9]

Nedodržení posledního kroku je přípustné, není to ovšem doporučené. Pokud se stejný problém vyskytne i v budoucnu, existující dokumentace přispěje k jeho rychlejšímu odstranění.

4.1.2 Configuration Management

Některé chyby se mohou vyskytnout v důsledku výměny některého ze zařízení za nové nebo kvůli změně v konfiguračních souborech. Takovéto chyby pak mohou být těžce odhalitelné, zvlášť v situacích, kdy se projeví až po určitý čas od provedení změny. Systém správy konfigurace eviduje všechny provedené změny a umožňuje rychle vrátit systém do provozuschopného stavu. Údaje o změnách můžou být uloženy ve speciální služebné databázi. Obvykle se evidují tyto údaje:

- Verze operačního systému a aplikací,
- Počet a rychlost síťových rozhraní,
- Počet pevných disků,
- Počet procesoru,
- Velikost paměti.[9]

4.1.3 Accounting Management

Systém správy účtů slouží k zajištění spravedlivého přiřazení systémových zdrojů uživatelům. Prostředkem k tomu slouží především různé kvóty. Pokud systém nějakým způsobem účtuje uživatelské poplatky, tato činnost také probíhá v rámci daného systému.

4.1.4 Performance Management

Systém řízení výkonnosti umožňuje porovnávat míru aktuálního zatížení systému vůči hodnotám, které jsou považovány za normální pro tento systém. Nejdřív probíhá sběr dat o zatížení systému v rámci obecného provozu. Pak probíhá analýza nasbíraných údajů s cílem odvození hodnot, které se budou považovat za normální. Dál se definují prahy kritických hodnot pro každou ze sledovaných charakteristik. Pokud během dalšího provozu bude některý z prahů překonán, může být správce na tuto skutečnost upozorněn. Každá

takováto anomálie by měla být důsledně prozkoumána, neboť se může jednat o problém, který může zhoršit výkon celého systému.

4.1.5 Security Management

Systém řízení bezpečnostní má několik rozdílných úkolů. Prvním z nich je omezení přístupu uživatelů k některým prostředkům systému. Může se jednat o data nebo například o systémové zdroje. Cílem je znemožnit nebo alespoň ztížit provádění útoků a krádež informací. Typickým prostředkem pro tento účel slouží nastavení uživatelům příslušných prav. Kromě obrany před útokem práva také chrání systém před uživatelskými chybami. Například chybný příkaz nebo nesprávné nastavení konfigurace nebudou moct ovlivnit celý systém, ale pouze jeho část.

Další nezbytností je šifrování dat a ověření jejich pravosti. K tomu slouží algoritmy šifrování a různé podoby elektronických certifikátů.

Je také zapotřebí mít schopnost detekovat útok a přijmout adekvátní protiopatření. Prostředkem k tomu slouží firewally, antivirové aplikace a speciální software pro detekci a reakci na útoky.

4.2 Objekty monitoringu

Tato podkapitola vymezuje některé důležité části podnikového systému a poukazuje na několik významných indikátorů, které by měly být sledovány.

4.2.1 Aplikace

Prvním důležitým bodem je monitoring stavu podnikových aplikací. Aplikace může být používána zákazníky podniku (např. elektronický obchod) nebo sloužit pro vyřízení interních záležitosti firmy. V každém případě firma spoléhá na bezproblémový běh aplikací a jejich výpadek může mít za následek ztrátu peněz a reputace v očích zákazníku. Proto je nezbytné kontrolovat stav běžících aplikací a mít možnost rychle vrátit aplikace do provozuschopného stavu.

Toto není jednoduchý úkol. Spousta firem totiž provozuje své aplikace metodou „černé skříňky“ a nemá přehled o tom, jakým způsobem aplikace plní své úkoly. V případě selhání aplikace pak nemají zaměstnance firmy možnost rychle opravit závadu a musejí

se obracet na vývojáře nebo na technickou podporu. Čas nedostupnosti služby se kvůli tomu protahuje, což jen zvyšuje ztráty podniku.

Jedním z nejefektivnějších způsobů sledování stavu aplikace je integrace systému generování metrik do cílové aplikace. Nedostatkem tohoto způsobu je jeho nákladnost, neboť tato činnost musí probíhat již během vývoje aplikace a vyžaduje komunikaci mezi vývojáři a představiteli firmy. Velkou výhodou tohoto přístupu je možnost zkoumat stav běžící aplikace a všimnout si její anomálního chování ještě předtím, než dojde ke skutečnému selhání.[13]

Jednodušší verzí předchozího způsobu je vytvoření speciálního endpointu, který slouží pro zobrazení aktuálního stavu aplikace. Typickou adresou pro tento endpoint bývá obvykle */health*. Dotazování koncového bodu lze automatizovat a nasbírané údaje pak mohou sloužit k analýze a detekci anomálií. Musí být vzata v potaz i bezpečnostní stránka, neboť firma obvykle nechce, aby údaje o stavu jejích aplikací byly přístupné široké veřejnosti.[13]

4.2.2 Servery

Kromě samotných bežících aplikací by měly být monitorovány i stroje, na nichž aplikace běží. Nejčastěji jsou sledovány základní systémové údaje jako například:

- zatížení procesoru,
- využití paměti,
- využití diskového prostoru a rychlost operací čtení a zápisu,
- střední počet procesů čekajících na zpracování.

Tyto údaje jsou důležité, neměly by však sloužit jako důvod pro spuštění poplachů, jinak hrozí zahlcení technického personálu proudem upozornění a varování, které reflektuje jen krátkodobý stav zařízení a obvykle není zapotřebí je řešit. Každopádně by ale měly být tyto údaje ukládány, protože v případě selhání některého ze zařízení můžou poskytnout informace o tom, zda-li problém vznikl v důsledku nedostatečné výkonnosti hardwaru.

U webových serveru je vhodné monitorovat počet dotazu za sekundu. Tento údaj je nejdůležitější pro kontrolu výkonnosti serveru. Jiným významným ukazatelem jsou kódy odpovědí serveru. Každá HTTP odpověď zahrnuje speciální kód, který popisuje její stav. Pokud dotaz byl patřičným způsobem vyřízen, nejčastěji vraceným kódem je *200 OK*. V případě, že během zpracování dotazu se vyskytla chyba, bude klientovi zpět zaslán kód z rodiny *400 Client Error* nebo *500 Server Error*. Právě tyto skupiny kódů by měly být

správci bedlivě sledovány, neboť jejich výskyt může znamenat existenci chyb v podnikových aplikacích. Kromě toho, velký počet kódu *401 Not Authorized* může být signálem probíhajícího útoku.[13]

Pro sledování výkonnosti databázového serveru může být použít počet současných připojení. Tento údaj ale není úplně vypovídající, protože nereflektuje reální míru zatížení serveru. Lepší variantou je sledování počtu dotazů za sekundu. Drtivá většina databázových serveru umožňuje ukládat kód samotného dotazu a čas strávený jeho zpracováním. Pomocí těchto údajů lze identifikovat příliš pomalé dotazy a následně se zaměřit na jejich optimalizaci.[13]

4.2.3 Síť

Síťová infrastruktura je páteřní komponentou každé firmy. Nelze zajistit větší míru dostupnosti služeb než dovoluje síť, pomocí které veškeré uzly spolu komunikují. Nejdůležitějším údajem o síťovém zařízení je jeho dosažitelnost. Pokud některý uzel neodpovídá, může to znamenat buď nesprávné zadaní konfigurace nebo selhání hardwaru. Dalším důvodem nedosažitelnosti může být chybná konfigurace dynamických směrovacích protokolů. V tomto případě je důležité zkontrolovat přístupnost uzlu hned z několika různých částí sítě, neboť chyba v konfiguraci nemusí být zadaná přímo na neodpovídajícím zařízení.

Velmi významnou části monitoringu sítě je evidence konfigurace. Jedním z nejčastějších důvodů nefunkčnosti uzlu nebo úseku sítě je nesprávné zadání nastavení. Pokud jsou všechny změny dohledatelné, lze rychle vrátit nastavení zpět a tím znovu zpřístupnit nedosažitelnou část sítě.

Dotazování aktivních síťových prvků obvykle probíhá s využitím protokolu SNMP. Každý výrobce může poskytovat přístup k různým údajům svých zařízení. Nejzajímavější z hlediska užitečných informací části aktivního síťového prvku jsou jeho rozhraní a většina výrobců umožňuje získávat z nich aktuální hodnoty. Z pohledu udržování výkonnosti infrastruktury jsou důležité zejména následující údaje:

- Přenosová kapacita. Teoretické maximum rychlosti datového přenosu tímto rozhraním. Statický údaj, který není zapotřebí aktivně monitorovat, slouží ale jako reference k použití následující charakteristiky,
- Propustnost. Reálná přenosová rychlost linky. Vždycky je menší než maximální kapacita. Nestačí jen občas získávat aktuální hodnotu propustnosti, toto měření by

reflektovalo stav linky jen v okamžiku samotného měření. Místo toho je doporučeno používat speciální nástroje jako *ipref2* nebo *bwctl*. Snížení propustnosti může být důsledkem výskytů chyb při přenosu. Nejčastěji se to stává kvůli nevhodnému nastavení parametrů linky,

- Latence. Doba potřebná k přenosu výchozího paketu k cíli a odpovědi zpět ke klientovi. Vysoká latence může mít negativní dopad na spokojenost uživatelů služby,
- Výskyt chyb, velký počet chyb na rozhraní obvykle slouží buď signálem nesprávné konfigurace nebo selhání pasivních síťových prvků (kabely, konektory atd.),
- Jitter. Tento údaj popisuje kolísání latence. Obvykle není zapotřebí tento údaj aktivně monitorovat, neboť je odvoditelný od konzistence hodnoty latence. Pokud se latence v čase nemění, hodnota jitter je nulová. [13]

Většina rozhraní v síti dostatečně velkého podniku slouží k připojení koncových pracovních stanic zaměstnanců k firemní sítě. Tato rozhraní obvykle není nutno kontrolovat. Jsou k tomu dva důvody. Prvním důvodem je náchylnost tohoto druhu rozhraní k chybám. Druhým je jejich relativně mála kritičnost v rámci celé sítě. Monitorování koncových rozhraní by přispělo spíš jen k zahlcení personálu velkým proudem málo významných upozornění a ztížilo by detekci a lokalizaci důležitějších problémů. [13]

Rozhraní, která vedou k serverům a na vnější sítě, jsou naopak kritické pro nerušený provoz podniku. Tato rozhraní by měla být pečlivě monitorována.

5 NÁSTROJ GNS3

Testování monitorovací strategie v praxi může být poměrně obtížnou záležitostí. Zavádění systému tak velkého rozsahu vyžaduje spoustu konfiguračních změn, každá ze kterých se může stát zdrojem nežádoucích chyb. Použití nevhodných řešení může jen zkomplikovat stávající situaci. Kromě toho nepromyšlené nasazení nějakého systému může zasáhnout i výkonnost sítě. Konečně i úspěšně implementována řešení můžou neodpovídat úkolům a jen zbytečně komplikovat život uživatelům.

Z uvedených výše důvodu je vždy lepší nejdřív testovat navržené řešení v rámci emulované sítě. Tento experiment je řadově levnější a jednodušší než zkouška „naostro“ v reálném prostředí. Chyby a nedostatky řešení lze pak odstranit a připravit k nasazení hned o něco víc optimalizovanou verzi.

Existuje několik programových prostředku, pomocí kterých lze emulovat síť a zařízení v ní. Tato kapitola pojednává o nástroji GNS3, jednom z nejpopulárnějších volně šiřitelných nástrojů pro emulaci sítě. Pomocí GNS3 lze emulovat nejen aktivní síťové prvky, ale i plnohodnotné počítače uvnitř emulované síti. Na těchto počítačích lze zprovoznit jak např. software pro poskytnutí síťových služeb, tak i monitorovací prostředky. Tím lze maximálně přiblížit emulované prostředí reálným budoucím podmínkám provozu.

Nejzajímavější vlastnosti GNS3 je jeho schopnost úplně emulovat některý stroj. Toto ho odlišuje například od nástroje Packet Tracer, který emuluje jen omezené množství příkazů. Rozdíl spočívá v tom, že Packet Tracer je spíš omezeným interpretem, než skutečným virtuálním strojem. GNS3 naopak emuluje virtuální hardware a umožňuje spustit na něm reálný software. Tento způsob poskytuje mnohem věcnější pohled na emulovaný objekt, je však náročnější na systémové prostředky. Velikost emulovaného systému je omezena výkonnosti stroje, na němž emulace probíhá. Tuto nepříjemnost lze do jisté míry zahladit rozdělením zátěže mezi několika reálnými stroji. GNS3 se řídí modelem klient-server, kde klientem je uživatelské rozhraní, a serverem – část softwaru, která se stará o emulaci.

Další silnou stránkou GNS3 je možnost propojit emulovanou síť s reálným vnějším síťovým prostředím počítače, na kterém emulace běží. Díky tomu nezůstává emulovaná síť v izolaci s ostatním světem a může tím pádem sloužit jako plnohodnotná část Internetu.

Nástroj GNS3 je kvůli svým vlastnostem využit v praktické části této práce pro emulaci síťového prostředí firmy.

5.1 Emulace síťových zařízení

Emulace síťových prvků je jeden z nejdůležitějších úkolů GNS3. Emulace všech existujících na trhu zařízení není možná vzhledem k vysokému počtu výrobců a rozmanitosti použitých technologických řešení. Na trhu však existuje několik „velkých hráčů“ – společností, které dodávají své výrobky široké sféře zákazníku a které přímo a nepřímo ovlivňují vývoj síťových standardů.

K takovým hráčům patří společnosti Cisco a Juniper. Jejich zařízení se řadí mezi nejčastěji používaná v rámcích podnikových sítí. GNS3 podporuje emulaci některých zařízení obou společností, dělá to pro každou společnost jiným způsobem.

5.1.1 Zařízení Cisco

Emulace zařízení od společnosti Cisco je prováděna s využitím softwaru Dynamips. Tento software umožňuje emulovat zařízení z následujících rodin: *1700*, *2600*, *3600*, *3700*, *7200*. Emulace každé rodiny není úplně dokonalá a v každé z nich se může za nepředvidatelných okolností dojít k nečekané chybě. Stabilita emulace spočívá i na verzi IOS, jako nejlepší z tohoto hlediska se ukázaly verze *c36xx*, *c37xx* a *c7200* (s výjimkou c7200p).[14]

Dynamips emuluje jenom fyzickou část zařízení. Ke své činnosti potřebuje ještě operační systém Cisco IOS. Z licenčních důvodu tento software není součásti instalace Dynamips ani GNS3 a je vlastnictvím společnosti Cisco. Doporučeným způsobem je kopírování systémového softwaru ze zařízení Cisco, které uživatel vlastní.[14]

Nedostatkem Dynamips je to, že ne všechny verze IOS jsou s ním kompatibilní. To může vyvolat situaci, kdy některá funkcionalita je dodávána jen s některými verzemi IOS, které nemusí být s GNS3 slučitelné. Řešením tohoto problému je použití jiného prostředku pro emulaci: Cisco IOU.

Cisco IOU je speciální verze systému IOS, která může byt spuštěna na UNIXových systémech. Hlavní výhodou IOU je to, že tento systém vznikal za účelem emulace funkcionality původního systému na rozdíl od Dynamips, který se zaměřuje na emulaci hardwaru. Funkce L3 switchů jsou IOU také podporovaný, Dynamips toto kvůli složitosti konstrukce L3 switchů neumí.[14]

5.1.2 Zařízení Jupiter

Zařízení od společností Jupiter můžou být emulovány pomocí speciální verze operačního systému Juniper Olive. Tato verze standardního pro všechna zařízení operačního systému Junos OS může být spuštěna na strojích, které jsou kompatibilní se systémem FreeBSD. Takové prostředí může být poskytnuto pomocí emulátorů QEMU a VirtualBox. Oba tyto prostředky jsou kompatibilní s GNS3 a kvůli tomu umožňují v případě potřeby začlenit do sítě i zařízení Jupiter.[14]

5.2 Pracovní stanice a servery

Nejjednodušším způsobem emulace uživatelských počítačů je použití vestavěného emulátoru VPCS. Tento prostředek je velmi šetrný na systémové prostředky a nemusí být před použitím dodatečně konfigurován. Možnosti VPCS jsou ale docela omezené. Software poskytuje podporu jen základních příkazů, které můžou být použity k testování konektivity. VPCS je vhodným řešením pro emulaci uživatelských pracovních stanic.

Jiným prostředkem může sloužit nástroj VirtualBox. Tento nástroj umožňuje rozmístit na emulovaném stroji distribuci většiny široko používaných operačních systému např. Windows, Linux, FreeBSD atd. Na systém lze dále nainstalovat kterýkoliv odpovídající software.

Nedostatkem VirtualBox je jeho náročnost na prostředky. Každá instance VirtualBox je plnohodnotný stroj, který aktivně zatěžuje paměť a procesor hostitele. Z toho důvodu VirtualBox je dobrým řešením pro emulaci strojů odpovědných za provoz různých síťových služeb. Není však doporučené používat VirtualBox pro emulaci všech koncových uzlu v síti.[14]

Z webové stránky GNS3 lze také stáhnout i některé předem konfigurované virtuální stroje. Výhodou jejich použití je jejich optimalizace pro práci v prostředí GNS3 a zjednodušení nastavení. Je nutno zmínit, že emulace je v tomto případě prováděna zase pomocí VirtualBox, což znamená, že i navzdory optimalizaci je tento způsob emulace poměrně náročný na prostředky. V případě emulace serverů a jiných významných uzlu sítě je však použití předem konfigurovaných strojů rozumnou alternativou ruční instalaci a konfiguraci potřebného systému.

6 ZAVEDENÍ MONITORINGU VE FIREMNÍM PROSTŘEDÍ

Tato kapitola popisuje praktické zavedení monitoringu v prostředí fiktivní firmy. Prostředí je emulováno pomocí nástroje GNS3. Firma poskytuje interní a externí síťové služby, které jsou typické pro reálné podniky. K takovým patří například HTTP server, který poskytuje přístup na webové stránky podniku, databázový server, který slouží úložištěm informací atd.

Jako nástroj pro zajištění monitoringu byl zvolen Nagios Core zejména kvůli velkému počtu pluginů, které jsou flexibilní a umožňují kontrolovat stavy široké škály různých zařízení.

6.1 Topologie průmyslové sítě

Celkovou síť podniku tvoří několik podsítí. Dvě podsítě jsou vyhrazeny pro koncové uživatelské pracovní stanice. IP adresy uzlů v těchto dvou sítích jsou přidělovány automaticky pomocí služby DHCP. DHCP služba běží na směrovačích, které sousedí se zmíněnými sítěmi. Všechny podsíti jsou propojeny mezi sebou s využitím směrovacího protokolu OSPF.

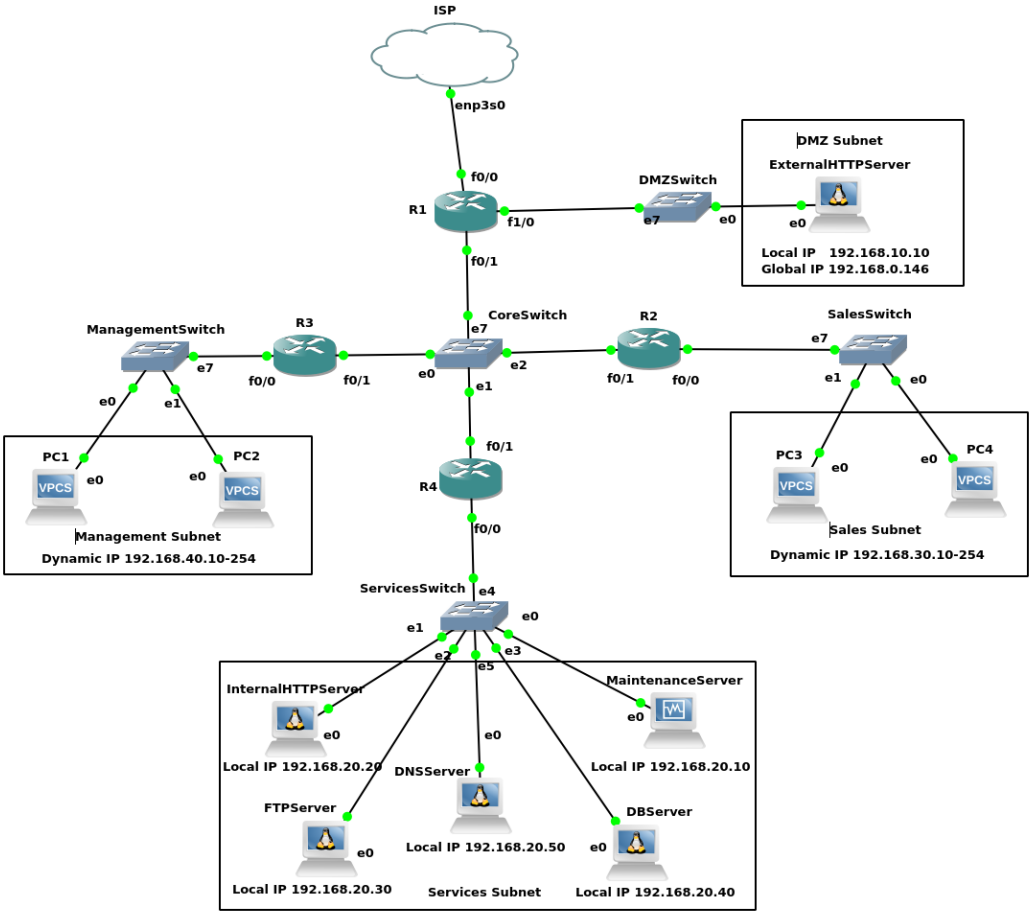
Nejdůležitější z hlediska poskytování služeb podsíti je *192.168.20.0/24*. Právě tam je rozmístěna většina strojů poskytujících služby. Konkretně se jedná o:

- *192.168.20.10* – server Nagios Core a server syslog,
- *192.168.20.20* – HTTP server pro interní aplikaci podniku,
- *192.168.20.30* – FTP server pro sdílení souborů mezi zaměstnanci firmy,
- *192.168.20.40* – databázový server, poskytuje přístup k uloženým datům prostřednictvím webových aplikací,
- *192.168.20.50* – DNS server, jehož úkolem je zjednodušení práce s interními služby pomocí přiřazení jim snadněji zapamatovatelných doménových jmen.

Všechny stroje v této podsíti jsou přístupné pouze zevnitř podnikové sítě a jsou pro zbytek světa neviditelné.

Pro rozmístění síťových služeb, které by měly byt přístupné uživatelům, slouží speciální podsíť *192.168.10.0/24*. Každému uzlu v této podsíti musí být přiřazena svá unikátní veřejná IP adresa. Firewall na směrovači *R1* umožňuje průchod paketů zvenku pouze v tom

případě, pokud cílem je stroj v této podsíti. V rámci emulované sítě je tady rozmístěn stroj *192.168.10.10*, který má přiřazenou „veřejnou“ IP adresu *192.168.0.146*. Emulace této adresy je zajištěna pomocí přidání záznamu do routovací tabulky směrovače, který řídí přenos dat pro reálný stroj, na kterém probíhá emulace.



Obrázek 4: Topologie podnikové sítě

6.2 Pracovní prostředí monitoringu

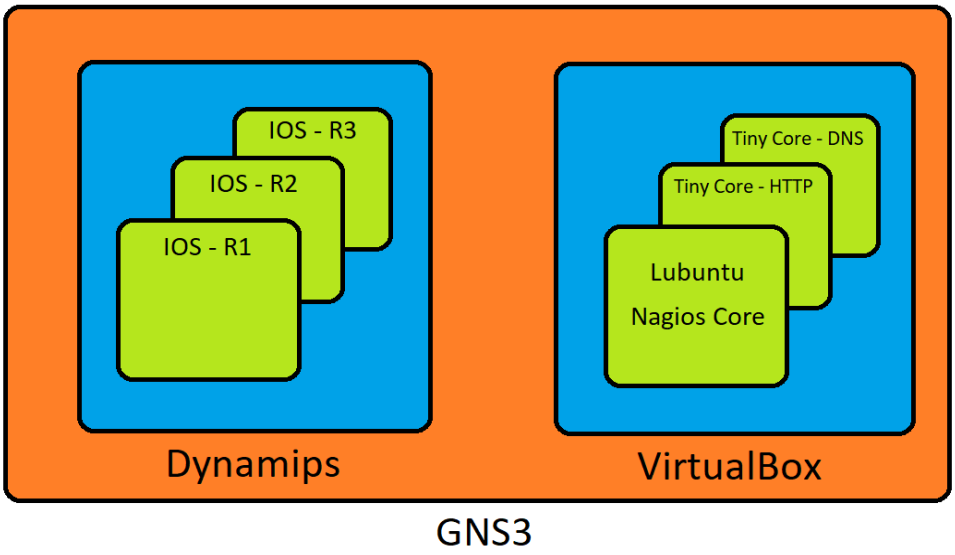
Pracovní prostředí monitoringu se sestává z několika rozdílných prvků, které komunikují mezi sebou pro zajištění emulace běhu síťových služeb a monitorovacího nástroje.

Prvním a asi nejvýznamnějším prvkem je samotný emulátor GNS3. Emulátor poskytuje prostředí, které obsahuje v sobě ostatní komponenty, a emuluje pasivní síťové prvky. Ostatní komponenty mohou díky tomu navázat spojení jak mezi sebou, tak i s okolním světem.

Software směrovačů je emulován pomocí emulátoru Dynamips. Tento nástroj může být provozován i zcela samostatně a není součásti GNS3. Je ale obvykle dodáván společně s GNS3, pokud instalace byla provedena pomocí balíčkovacího systému nebo instalačního programu. Dynamips umožňuje zprovoznit Cisco IOS a emulovat tím chování směrovačů v síti.

Dalším významným prvkem je virtualizační nástroj Oracle VM VirtualBox. Tento nástroj musí byt nainstalován samostatně. GNS3 umí s tímto nástrojem komunikovat. VirtualBox umožňuje emulovat chování téměř jakéhokoliv operačního systému. Kromě toho na oficiálních stránkách GNS3 lze stáhnout předpřipravená zařízení, která byla speciálně upraveny tak, aby jejich kompatibilita s GNS3 byla co nejvyšší.

Na instancích VirtualBox byly pak rozmístěny operační systémy a příslušný software. O zvolených možnostech pojednává následující podkapitola.



Obrázek 5: Pracovní prostředí monitoringu

6.3 Simulační prostředky

Směrovače byly emulovány s využitím softwaru Dynamips, který je součásti GNS3. Byla zvolena verze IOS *c3745*, zejména díky malému riziku vzniku u ní problémů s kompatibilností. Emulace switchů je ale dlouhodobým problémem GNS3. Důvodem k tomu je použití ve switchích speciálních zákaznických integrovaných obvodů, které jsou špatně emulovatelné. Jelikož emulace sítě nevyužívá pokročilých funkcí switchů (VLAN atd.),

jako prostředek pro emulaci switchů byly zvoleny Ethernet switche GNS3. Tyto switche poskytují jen základní funkcionalitu, která je ale pro naše účely zcela dostatečná.

Pro emulaci koncových uživatelských stanic je použit nástroj VPCS, který je součástí GNS3. Pracovní stanice nejčastěji poskytují přístup k síťovým službám pomocí webového prohlížeče, který ale vyžaduje i grafické prostředí. Emulace několika takových stanic by byla příliš náročná na systémové prostředky a u pracovních stanic nás jako správce zajímá především to, jestli konkrétní stroj může navázat spojení se serverem. K takovým účelům nám bohatě vystačí VPCS, který navíc umožňuje ušetřit prostředky pro jiné zajímavější stroje.

K takovým patří například stroje v podsíti *192.168.20.0/24*. Každý stroj v této podsíti již poskytuje nějakou službu. Pro jejich emulaci je využita distribuce Tiny Core Linux. Tato minimalistická distribuce poskytuje minimální sadu nástrojů, mezi které patří ale i balíčkovací systém. Pomocí něj je možné nainstalovat software, který se bude starat o poskytování příslušných služeb. Všechny uvedené dál balíčky jsou přístupné pomocí nástroje *tce-load* a instalace balíčku probíha zadáním příkazu „tce-load -wi <název balíčků>.tcz“.

Pro zprovoznění dvou lokálních HTTP serverů (*192.168.20.20* a *192.168.10.10*) byl použit balíček *busybox-httpd*. Výchozí nastavení Tiny Core způsobuje to, že drtivá většina složek a souborů není ukládána a restart stroje způsobí jejich ztrátu. Týká se to i složky */usr/local/httpd/bin*, kde jsou obvykle rozmístěny webové dokumenty HTTP serveru. Aby se tomuto problému předešlo, je zapotřebí vytvořit novou složku (například */mnt/sda1/wwwsite/*), která nebude vymazaná po vypnutí počítače. Pak je zapotřebí při každém startu zkopírovat obsah této složky do složky */usr/local/httpd/bin*.

Na strojích, které plní funkce HTTP serverů, byly také zprovozněny SFTP servery, které umožňují vzdálený přenos souborů na tyto stroje. Výchozí instalace Tiny Core již zahrnuje prostředky pro spuštění TFTP serveru. Pro bezpečnější interakci s TFTP serverem přes protokol SSH je využit balíček *openssh*. Po stažení veškerých potřebných balíčků je zapotřebí nastartovat všechny požadované služby. Aby tyto služby byly spuštěny automaticky při startu počítače, je třeba přidat do souboru */opt/bootlocal.sh* (příkazy v tomto souboru budou vykonány automaticky po restartu) následující řádky:

```
cp -R /mnt/sda1/wwwsite/cgi-bin /usr/local/httpd/bin/
cd /usr/local/httpd/bin/
sudo ./busybox httpd -p 80 -h /usr/local/httpd/bin/
```

```
cd /usr/local/etc/init.d/
./openssh start
cd /etc/init.d/services/
./tftpd start
```

Reálné webové aplikace obvykle spoléhají na databázi, kde jsou uložena data, přístup ke kterým je pak poskytován prostřednictvím samotné aplikace. Aplikace jsou vyvíjeny v jazycích, které mají prostředky pro interakci s databázemi (např. Java nebo PHP). Tato interakce je v rámci emulované sítě imitována pomocí CGI skriptu, který se nejdřív obrací na databázi s dotazem a pak vrací výsledek dotazu jako součást webového dokumentu. Aby zmíněný CGI skript fungoval, je nezbytné nainstalovat na HTTP servery i klientskou aplikaci pro přístup k DB. V práci pro tyto účely byl použit balíček *mariadb-client*.

Samotná databázová aplikace byla rozmístěna na uzlu *192.168.20.40* a k její zprovoznění byl využit balíček *mariadb*. V rámci databáze byla vytvořena testovací tabulka a několik uživatelských účtu pro přístup z jiných uzlů (HTTP servery a monitorovací stanice). Stejně jako u HTTP serverů (i všech ostatních serverů, které běží na Tiny Core) je zapotřebí přidat cesty ke složkám, ve kterých MariaDB uchovává své soubory, do souboru */opt/.filetool.lst*. V našem případě jedná se zejména o složku */usr/local/mysql/data*. Aby se projevili změny v samotném souboru */opt/.filetool.lst* je také nezbytné provést příkaz „filetool.sh -b“. Příkaz „/usr/local/mysql/bin/mysqld_safe &“, který spouští databázovou aplikaci byl přidán do souboru */opt/bootlocal.sh* pro automatický start DB.

Pro sdílení libovolných souborů v rámci sítě byl zprovozněn i běžný FTP server přístupný na adrese *192.168.20.30*. Funkcionalita FTP serveru je obsažena v balíčku *bftpd*. Příkaz pro spuštěni služby je „bftpd -d -c <cesta ke konfiguračnímu souboru>“. Cesta ke konfiguračnímu souboru je */mnt/sda1/bftpd.conf*. Kontrola správnosti provozu všech FTP služeb (včetně SFTP na HTTP serverech) byla provedena na stroje *192.168.20.10* s využitím aplikace *FileZilla*.

V síti byl také zprovozněn DNS server, který má usnadnit práci uživatelům. Jeho úkolem je převod doménových jmen intranetových zdrojů na IP adresy. V případě, že záznam o jménu nebude na lokálním DNS serveru nalezen, bude dotaz přesměrován na veřejný server (obvykle je k tomu použit server poskytovatele připojení, v rámci emulované sítě je použít veřejný server Google – *8.8.8.8*). Pro zprovoznění DNS serveru byl použit balíček *bind*. Spuštění služby probíhá s využitím příkazu „/usr/local/sbin/named -

c <cesta ke konfiguračnímu souboru>“. Kromě toho cesta k konfiguračnímu souboru (v našem případě se jedná o soubor */usr/local/etc/bind/named.conf*) byla přidána také do souboru */opt/.filetool.lst*, aby se předešlo jeho vymazání při vypnutí stroje.

Jediný stroj, který nepoužívá Tiny Core Linux je stroj *192.168.20.10*. Na tomto stroji běží software Nagios Core, který mimo jiné poskytuje webové rozhraní pro zobrazení aktuálního stavu systému. Sestavení Nagios Core ze zdrojových kódů vyžaduje přítomnost dodatečných knihoven, které jsou součásti větších distribucí Linux ale nemusí být dostupné jako balíčky Tiny Core. Kromě toho, je také nutné mít v systému alespoň jeden stroj s grafickým prostředím, kde by bylo možné spustit webový prohlížeč. Z těchto důvodu je na stroji *192.168.20.10* rozmístěna distribuce Lubuntu, který je odlehčenou verzi populární distribuce Ubuntu.

Pro instalaci Nagios bylo zapotřebí nejdříve zkompilovat zdrojové kódy aplikace a pluginů. Podrobný návod lze nalézt na oficiální webové stránce projektu. Součásti Nagios je i webové rozhraní, pomocí kterého lze vizualizovat výsledky monitoringu. Aby toto rozhraní fungovalo, je třeba nejdříve nainstalovat HTTP server (v tomto případě byl použít *apache2*) a interpret jazyka PHP.

6.4 Implementace

Nagios Core uchovává svá nastavení v podobě konfiguračních souborů. To znamená, že veškerá konfigurace spočívá v editaci těchto souborů, kde jsou umístěny definice objektů. Objektem může být host, běžící služba, skupiny hostů a služeb, kontaktní údaje pro zaslaní upozornění, příkazy pro vykonaní kontrol atd. Hlavním konfiguračním souborem je *etc/nagios.cfg* (tato a dále uvedené cesty k souborům jsou relativní). Nejdůležitějšími proměnnými v tomto souboru jsou *cfg_file* a *cfg_dir*. Díky ním může Nagios nalézt cesty k souborům s definicemi objektů. Definice objektů může být umístěna kdekoliv, pokud dotyčný soubor nebo složka je uvedena v hlavním konfiguračním souboru. Výchozí nastavení Nagios zahrnuje složku *etc/objects*, kde jsou uloženy šablony. Pro oddělení konfigurace podnikových objektů od šablon jsou vytvořeny složky *etc/objects/hosts*, *etc/objects/services*, *etc/objects/contacts/contacts* a *etc/objects/timeperiods*. Cesta ke složkám byla vložena do souboru *etc/nagios.cfg*.

Jak již bylo zmíněno dříve, všechny kontroly jsou prováděny pomocí externích pluginů, které nejsou součásti výchozí instalace. Nicméně existuje oficiální balíček pluginu, který

zahrnuje více než 50 různých nástrojů na kontrolu běžně používaných služeb. Každý plugin je v podstatě malý skript, který lze spustit i ručně z příkazového řádku. Pro umístění pluginu slouží složka *libexec*.

Propojit Nagios a externí plugin lze pomocí definice příkazu, která může vypadat například takto:

```
define command {
    command_name check_rsync
    command_line $USER1$/check_rsync -H $HOSTADDRESS$
}
```

V tomto případě *command_name* je jménem příkazu, na které se pak bude odkazovat a *command_line* je jeho obsahem. Elementy *$USER1$* a *$HOSTADDRESS$* jsou makra, která jsou rozvinuta předtím, než kontrola bude vykonána. Díky tomu lze používat jednu definici pro kontrolu libovolného čísla hostů a služeb. Z *$USER1$* se tak stane *libexec* a *$HOSTADDRESS$* bude odkazovat na IP adresu požadovaného zařízení, například *192.168.20.20*. Po rozvinutí maker bude příkaz vypadat následovně:

```
command_line libexec/check_rsync -H 192.168.20.20
```

Příkaz v této formě lze spustit již i z příkazového řádku a vyzkoušet tak jeho funkčnost nebo získat nápovědu.

Po editaci konfiguračních souboru je vhodné nejdřív ověřit jejich správnost pomocí následujícího příkazu:

```
<cesta ke spustitelnému souboru Nagios> -v \
<cesta k hlavnímu konfiguračnímu souboru>
```

Pokud tento příkaz nehlásí chyby, je možné službu bezpečně restartovat. Všechny změny se projeví samy.

6.4.1 Sledování stavu hostů a služeb

Pro monitoring přístupnosti hostu stačí jen zadat definici příslušného hostu do konfigurace. Definice hostu může vypadat takto:

```
define host {
```

```
    host_name             databaseHost
    alias                 Database machine
    address               192.168.20.40
    max_check_attempts    3
    check_period          24x7
    check_command         check-host-alive
    contacts              admin
    contact_groups        admins
    notification_interval 60
    notification_period   24x7
}
```

Definice se skládá z následujících části:

- *host_name* – jméno, na které se pak odkazuje,
- *alias* – jméno, které se zobrazí ve webovém rozhraní a notifikacích,
- *adress* – IP adresa nebo doménové jméno stroje,
- *max_check_attempts* – kolikrát se provede kontrola předtím, než změna stavu bude zafixována,
- *check_period* – uvádí časový úsek, ve kterém je host průběžně kontrolován,
- *check_command* – odkaz na dříve definovaný příkaz, který bude proveden pro zjištění stavu uzlu,
- *contacts* – odkaz na kontaktní údaje, kam bude zasláno upozornění (o notifikacích, kontaktních údajech a časových úsecích vizte dále),
- *contact_groups* – odkaz na skupinu kontaktů, které budou upozorněny na změnu stavu,
- *notification_interval* – kolik času musí uplynout od zaslání předchozí notifikace, aby byla zaslaná nová (s podmínkou toho, že stav hostu se nezmění),
- *notification_period* – uvádí časový úsek, během kterého může být odeslána notifikace.

Přidaní službu probíhá stejným způsobem a většina nastavení je úplné stejná s nastaveními hostů. Důležitou položkou je *host_name*, která vytváří asociaci mezi službou a dříve definovaným hostem. Definice kontroly databázové služby pak vypadá následovně:

```
define service {
```

```
    host_name             databaseHost
    service_description   MySQLDatabase
    check_command         check_mysqld
    max_check_attempts    3
    check_interval        5
    retry_interval        1
    check_period          24x7
    contacts              admin
    contact_groups        admins
    notification_interval 60
    notification_period   24x7
}
```

Tato definice obsahuje několik nových položek, které nebyly ještě uvedeny. K takovým patří:

- *service_description* – popis služby,
- *check_interval* – jak často se provádí kontrola, pokud výsledkem minulé kontroly byl stav *OK*,
- *retry_interval* – jak často se provádí kontrola, pokud výsledkem minulé kontroly nebyl stav *OK*.

Je nutné uvést, že proměnné *check_interval* a *retry_interval* lze použit i v definici hostů.

Za účelem usnadnění orientace lze seřadit hosty a služby do skupin, např skupina „DMZServices“ může zahrnovat ty podnikové služby, které jsou přístupné zákazníkům zvenčí. Její definice vypadá takto:

```
define servicegroup {
    servicegroup_name dmzServices
    alias             Services that are located in DMZ
    members           externalWebServerHost,HTTP
}
```

a skupina hostů, která zahrnuje všechny směrovače je definována jako:

```
define hostgroup {
    hostgroup_name    routers
```

```
    alias               Enterprise routers
    members             salesRouter, managementRouter,
                        DMZRouter, servicesRouter
}
```

Uváděné údaje jsou podobné a skládají se z:

- *servicegroup_name/hostgroup_name* – jméno příslušné skupiny,
- *alias* – podrobnější popis,
- *members* – výčet prvků skupiny, příslušnost ke skupině může být zadána i přímo v konfiguraci hostu nebo služby jako obsah proměnné *hostgroups* u hostů a *servicegroups* u služeb.

Pro kontrolu stavů hostů a služeb byly vytvořeny příslušné definice objektů (podrobné definice jsou uvedené v příloze 1). Nejdůležitější části definice je příkaz, který odkazuje na plugin. Stav všech hostů je kontrolován pomocí jednoduchého pluginu *check-host-alive*, který zasílá ICMP echo zprávu a čeká na odpověď. Pokud odpověď dorazí, bude host považován za přístupný.

Definice služeb je poněkud složitější záležitost. Pro každý druh služby (HTTP, FTP, DNS atd.) musí být použít jiný plugin, který bude testovat hodnoty spojené s konkrétní službou. Pro kontrolu HTTP serverů byl použit plugin *check-http*, který zasílá dotaz shodný s tím, co posílá webový prohlížeč při pokusu o načtení stránky. Tento plugin očekává obdržet HTTP odpověď s kódem 200, tento výsledek bude považován za úspěšný.

Stav SQL databáze je kontrolován s využitím pluginu *check-mysqld*. Tento plugin dokáže získat podrobné informace o stavu databáze. V rámci emulace nás zajímá především to, zda databázová aplikace běží. Proto nejvýznamnějším údajem je přítomnost či absence odpovědi, hodnoty jednotlivých parametrů nejsou pro nás v tuto chvíli podstatné.

Kontrola stavu služby DNS je prováděna s využitím pluginu *check-dns*. V definici příkazu je třeba uvést doménové jméno a výslednou IP adresu. Pokud IP adresa v odpovědi je shodná s očekávanou hodnotou, bude výsledek kontroly úspěšný.

O zjištění stavu FTP serveru se stará plugin *check-ftp*. Tento plugin se snaží navázat spojení s FTP serverem. Při úspěšném navázaní spojení bude vracen kladný výsledek kontroly.

Kontrola aktivních síťových prvků probíhá s využitím protokolu SNMP. Plugin *check-snmp* dovoluje zaslat SNMP dotaz na uzel a porovnat odpověď s očekávanou hodnotou.

V rámci této práce se na každém směrovači zkoumá hodnota Uptime, která vyjadřuje počet sekund od startu směrovače. Tím lze ověřit, jestli směrovač je zapnutý a schopen odpovídat na SNMP dotazy. Kromě toho je kontrolován i stav vybraných kritických Ethernet rozhraní. Switche nejsou kontrolovány, protože jejich omezená emulace nedovoluje jim odpovídat na SNMP dotazy.

6.4.2 Notifikace

Nagios umí nejen průběžně kontrolovat stav prvků, ale také i zasílat upozornění v případě toho, že host nebo služba změní svůj stav. Prvním krokem v nastaveni notifikací je definice kontaktních údajů, které budou použity pro zaslaní upozornění. Definice objektu kontaktních údajů vypadá takto:

```
define contact {
    contact_name                    admin
    alias                           Administrator
    email                           example@mail.com
    host_notification_commands      notify-host-by-email
    host_notification_options       d,u,r
    host_notification_period        24x7
    service_notification_commands   notify-service-by-email
    service_notification_options    w,u,c,r
    service_notification_period     24x7
}
```

a zahrnuje následující nastavení:

- *contact_name* – vnitřní jméno kontaktu,
- *alias* – podrobnější popis,
- *email* – adresa pro zaslaní mailů,
- *host_notification_commands* – definuje jakým způsobem bude probíhat notifikace o změně stavu hostu,
- *host_notification_options* – notifikace bude probíhat, pokud host je v jednom z uvedených stavů (UP, DOWN, RECOVERY),
- *host_notification_period* – uvádí časový úsek, během kterého může být odeslána notifikace o změně stavu hostů,

- *service_notification_commands* – definuje jakým způsobem bude probíhat notifikace o změně stavu služby,
- *service_notification_options* – notifikace bude probíhat, pokud služba je v jednom z uvedených stavů (OK, WARNING, UNKNOWN, CRITICAL, RECOVERY),
- *service_notification_period* – uvádí časový úsek, během kterého může být odeslána notifikace o změně stavu služby.

Definované takovým způsobem kontakty pak můžou být odkazovány z konfiguračních souboru hostů a služeb. Nejčastějším způsobem notifikace je zaslaní e-mailů. Pro odeslání upozornění Nagios využívá služeb externích aplikací, v případě e-mailů se jedná o soubor */bin/mail*.

Pro určení časových úseku, během kterých by měl být technický personál upozorňován na změny stavu, existuje speciální druh objektu – timeperiod. Na tento objekt je možné odkazovat nejen v definicích kontaktů, ale také i v definicích hostů a služeb. Tím lze specifikovat, kdy budou probíhat kontroly příslušných objektů. Definice časového úseku vypadá následovně:

```
define timeperiod {
    timeperiod_name weekdays
    alias Weekdays
    monday 00:00-24:00
    tuesday 00:00-24:00
    wednesday 00:00-24:00
    thursday 00:00-24:00
    friday 00:00-24:00
}
```

Přidání definicí kontaktních údajů nevyžaduje změny v definicích hostů a služeb. Díky tomu lze relativně jednoduše měnit tyto údaje, aniž by byla ovlivněna logika provádění samotných kontrol.

Zaslání upozornění pak probíhá úplně automaticky na základě uvedených v definicích podmínek. Příklad takového upozornění v případě selhání služby a její následného návratu do OK stavu je ukázán v příloze 2.

6.4.3 Vizualizace výsledků monitoringu

Pro zobrazení výsledků monitoringu poskytuje Nagios webové rozhraní. Rozhraní je přístupné na adrese *localhost/nagios*. Pro úspěšné přihlášení je zapotřebí uvést údaje uživatele, který byl vytvořen během instalace Nagios.

Webové prostředí umožňuje provádět rutinní operace bez zásahů do konfiguračních souborů. K takovým operacím patří:

- dočasné přerušování provádění kontrol,
- změna rozvrhu provádění kontrol,
- vymezování časových období, kdy nebudou zasílána upozornění,
- atd.

Kromě toho webové rozhraní umožňuje sledovat stav všech kontrolovaných prvků nebo stav členů některé ze skupin hostů a služeb.

Ukázka práce s webovým rozhraním Nagios je předvedena v příloze 3.

Pro ověření funkčnosti monitorovacího systému bylo zmodelováno několik situací, jejichž pravděpodobnost výskytu během reálného provozu je relativně vysoká. Ve všech případech byla provedena kontrola výstupních informací webového rozhraní Nagios.

- Situace A. Normální stav sítě.

  Situace A slouží referenčním bodem a reflektuje normální stav sítě, kdy každý prvek je úspěšně zprovozněn.

Service Status Details For All Hosts

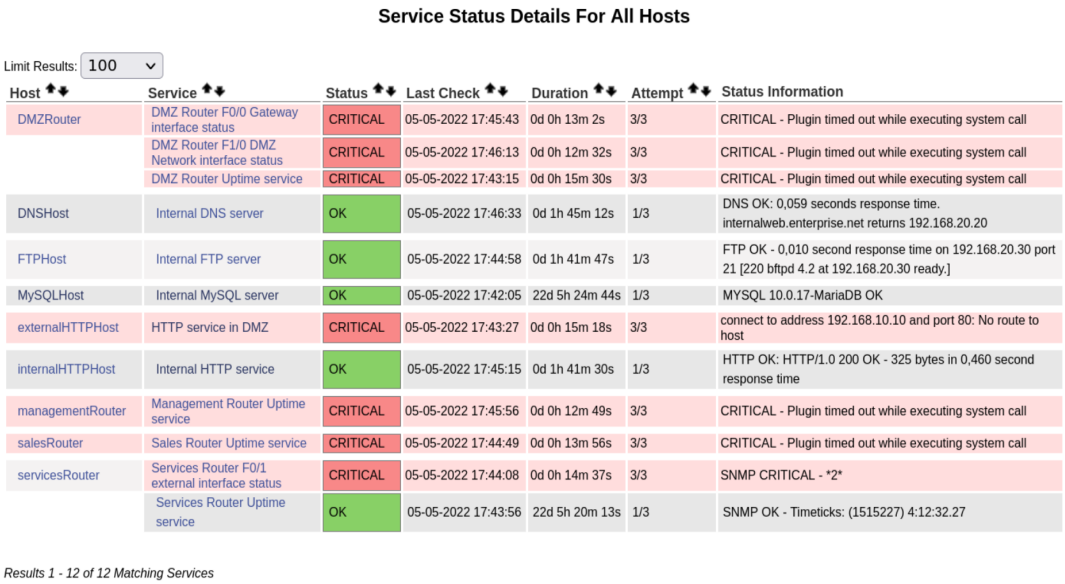
Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
DMZRouter	DMZ Router F0/0 Gateway interface status	OK	05-05-2022 17:13:43	0d 0h 7m 58s	1/3	SNMP OK - 1
	DMZ Router F1/0 DMZ Network interface status	OK	05-05-2022 17:14:13	0d 0h 17m 28s	1/3	SNMP OK - 1
	DMZ Router Uptime service	OK	05-05-2022 17:16:15	22d 5h 1m 13s	1/3	SNMP OK - Timeticks: (1349263) 3:44:52.63
DNSHost	Internal DNS server	OK	05-05-2022 17:11:33	0d 1h 15m 8s	1/3	DNS OK: 0,058 seconds response time. internalweb.enterprise.net returns 192.168.20.20
FTPHost	Internal FTP server	OK	05-05-2022 17:14:58	0d 1h 11m 43s	1/3	FTP OK - 0,010 second response time on 192.168.20.30 port 21 [220 bftpd 4.2 at 192.168.20.30 ready.]
MySQLHost	Internal MySQL server	OK	05-05-2022 17:12:05	22d 4h 54m 40s	1/3	MYSQL 10.0.17-MariaDB OK
externalHTTPHost	HTTP service in DMZ	OK	05-05-2022 17:16:27	0d 1h 15m 14s	1/3	HTTP OK: HTTP/1.0 200 OK - 324 bytes in 0,659 second response time
internalHTTPHost	Internal HTTP service	OK	05-05-2022 17:15:15	0d 1h 11m 26s	1/3	HTTP OK: HTTP/1.0 200 OK - 325 bytes in 0,358 second response time
managementRouter	Management Router Uptime service	OK	05-05-2022 17:13:56	10d 1h 38m 22s	1/3	SNMP OK - Timeticks: (1335233) 3:42:32.33
salesRouter	Sales Router Uptime service	OK	05-05-2022 17:12:49	10d 1h 38m 11s	1/3	SNMP OK - Timeticks: (1328546) 3:41:25.46
servicesRouter	Services Router F0/1 external interface status	OK	05-05-2022 17:12:08	0d 0h 19m 33s	1/3	SNMP OK - 1
	Services Router Uptime service	OK	05-05-2022 17:13:56	22d 4h 50m 9s	1/3	SNMP OK - Timeticks: (1335224) 3:42:32.24

Obrázek 6: Přehled stavu hostů a služeb ve webovém prostředí Nagios

- Situace B. Nedostupné interní služby.

  Zaměstnanci firmy nemají přístup k interním službám sítě. Vnější síťové destinace jsou dosažitelné při adresaci pomocí IP adres, překlad doménových jmen neprobíhá. Zkoumaní stavu systému ve webovém prostředí umožňuje nalézt zdroj problému. V tomto případě viníkem nastalé situace je host *serviceRouter*. Samotný směrovač je dosažitelný, ale kontrola stavu síťového rozhraní *F0/1* vrací neočekávanou hodnotu. Stává se to v důsledku toho, že zmíněné rozhraní bylo vypnuté a zamezovalo tak datovému přenosu mezi stroji zaměstnanců a částí síti, kde je umístěna většina služeb včetně monitorovací stanice. Host s DNS službou je úplně funkční prvek z pohledu monitorovací stanice, i když není dosažitelný pro stroje v jiných částech sítě.



Service Status Details For All Hosts

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
DMZRouter	DMZ Router F0/0 Gateway interface status	CRITICAL	05-05-2022 17:45:43	0d 0h 13m 2s	3/3	CRITICAL - Plugin timed out while executing system call
	DMZ Router F1/0 DMZ Network interface status	CRITICAL	05-05-2022 17:46:13	0d 0h 12m 32s	3/3	CRITICAL - Plugin timed out while executing system call
	DMZ Router Uptime service	CRITICAL	05-05-2022 17:43:15	0d 0h 15m 30s	3/3	CRITICAL - Plugin timed out while executing system call
DNSHost	Internal DNS server	OK	05-05-2022 17:46:33	0d 1h 45m 12s	1/3	DNS OK: 0,059 seconds response time. internalweb.enterprise.net returns 192.168.20.20
FTPHost	Internal FTP server	OK	05-05-2022 17:44:58	0d 1h 41m 47s	1/3	FTP OK - 0,010 second response time on 192.168.20.30 port 21 [220 bftpd 4.2 at 192.168.20.30 ready.]
MySQLHost	Internal MySQL server	OK	05-05-2022 17:42:05	22d 5h 24m 44s	1/3	MYSQL 10.0.17-MariaDB OK
externalHTTPHost	HTTP service in DMZ	CRITICAL	05-05-2022 17:43:27	0d 0h 15m 18s	3/3	connect to address 192.168.10.10 and port 80: No route to host
internalHTTPHost	Internal HTTP service	OK	05-05-2022 17:45:15	0d 1h 41m 30s	1/3	HTTP OK: HTTP/1.0 200 OK - 325 bytes in 0,460 second response time
managementRouter	Management Router Uptime service	CRITICAL	05-05-2022 17:45:56	0d 0h 12m 49s	3/3	CRITICAL - Plugin timed out while executing system call
salesRouter	Sales Router Uptime service	CRITICAL	05-05-2022 17:44:49	0d 0h 13m 56s	3/3	CRITICAL - Plugin timed out while executing system call
servicesRouter	Services Router F0/1 external interface status	CRITICAL	05-05-2022 17:44:08	0d 0h 14m 37s	3/3	SNMP CRITICAL - *2*
	Services Router Uptime service	OK	05-05-2022 17:43:56	22d 5h 20m 13s	1/3	SNMP OK - Timeticks: (1515227) 4:12:32.27

Results 1 - 12 of 12 Matching Services

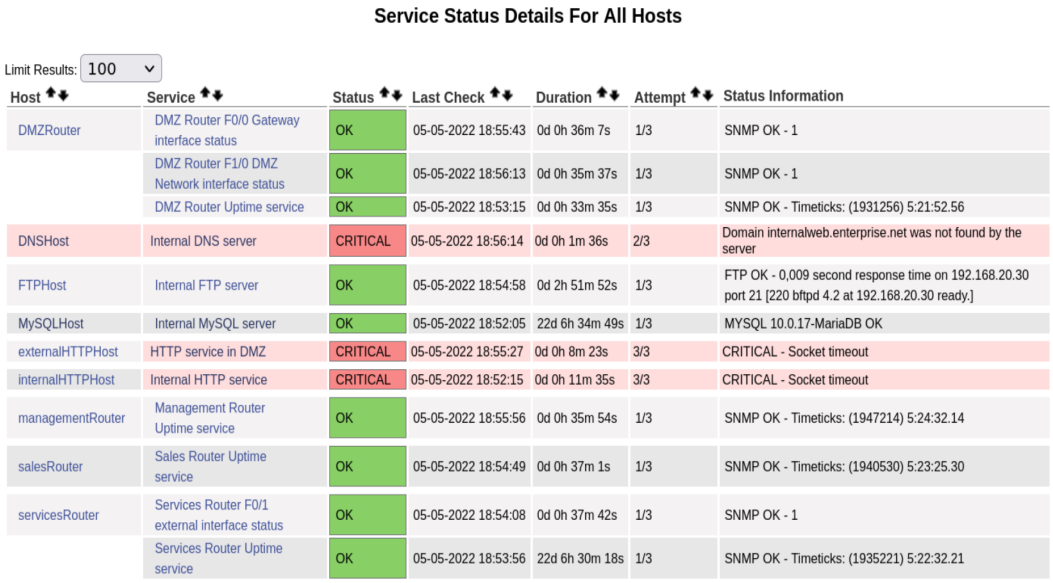
Obrázek 7: Vzhled okna stavu služeb v situaci B

- Situace C. Externí aplikace podniku není přístupná pro klienty.

  Klienti firmy nemají přístup k firemnímu webu. Kromě toho zaměstnanci firmy mají stejné problémy jako v situaci B. Můžou ale tentokrát navázat spojení v případě adresace pomocí IP adres.

  Z přehledu stavu služeb lze odvodit, že tentokrát prvkem zodpovědným za selhání je host *DNSHost*. Firemní HTTP servery v průběhu vyřízení uživatelského dotazu se obracejí na databázi s cílem získat potřebná data. V CGI skriptech je databáze odkazována pomocí její doménového jména. Aby HTTP server mohl zaslat dotaz

na databázi, musí být nejdřív zjištěna její IP adresa. V důsledku vypnutí DNS serveru tato informace je pro HTTP server nedostupná, což vede k tomu, že zpracování přicházejícího HTTP požadavku skončí neúspěšně. Na obrázku lze vidět, že stav služby je považován za kritický v důsledku vypršení časového limitu pro zaslaní HTTP odpovědi s kódem *200*. Hosty, na kterých běží HTTP servery jsou ale pořád považované za přístupné, protože výpadek DNS serveru žádným způsobem neovlivnil jejich schopnost odpovídat na Echo zprávy.



Service Status Details For All Hosts

Limit Results: 100

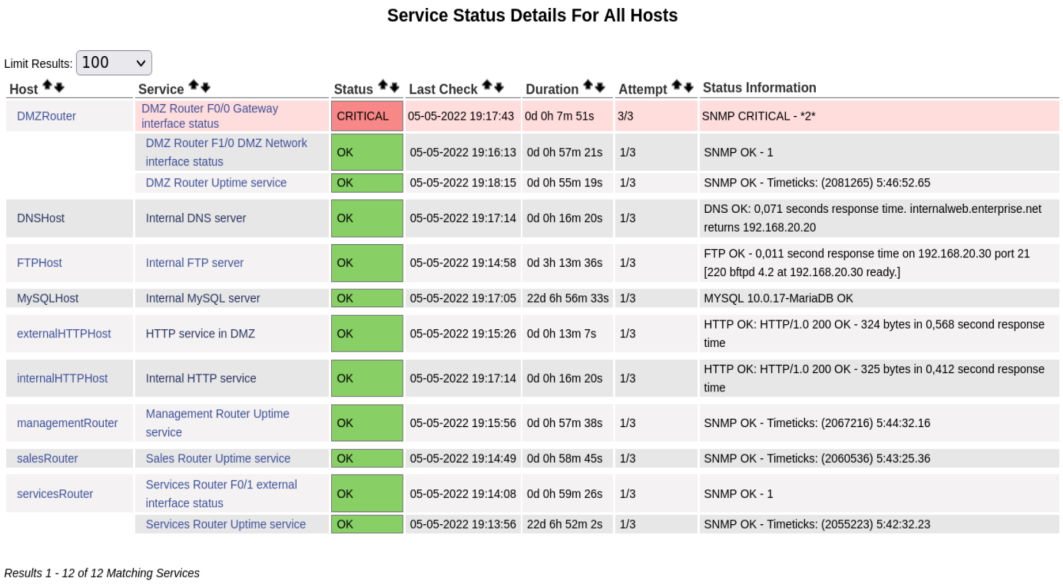
Host	Service	Status	Last Check	Duration	Attempt	Status Information
DMZRouter	DMZ Router F0/0 Gateway interface status	OK	05-05-2022 18:55:43	0d 0h 36m 7s	1/3	SNMP OK - 1
	DMZ Router F1/0 DMZ Network interface status	OK	05-05-2022 18:56:13	0d 0h 35m 37s	1/3	SNMP OK - 1
	DMZ Router Uptime service	OK	05-05-2022 18:53:15	0d 0h 33m 35s	1/3	SNMP OK - Timeticks: (1931256) 5:21:52.56
DNSHost	Internal DNS server	CRITICAL	05-05-2022 18:56:14	0d 0h 1m 36s	2/3	Domain internalweb.enterprise.net was not found by the server
FTPHost	Internal FTP server	OK	05-05-2022 18:54:58	0d 2h 51m 52s	1/3	FTP OK - 0,009 second response time on 192.168.20.30 port 21 [220 bftpd 4.2 at 192.168.20.30 ready.]
MySQLHost	Internal MySQL server	OK	05-05-2022 18:52:05	22d 6h 34m 49s	1/3	MYSQL 10.0.17-MariaDB OK
externalHTTPHost	HTTP service in DMZ	CRITICAL	05-05-2022 18:55:27	0d 0h 8m 23s	3/3	CRITICAL - Socket timeout
internalHTTPHost	Internal HTTP service	CRITICAL	05-05-2022 18:52:15	0d 0h 11m 35s	3/3	CRITICAL - Socket timeout
managementRouter	Management Router Uptime service	OK	05-05-2022 18:55:56	0d 0h 35m 54s	1/3	SNMP OK - Timeticks: (1947214) 5:24:32.14
salesRouter	Sales Router Uptime service	OK	05-05-2022 18:54:49	0d 0h 37m 1s	1/3	SNMP OK - Timeticks: (1940530) 5:23:25.30
servicesRouter	Services Router F0/1 external interface status	OK	05-05-2022 18:54:08	0d 0h 37m 42s	1/3	SNMP OK - 1
	Services Router Uptime service	OK	05-05-2022 18:53:56	22d 6h 30m 18s	1/3	SNMP OK - Timeticks: (1935221) 5:22:32.21

Obrázek 8: Vzhled okna stavu služeb v situaci C

- Situace D. Stroje ve firemní sítě nemůžou navázat spojení s jinými sítěmi.

  Zaměstnanci firmy nemůžou navázat spojení se strojí v jiných sítích. Klienti rovněž nemají přístup k externí aplikaci firmy.

  Webové rozhraní umožňuje rychle odhalit zdroj problému. V tomto případě jim je rozhraní *F0/0* na směrovači *DMZRouter*.



Service Status Details For All Hosts

Limit Results: 100

Host	Service	Status	Last Check	Duration	Attempt	Status Information
DMZRouter	DMZ Router F0/0 Gateway interface status	CRITICAL	05-05-2022 19:17:43	0d 0h 7m 51s	3/3	SNMP CRITICAL - *2*
	DMZ Router F1/0 DMZ Network interface status	OK	05-05-2022 19:16:13	0d 0h 57m 21s	1/3	SNMP OK - 1
	DMZ Router Uptime service	OK	05-05-2022 19:18:15	0d 0h 55m 19s	1/3	SNMP OK - Timeticks: (2081265) 5:46:52.65
DNSHost	Internal DNS server	OK	05-05-2022 19:17:14	0d 0h 16m 20s	1/3	DNS OK: 0,071 seconds response time. internalweb.enterprise.net returns 192.168.20.20
FTPHost	Internal FTP server	OK	05-05-2022 19:14:58	0d 3h 13m 36s	1/3	FTP OK - 0,011 second response time on 192.168.20.30 port 21 [220 bftpd 4.2 at 192.168.20.30 ready.]
MySQLHost	Internal MySQL server	OK	05-05-2022 19:17:05	22d 6h 56m 33s	1/3	MYSQL 10.0.17-MariaDB OK
externalHTTPHost	HTTP service in DMZ	OK	05-05-2022 19:15:26	0d 0h 13m 7s	1/3	HTTP OK: HTTP/1.0 200 OK - 324 bytes in 0,568 second response time
internalHTTPHost	Internal HTTP service	OK	05-05-2022 19:17:14	0d 0h 16m 20s	1/3	HTTP OK: HTTP/1.0 200 OK - 325 bytes in 0,412 second response time
managementRouter	Management Router Uptime service	OK	05-05-2022 19:15:56	0d 0h 57m 38s	1/3	SNMP OK - Timeticks: (2067216) 5:44:32.16
salesRouter	Sales Router Uptime service	OK	05-05-2022 19:14:49	0d 0h 58m 45s	1/3	SNMP OK - Timeticks: (2060536) 5:43:25.36
servicesRouter	Services Router F0/1 external interface status	OK	05-05-2022 19:14:08	0d 0h 59m 26s	1/3	SNMP OK - 1
	Services Router Uptime service	OK	05-05-2022 19:13:56	22d 6h 52m 2s	1/3	SNMP OK - Timeticks: (2055223) 5:42:32.23

Results 1 - 12 of 12 Matching Services

Obrázek 9: Vzhled okna stavu služeb v situaci D

ZÁVĚR

Cílem této práce bylo zavedení monitoringu v prostředí fiktivní firmy. Byly zmíněny nejrozšířenější protokoly pro vykonání monitorovacích činností a konkrétní softwarová řešení, která můžou být nasazená pro zajištění monitoringu. Byla také popsaná metodika FCAPS, která může sloužit výchozím bodem pro definování monitorovací strategie podniku.

Bylo navrženo prostředí monitoringu, které se sestává ze dvou základních samostatných komponent – GNS3 a Oracle VM Virtualbox. Uvnitř emulované sítě byly zprovozněny služby, které jsou typické pro podnikové sítě – HTTP, FTP, DNS, SQL databáze. Rozsah emulované sítě byl omezen výkonem stroje, na kterém probíhala emulace.

Byl zmíněn také samotný nástroj GNS3. V práci byl také uveden popis tohoto emulačního softwaru a jeho přístup k emulaci různých síťových prvků.

Následně byla zajištěna průběžná kontrola prvku v síti prostřednictvím nástroje Nagios. Pro zavedení monitoringu jsem se musel seznámit s formátem zadávaní definicí a příkazů tohoto nástroje. Definice všech použitých objektu je zahrnuta do práce jako příloha 1.

Byl ukázán postup pro zprovoznění zasílaní upozornění, definice potřebných kontaktních údajů je součástí přílohy 1. Kromě toho ukázky notifikačních zpráv jsou zobrazeny v příloze 2. Uživatelská interakce s webovým prostředím Nagios Core je ukázaná v příloze 3.

Výslednou emulaci lze dále v případě potřeby rozšiřovat a přidávat i nové služby. Projekt emulované sítě je vložen do práce jako příloha 4. Tento projekt je možné přemístit na jiný stroj a tím zreplikovat síť. GNS3 bohužel neumožňuje provést export uzlů, které jsou spravované pomocí VirtualBox. V důsledku toho stroj, který zastává funkci monitorovací stanice, není součásti přílohy. Nicméně definice všech objektů v příloze 1 umožňují snadno nakonfigurovat Nagios po nové instalaci na libovolný operační systém.

POUŽITÁ LITERATURA

[1] TANENBAUM, Andrew S. a David J. WETHERALL *Computer Networks*. Vyd. 5. Pearson, 2010 ISBN 978-0132126953.

[2] INFORMATION SCIENCES INSTITUTE UNIVERSITY OF SOUTHERN CALIFORNIA *RFC 791* [online]. 9-1981 [cit. 18-02-2022]. Dostupné z `https://datatracker.ietf.org/doc/html/rfc791`

[3] NETWORK WORKING GROUP *RFC 2460* [online]. 12-1998 [cit. 18-02-2022]. Dostupné z `https://datatracker.ietf.org/doc/html/rfc2460`

[4] PEARSON EDUCATION, CISCO PRESS *Types of Routing Protocols (3.1.4) > Cisco Networking Academy's Introduction to Routing Dynamically | Cisco Press* [online]. 24-03-2014[cit. 20-02-2022]. Dostupné z `https://www.ciscopress.com/articles/article.asp?p=2180210&seqNum=7`

[5] PEARSON EDUCATION, CISCO PRESS *BGP Fundamentals > Border Gateway Protocol | Cisco Press* [online]. 1-01-2018[cit. 20-02-2022]. Dostupné z `https://www.ciscopress.com/articles/article.asp?p=2756480`

[6] GRAZIANI, Rick a Allan JOHNSON *Routing Protocols and Concepts, CCNA Exploration Companion Guide*. Vyd. 2. Cisco Press, 2008 ISBN 978-1-58713-206-3

[7] FOROUZAN, Behrouz A. *TCP/IP protocol suite*. Vyd. 4. McGraw-Hill, 2010 ISBN 978-0-07-337604-2

[8] NETWORK WORKING GROUP *RFC 792* [online]. 9-1998 [cit. 13-03-2022]. Dostupné z `https://datatracker.ietf.org/doc/html/rfc792`

[9] MAURO, Douglas R. a Kevin J. SCHMIDT. *Essential SNMP*. Vyd. 2. O'Reilly Media, Inc., 2005 ISBN 978-0-596-00840-6

[10] KUNDU, Dinangkur a S. M. Ibrahim LAVLU *Cacti 0.8 Network Monitoring*. Packt Publishing Ltd., 2009 ISBN 978-1-847195-96-8

[11] KOCJAN, Wojciech a Piotr BELTOWSKI *Learning Nagios 3.0*. Birmingham, UK: Packt Publishing Ltd., 2016 ISBN 978-1-78588-595-2

[12] KEARY, Tim *Nagios Core vs Zabbix Comparison* [online]. 28-08-2020[cit. 14-03-2022]. Dostupné z `https://www.comparitech.com/net-admin/nagios-vs-zabbix`

[13] JULIAN, Mike *Practical Monitoring: Effective Strategies for the Real World*. O'Reilly Media, 2017 ISBN 978-1491957356

[14] C. NEUMANN, Jason *The Book of GNS3: Build Virtual Network Labs Using Cisco, Juniper, and More.*. San Francisco: William Pollock, 2014 ISBN 978-1-59327-554-9

SEZNAM PŘÍLOH

PŘÍLOHA 1 – DEFINICE OBJEKTŮ

```
#Contacts

define contact {
contact_name admin
alias Administrator
email example@mail.com
host_notification_commands notify-host-by-email
host_notification_options d,u,r
host_notification_period 24x7
service_notification_commands notify-service-by-email
service_notification_options w,u,c,r
service_notification_period 24x7
}

#Hosts

define host {
host_name  DMZRouter
alias  DMZ router and default gateway
address  192.168.250.1
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
```

```
host_name  DNSHost
alias  DNS host
address  192.168.20.50
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  externalHTTPHost
alias  HTTP host in DMZ
address  192.168.10.10
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  FTPHost
alias  FTP host
address  192.168.20.30
max_check_attempts 3
check_period 24x7
check_command check-host-alive
```

```
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  internalHTTPHost
alias  HTTP host for internal application
address  192.168.20.20
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  managementRouter
alias  Management subnet router
address  192.168.40.1
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}
```

```
define host {
host_name  MySQLHost
alias  MySQL host
address  192.168.20.40
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  salesRouter
alias  Sales subnet router
address  192.168.30.1
max_check_attempts 3
check_period 24x7
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

define host {
host_name  servicesRouter
alias  Services subnet router
address  192.168.20.1
max_check_attempts 3
check_period 24x7
```

```
check_command check-host-alive
contacts  admin
contact_groups
notification_interval 60
notification_period 24x7
}

#Hostgroups

define hostgroup {
hostgroup_name  allHostgroup
alias All hosts in network
members internalHTTPHost, externalHTTPHost, FTPHost,
DNSHost, MySQLHost, DMZRouter, salesRouter,
managementRouter, servicesRouter
}

define hostgroup {
hostgroup_name  DMZHostgroup
alias All hosts running DMZ services
members externalHTTPHost
}

define hostgroup {
hostgroup_name  routersHostgroup
alias All routers in network
members DMZRouter, salesRouter, servicesRouter, managementRouter
}

define hostgroup {
hostgroup_name  internalHostgroup
alias All hosts running internal services
```

```
members internalHTTPHost, FTPHost, DNSHost, MySQLHost
}

#Services

define service {
host_name  DMZRouter
service_description  DMZ Router Uptime service
check_command  check_snmp!-C public -o 1.3.6.1.2.1.1.3.0
                                        --protocol=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  DNSHost
service_description  Internal DNS server
check_command  check_dns
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
```

```
}

define service {
host_name  externalHTTPHost
service_description  HTTP service in DMZ
check_command  check_http
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  FTPHost
service_description  Internal FTP server
check_command  check_ftp
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  internalHTTPHost
```

```
service_description  Internal HTTP service
check_command  check_http
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  managementRouter
service_description  Management Router Uptime service
check_command  check_snmp!-C public -o 1.3.6.1.2.1.1.3.0
                                        --protocol=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  MySQLHost
service_description  Internal MySQL server
check_command  check_mysqld
max_check_attempts 3
```

```
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  salesRouter
service_description  Sales Router Uptime service
check_command  check_snmp!-C public -o 1.3.6.1.2.1.1.3.0
                                        --protocol=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  servicesRouter
service_description  Services Router Uptime service
check_command  check_snmp!-C public -o 1.3.6.1.2.1.1.3.0
                                        --protocol=1
max_check_attempts 3
check_interval 5
retry_interval 1
```

```
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  DMZRouter
service_description  DMZ Router F1/0 DMZ Network interface status
check_command  check_snmp!-C public -o 1.3.6.1.2.1.2.2.1.7.1
                                        --protocol=1 --string=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60

notification_period 24x7
}

define service {
host_name  DMZRouter
service_description  DMZ Router F0/0 DMZ Gateway interface status
check_command  check_snmp!-C public -o 1.3.6.1.2.1.2.2.1.7.2
                                        --protocol=1 --string=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
```

```
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

define service {
host_name  DMZRouter
service_description  DMZ Router F0/! external interface status
check_command  check_snmp!-C public -o 1.3.6.1.2.1.2.2.1.7.2
                                        --protocol=1 --string=1
max_check_attempts 3
check_interval 5
retry_interval 1
check_period 24x7
contacts admin
contact_groups
notification_interval 60
notification_period 24x7
}

#Servicegroups

define servicegroup {
servicegroup_name allServicegroup
alias All services
members DMZRouter, DMZ Router Uptime service, salesRouter,
Sales Router Uptime service, managementRouter,
Management Router Uptime service , servicesRouter,
Services Router Uptime service, internalHTTPHost,
Internal HTTP service, externalHTTPHost,
HTTP service in DMZ, DNSHost, Internal DNS server,
```

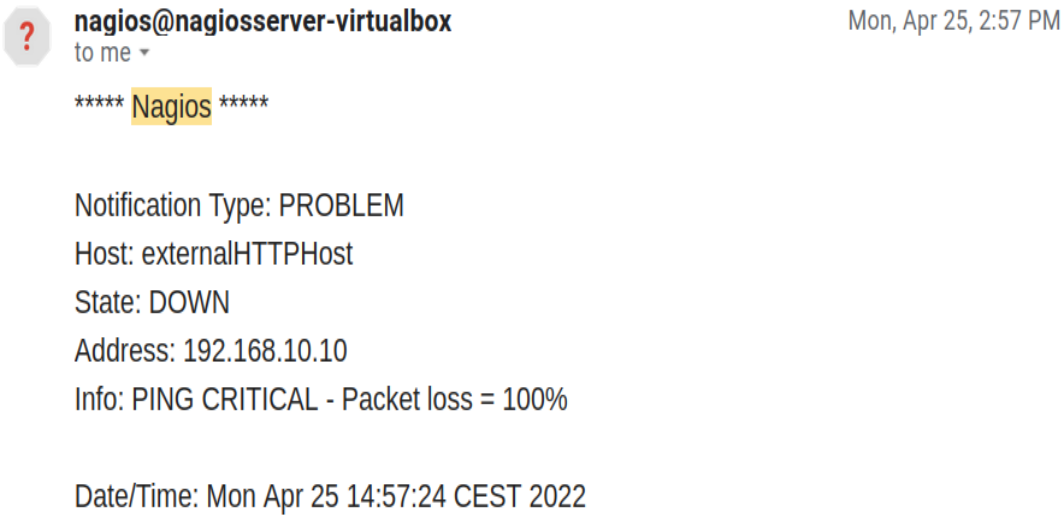
```
FTPHost, Internal FTP server, MySQLHost,
Internal MySQL server
}

define servicegroup {
servicegroup_name routersServicegroup
alias Routers SNMP service
members DMZRouter, DMZ Router Uptime service, salesRouter,
Sales Router Uptime service, managementRouter,
Management Router Uptime service , servicesRouter,
Services Router Uptime service
}

define servicegroup {
servicegroup_name internalServicegroup
alias All services in internal network
members internalHTTPHost, Internal HTTP service, DNSHost,
Internal DNS server, FTPHost, Internal FTP server,
MySQLHost, Internal MySQL server
}

define servicegroup {
servicegroup_name DMZServicegroup
alias Services in DMZ
members externalHTTPHost, HTTP service in DMZ, DNSHost
}
```

PŘÍLOHA 2 – UKÁZKA NOTIFIKACE



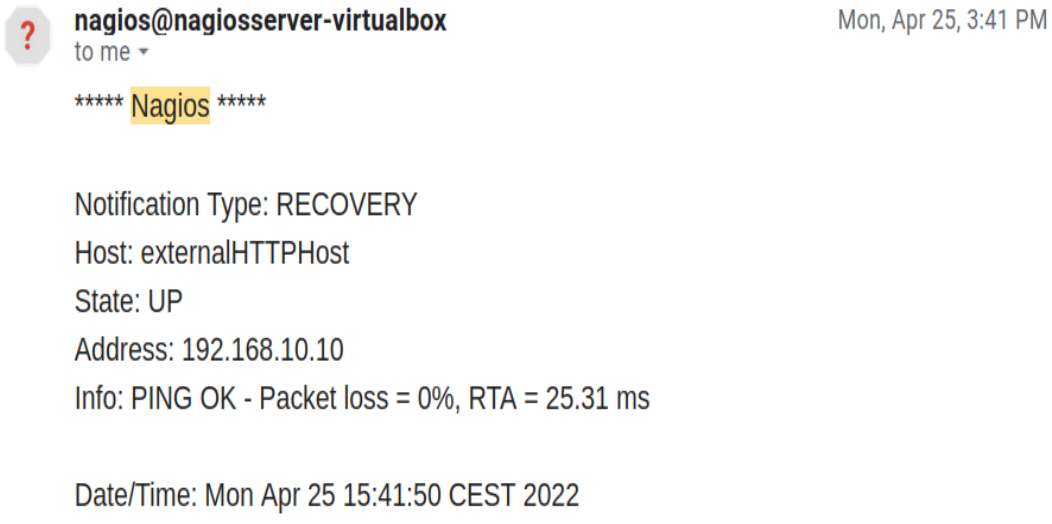
nagios@nagiosserver-virtualbox Mon, Apr 25, 2:57 PM
to me

***** Nagios *****

Notification Type: PROBLEM
Host: externalHTTPHost
State: DOWN
Address: 192.168.10.10
Info: PING CRITICAL - Packet loss = 100%

Date/Time: Mon Apr 25 14:57:24 CEST 2022

Obrázek 10: Notifikace o výpadku hosta nebo služby



nagios@nagiosserver-virtualbox Mon, Apr 25, 3:41 PM
to me

***** Nagios *****

Notification Type: RECOVERY
Host: externalHTTPHost
State: UP
Address: 192.168.10.10
Info: PING OK - Packet loss = 0%, RTA = 25.31 ms

Date/Time: Mon Apr 25 15:41:50 CEST 2022

Obrázek 11: Notifikace o návratu hosta nebo služby do OK stavu

PŘÍLOHA 3 – UKÁZKA INTERAKCE S WEBOVÝM ROZHRANÍM

Přílohou je soubor *WebInterfaceScreencast.mp4* odevzdaný jako součást práce.

PŘÍLOHA 4 – PROJEKT EMULOVANÉ SÍTĚ V GNS3

Přílohou je soubor *projektBPErmolaev.zip* odevzdaný jako součást práce.