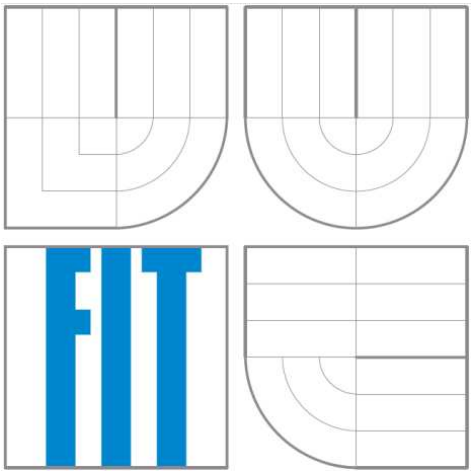


Dokumentace k projektu do předmětu ISA

# Programování síťové služby
# Sniffer CDP a LLDP

21. listopadu 2011

Autor: Radim Loskot, xlosko01@stud.fit.vutbr.cz
Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

# Obsah

# 1 Úvod

Tento dokument analyzuje používané protokoly LLDP a CDP v oblasti počítačových sítí a popisuje návrh a implementaci aplikace, která umožňuje zachytávání nebo generování paketů těchto protokolů.

Jelikož aplikace potřebuje ke svému běhu umět zpracovávat pakety protokolů LLDP a CDP, je nejprve nutné definovat, na jakých vrstvách lze tyto pakety odchytávat či posílat a jakou mají strukturu. Tato problematika je detailně popsána hned v následující kapitole 2 Analýza problému a princip jeho řešení.

V dalších kapitolách je uveden nejprve návrh aplikace (kap. 3 Návrh řešení problému), její konečná implementace (kap. 4 Popis řešení) a popis ovládání (kap. 5 Návod na použití programu).

# 2 Analýza problému a princip jeho řešení

V této kapitole je uvedena specifikace řešeného problému a popsána struktura paketů protokolů LLDP a CDP.

Oba protokoly jsou si funkčně podobné a slouží pro zasílání zpráv o službách na zařízení, které pakety zasílá, nebo o ostatních zařízeních, které jsou na LAN síti.

## 2.1 Zadání problému

Cílem projektu je vytvořit na síťovém rozhraní program, který umožní zpracovávat data protokolů CDP a LLDP. Aplikace má pracovat ve dvou režimech, kdy bude aplikace buď odposlouchávat pakety (režim listener) nebo pakety odesílat (režim sender). Dále je specifikována možnost uživatelské definice pro režim sender, kdy může uživatel zvolit interval odesílání paketů nebo dobu, po kterou je daný paket platný (Time To Live). Program v režimu sender je funkčně závislý na povoleném IPv4 nebo IPv6 forwardingu, v opačném případě bude program ukončen.

## 2.2 Protokol LLDP

Protokol LLDP [2] je protokolem 2. vrstvy (linkové) ISO/OSI modelu. Používá se především s Ethernetovou (Ethernet II) linkovou vrstvou., která slouží i jako transportní protokol – lze v ní přímo specifikovat jaký protokol je přenášen Ethernetovým rámcem položkou etherType v hlavičce Ethernetového rámce.

Pro ostatní protokoly z rodiny Ethernetu (IEEE 802.3), které položku etherType nemají, nebo ostatní linkové vrstvy sítí IEEE 802 je nutné použít mezivrstvu protokolu LLC se SNAP hlavičkou [3]. My se ale v rámci aplikace omezíme pouze na linkovou vrstvu Ethernetu II.

Jelikož předpokládáme odesílání paketů LLDP pouze na linkové vrstvě Ethernetu II, bude našemu paketu předcházet Ethernetová hlavička. Pro tuto hlavičku platí určitá pravidla, která zohledníme při odchytávání paketu a nebo která je nutné dodržet při generování paketu a jeho následném odesílání. (Obrázek 1)

Ethernetová hlavička LLDP paketu obsahuje jako cílovou MAC adresu speciální multicastovou adresu, která není přeposílána MAC mosty. Nejvíce používanou cílovou adresou je 01-80-C2-00-00-0E. Dále hlavička obsahuje MAC adresu odesílatele a položku typu Ethernetu, která pro LLDP pakety vždy nabývá hodnoty 88-CC.

| Cílová MAC | Zdrojová MAC | Typ Ethernetu | Data + zarovnání | Kontrolní součet |
|---|---|---|---|---|
| LLDP_Multicast address | MAC address | 88-CC | LLDPDU | FCS |
| 6 oktetů | 6 oktetů | 2 oktety | 1500 oktetů | 4 oktety |

Obrázek 1 Struktura paketu LLDP na linkové vrstvě Ethernetu

Za Ethernetovou hlavičkou následuje ihned datová jednotka (data) LLDP. Datová jednotka je složena z posloupnosti TLV (Type-Length-Value) záznamů (Obrázek 2), z nichž některé jsou povinné a musí být v datové jednotce vždy uvedeny. Jako poslední TLV záznam je záznam, který značí ukončení datové jednotky a musí být vždy přiložen na konci.

| **Chassis ID TLV** | **Port ID TLV** | **Time To Live TLV** | Optional TLV | ... | Optional TLV | **End Of LLDPDU TLV** |
|---|---|---|---|---|---|---|

Obrázek 2 Datová jednotka LLPD (LLDPDU)

Základní záznam TLV je určen třemi položkami (Obrázek 3). Položkám typu TLV a počtu znaků přenášené informace lze též říkat hlavička TLV, neboť jsou obsaženy v každém záznamu TLV. Zato přenášená informace může být i nulové délky. Je tomu tak např. u TLV záznamu signalizujícího konec posloupnosti TLV záznamů.

| **Typ TLV záznamu** | **Počet znaků informace** | Přenášená informace |
|---|---|---|
| **7 bitů** | **9 bitů** | 0 ≤ n ≤ 511 oktetů |

Obrázek 3 Struktura TLV záznamu LLDP paketu

Jednotlivé typy TLV a jejich nosná informace jsou blíže popsány v literatuře [1].

## 2.3 Protokol CDP

Protokol CDP [4] je taktéž protokolem 2. vrstvy ISO/OSI modelu. Nejvíce se používá s Ethernetem IEEE 802.3. V sítích IEEE 802 je závislý na mezivrstvě LLC se SNAP hlavičkou [3]. Lze však použít i na linkových vrstvách Cisco HDLC nebo PPP [5].

My se opět zaměříme na nejpoužívanější typ přenosu CDP paketu na sítích IEEE 802, konkrétně s využitím Ethernetu IEEE 802.3 a LLC mezivrstvy. (Obrázek 4)

| **Cílová MAC** | **Zdroj. MAC** | **Délka** | **DSAP** | **SSAP** | **Control** | **OUI** | **Typ Ethernetu** |
|---|---|---|---|---|---|---|---|
| **01-00-0C-CC-CC-CC** | **MAC address** | **length** | **AA** | **AA** | **03** | **00000C** | **2000** |
| **6 oktetů** | **6 oktetů** | **2 oktety** | **1 oktet** | **1 oktet** | **1 oktet** | **3 oktety** | **2 oktety** |
| **IEEE 802.3** | | | | | **IEEE 802.2 LLC a SNAP** | | |

Obrázek 4 Transportní protokoly protokolu CDP

Všechny konstantní údaje, které jsou obsaženy protokolech nám poslouží jako filtr pro zachytávání CDP paketů. Za hlavičkou LLC protokolu již následuje paket CDP.

Na rozdíl od LLDP paketu CDP paket obsahuje hlavičku, kde je uvedena verze protokolu, doba platnosti paketu a kontrolní součet celého paketu. (Obrázek 5)

| **Verze CDP** | **Doba platnosti** | **Kontrolní součet** | **Záznamy TLV** |
|---|---|---|---|
| **1 oktet** | **1 oktet** | **2 oktety** | |

Obrázek 5 Struktura CDP paketu

Základní záznam TLV CDP paketu je opět určen třemi položkami (Obrázek 6). Položky CDP jsou ovšem rozdílných datových velikostí, než tomu bylo u LLDP paketu.

| **Typ TLV záznamu** | **Počet znaků informace** | Přenášená informace |
|---|---|---|
| **2 oktety** | **2 oktety** | 0 ≤ n ≤ 1490 oktetů |

Obrázek 6 Struktura TLV záznamu CDP paketu

Jednotlivé typy TLV a jejich nosná informace jsou blíže popsány v literatuře [4].

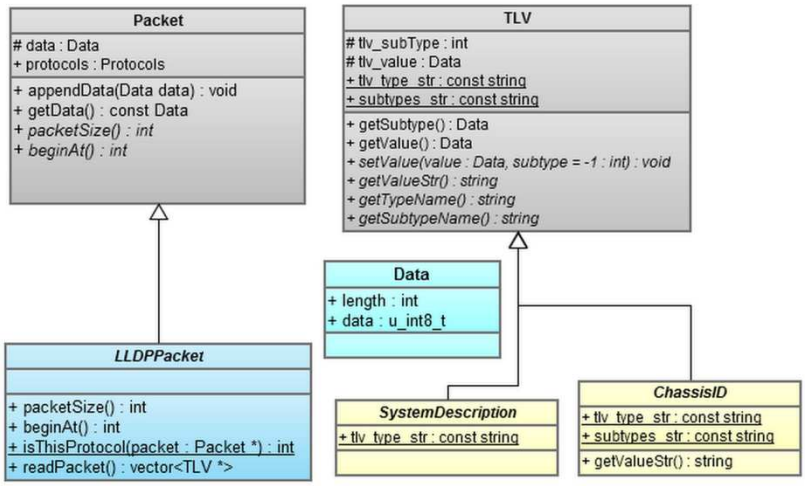
# 3 Návrh řešení problému

V předchozí kapitole jsme definovali zadání problému a strukturu jednotlivých paketů. Nyní navrhneme zjednodušenou třídní koncepci, jak lze danou aplikaci implementovat.

## 3.1 Pakety s TLV záznamy

Struktura TLV záznamů protokolu LLDP a CDP se ve svém základu neliší, proto můžeme použít obecnou `TLV` třídu (Obrázek 1) pro zachycení většiny případů užití pro TLV záznamy obou protokolů.

Základním prvkem třídy `TLV` je atribut typu TLV záznamu `tlv_type` a atribut hodnoty a délky `tlv_value`. Za povšimnutí stojí i atribut podtypu, který může být také užitečný zejména v případě LLDP záznamů. Další atributy slouží už pro generování formátovaného výstupu tj. atributy názvů typů `tlv_type_str` a podtypů `subypes_str`. Přístup k atributům je prováděn přes implementované metody.




Obrázek 7 Třídy TLV záznamů a protokolových paketů

Další základní třídou je třída `Packet`, která zaobalí data paketu a umožní nad nimi základní operace, jako je zjištění, kde daný paket začíná v rámci rámce `beginAt()` nebo jak je velký `getSize()`. Informaci o jednotlivých vrstvách ponese atribut `protocols`.

V derivovaných třídách bude potřeba kromě předefinování některých metod přidat i nové metody např. metody čtení hlavičky paketu či získání pole záznamů TLV `readPacket()`. Důležitou funkcí bude plnit i metoda validace, zda paket odpovídá protokolu a je vložen na správných vrstvách `isThisProtocol()`.

## 3.2 Hlavní program a třída Sniffers

Z hlavního programu `CDP & LLDP Sniffer` (Obrázek 8) budou volány metody pro spuštění odposlouchávání nebo odesílání paketů a to prostřednictvím instance třídy `Sniffers`.

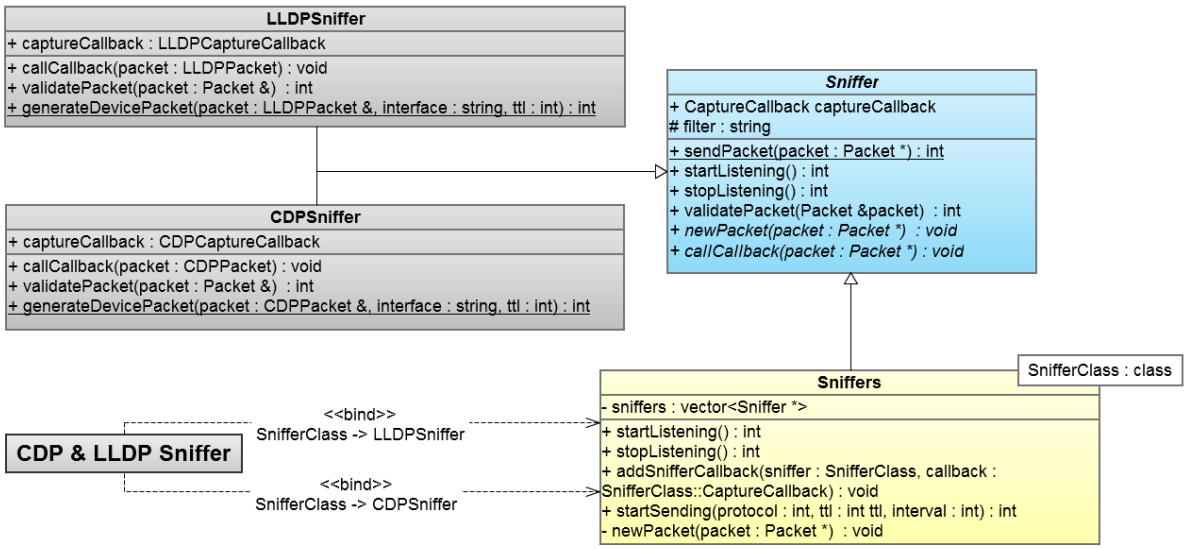
Pro odposlech bude nejprve nutné provést registraci odposlechu určitého protokolu. K tomu bude sloužit metoda `addSnifferCallback()`, která uloží novou instanci `LLDPSniffer` nebo `CDPSniffer` do pole `sniffers`. V instanci taktéž nastaví callback funkci pro zpracování zachyceného paketu.

Jelikož callback funkce předávaná v registrační metodě bude pokaždé jiná, je tato metoda šablonová. Na rozdíl od metody odesílání paketů `startSending()`.

V režimu listener bude po registraci všech odposlouchávaných paketů zavolána metoda `startListening()`, která projde pole zaregistrovaných snifferů a složí výsledný filtr z filtrů, které má každý sniffer ve své struktuře staticky definované.

Při zachycení nového paketu tzn. zavolání virtuální metody `newPacket()`, bude nutné opět projít celé pole `sniffers` a postupně ověřovat zda došel paket, který je pojen s daným snifferem pomocí metody `validatePacket()`. Pokud tomu tak bude, bude zavolána callback funkce.

Poslední základní metodou z hlediska funkčnosti programu je metoda pro generování paketů v režimu sender `generateDevicePacket()`.




Obrázek 8 Popis třídy Sniffers a jednotlivých sniferrů

# 4 Popis řešení

Implementace aplikace byla provedena v jazyce C++ s využitím standardních knihoven a knihovny pcap pro zachytávání a odesílání paketů.

Aplikace byla implementována podle návrhu uvedeného v předchozí kapitole, takže k podrobnostem reprezentace tříd se již nebudeme vracet a popíšeme si spíše jakých funkcí knihovny pcap bylo použito a co, jak a v jakém rozsahu bylo nakonec implementováno.

## 4.1 Hlavní modul

Hlavní modul (obsahující funkci `main`) aplikace se nachází v adresáři "src" a v souboru nazvaném "cdp_lldp_sniffer.cpp".

V `main` funkci je volána nejprve funkce pro zpracování parametrů `getFlags()`, která vrací asociativní pole přepínačů s přiřazenými argumenty. Toto pole je poté předáno funkci `runSniffer()`, která už zajistí předání parametrů instanci třídy `Sniffers` a běh CDP a LLDP snifferu.

V hlavní modulu se také nacházejí callback funkce, které jsou volány při příchodu nového paketu `callback_LLDPPacket()` a `callback_CDPPacket()`. Tyto funkce zajišťují výpis zachycených paketů na standardní výstup.

## 4.2 Režim listener

Pro odposlech paketů muselo být využito výhradně funkce `pcap_next_ex()`. Ostatní funkce knihovny pcap požadují pro obsloužení zachycení paketu uvedení callback funkce. V C++ ovšem nelze definovat callback na metodu některého objektu.

Výstup zachycení paketu v listeneru má následující podobu:

```
-------------------------------------------------------------------------------
 Captured packet: <int> ([LLDP|CDP] packet)
-------------------------------------------------------------------------------
<TLV STRUCTURES>
      <tlv type1>: <tlv value1>
      <tlv type2>: <tlv value2>
      ...
      <tlv typeN>: <tlv valueN>
```

Nejprve je tištěna informace o typu příchozího paketu a poté následuje buď ihned skupina TLV záznamů `<TLV STRUCTURE>` v případě LLDP paketu a nebo skupina hlavičky `<HEADER>` v případě CDP paketu a až poté skupina TLV záznamů.

Ukázkový výstup listeneru je zobrazen v příloze A Ukázka výstupu listeneru a seznam podporovaných typů a podtypů TLV záznamů je uveden v příloze B Podporované typy TLV záznamů.

## 4.3 Režim sender

K odesílání paketů na síťové rozhraní byla použita funkce `pcap_inject()`, která přebírá celý odesílaný rámec vytvořený dle definovaných kritérií v kapitolách 2.2 Protokol LLDP a 2.3 Protokol CDP.

Abychom mohli Ethernetový rámec odeslat, musíme nejprve zjistit MAC adresu zdrojového zařízení. Pro zjištění zdrojové MAC adresy v Linuxu je nejprve otevřen na zařízení socket funkcí `socket()` a poté funkcí `ioctl()` zjištěna MAC adresa zařízení. Naproti tomu ve FreeBSD je využito funkce `getifaddrs()`, která vrací seznam všech dostupných zařízení. Z tohoto seznamu je následně vybráno naše hledané zařízení a navrácena jeho MAC adresa.

Položky, které jsou odesílány v jednotlivých paketech, jsou uvedeny v tabulce níže.

| LLDP paket | CDP paket |
|---|---|
| Chassis ID | Device ID |
| Port ID | Software version |
| TTL | Platform |
| System name | Port ID |
| System description | Capabilities |
| System capabilities | |

Tabulka 1 Zasílané TLV typy

## 4.4 Testování

K testování správnosti příjmu paketů bylo využito aplikace ladvd (pro protokoly CDP a LLDP) a aplikace mz (pro protokol CDP). Tyto aplikace umožnily generování paketů a v případě aplikace mz i definovaní vlastních položek v paketu CDP.

Díky knihovně šlo realizovat i offline testování načtením souboru se zachyceným síťovým provozem. Testy byly provedeny nad ukázkovými soubory z dokumentace Wiresharku [5][6].

Správnost odesílání paketů byla ověřena aplikací Wireshark.

# 5 Návod na použití programu

## 5.1 Překlad programu

Pro překlad aplikace je zapotřebí mít nainstalovaný GNU make. GNU makefile pro překlad aplikace je umístěn v souboru "Makefile.am". Pokud je GNU make asociovaný s příkazem `make`, lze překlad provést následujícím způsobem:

```
$ make -f Makefile.am
```

Pro zjednodušení překladu byl vytvořen univerzální makefile, který dokáže zpracovat jak GNU make, tak i FreeBSD make. V tomto univerzálním makefilu je volán skript "run_make.sh", jenž v závislosti na operačním systému spouští GNU make. Předpokládá se, že v distribucích Linuxu je GNU make asociovaný přímo s příkazem `make` a na FreeBSD s příkazem `gmake`. Ve výsledku je tedy překladu dosaženo pouhým příkazem `make`.

## 5.2 Použití programu

Pro běh programu musí být vždy uveden režim, v jakém má program běžet (přepínač `-s` nebo `-l`) a rozhraní na jakém má docházet k odposlouchávání či odesílání paketů (přepínač `-i`).

**Popis parametrů s jakými lze program spustit:**

```
Použití:
      xlosko01 [-l|-s] -i <rozhraní> [-c] [-t <int>] [-r <int>]

Přepínače:
      -i název rozhraní
      -s režim zasílání paketů (bez přepínače -c zasílání LLDP paketů)
      -l režim naslouchání na rozhraní
      -c zasílání CDP paketů
      -t doba běhu programu v režimu zasílání paketů (v sekundách)
      -r interval odesílání paketů (v sekundách)
```

# 6 Závěr

Během řešení projektu jsem se snažil splnit veškeré požadavky plynoucí ze specifikace projektu. Při vypracování se nevyskytlo příliš mnoho závažnějších problémů, které by bylo nutné řešit. Bylo ovšem potřeba provést lokalizaci aplikace pro platformu FreeBSD, neboť aplikace byla původně vyvíjena v prostředí Linuxu.

Aplikaci lze stále v mnoha ohledech rozvíjet. Nyní sice je podporováno zobrazení většiny typů a podtypů, které jsou uvedeny ve standardech protokolů, ale ne všechny jejich hodnoty jsou plnohodnotně formátovány na text. Dále existuje celá řada TLV záznamů, které ovšem nejsou zdokumentovány, ale běžně se používají.

Aplikace byla testována na operačním systému Linux a FreeBSD.

## Literatura


[1] *Station and Media Access Control Connectivity Discovery* [online]. [cit. 2011-11-23]. Dostupný z WWW: <http://standards.ieee.org/getieee802/download/802.1AB-2005.pdf>

[2] *Address Family Numbers* [online]. [cit. 2011-11-23]. Dostupný z WWW: < http://www.iana.org/assignments/address-family-numbers/address-family-numbers.xml >

[3] *IIE 802.2 LLC* [online]. [cit. 2011-11-25]. Dostupný z WWW: <http://ckp.made-it.com/ieee8022.html >

[4] *Frame Formats : CDP Packet Format* [online]. [cit. 2011-11-25]. Dostupný z WWW: <http://www.cisco.com/univercd/cc/td/doc/product/lan/trsrb/frames.htm#xtocid12 >

[5] *Cisco Discovery Protocol (CDP)* [online]. [cit. 2011-11-25]. Dostupný z WWW: <http://wiki.wireshark.org/CDP>

[6] *Link Layer Discovery Protocol (LLDP)* [online]. [cit. 2011-11-25]. Dostupný z WWW: < wiki.wireshark.org/LinkLayerDiscoveryProtocol>

# Přílohy

## A. Ukázka výstupu listeneru

```
------------------------------------------------------------------------------
 Captured packet: 0 (LLDP packet)
------------------------------------------------------------------------------
<TLV STRUCTURES>
       Chassis ID (MAC address): 00:01:30:f9:ad:a0
       Port ID (Interface name): 1/1
       Time To Live: 120 s
       Port description: Summit300-48-Port 1001
       System name: Summit300-48
       System description: Summit300-48 - Version 7.4e.1 (Build 5) by Release_Master
05/27/05 04:53:11
       System capabilities: MAC bridge, router (enabled: MAC bridge, router)
       Management address (all 802): 00:01:30:f9:ad:a0 (ifIndex: 1001)

------------------------------------------------------------------------------
 Captured packet: 1 (CDP packet)
------------------------------------------------------------------------------
<HEADER>
       Version: 2
       Time To Live: 180 s
       Checksum: 0x7f3d [OK]

<TLV STRUCTURES>
       Software version: BCM1100
       Device ID: 0060B9C14027
       Capabilities: host
       Platform: BCM91100

------------------------------------------------------------------------------
 Captured packet: 2 (CDP packet)
------------------------------------------------------------------------------
<HEADER>
       Version: 2
       Time To Live: 180 s
       Checksum: 0x1827 [OK]

<TLV STRUCTURES>
       Device ID: sw1B1
       Software version: Cisco IOS Software, CBS31X0 Software (CBS31X0-UNIVERSALK9-
M), Version 12.2(58)SE1, RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2011 by Cisco Systems, Inc.
Compiled Thu 05-May-11 04:08 by prod_rel_team
       Platform: cisco WS-CBS3130G-S-F
       Addresses: 1 address(es) (10.0.161.13)
       Port ID: GigabitEthernet1/0/19
       Capabilities: switch, IGMP capable
       Duplex: full

==============================================================================
Captured packets: 3
Processed bytes [B]: 807
```

# B. Podporované typy TLV záznamů

## Podporované TLV typy a podtypy LLDP listeneru

| Č. typu | Název | Rozpoznání podtypů[1] | Formátování hodnoty[2] |
|---|---|---|---|
| **0** | End of LLDPDU | (nemá podtypy ani hodnotu) | |
| **1** | Chassis ID | všechny podle standardu [1] | MAC address, Interface name, Interface alias |
| **2** | Port ID | všechny podle standardu [1] | MAC address, Interface name, Interface alias |
| **3** | Time To Live | (nemá podtypy) | (ano) |
| **4** | Port Description | (nemá podtypy) | (ano) |
| **5** | System Name | (nemá podtypy) | (ano) |
| **6** | System Description | (nemá podtypy) | (ano) |
| **7** | System Capabilities | rozsah dle standardu [1] | rozsah dle standardu [1] |
| **8** | Management address | Management address podtypy: od hodnot 0x00 do 0x10 [2]<br>Intarface numbering podtypy: všechny podle standardu [1]<br>OID není implementováno | Management address podtypy [2]: IP, 802<br>Interface numbering podtypy: vše dle [1] |

## Podporované TLV typy a podtypy CDP listeneru

| Č. typu | Název | Rozpoznání podtypů[1] | Formátování hodnoty[2] |
|---|---|---|---|
| **1** | Device ID | (nemá podtypy) | (ano) |
| **2** | Addresses | Protocol type: NLPID a 802.2<br>Protocol: IP | Protocol type: NLPID<br>Protocol: IP |
| **3** | Port ID | (nemá podtypy) | (ano) |
| **4** | Capabilities | Router, Transparent Bridge, Source Route Bridge, Switch L2, Host, IGMP Capable, Repeater | (ano pro rozpoznané podtypy) |
| **5** | Software Version | (nemá podtypy) | (ano) |
| **6** | Platform | (nemá podtypy) | (ano) |
| **7** | Duplex | (všechny) | (ano) |
| **8** | MTU | (nemá podtypy) | (ano) |
| **9** | System Name | (nemá podtypy) | (ano) |

[1] Ve sloupci rozpoznání podtypů, je uvedeno pro jaké podtypy budou zobrazeny jejich odpovídající názvy ve výpisu listeneru. Pokud není podtyp rozpoznán bude zobrazeno "unknown".
[2] Ve sloupci formátování hodnoty, je uvedeno jaké hodnoty budou převedeny do textové podoby. Pokud má typ více podtypů, je uvedeno pro jaké podtypy proběhne konverze na text. Když nelze zobrazit hodnotu jako text, je zobrazena v posloupnosti hexadecimálních číslic ve formátu "unknown (hex) - 0a 00 a1 0d ".