Anotace

V bakalářské práci je rozebírána možnost monitorování aktivních prvků firmy Cisco, VLAN sítí sestavených mezi nimi a vytíženosti jednotlivých linek. Všechny informace týkající se zařízení jsou ukládány do MIB databáze. Navržený a následně implementovaný systém využívá, pro získání potřebných informací pro tuto práci, dotazování MIB databáze na zařízení pomocí SNMP protokolu. Pro aplikaci SNMP protokolu je použit programovací jazyk Perl. Získaná data jsou zpracována a následně ukládána do MySQL databáze. Navržený systém není závislý na platformě ani na této implementaci. Veškeré použité nástroje a technologie jsou v práci popsány a rozebrány. V práci je detailně popsáno a ukázáno, jak se zařízení dotazovat i s výsledky dotazování.

Klíčová slova

SNMP, MIB, CDP, VLAN, monitorování sítě, mapování sítě

Annotation

The bachelor thesis deals witch possibility of monitoring active items of Cisco Company, VLAN networks built between them and utilization of particular lines. All the information about items is stored in MIB database. Designed and then implemented system uses pooling MIB database using SNMP protocol, to obtain required information. For application SNMP protocol, programming language Perl is used. Obtained and processed data are stored in MySQL database. Designed system is not dependent either on any platform or this implementation. All the technologies and tools used described in the thesis. In the thesis there is described and shown, how is possible pooling items including results of pooling.

Key words

SNMP, MIB, CDP, VLAN, monitoring network, mapping network

Citace

HERMAN, V. *Monitorování ethernetové sítě.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 39 s. Vedoucí bakalářské práce Ing. Jan Jeřábek.

Prohlášení

Prohlašuji, že svoji bakalářskou práci na téma Mapování ethernetové sítě jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením tohoto projektu jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestně právních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

V Brně dne …… ……

podpis autora

Poděkování

Chtěl bych poděkovat Ing. Zdeňkovi Roubíčkovi z T-Systems Czech Republic za velice zajímavé téma a cenné rady. Firmě T-Systems Czech Republic za poskytnutí hardwaru na testování. Ing. Janu Jeřábkovi za umožnění psaní práce a za odborné vedení. Mgr. Martinu Kyselákovi za obětavou pomoc, trpělivost a cenné rady programátora.

Velké díky také patří rodičům a přítelkyni Bětce za umožnění studia, vytváření příznivých podmínek při studiu, trpělivost a morální podporu.

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod

Ethernetová síť může být distribuována do více než jednoho místa pomocí aktivních prvků, jako jsou směrovače a přepínače. Tyto prvky je nutno nějakým způsobem dohledovat. To lze dělat lokálně u každého aktivního prvku, avšak nejlepší forma managementu sítě je správu sítě centralizovat. Pokud je síť spravována z jednoho centra, díky vzdálenému přístupu na aktivní prvky, sníží se náklady a díky dohledovému systému lze předcházet výpadkům. Toto opatření zvyšuje spolehlivost sítě, což je pro poskytovatele datových služeb v dnešní době nutností. Nejvhodnější způsob managementu sítě je systém, postavený na standardním protokolu, konkrétně SNMP (Simple Network Management Protocol) díky jeho strojové nezávislosti. V práci bude protokol rozebírán a používán k získávání dat od zařízení. Na aplikaci tohoto protokolu, tj. dotazování jednotlivých zařízení, se zdá být nejvhodnější programovací jazyk Perl. Především díky své nezávislosti na platformě a jeho jednoduchosti ve zpracování textu. Programovací jazyk Perl bude v práci používán pro aplikaci SNMP protokolu. I když je SNMP protokol strojově nezávislý, v práci bude rozebírán a aplikován pouze na aktivní prvky firmy Cisco, dle zadání práce.

V první polovině práce je teoreticky rozebírán způsob dotazování zařízení. V této polovině práce jsou také popisovány možnosti, jak se dostat k informacím o zařízení a kde jsou tyto informace uloženy. Druhá polovina práce je zaměřena na implementaci, ukázky a testování navrženého systému.

1.1 Požadavky na navrhovaný systém

Navrhovaný systém pro firmu T-Systems Czech Republic by měl umožnit dohledovat ethernetovou síť, monitorovat jednotlivá zařízení, sbírat z nich data a umožňovat přístup k jejich stavům v čase. Systém by měl splňovat tato kritéria:

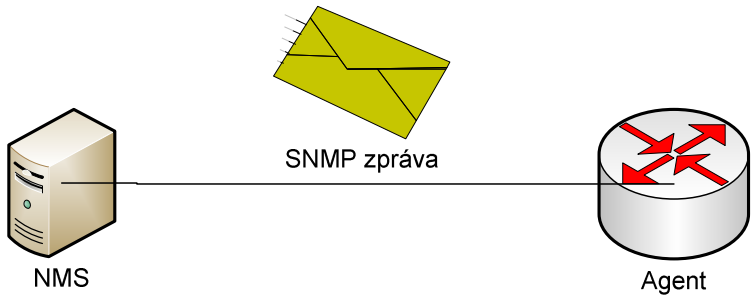
- získání topologie sítě
- získání informací o sestavených VLAN sítích
- získání informací o vytíženosti jednotlivých linek
- získání informací o stavu zařízení
- získání informací o jednotlivých portech na zařízení
- jednotlivé stavy uchovávat

2 Simple Network Management Protocol

Protokol SNMP je nejznámější a nejpoužívanější protokol pro správu sítí. Poprvé byl formalizován v roce 1988. SNMP je jednoduchý protokol pro management architektury TCP/IP (Transmission Control Protocol/Internet Protocol). Pracuje na aplikační vrstvě. K posílání zpráv využívá protokol UDP (User Datagram Protocol). Distribuce SNMP zpráv je zprostředkována NMS (Network Management Station) a agenty, viz obr. 1.

NMS, obvykle server, řídí všechny agenty, které má spravovat. Dotazuje se jednotlivých agentů, shromažďuje informace od těchto agentů a tím vytváří centralizovanou databázi. V rozsáhlejší síti, je možno umístit více NMS. Potom každá NMS má svoji skupinu agentů. Po získání informací od svěřených agentů si NMS musí vyměnit informace mezi sebou.

Agent představuje spravované zařízení, jehož úkolem je odpovídat na dotazy NMS pomocí programu, který je umístěn na zařízení.



Obr. 1: Práce NMS vs. agent

Pokud by na každém zařízení byl umístěn takový agent, je možno jakoukoli síť dohledovat a spravovat pomocí stavových informací od jednotlivých agentů a s těmito centralizovanými informacemi pracovat na NMS.[7]

2.1 Verze SNMP

Protokol SNMP existuje ve třech verzích: [5]

- SNMP verze 1 – označován jako SNMPv1, specifikován v RFC (Request For Comments) 1155 v roce 1990. Pracuje se dvěma příkazy: *Get (přečti hodnotu proměnné)* a *Set (nastav hodnotu proměnné).*
- SNMP verze 2 – označován jako SNMPv2, specifikován v RFC 1441-1452 v roce 1991. Na rozdíl od svého předchůdce si může vyžádat od agenta naráz všechny informace, které má agent k dispozici. V této verzi je zahrnuta možnost komunikace NMS mezi sebou, kvůli komplexnímu dohledu nad sítí.
- SNMP verze 3 – označován jako SNMPv3, specifikován v RFC 3411-3418 v roce 2002. Ve verzi 3 je SNMP protokol obohacen o bezpečnostní mechanizmus zpráv, řízení přístupu k síťovým zařízením, ochranu SNMP zpráv při cestě po síti a ověření zdroje SNMP zpráv.

2.2 Dotazování v SNMP

Dotazování používá model dotaz/odpověď. Standardně se NMS dotazuje agentů a žádá od nich nové informace. Četnost dotazování závisí na nastavení. Dotazy se dělí na tři skupiny:[7]

- monitorovací dotazování – zjišťují dosažitelnost zařízení
- prahové dotazování – dotazy porovnávající základní nastavení a provoz zařízení, prahové hodnoty záleží na konkrétním nastavení
- výkonové dotazování – umožňují sledovat vytíženost sítě

Protokol SNMP má pouze tyto základní příkazy:[5]

- *Get, GetNext a Walk* – těmito příkazy získává NMS informace od agentů
- *GetBulk* – v SNMPv2, umožňuje získat všechny informace najednou
- *Set* – umožňuje NMS nastavit/změnit hodnotu proměnné, operace vyžaduje zabezpečení přístupu a autorizaci
- *Trap* – jediný příkaz, který dovoluje agentovi posílat informace bez předchozí výzvy týkající se urgentních událostí na zařízení
- *Inform* – v SNMPv2, umožňuje NMS posílat informace o přidělených agentech další NMS

2.3 SNMP – strojově nezávislý protokol

SNMP protokol byl navržen tak, aby byl strojově nezávislý. Tedy aby se dal aplikovat na jakékoli zařízení v síti a tím byl umožněn dohled zařízení. Je možné jej použít na:

- síťové prvky (směrovače, přepínače, rozbočovače)
- servery (nezávisle na platformě)
- osobní počítače (nezávisle na platformě)
- tiskárny
- UPS (Uninterruptible Power Supply) – záložní zdroje napájení
- ostatní síťová zařízení

Strojovou nezávislost má SNMP protokol díky tomu, že deklaruje vlastní datové proměnné podle standardu SMI (Structure od Management Information), tzn. strukturované informace pro správu. SMI definuje strojově nezávislou syntaxi datových typů. Datové typy potom nejsou závislé na datových typech použitých na jakémkoli zařízení. SMI deklaruje datové proměnné typu: identifikátor objektu, čítač, řádek, tabulka, řetězec oktetu, síťová adresa a další části protokolu SNMP.

MIB (Management Information Base), která bude popisována dále, je programována pomocí programovacího jazyku ASN.1 (Abstract Syntax Notation One), což je standard OSI (International Organization for Standardization). Tento programovací jazyk byl stvořen pro programovaní SMI, tedy nezávislých datových proměnných. Program by měl být napsán podle zásady „napsat jednou, spouštět kdekoliv“, což je stejná myšlenka nezávislosti jako u SMI.[7]

Pro rozdělení různých zařízení do skupin, se používá SNMP comunita. SNMP comunita je vytvoření virtuální množiny, do které mohou zařízení náležet. Při dotazování SNMP protokolem se udává také SNMP comunita a tím se tedy mohu dotazovat pouze zařízení, které do právě zadané comunity patří.

3 Management Information Base

MIB představuje informační databázi, která je strukturována do logického stromu. Do MIB agent ukládá veškeré informace ve formě proměnných. Informace se definují jedinečně a jednoznačně pomocí: [7]

- *Object Identifier* – jedinečné a jednoznačné pojmenování proměnné, pomocí textového jména platné v celé MIB. Slouží k větší srozumitelnosti, např.: *sysName* (hostname - název zařízení).
- *Object Descriptor* – jedinečné a jednoznačné pojmenování proměnné, pomocí posloupnosti přirozených čísel platné v celé MIB. S tímto pojmenováním pracuje SNMP protokol, např.: 1.3.6.1.2.1.1.5

Pokud mám textové jméno určité proměnné a potřebuji získat posloupnost čísel, tedy formát, se kterým pracuje protokol, stačí textové jméno zadat do MIB překladače, který jméno vyhledá v logickém stromu a vypíše posloupnost čísel dle pozice v logickém stromu, jenž bude vysvětlen dále.

Pokud do MIB překladače zadám textové jméno, např.: *sysName*, bude vrácena tato posloupnost čísel: 1.3.6.1.2.1.1.5.[1]

V MIB rozlišujeme několik druhů proměnných: [7]

- jednoduché typy – primitiva *INTEGER* (celé číslo), *OCTET STRING* (řetězec znaků), *OBJECT IDENTIFIER* (řetězec znaků, pojmenování), *NULL* (označení pro žádnou hodnotu)
- složené typy – tabulka *SEQUENCE OF* primitiv
- definované typy – *IpAddress* jako *OCTET STRING* v délce 4 slabik, *TimeTicks* jako *INTEGER*

Vlastní hodnota proměnné je de facto *skalár*. Pokud se jedná o tabulku, je reprezentována *koncepční tabulkou* skládající se ze *sloupcových objektů*, které opět obsahují skalární objekty. Identifikace v sloupcových objektech probíhá pomocí přípony, která určuje pozici v tabulce, tedy sloupec a řádek.[5]

3.1 Typy MIB

Základními kategoriemi proměnných v MIB jsou: [7]

- konfigurace – podává základní informace o zařízení,
  např.: *sysName* (hostname - název zařízení)
- četnost chyb – monitoruje určité síťové rozhraní,
  např.: *ifInErrors* (počet příchozích chyb na rozhraní)
- šířka pásma – pracuje se šířkou pásma,
  např.: *icmpInEchos* (šířka pásma směrem dovnitř)
- tok síťového provozu – monitorování toku síťového provozu,
  např.: *locIfInBitSec* (bitový provoz na rozhraní směrem dovnitř)
- nedosažitelné adresy – monitoruje snahu o dosažitelnost adres,
  např.: *icmpOutDesUnreachs* (měří čas žádostí o dotazování dosažitelnosti adres)
- data SNMP – monitoruje zatížení zařízení SNMP dotazy,
  např.: *snmpInGetRequest* (zatížení přicházejícími SNMP dotazy)

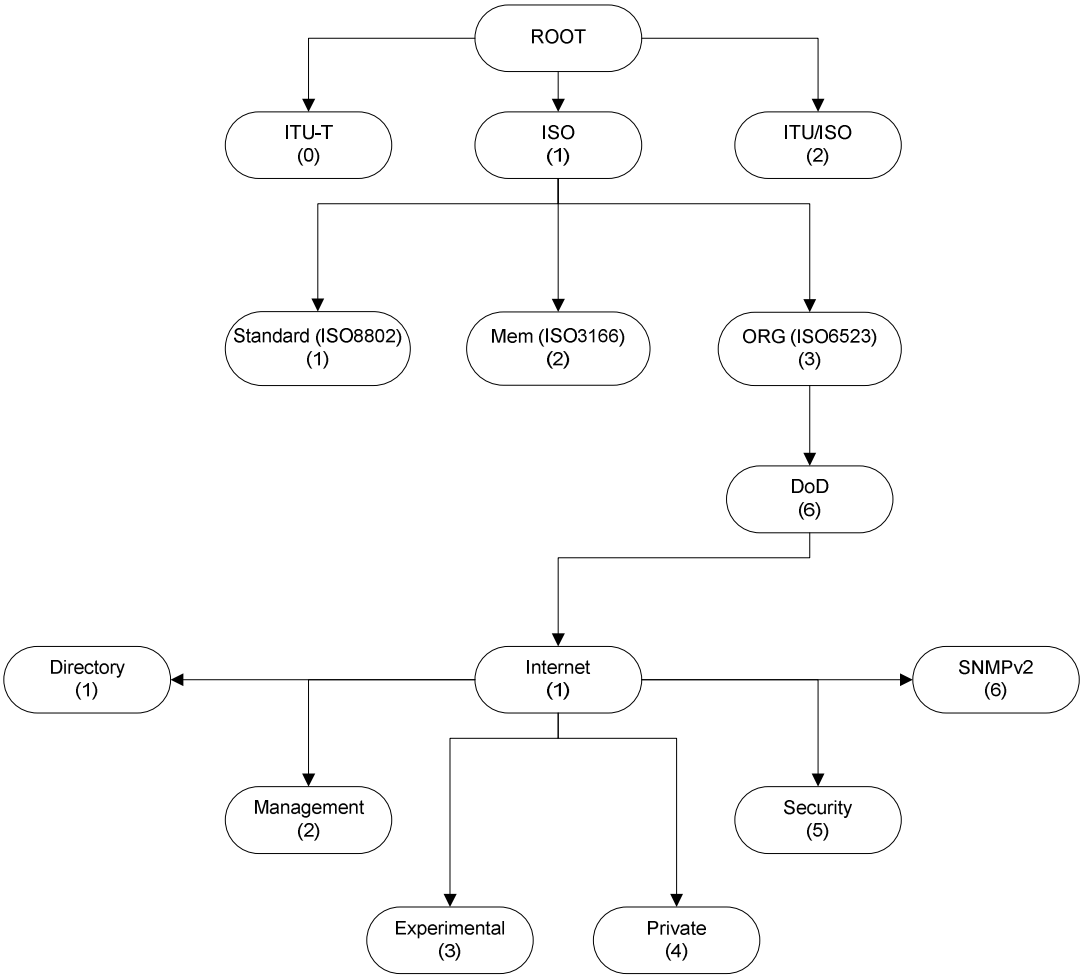
Pro lepší orientaci je dobré vědět, že *Object Identifier* se vlastně skládá z anglických zkratek. Tedy *if* je z anglického slova *interface*, dále třeba *loc* je z anglického slova *local*. Zkratka *sys* je z anglického slova *system*. Výraz *In* popisuje příchozí provoz, naopak výraz *Out* popisuje provoz odchozí.

3.2 Hierarchie MIB

MIB je strukturována do logického stromu. Současná verze definuje více než 200 standardních objektů a více jak 400 privátních, soukromých objektů. V práci bude rozebírána především soukromá větev firmy Cisco. Kompletní seznam privátní větve má pouze výrobce konkrétního zařízení. Výrobce také většinou poskytuje MIB překladač, kde lze kompletní strom procházet. Překládat lze *Object Identifier* na *Object Descriptor* a naopak. V této kapitole bych chtěl naznačit, jak systém funguje a pozastavím se nad základními objekty a dále také objekty důležitými pro navrhovaný systém. Na obrázcích bude vyobrazen logický strom s větvemi. Každý objekt na obrázku, mimo *root*, obsahuje jméno a číslo. Jméno slouží k sestavení *Object Identifier* a z čísel je potom sestaven *Object Descriptor*, podle logické struktury stromu. Popis se bude týkat pouze základní větvě, resp. důležitých částí. [7]

3.2.1 Základní hierarchie MIB

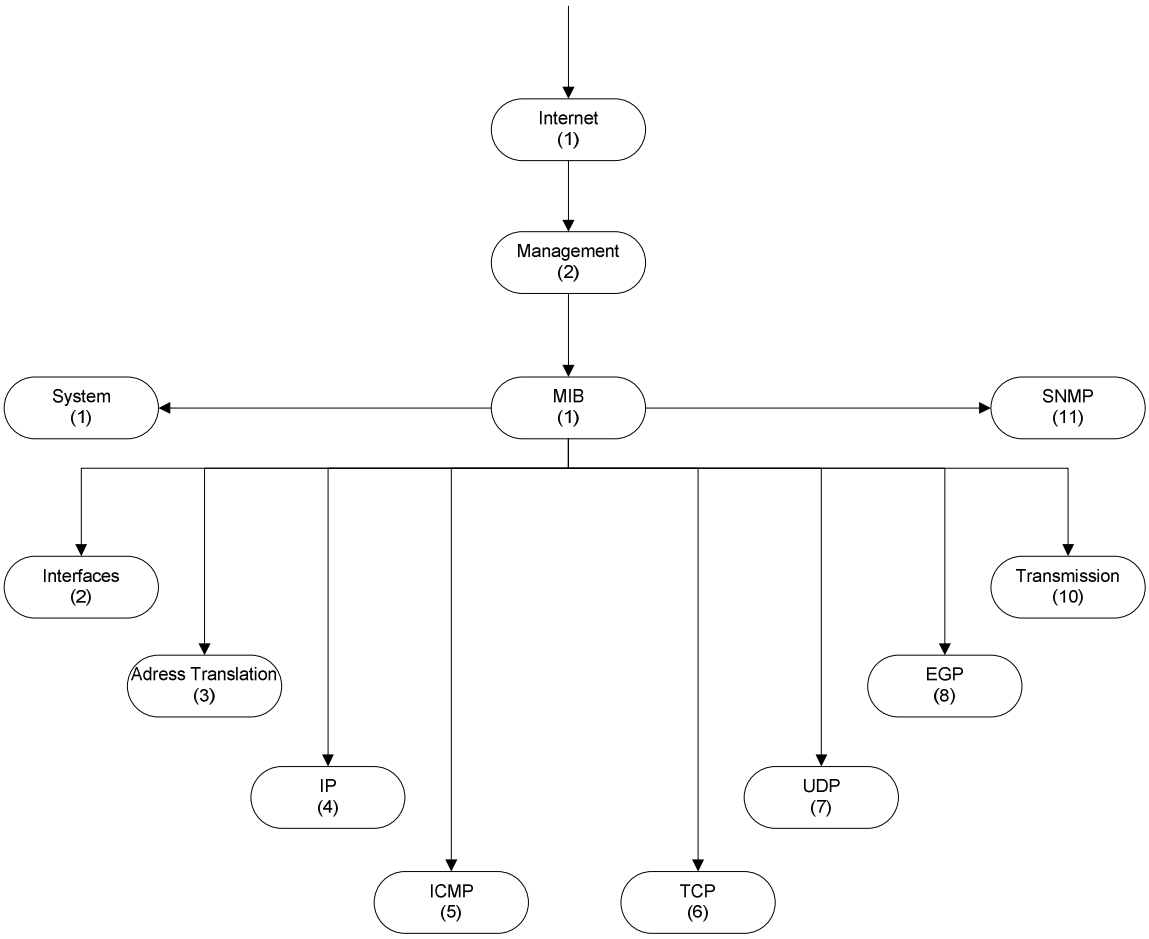
Na obr. 2 je vyobrazena základní hierarchie MIB. Na vrcholu je kořen stromu, nejvíce rozvinutá větev pokračuje objektem *ISO*, což je organizace pro standardy. Dále je objekt *ORG*, představující zkratku pro organizace. Následuje objekt *DoD* (Department of Defense), vyjadřující americké ministerstvo obrany, které vše odstartovalo svým projektem ARPANET (Advanced Research Projects Agency Network). Další objekt je *Internet*. Zde základní, nejvíce rozvinutá větev končí a strom se dělí na důležitých šest větví. První je objekt *Directory*, neboli adresář, který je vyhrazen pro budoucí účely. Další větví objektů je *Management*, což je větev standardních objektů pro nástroje na správu. Třetí větví objektů je větev *Experimental*, která slouží k experimentálním účelům. Čtvrtá větev *Private* je nejdůležitější pro výrobce zařízení, protože si v této větvi mohou vytvořit cokoli, samozřejmě při dodržení určitých pravidel. Pátá větev *Security* je věnována zabezpečení. Poslední znázorněná větev *SNMPv2* je určena pro SNMP protokol verze 2. [1], [7]



Obr. 2: Základní hierarchie MIB

3.2.2 Hierarchie MIB management

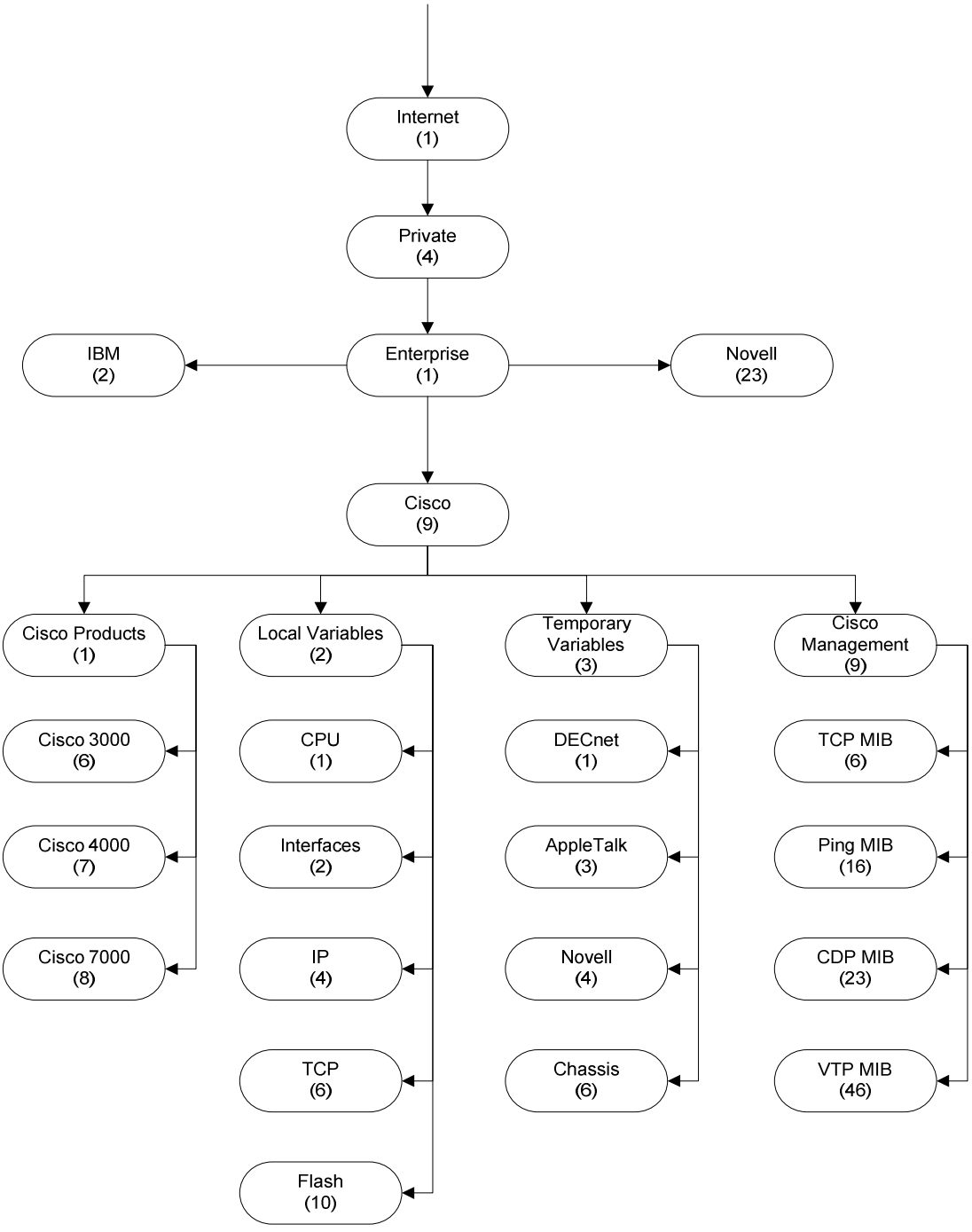
Na obr. 3 je znázorněna pokračující, rozšiřující se větev *Managementu* z obr. 2. Tato větev je standardem pro všechny a je tedy shodná v každé verzi MIB. Po objektu *Management* následuje v této větvi objekt *MIB*. Zde se větev rozděluje na jedenáct důležitých větví. První je *System*, kde můžeme nalézt proměnné týkající se stavu zařízení nebo základního nastavení. Druhá větev, *Interfaces*, je učena pro proměnné týkající se rozhraní, tedy chyby rozhraní nebo stavy rozhraní apod. Další větev, kterou musím zmínit, je *ICMP* (Internet Control Message Protocol). V této větvi je možno nalézt proměnné týkající se odezvy, šířky pásma a dostupnosti konkrétního zařízení. Další větve jsou sice také důležité, avšak nemají další využití v této práci. Uvedl jsem je jen pro úplnost, viz obr. 3. [1], [7]



Obr. 3: Hierarchie MIB management

3.2.3 Privátní hierarchie MIB firmy Cisco

Na obr. 4 je naznačena částečná struktura logického stromu soukromé větve firmy Cisco. Není potřeba obrázek blíže popisovat, protože je jednoznačný a jednalo by se jen o překlad. Styl, jakým jsou obrázky tvořeny, již popsán byl. Potřebné objekty budou popisovány dále. [1], [7]



Obr. 4: Privátní hierarchie MIB firmy Cisco

4 Analýze sítě, Cisco technologie

Navrhovaný systém by měl umět sestavit topologii sítě a získat informace o sestavených VLAN (Virtual Local Area Network) sítích v síti. Tato kapitola je věnována popisu, jak lze tyto informace najít na zařízeních firmy Cisco.

4.1 Cisco Discovery Protocol

CDP (Cisco Discovery Protocol) je protokol umožňující zařízením firmy Cisco vyhledávat a následně komunikovat se sousedními zařízeními a vyměňovat si tak informace. Neváže se na síťové vrstvy OSI modelu, takže spolu zařízení mohou komunikovat i bez nastavených IP adres. Tento protokol není závislý na používaném médiu a ve výchozím nastavení je zapnut na každém zařízení od firmy Cisco. Pomocí CDP tedy lze zjistit sousedy zařízení a tak postupně odhalit celou topologii sítě.[7]

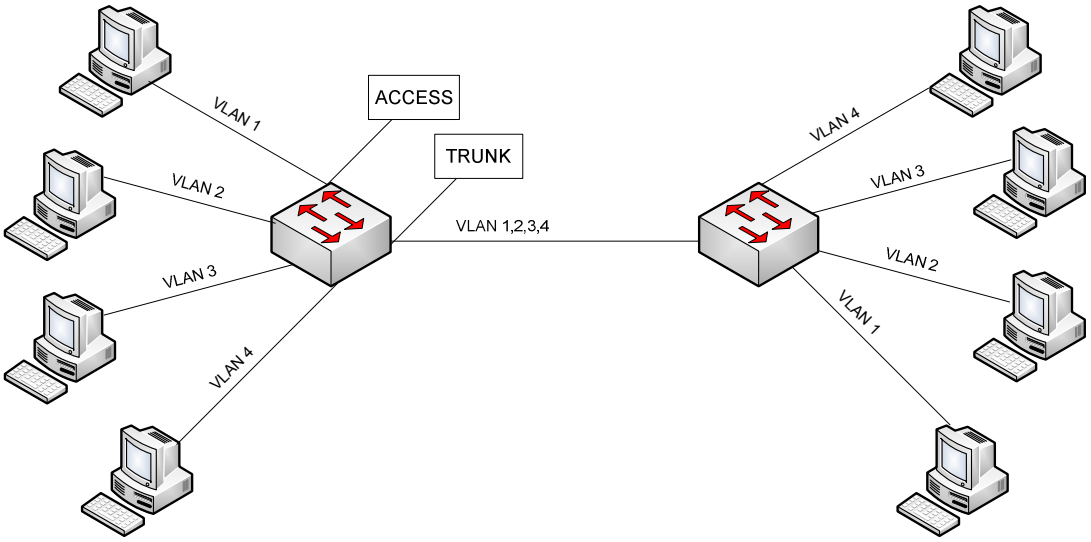
4.2 VLAN sítě

VLAN je zkratka pro Virtuální LAN sítě. Pomocí VLAN sítě lze vytvořit logickou síť LAN, která následně spojuje uzly/koncová zařízení, jež mohou být připojeny přes různá fyzická připojení. Jednotlivá koncová zařízení se tedy mohou nacházet v různých fyzických LAN sítích, avšak jako by byli v jediné síti LAN. VLAN sítě mnohou být: [7], [8]

- definované VLAN sítě pomocí portů
- definované VLAN sítě pomocí MAC (Media Access Control) adres
- definované VLAN sítě pomocí protokolů nebo adres
- definované VLAN sítě pomocí skupinového vysílání

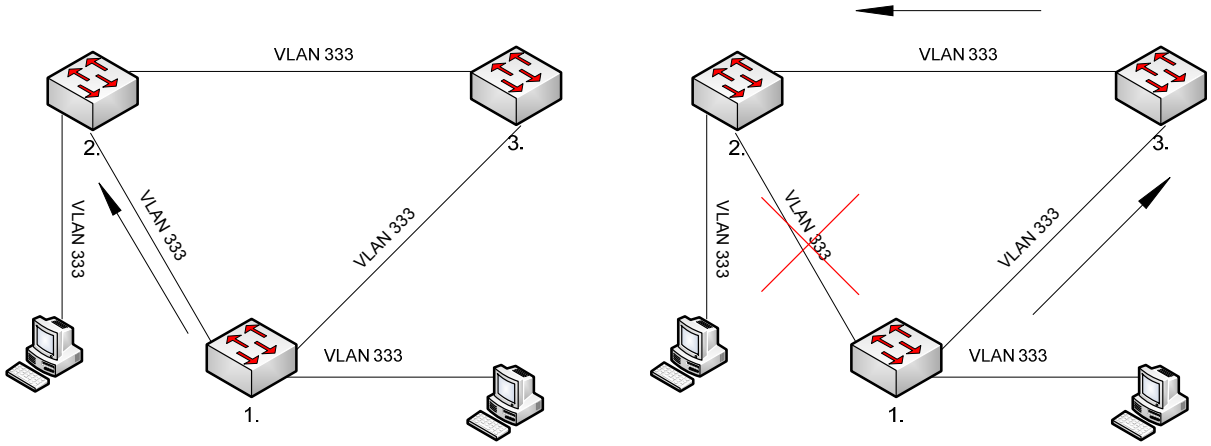
Pro navrhovaný systém je důležité definovaní VLAN pomocí portů, protože tento způsob je nejrozšířenější. Názorná ukázka zapojení a vytvoření logické sítě, využití jednoho fyzického média, je na obr. 5. Jak je také vidět na obr. 5, konfigurace rozhraní přepínače umožňuje dvě varianty. První varianta je rozhraní nastavit do stavu ACCESS. Tento stav rozhraní umožňuje povolení pouze jedné VLAN sítě. Nastavení do tohoto stavu se používá na rozhraních, kde jsou připojeny koncová zařízení. Druhá varianta je nastavit rozhraní přepínače do stavu

TRUNK. Tento stav umožňuje na rozhraní nakonfigurovat více VLAN sítí najednou. Nastavení rozhraní do stavu TRUNK se používá na portech, kterými jsou přepínače propojeny mezi sebou. Těmito porty je tak umožněno vytváření logických sítí jakýchkoli rozměrů.



Obr. 5: Ukázka VLAN sítí

Na obr. 6 je znázorněno řešení v případě chyby v síti. Při konfiguraci VLAN v sítích by se mělo postupovat podle určitých pravidel. Jedno z nich zní, vytvářenou VLAN síť na přepínačích zakruhovat, viz obr. 6. Tedy pokud spadne linka mezi *1. přepínačem* a *2. přepínačem* a *VLAN 333* je nakonfigurována i na *3. přepínači*, potom se provoz odkloní přes *3. přepínač*, viz obr. 6.

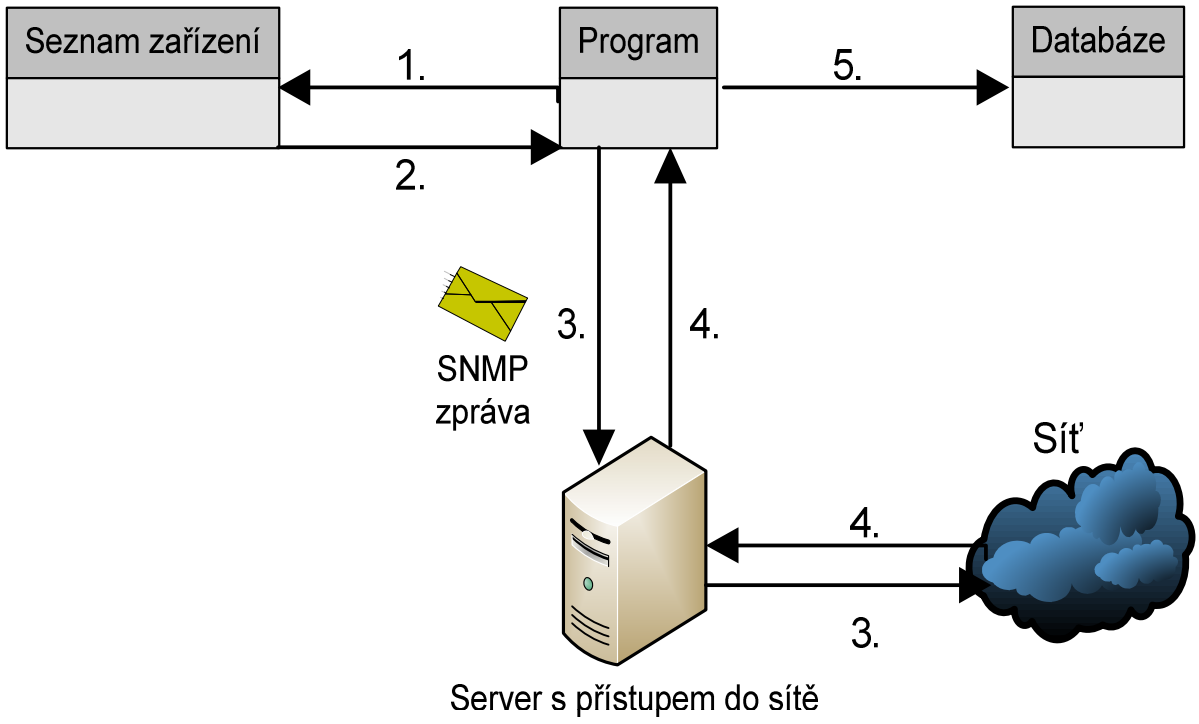


Obr. 6: Ukázka konfigurace a funkčnosti VLAN sítí

5 Návrh systému

Aby navrhovaný systém pro firmu T-Systems Czech Republic mohl fungovat, je zapotřebí čtyř částí. První částí jsou vybrané MIB objekty, ze kterých lze vyčíst určitou informaci tak, aby systém splňoval výše zmíněné požadavky. Druhou částí, je program/programovací jazyk, díky kterému bude systém pracovat na vyhledávání/dotazování agentů a získávání informací od agentů. Třetí částí je databáze, do které se budou získaná data ukládat. Čtvrtou částí je seznam zařízení, kterých se má systém dotazovat a získávat od nich informace. Podmínkou funkčnosti systému je spuštění na serveru, který má přístup ke všem aktivním prvkům v seznamu zařízení, tedy NMS. Takovéto servery se ve velkých sítích poskytovatelů služeb využívají na vzdálenou správu zařízení.

Tyto čtyři části budou spolupracovat tak, jak je naznačeno na obr. 7. Program/NMS je ústřední částí komunikující se všemi ostatními částmi. Program/NMS získá adresu/jméno agenta ze seznamu zařízení (1.),(2.). Na tuto adresu/jméno vyšle program SNMP zprávu s dotazem na potřebné údaje (3.). Agent, umístěný na zařízení v síti, přečte požadovanou informaci z MIB a vyšle ji zpět k NMS/programu (4.). Program/NMS získanou informaci zpracuje a odešle ji do databáze, kde se údaj uloží pro další použití (5.)



Obr. 7: Schéma jednotlivých částí systému

5.1 Potřebné MIB objekty

V této kapitole budou popsány MIB objekty, které jsou nezbytné k zjištění požadovaných informací, a způsob, jak s nimi zacházet a porozumět jim. U každého MIB objektu bude popsán typ proměnné, jaký SNMP dotaz vrací, a příklad vrácené hodnoty ze zařízení. Informace o jednotlivých objektech jsem našel ve výpisu odpovědí ze zařízení a MIB překladači firmy Cisco.

Dotazy se provádí tímto způsobem:

```
snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.2.1.1.5.0
```

Kde `–v2c` je verze SNMP protokolu, `–c public` je komunita ve které se zařízení nachází, následuje IP adresa dotazovaného zařízení a *Object Descriptor*.

Odpověď od zařízení je pak následující:

```
SNMPv2-MIB::sysName.0 = STRING: router
```

Kde `SNMPv2` je verze SNMP protokolu, `MIB::sysName.0` je popis v MIB stromu, `STRING` typ proměnné a `router` je samotná odpověď.

MIB objekty týkající se zařízení:

- *sysName* - 1.3.6.1.2.1.1.5.0
  význam: hostname (jméno) zařízení
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.2.1.1.5.0
  odpověď: SNMPv2-MIB::sysName.0 = STRING: router

- *sysLocation* - .1.3.6.1.2.1.1.6.0
  význam: umístění zařízení
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.2.1.1.6.0
  odpověď: SNMPv2-MIB::sysLocation.0 = STRING: Brno

- *sysUpTime* - .1.3.6.1.2.1.1.3.0
  význam: doba po kterou je zařízení v provozu
  typ proměnné: timeticks
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.2.1.1.3.0
  odpověď: DISMAN-EVENT-MIB::sysUpTimeInstance = Timeticks: (49746) 0:08:17.46

MIB objekty týkající se zařízení nalezených přes CDP, tedy sousedů zařízení:

- *cdpCacheAddressType* - .1.3.6.1.4.1.9.9.23.1.2.1.1.3
  význam: typ adresy sousedního zařízení, 1=síťová adresa
  typ proměnné: integer
  dotaz: snmpwalk -c public 192.168.0.1 .1.3.6.1.4.1.9.9.23.1.2.1.1.3
  odpověď: SNMPv2-SMI::enterprises.9.9.23.1.2.1.1.3.10101.1 = INTEGER: 1

- *cdpCacheAddress*- .1.3.6.1.4.1.9.9.23.1.2.1.1.4
  význam: IP adresa sousedního zařízení
  typ proměnné: hex-string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.4.1.9.9.23.1.2.1.1.4
  odpověď:SNMPv2-SMI:enterprises.9.9.23.1.2.1.1.4.1.1 = Hex-STRING: C0 A8 00 02
  Po převodu z šestnáctkové soustavy je IP adresa: 192.168.0.2

- *cdpCacheDeviceId*- .1.3.6.1.4.1.9.9.23.1.2.1.1.6
  význam: hostname (jméno) sousedního zařízení
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.4.1.9.9.23.1.2.1.1.6
  odpověď: SNMPv2-SMI::enterprises.9.9.23.1.2.1.1.6.1.1 = STRING: "switch"

- *cdpCacheDevicePort* - .1.3.6.1.4.1.9.9.23.1.2.1.1.7
  význam: rozhraní na sousedním zařízení, přes které je zařízení připojeno
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.4.1.9.9.23.1.2.1.1.7
  odpověď:
  SNMPv2-SMI::enterprises.9.9.23.1.2.1.1.7.1.1 = STRING: "GigabitEthernet0/1"

- *cdpCacheDevicePlatform* - .1.3.6.1.4.1.9.9.23.1.2.1.1.8
  význam: typ sousedního zařízení
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.1 .1.3.6.1.4.1.9.9.23.1.2.1.1.8
  odpověď:
  SNMPv2-SMI::enterprises.9.9.23.1.2.1.1.8.1.1 = STRING: "cisco WS-C2960-8TC-L"

MIB objekty týkající rozhraní na zařízení:

- *ifIndex*- .1.3.6.1.2.1.2.2.1.1
  význam: indexy jednotlivých rozhraní
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.2.1.2.2.1.1
  odpověď: IF-MIB::ifIndex.1 = INTEGER: 1
  IF-MIB::ifIndex.6 = INTEGER: 6
  IF-MIB::ifIndex.10 = INTEGER: 10
  IF-MIB::ifIndex.14 = INTEGER: 14
  IF-MIB::ifIndex.20 = INTEGER: 20
  IF-MIB::ifIndex.10001 = INTEGER: 10001
  IF-MIB::ifIndex.10002 = INTEGER: 10002
  IF-MIB::ifIndex.10003 = INTEGER: 10003
  IF-MIB::ifIndex.10004 = INTEGER: 10004
  IF-MIB::ifIndex.10005 = INTEGER: 10005
  IF-MIB::ifIndex.10006 = INTEGER: 10006
  IF-MIB::ifIndex.10007 = INTEGER: 10007
  IF-MIB::ifIndex.10008 = INTEGER: 10008
  IF-MIB::ifIndex.10101 = INTEGER: 10101
  IF-MIB::ifIndex.10501 = INTEGER: 10501

Odpověď je od 8. portového switche s jedním propojovacím portem. Indexy do 10000 reprezentují VLAN sítě, které se na zařízení nacházejí. Indexy od 10000 výše již reprezentují porty. Avšak poslední dva indexy 10101 a 10501 díky stejné konečné 1 reprezentují identický port s možností připojení metalického kabelu nebo optického kabelu.
Dále je nutno pracovat s těmito indexy, protože jednotlivé rozhraní v MIB stromu mají přiřazeny tyto indexy.

Dále budu pro dotazování používat index 10008, tedy 8. port na dotazovaném zařízení.

- *ifOperStatus* - .1.3.6.1.2.1.2.2.1.8
  význam: stav rozhraní, 1=up nebo 2=down
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.2.1.2.2.1.8.10008
  odpověď: IF-MIB::ifOperStatus.10008 = INTEGER: up(1)

- *ifDescr* - .1.3.6.1.2.1.2.2.1.2
  význam: název rozhraní
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.2.1.2.2.1.2.10008
  odpověď: IF-MIB::ifDescr.10008 = STRING: FastEthernet0/8

- *ifName* - .1.3.6.1.2.1.31.1.1.1.1
  význam: zkratka rozhraní
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.2.1.31.1.1.1.1.10008
  odpověď: IF-MIB::ifName.10008 = STRING: Fa0/8

- *locIfDescr* - .1.3.6.1.4.1.9.2.2.1.1.28
  význam: popis (description) rozhraní
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.2.2.1.1.28.10008
  odpověď: SNMPv2-SMI::enterprises.9.2.2.1.1.28.10008 = STRING: "port pro pc"

- *ifSpeed* - .1.3.6.1.2.1.2.2.1.5
  význam: rychlost rozhraní v bit/s
  typ proměnné: gauge32
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.2.1.2.2.1.5.10008
  odpověď: IF-MIB::ifSpeed.10008 = Gauge32: 100000000
  100 000 000 bit/s = 100 Mbit/s

- *locIfInBitsSec* - .1.3.6.1.4.1.9.2.2.1.1.6
  význam: průměrný datový tok za 5 minut v bit/s směrem k zařízení
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.2.2.1.1.6.10008
  odpověď: SNMPv2-SMI::enterprises.9.2.2.1.1.6.10008 = INTEGER: 47000
  47 000 bit/s = 47 kbit/s

- *locIfOutBitsSec* - .1.3.6.1.4.1.9.2.2.1.1.8
  význam: průměrný datový tok za 5 minut v bit/s směrem od zařízení
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.2.2.1.1.8.10008
  odpověď: SNMPv2-SMI::enterprises.9.2.2.1.1.8.10008 = INTEGER: 46000
  46 000 bit/s = 46 kbit/s

MIB objekty týkající se VLAN sítí:
V tomto případě pro dotazování budu používat index 10 tedy VLAN 10.

- *vtpVlanState* - 1.3.6.1.4.1.9.9.46.1.3.1.1.2.1
  význam: stav VLAN sítě, 1=aktivní
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.3.1.1.2.1.10
  odpověď: SNMPv2-SMI::enterprises.9.9.46.1.3.1.1.2.1.10 = INTEGER: 1

- *vtpVlanType* - .1.3.6.1.4.1.9.9.46.1.3.1.1.3.1
  význam: typ VLAN sítě, 1=ethernet
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.3.1.1.3.1.10
  odpověď: SNMPv2-SMI::enterprises.9.9.46.1.3.1.1.3.1.10 = INTEGER: 1

- *vtpVlanName* - .1.3.6.1.4.1.9.9.46.1.3.1.1.4.1
  význam: jméno VLAN sítě
  typ proměnné: string
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.3.1.1.4.1.10
  odpověď: SNMPv2-SMI::enterprises.9.9.46.1.3.1.1.4.1.10 = STRING: "mgmt"

- *vmVlan* - .1.3.6.1.4.1.9.9.68.1.2.2.1.2
  význam: číslo VLAN sítě, které je použito na rozhraní, pouze v ACCES módu
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.68.1.2.2.1.2.10008
  odpověď: SNMPv2-SMI::enterprises.9.9.68.1.2.2.1.2.10008 = INTEGER: 10
  Port 8 je nakonfigurován na ACCES s VLAN sítí 10.

- *vtpTrunkPortDynamicState* - .1.3.6.1.4.1.9.9.46.1.6.1.1.13
  význam: zjistí, jestli je rozhraní konfigurováno jako TRUNK
  1= ON; 2=OFF; 3=Desirable; 4=AUTO; 5= OnNoNegotiate
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.6.1.1.13.10008
  odpověď: SNMPv2-SMI::enterprises.9.9.46.1.6.1.1.13.10008 = INTEGER: 2

- vtpTrunkPortDynamicStatus - .1.3.6.1.4.1.9.9.46.1.6.1.1.14
  význam: zjistí, jestli je dané rozhraní v TRUNK módu nebo ne, podmínka je aby rozhraní bylo UP; 1=TRUNK; 2=non TRUNK
  typ proměnné: integer
  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.6.1.1.14.10008
  odpověď: SNMPv2-SMI::enterprises.9.9.46.1.6.1.1.14.10008 = INTEGER: 2

Pro další dotaz jsem použil index 10003 pro 3. port, který je konfigurován jako TRUNK, a index 10008 pro 8. port, který je konfigurován jako ACCES. Na 3. portu jsou nastaveny VLAN sítě 10, 17, 20 a 26.
V MIB struktuře se občas vyskytuje proměnná OCTET STRING, což není nic jiného než řetězec bitů, kde každý bit reprezentuje jednu VLAN síť. Do prvního oktetu náleží VLAN sítě od 0 do 7, do druhého oktetu náleží VLAN sítě od 8 do 15 atd. Převod z oktetů na čísla VLAN sítí je naznačen v tab. 1.

Tab. 1.: Grafická ukázka proměnné OCTET STRING

oktety (16)	0				0				0				0				0				0			
VLAN sítě	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
bity (2)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

- *vtpVlanTrunkPortVlansEnabled* - .1.3.6.1.4.1.9.9.46.1.6.1.1.4

  význam: použité VLAN sítě na rozhraní, VLAN sítě od 0 do 1023

  typ proměnné: octet string

  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.6.1.1.4.10003

  odpověď:

  00 20 48 20 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Převod této odpovědi na čísla VLAN sítí nakonfigurovaných na 3. portu je graficky znázorněn v tab. 2.

Tab. 2.: Grafický výpočet použitých VLAN sítí na rozhraní

oktety	0				0				2				0				4				8			
VLAN sítě	0	1	2	3	4	5	6	7	8	9	**10**	11	12	13	14	15	16	**17**	18	19	**20**	21	22	23
bity	0	0	0	0	0	0	0	0	0	0	**1**	0	0	0	0	0	0	**1**	0	0	**1**	0	0	0

  dotaz: snmpwalk -v2c -c public 192.168.0.2 .1.3.6.1.4.1.9.9.46.1.6.1.1.4.10008

  odpověď:

  7F FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

  FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF

Odpověď na druhý dotaz je opět OCTET STRING, avšak tentokrát je prázdný. Nula a tři jedničky na začátku naznačují, že rozhraní nepracuje v TRUNK módu.

- *vtpVlanTrunkPortVlansEnabled2k* .1.3.6.1.4.1.9.9.46.1.6.1.1.17
  význam: použité VLAN sítě na rozhraní, VLAN sítě od 1024 do 2047
  typ proměnné: octet string

- *vtpVlanTrunkPortVlansEnabled3k* .1.3.6.1.4.1.9.9.46.1.6.1.1.18
  význam: použité VLAN sítě na rozhraní, VLAN sítě od 2048 do 3071
  typ proměnné: octet string

- *vtpVlanTrunkPortVlansEnabled4k* .1.3.6.1.4.1.9.9.46.1.6.1.1.19
  význam: použité VLAN sítě na rozhraní, VLAN sítě od 3072 do 4095
  typ proměnné: octet string

5.2 Programové vybavení

Pro aplikaci SNMP protokolu je nejvhodnější programovací jazyk Perl. Perl byl vytvořen Larry Wallem v roce 1987. Tento programovací jazyk má spoustu výhod, které lze využít při tvorbě navrhovaného systému: [2], [6]

- interpretovaný jazyk, není nutno kompilovat
- svobodný software, licencován pod Artistic License nebo GNU GPU (General Public License), lze jej zdarma používat i při komerčních projektech
- dobrá dokumentace díky komunitě, která je okolo Perlu vytvořena
- platformová nezávislost (Linux, Unix, MS Windows, atd.)
- díky rozhraní pro databáze, DBI, podporuje Perl všechny známé databáze, např.: MySQL

Největší výhodou tohoto programovacího jazyka jsou volně dostupné moduly CPAN (Comprehensive Perl Archive Network), které může vyvíjet kdokoli, dle jistého standardu. Proto programovací jazyk zůstává stejný. Vývoj Perlu jde dopředu díky balíčkům a tak si Perl udržuje kompletní funkčnost. Díky této vlastnosti Perlu stačí vyhledat balíček, který umí pracovat s protokolem SNMP. Tento balíček má název *Net:SNMP* a jeho poslední verze je 5.2.0. Modul interpretuje získávání standardních hodnot z MIB protokolu SNMP.[6]

5.3 Databáze

Pro ukládání získaných dat se zdá být nejvhodnější databází MySQL. Databáze byla vytvořena autory Michaelem Wideniusem a Davidem Axmanem. Důvody, proč je databáze vhodná pro navrhovaný systém: [3]

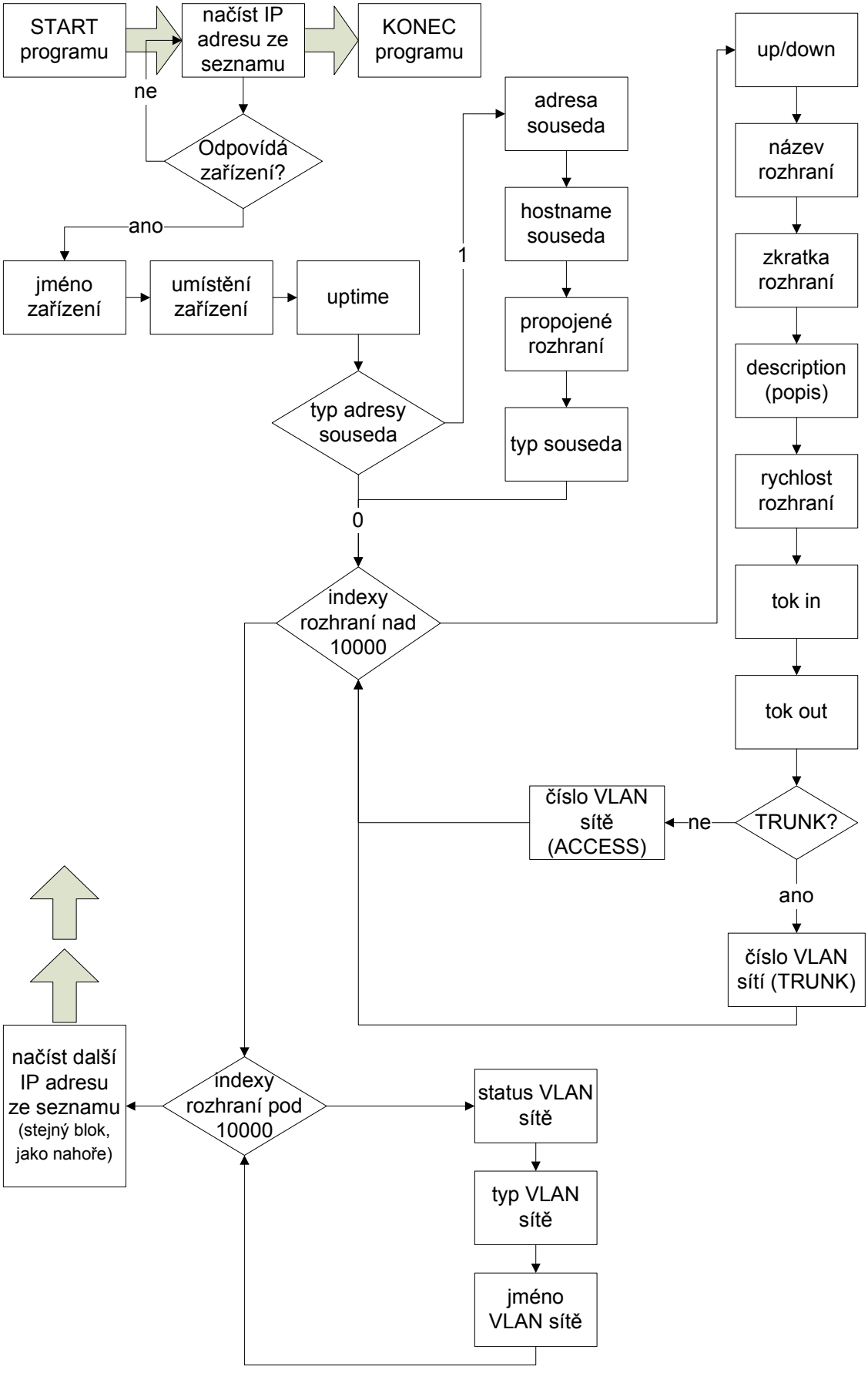
- dostupná bezplatně pod licencí GPL (a též jako komerční placená verze)
- platformová nezávislost (Linux, Unix, MS Windows, atd.)
- rozšířená/používaná databáze
- standardně pracuje s programovacím jazykem Perl

5.4 Seznam zařízení

Další důležitou částí systému je seznam zařízení, kterých se má systém dotazovat a tím od nich získávat požadovaná data. Seznam musí obsahovat IP adresu zařízení a také jeho hostname, tedy název. Takové seznamy jsou pro poskytovatele datových služeb běžné, protože je využívají jiné dohledové systémy a používají se také k evidenci. Seznam zařízení je veden také ve formě databáze, ale na obr. 7 je znázorněn jako *seznam zařízení* jen pro odlišení od výsledné databáze.

5.5 Vývojový diagram systému

Díky spojení programovacího jazyka Perlu, databáze MySQL a operačního systému Linux je samotné programování velice efektivní. V některých případech je však možno narazit na nezdokumentované problémy, které následně značně práci zkomplikují. Aby program nebyl závislý pouze na mé implementaci v jazyce Perl a databázi MySQL, vytvořil jsem univerzální vývojový diagram, který je znázorněn na obr. 8. Vývojový diagram ve spojení s vyjmenovanými/potřebnými MIB objekty a příklady SNMP dotazování se dá implementovat jakkoli. Vývojový diagram lze použít pro vlastní implementaci za předpokladu, že je k dispozici seznam IP adres zařízení a dotazování zařízení bude probíhat postupně.



Obr. 8.: Vývojový diagram systému

6 Testování systému

Pro bezproblémový běh systému je zapotřebí vzít na vědomí několik důležitých faktorů:

- na zařízeních, která mají být dotazována, musí být povoleno SNMP dotazování
- zařízení, která mají být dotazována, musí být ve stejné SNMP komunitě
- i když je SNMP protokol univerzální, systém byl navržen pouze pro zařízení firmy Cisco
- navržený systém zjišťuje údaje i o rozhraních, která nejsou ve stavu up
- navržený systém monitoruje pouze VLAN sítě typu ethernet
- aby rozhraní bylo prohlášeno za TRUNK, musí tak být nakonfigurováno

Systém byl napsán a otestován na notebooku DELL D510 s operačním systémem Linux Debian 5.0. Databáze byla nainstalovaná taktéž na tomto notebooku. Připojení do sítě bylo realizováno přes FastEthernet tedy 100 Mbit/s připojení. Protože dotazy neprobíhají paralelně, není tedy žádný vysoký nárok na hardwarovou vybavenost. Není důvod se domnívat, že by notebook jakkoli systém brzdil. Systém byl od začátku navrhován tak, že poběží na dohledovém serveru pod operačním systémem FreeBSD.

Samotné testování probíhalo ve dvou fázích. První fáze testování probíhala již během programování a druhá fáze testování systému jako celku. Bohužel jsem měl k dispozici omezený počet zařízení. Zahrnutý výstup systému je také z tohoto testování. Jednalo se pouze o router a switch.

Druhé testování probíhalo ve firmě T-Systems Czech Republic na části páteřní sítě. Testování se týkalo privátní sítě a výsledky zahrnovali velice důvěrná data o síti. Aby byla zachována bezpečnost informací, nemohu výstup z tohoto testování publikovat.
Jako přínos je ovšem chování systému v rámci větší sítě. V testované síti se nacházelo 5 směrovačů a 20 přepínačů. I když to již bylo mnoho informací, systém rychle a spolehlivě dokázal data ukládat do databáze. Velice důležité bylo také odmítnutí dotazování VoIP (Voice over Internet Protocol) telefonů, které se v síti také nacházeli.

6.1 Výstup systému

Protože jsem zvolil databázi MySQL pracující v operačním systému Linux, není grafická podoba moc uživatelsky přívětivá. Proto jsem výsledky přepracoval a upravil do lepší grafické podoby. Jedná se vlastně o čtyři navzájem provázané tabulky. Také seznam IP adres dotazovaných zařízení jsem umístil do databáze. Výstup systému je znázorněn v tab. 3 – 8.

Tab. 3.: Seznam dotazovaných zařízení

Number	IP	Name
1	192.168.0.1	router
2	192.168.0.2	switch

Tab. 4.: Výpis z části Device

DeviceID	IP	Name	Location	Uptime
1	192.168.0.1	router	Brno	1:10
2	192.168.0.2	switch	Brno	1:09

Tab. 5.: Výpis z části Link

LinkID	Device1ID	Device2ID	Port1ID	Port2ID
1	1	2	FastEthernet0/1	GigabitEthernet0/1
2	2	1	GigabitEthernet0/1	FastEthernet0/1

Tab. 6.: Výpis z části VLAN

VLANID	PortID	Name	Status	Type
1	2	default	off	ethernet
1	3	default	off	ethernet
10	4	mgmt	on	ethernet
17	4	test	off	ethernet
20	4	VoIP	off	ethernet
1	5	default	off	ethernet
10	6	mgmt	on	ethernet
17	6	test	off	ethernet
20	6	VoIP	off	ethernet
1	7	default	off	ethernet
1	8	default	off	ethernet
10	9	mgmt	on	ethernet
10	10	mgmt	on	ethernet

Tab. 7.: Výpis z části Port – 1. část

PortID	Status	Name	Abbreviation	Description
1	up	FastEthernet0/1	Fa0/1	pripojeni k switchi
2	down	FastEthernet0/1	Fa0/1	
3	down	FastEthernet0/2	Fa0/2	
4	down	FastEthernet0/3	Fa0/3	nakonfigurovano trunk
5	down	FastEthernet0/4	Fa0/4	
6	down	FastEthernet0/5	Fa0/5	nakonfigurovano trunk
7	down	FastEthernet0/6	Fa0/6	
8	down	FastEthernet0/7	Fa0/7	
9	up	FastEthernet0/8	Fa0/8	port pro pc
10	up	GigabitEthernet0/1	Ga0/1	propojeni k routeru

Tab. 8.: Výpis z části Port – 2. část

PortID	Speed	InputRate	OutputRate	Type	DeviceID
1	100000000	58000	62000	ACCESS	1
2	100000000			ACCESS	2
3	100000000			ACCESS	2
4	100000000			TRUNK	2
5	100000000			ACCESS	2
6	100000000			TRUNK	2
7	100000000			ACCESS	2
8	100000000			ACCESS	2
9	100000000	47000	46000	ACCESS	2
10	100000000	55000	56000	ACCESS	2

Je zřejmé, že výstup systému je z databáze. Zde nemohu ukázat jednotlivé návaznosti mezi sebou, avšak názvy jednotlivých sloupců napoví, že jednotlivé tabulky jsou provázané mezi sebou. Na první pohled je v uložených datech relativní chaos. A to jsou dotazovány pouze dvě zařízení. Pokud by switch byl 48 portový, výpis by se již na stránku nevešel. Při druhém testování ve firmě T-Systems Czech Republic byly výpisy z databáze velice obsáhlé. Díky vhodně zvolené databázi nebyl problém se ve výstupu systému zorientovat a výpisy procházet a hledat v nich chyby.

Uživatelsky přívětivý výsledek je však pouze ten grafický. Protože jsem při implementaci systému narazil na problém s ukládáním proměnné typu OCTET STRING do databáze, což jsem nikde nenašel zdokumentováno a tento problém mi vzal více času, než jsem předpokládal, grafický modul jsem nedokončil. Tyto výpisy se za uživatelsky přívětivý výsledek považovat rozhodně nedají.

7 Instalace systému

Jak již bylo zmíněno, program byl napsán a následně otestován na notebooku DELL D510 s procesorem Intel Pentium 1,73GHz, 512MB RAM. Jako operační systém byl zvolen Linux, konkrétně distribuce Debian 5.0 Lenny. Debian byl zvolen s ohledem na dobrou funkčnost s notebooky firmy DELL.

Pro SNMP komunikaci je nezbytný balík Net-SNMP. Tento balík ovšem pro Debian jako samoinstalační balíček neexistuje. Lze jej stáhnout z internetu a podle podrobného návodu (pouze v angličtině) zkompilovat a nainstalovat. [4]

Jako programovací jazyk byl zvolen Perl, je tedy nutné mít nainstalovaný interpret Perlu. V současné době je k dispozici verze 5.10.0-19 jako balíček do Debianu.

Jako databázový model jsem zvolil databázi MySQL. Ze správce balíčků Debianu lze nainstalovat databázový server a klienta MySQL databáze ve verzi 5.0.51a-24. Je možné také nainstalovat grafický nástroj pro snadnější správu databáze. Na přiloženém CD je soubor *databaze*, který vytvoří potřebnou databázovou strukturu v již nainstalovaném a spuštěném MySQL serveru.

Nutností je správná konfigurace počítače do sítě a zajištěná komunikace s dotazovanými zařízeními. V databázové struktuře se také musí upravit seznam IP adres, kterých se má dotazování týkat. Nejlépe do databáze naimportovat.

Pro otestování správné SNMP komunikace, což je nezbytné před samotným spuštěním systému, je dobré použít příklad použitý již dříve a to při výčtu potřebných MIB objektů. Po upravení IP adresy a komunity je možno příkaz zadat rovnou do příkazové konzole. Pokud je dotaz úspěšný, vrátí se odpověď a SNMP komunikace tedy funguje. Nyní je možné spustit z přiloženého CD soubor *monitor* a počkat než systém provede svoji úlohu.

Soubor *monitor* není možné upravovat, protože funguje jako „spustitelný program“ a potřebné moduly má zabaleny v sobě.

8 Závěr

V bakalářské práci měl být nalezen a implementován nejvhodnější způsob pro získávání údajů o aktivních prvcích firmy Cisco a mezi nimi sestavenými VLAN sítěmi. Nejvhodnější způsob pro toto dotazování je využití SNMP protokolu. V práci je rozebíráno jak používat SNMP protokol i jak se dostat s jeho pomocí k MIB objektům uložených v MIB struktuře. V práci jsou detailně vybrány a popsány MIB objekty potřebné pro činnost navrženého systému s důrazem na platformovou nezávislost. Při implementaci navrženého systému a vybrání jednotlivých prostředků, se kterými bude systém pracovat, jsem kladl důraz na volně šiřitelné a dobře zdokumentované nástroje. Dále jsem navrhl běh systému, který je nezávislý na implementaci. V práci jsou také zmíněny některé základní údaje týkajících se zařízení firmy Cisco jako je např. CDP prokotol nebo fungování a nastavování VLAN sítí.

Navržený a implementovaný systém byl otestován a výsledky zahrnuty do práce. Systém se během testování choval dle požadavků zadání. Získal údaje od zařízení, které měl zadané v seznamu, a výsledky byly uloženy do databáze. Výsledky dotazování bylo možno nadále analyzovat a zpracovávat. Výstup systému sice není grafický, jak bylo očekáváno, ale stávající systém fungoval podle předpokladů. Při tvorbě grafického výstupu se objevil problém s umístěním vykreslovaných zařízení. Nepodařilo se mi nalézt optimální řešení pro vykreslení topologie a následné spojení sousedních zařízení. Tento problém vyžaduje velice podrobné prostudování a navržení řešení, které, jak se zdá, nebude jednoduché a přesahuje rozsah bakalářské práce.

Systém je možno dále rozšířit o grafický modul a modul pro dotazování nejen Cisco zařízení. Navržený systém budu nadále testovat a pracovat na jeho vylepšení a vývoji.

Seznam literatury

[1] *Cisco SNMP Object Navigator* [online]. c1992-2008 [cit. 2008-11-18]. Dostupný z WWW: http://tools.cisco.com/Support/SNMP/do/BrowseOID.do?local=en

[2] LEMAY, Laura. *Naučte se Perl za 21 dní.* Bogdan Kiszka, Jaroslav Černý. 1. vyd. Praha: Computer Press, c2002, 546 s, ISBN 80-7226-616-0

[3] *MySQL* [online]. c1995-2008 [cit. 2008-12-18]. Dostupný z WWW: http://www.mysql.com/

[4] *Net-SNMP* [online]. c1992-2007 [cit. 2009-4-11]. Dostupný z WWW: http://www.net-snmp.org/

[5] PUŽMANOVÁ, Rita. Moderní komunikační sítě od A do Z. 2. aktualiz. vyd. Brno: Computer Press, c2006. 430 s. ISBN 80-251-1278-0

[6] *The Perl Directory at Perl.org* [online]. c2002-2008 [cit. 2008-11-18]. Dostupný z WWW: http://www.perl.org/

[7] VELTE, Toby J. - VELTE, Anthony T. *Síťové technologie Cisco: Velký průvodce.* David Krásenský. 1. vyd. Brno: Computer Press, c2003. 800 s. ISBN 80-7226-857-0

[8] VERUŇÁK, Jiří. *Monitorovací systém pro sítě typu Ethernet.* [s. l.], 2006. 63 s. Vedoucí diplomové práce Ing. Martin Biško. ČVUT v Praze Fakulta elektrotechnická

Seznam zkratek

ARPANET	Advanced Research Projects Agency Network
ASN.1	Abstract Syntax Notation One
CDP	Cisco Discovery Protocol
CPAN	Comprehensive Perl Archive Network
DoD	Department of Defense
GPU	General Public License
ICMP	Internet Control Message Protocol
ISO	International Organization for Standardization
MAC	Media Access Control
MIB	Management Information Base
NMS	Network Management Station
RFC	Request For Comments
SMI	Structure od Management Information
SNMP	Simple Network Management Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
UPS	Uninterruptible Power Supply
VLAN	Virtual Local Area Network
VoIP	Voice over Internet Protocol

Seznam příloh

A Obsah CD

- bc_prace (práce v elektronické podobě)
- zadani_prace (zadání práce v elektronické podobě)
- titulni_strana (titulní strana v elektronické podobě)
- databaze (struktura databáze)
- monitor (monitorovací program)
- PICTURES (adresář, kde jsou uloženy obrázky použité v této práci)
- SOURCE_CODE (adresář, kde je uložen zdrojový kód monitorovacího programu)