Západočeská univerzita v Plzni – Katedra informatiky a výpočetní techniky

Semestrální práce z předmětu KIV/OS

Přidání ovladače aktivních sítí do protokolového zásobníku v systému Windows

Petr Jaroš
Václav Papež
28. 11. 2008

Obsah

1 Zadaný úkol

Zadaný úkol se skládá z několika částí. První nezbytnou částí je proniknutí do problematiky aktivních sítí, konkrétně do implementace SAN [1]. Hlavní je pochopení jejich principu na takové úrovni, abychom s nimi mohli dále pracovat. Druhým bodem je vytvoření ovladače pro vzorový síťový protokol. Tento ovladač musí obsahovat službu, která zajišťuje komunikaci klienta skrze vzorový protokol. Tento vzorový protokol bude posléze vyměněn za protokol komunikující pomocí SAN. Ovladač musí být kompatibilní se systémem Windows Vista. K ucelení tohoto bodu zadání je ještě třeba vytvořit INF file, neboli soubor s informacemi o ovladači. Na základě těchto informací bude ovladač nainstalován a služba jenž poskytuje bude zavedena. Poslední částí zadání je vytvoření programátorské dokumentace, která bude sloužit jako návod pro budoucí programátory, kteří se budou zabývat konkrétním zavedením protokolu SAN do systému.

2 Dvě možná řešení

2.1 Obecný úvod

Hlavní důvodem této práce, je nemožnost přímého nasazení ovladače. To plyne z toho, že zatím neexistuje specializovaný hardware a my se snažíme docílit toho, aby pomocí SAN mohl komunikovat i software, který k tomu není přímo uzpůsobený. !!!!!!Je proto těmto potřebám nutno přizpůsobit službu nad komunikačním protokolem, kterou bude poskytovat systém. Při zkoumání tohoto problému jsme přišli na řešení. To se ale ukázalo postupem času za sice použitelné, ale nedostačující. Zkoumání odstranění nedostatků dalo za vznik druhému řešení.

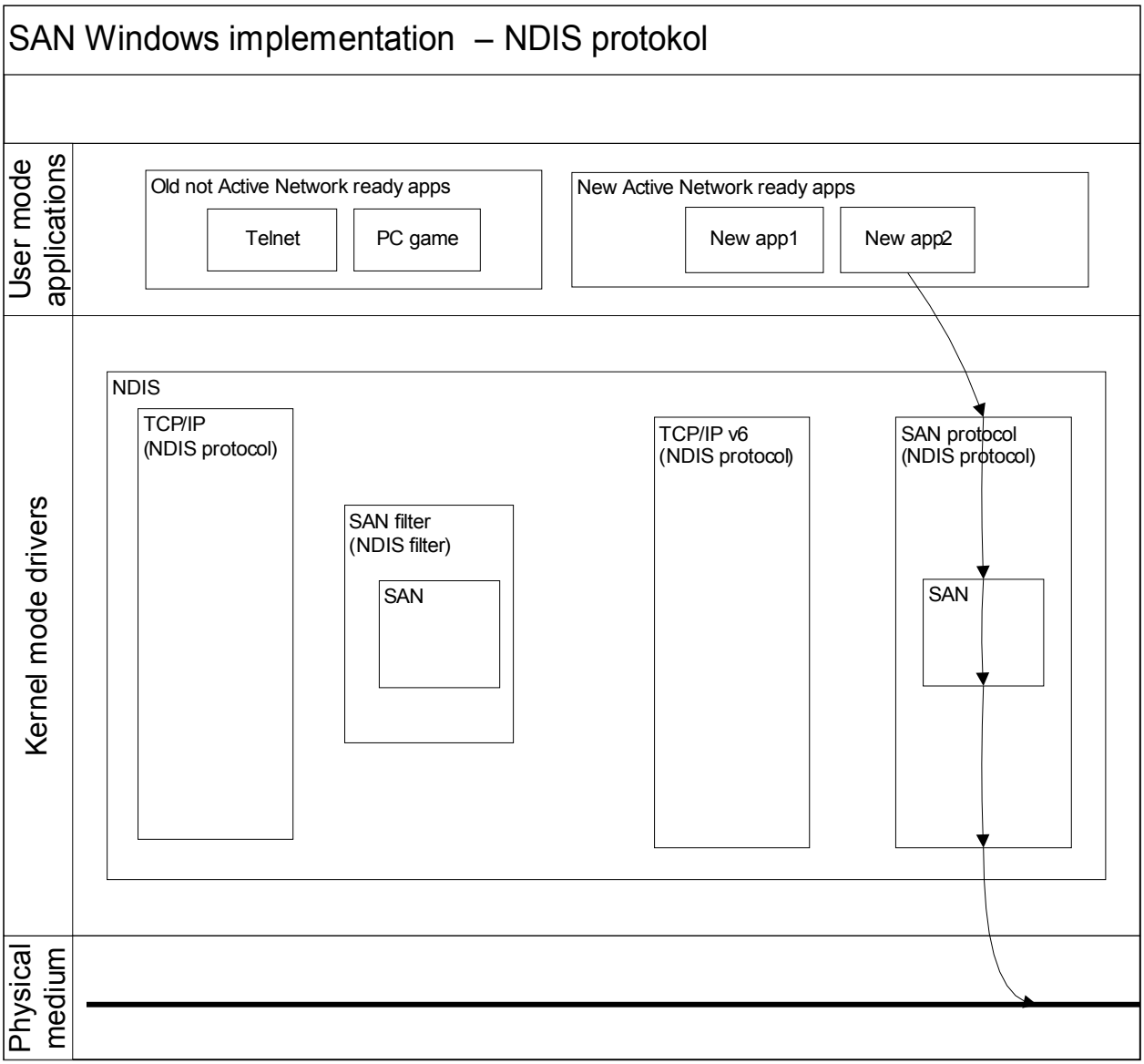
2.2 Řešení zavedením nového ovladače protokolu

2.2.1 Princip řešení

Řešení je vhodné v případě, že všechny uzly v síti (včetně hardwaru) mají nainstalovaný ovladač protokolu aktivních sítí. Po síti jsou posílány SAN capsule, kterým stávající TCP/IP hardware nerozumí. Řešení je tedy momentálně vhodné pro testování v izolované síti a pro budoucí použití, kdy bude aktivní sítě podporovat i hardware. Samotné řešení umožňuje připojit pouze nové aplikace, které jsou navrženy s ohledem pro využití aktivních sítí. Pro posílání dat ze stávajících aplikací (jako např.: telnet) je nutné řešení zkombinovat s filtrem paketů (viz následující kapitoly).

Řešení je založeno na vývoji ovladače protokolu aktivních sítí, který bude přidán do protokolového

zásobníku. Aplikace tedy budou mít na výběr, zda použít stávající protokoly (jako např.: TCP/IP, TCP/IPv6) anebo využít ovladač aktivních sítí. Na obrázku je naznačeno posílání dat, přijímání dat je analogické. Aplikace *New app2* otevře speciální virtuální soubor, který určuje, že se má použít právě SAN protokol, do kterého standardně zapíše data k poslání. SAN protokol data přijme, zabalí je do SAN capsule a pošle data na síť.



2.2.2 Popis programu

Před čtením tohoto textu je nejprve vhodné se seznámit s kapitolou Softwarové vybavení pro vývoj.

Ovladač síťového protokolu je ve Windows dle standardu *NDIS*. My budeme používat *NDIS* verze 6.0, která je dostupná pouze ve Windows Vista a Windows Server 2008, což ovšem vyhovuje

zadání. Pro starší Windows je ale řešení také možné. Při popisu *NDIS* filtru budeme vycházet ze vzorového příkladu, který je dodáván s vývojovým prostředím *WDK* (viz kapitola *Windows Driver Kit*). Příklad je umístěn ve složce *src\network\ndis\ndisprot\6.0.* Zde se nacházejí dvě složky. Ve složce *sys* se nachází samotný ovladač vzorového protokolu a ve složce *test* se nachází konzolová aplikace, sloužící k otestování funkčnosti ovladače. Obě složky obsahují zdrojové soubory v jazyce C.

Seznam nejdůležitějších souborů

ndisp.c - Obsahuje vstupní metodu ovladače, metody pro zaregistrování, odregistrování a obsluhu *IOCtl (Input Output Controls)* určené pro konfiguraci ovladače.

ndisbind.c - Obsahuje obslužné metody zodpovědné za připojování se a odpojování se od adaptérů

send.c - Obsahuje obslužné metody pro odesílání dat a rušení odesílání

recv.c - Obsahuje obslužné metody pro příjem dat

debug.c - Obsahuje debugovací funkce

ndisprot.inf - Konfigurační soubor ovladače, více v další kapitole.

Popis nejdůležitějších metod

`DriverEntry`

Vstupní bod ovladače. Tato metoda je to první, co se spustí při zavádění ovladače.

Hlavní úkol:

- vytvoří objekt ovladače
- nastavuje do datové struktury vlastnosti charakterizující ovladač, jako je název, verze *NDIS*, verze ovladače.
- do stejné datové struktury nastavuje ukazatele k obslužným metodám, které jsou pak volány při vyvolání události (jako je např. otevření kanálu pro zápis do protokolu)
- zaregistruje ovladač protokolu do systému.
- nastavuje obslužné metody spojené s čtením a zapisováním do souboru, který představuje datový tok mezi uživatelskou aplikací a ovladačem protokolu.

`NdisprotWrite`

Metoda pro obsluhu události zápisu, ve které dojde k příjmu dat od uživatelské aplikace a zabalení do hlaviček protokolu.

Hlavní úkol:

- Kontroluje správnost příchozích dat ze struktury *IRP*
- Vytváří strukturu *NET_BUFFER_LIST*
- Do *NET_BUFFER_LIST_CONTEXT* v *NET_BUFFER_LIST* struktuře si ukládá ukazatele aktuálně vytvořených objektů pro možné další použití v jiných funkcích.
- Ukládá data k odeslání do *NET_BUFFER_LIST* struktury
- Ukládá strukturu *NET_BUFFER_LIST* do fronty k odeslání

Zapisování pomocí uživatelské aplikace probíhá pomocí `WriteFile`.

`NdisprotRead`

Metoda pro obsluhu události čtení, která oddělí v příchozím paketu data od hlavičky, zachová se podle údajů v hlavičce.

Hlavní úkol:

- Validuje správnost příchozího paketu
- Uloží data do zásobníku, kde si jej můžu přečíst uživatelská aplikace

Čtení z uživatelské aplikace probíhá pomocí `ReadFile`.

`NdisprotIoControl`

Metoda, která obsluhuje kontrolní požadavky na ovladač.

Hlavní úkol:

- Zjistí o jaký kontrolní požadavek se jedná a podle toho spustí příslušnou funkci

Kontrolní požadavky na ovladač se zadávají pomocí metody `DeviceIoControl`.

Debugovací výpisy

Jak odchytávat debugovací výpisy je popsáno v kapitole DBGView. Uvedený driver má ale nastaveno, jak závažné chyby/upozornění se budou vypisovat. Pokud si chcete nechat vypisovat všechny debugovací výpisy, je nutné změnit hodnotu *ndisprotDebugLevel* na začátku souboru *debug.c* na *DL_EXTRA_LOUD*. Další konstanty pro vyplnění do proměnné *ndisprotDebugLevel* jsou definovány v souboru *debug.h.*

2.2.3 Potřebné nastavení k instalaci

K řádné instalaci je nutné mít k ovladači nastavovací soubor – *.inf (Setup Instalation File)*. O tom jak tyto instalační soubory fungují a co je v nich obecně třeba bude rozebráno dále v kapitole 3 Instalační soubory. Zde je pouze krátké shrnutí konkrétních direktiv, jejichž uvedení je nezbytné.

Nebude-li uvedeno jinak, platí to samé i pro druhé řešení. Z každé popisované sekce budou vždy vybrány jen důležité direktivy. Celý vzorový soubor je součástí přílohy.

Sekce s informacemi o verzi:

```
[version]
Signature = "$Windows NT$"      ;Ovladač je určený pro systém Windows založený na NT (NT, Server, XP, Vista)
Class = NetTrans                ;Třída do které ovladač spadá, NetTrans - komunikační protokoly
ClassGUID = {4d36e975-e325-11ce-bfc1-08002be10318}
                                ;ID třídy - viz. http://msdn.microsoft.com/en-us/library/ms791134.aspx
```

Sekce s údaji o přidávaném klíči:

```
[AddReg]
HKR,Ndi,Service,,"Ndisprot"                                 ;Informace, že se jedná o servis Ndisprot
HKR,Ndi\Interfaces, UpperRange,, noupper                    ;Spojení s nadřazeným rozhraním
HKR,"Ndi\Interfaces","LowerRange",,"ndis5,ndis4,ndis5_prot" ;Spojení s rozhraním o úroveň níže
```

Sekce s údaji o chování servisu:

```
ServiceType = 1 ;Typ servisu SERVICE_KERNEL_DRIVER (běží v jádře) viz. http://msdn.microsoft.com/en-us/library/ms794559.aspx
StartType = 1   ;Typ spouštění SERVICE_SYSTEM_START (při startu systému) viz. http://msdn.microsoft.com/en-us/library/ms794559.aspx
ErrorControl = 1 ;Reakce na chyby SERVICE_ERROR_NORMAL (Servis spadne, ale systém nastartuje a uživatele upozorní) viz. http://msdn.microsoft.com/en-us/library/ms794559.aspx
```

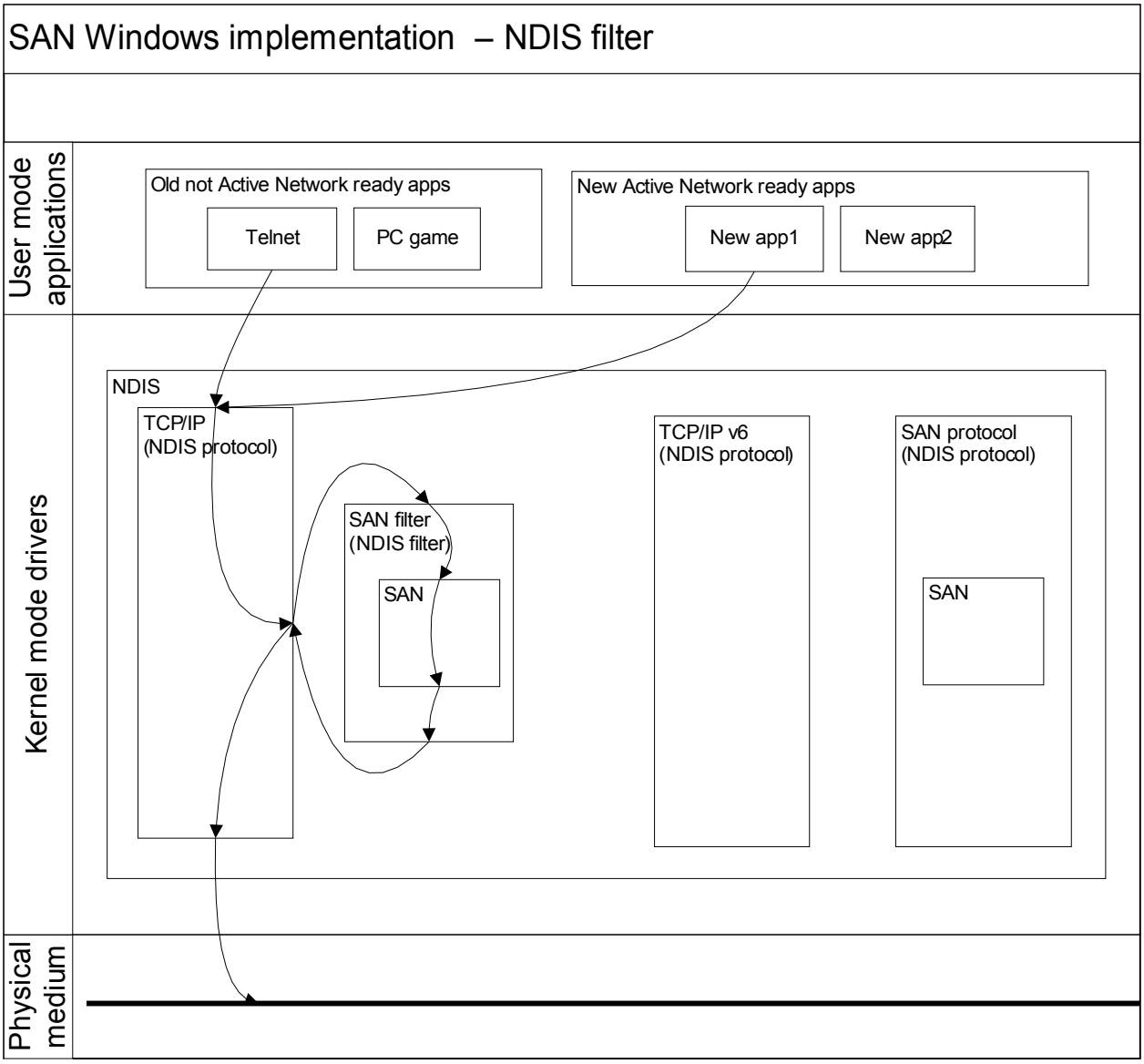
2.3 Řešení pomocí filtrování packetů

2.3.1 Princip řešení

Na rozdíl od předchozího řešení je tento postup možné použít i ve stávajících sítích. Stávající i nové programy posílají data standardně přes *TCP/IP* protokol. Data procházející protokoly je možné filtrovat. K filtrování použijeme *NDIS filter,* který umožňuje filtrování a hlavně také změny procházejících dat. Data nejprve zabalíme do *SAN* capsule a poté do *TCP/IP* paketu a až poté pošleme do sítě. Data tedy projdou bez problémů stávající sítí až na místo určení a pokud je na cílovém stroji nebo cestou nainstalován *SAN filter,* tak budou data vybalena a bude s nimi správně

naloženo. Na obrázku je znázorněno odesílání dat do sítě. Příjem dat je opět analogický.

Toto řešení je vhodné i pro starší programy, které nejsou koncipovány pro aktivní sítě. Pomocí filtru je možné specifikovat, které pakety budou zabaleny i do *SAN* capsule a které nebudou zabaleny a budou poslány standardním způsobem.



2.3.2 Popis programu

Před čtením tohoto textu je nejprve vhodné se seznámit s kapitolou Softwarové vybavení pro vývoj.

Filtrování se provádí pomocí filtru síťových dat. Ve Windows jde o síťový ovladač, který využívá standardu *NDIS*. My budeme používat *NDIS* verze 6.0, která je dostupná pouze ve Windows Vista a Windows Server 2008, což vyhovuje zadání. Pro starší Windows je ale řešení také

možné. Při popisu *NDIS* filtru budeme vycházet ze vzorového příkladu, který je dodáván s vývojovým prostředím *WDK* (viz kapitola Windows Driver Kit). Příklad je umístěn ve složce *src\network\ndis\ndisfilter*. Jedná se o zdrojové soubory v jazyce C.

Seznam nejdůležitějších souborů

device.c - Obsahuje metody pro zaregistrování, odregistrování a obsluhu *IOCtl (Input Output Controls*) určené pro konfiguraci ovladače – popis není obsahem tohoto textu.

filter.c - Obsahuje vstupní bod ovladače a veškerý kód spojený s filtrováním dat

flt_dbg.c - Obsahuje debugovací funkce

netlwf.inf - Konfigurační soubor ovladače, více v další kapitole.

Popis nejdůležitějších metod

`DriverEntry`

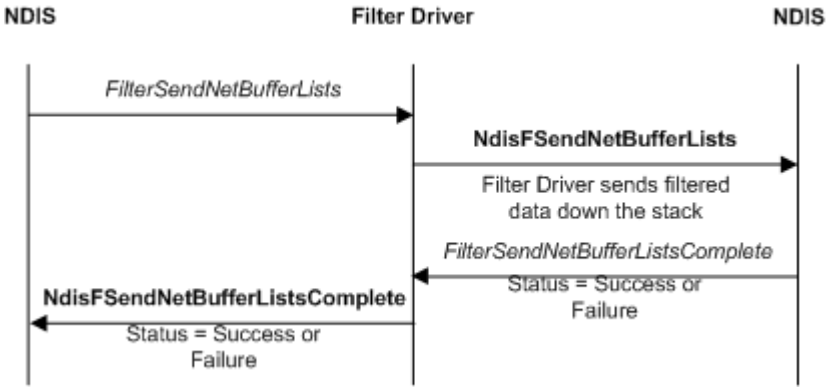
Vstupní bod ovladače. Tato metoda je to první, co se spustí při zavádění ovladače.

Hlavní úkol:

- nastavuje do datové struktury vlastnosti charakterizující ovladač, jako je název, verze *NDIS*, verze ovladače.
- Do stejné datové struktury nastavuje ukazatele k obslužným metodám, které jsou pak volány při vyvolání události (jako je např. příchod dat na ovladač protokolu)
- Zaregistruje filtr do systému.

Popis funkce filtru

Na následujícím obrázku je nastíněn proces filtrování

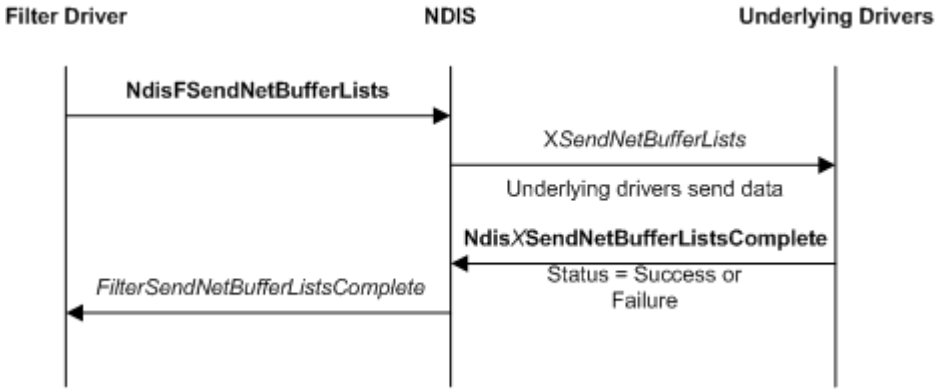


Ovladač kteréhokoliv protokolu v průběhu posílání dat, tedy ještě před odesláním dat na síť, zjistí přítomnost filtru. Pokud existuje filtr a má nastavenou obsluhu události `SendNetBufferListsHandler` zavolá obslužnou metodu ve filtru `FilterSendNetBufferLists`,

uvnitř této metody je možné udělat úpravu dat (což je cílem našeho řešení – zabalení dat do *SAN* capsule). Po dokončení úprav filtr uloží data do zásobníku k odeslání pomocí metody `NdisFSendNetBufferLists`. Po odeslání následuje podobná procedura. Ve filtru je volána obslužná metoda `FilterSendNetBufferListsComplete`, uvnitř které je vyhodnocen návratový kód a pomocí funkce filtr `NdisFSendNetBufferListsComplete` oznámí ovladači protokolu stav dokončení.

Pokud nemá ovladač nastavenou obsluhu události - má tedy nastaveno například `SendNetBufferListsHandler` na `null`, je obsluha ve filtru přeskočena a data jsou odeslána rovnou na zásobník.

Filtr může kromě filtrování datových paketů také sám od sebe iniciovat poslání dat. Tento postup je ilustrován na dalším obrázku.

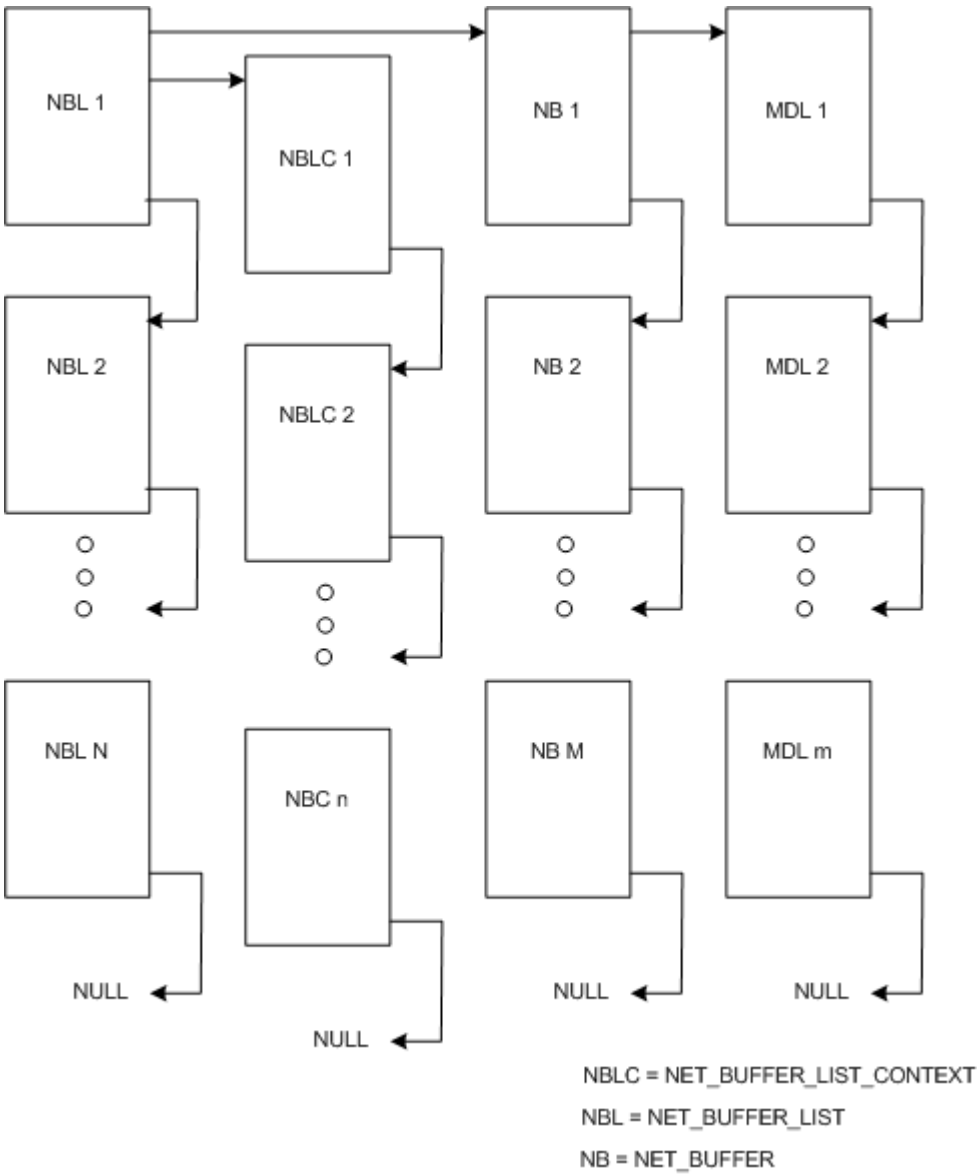


Posílání dat a oznámení o dokončení posílání je zde provedeno pomocí stejných metod jako u předchozího případu. S jediným rozdílem, že v datovém paketu, který je uložen ve struktuře *NET_BUFFER_LIST* (viz struktura datového paketu), je nastavena hodnota *SourceHandle* na stejnou hodnotu, jakou posíláme v parametru *NdisFilterHandle* do metody *NdisFSendNetBufferLists*.

Výše popsané funkce filtru jsou pro případ odesílání dat. Postup pro přijímání dat je analogický, metody mají ve svém názvu *Recieve* místo *Send*.

Struktura datového paketu

Data posílaná nebo přijímaná po síti se v *NDIS* ovladačích uchovávají v následující struktuře:



Data určená pro posílání po síti jsou uchovávána ve zřetězených strukturách *MDL*. Struktura *NET_BUFFER* odkazuje na první *MDL* v řetězu, obsahuje informace o celkové délce dat a offset, odkud začínají být platná data v *MDL* řetězu. *NET_BUFFER* je také zřetězená struktura. Odkaz na první strukturu *NET_BUFFER* v zřetězení je uložena ve struktuře *NET_BUFFER_LIST*. Ta může být opět zřetězena. *NET_BUFFER_LIST* struktura obsahuje odkaz na řetěz struktur *NET_BUFFER_LIST_CONTEXT*, kam si může ovladač uložit další data, která se přímo nebudou posílat po síti.

Následuje kód znázorňující výpis dat ze struktury (možno odchytit pomocí *DBGView*), například odesílaného paketu zachyceného filtrem.

```
PNET_BUFFER nb;
PMDL pMdl;
PUCHAR pData;
ULONG DataLength;

CurrNbl = NetBufferLists; // obsahuje naplněnou strukturu NET_BUFFER_LISTS
while (CurrNbl)
{
	nb = NET_BUFFER_LIST_FIRST_NB(CurrNbl);

	while(nb) {
		pMdl = NET_BUFFER_FIRST_MDL(nb);
		DEBUGP(DL_VERY_LOUD,
			("Write: MDL (offset: %d)\n", NET_BUFFER_DATA_OFFSET(nb)));
		while(pMdl) {
			NdisQueryMdl(pMdl, &pData, &DataLength, NormalPagePriority);
			DEBUGPDUMP(DL_VERY_LOUD, pData, DataLength);
			NdisGetNextMdl(pMdl, &pMdl);
		}
		nb = NET_BUFFER_NEXT_NB(nb);
	}
	CurrNbl = NET_BUFFER_LIST_NEXT_NBL(CurrNbl);
}
```

Uvedený kus kódu je možné vložit například do metody `FilterSendNetBufferLists` nad ukládání paketu k odeslání do zásobníku metodou `NdisFSendNetBufferLists`.

V případě úpravy dat je nutné pro nová data vytvořit novou datovou strukturu. To nejsnáze provedeme pomocí funkce *NdisAllocateNetBufferAndNetBufferList*.

Debugovací výpisy

Jak odchytávat debugovací výpisy je popsáno v kapitole DBGView. Uvedený driver má ale nastaveno, jak závažné chyby/upozornění se budou vypisovat. Pokud si chcete nechat vypisovat všechny debugovací výpisy, je nutné změnit hodnotu *filterDebugLevel* na začátku souboru *flt_dbg.c* na *DL_EXTRA_LOUD*. Další konstanty pro vyplnění do proměnné *filterDebugLevel* jsou definovány v souboru *flt_dbg.h*.

2.3.3 Potřebné nastavení k instalaci

Zde platí až na pár případů stejná nastavení jako v předchozím případě, jen je třeba se postarat ještě o filtr a jeho zavedení před všemi jinými službami.

Sekce s informacemi o verzi:

[version]

Class = NetService

ClassGUID = {4D36E974-E325-11CE-BFC1-08002BE10318}

Instalační sekce:

[install]

NetCfgInstanceId="{5cbf81bd-5055-47cd-9055-a76b2b4e3697}" ;ID je vzato z http://msdn.microsoft.com/en-us/library/aa503549.aspx Pro vygenerování jiného id lze použít program *Uuidgen.exe*, který je součástí WDK.

Sekce s údaji o přidávaném klíči:

[AddReg]

HKR, Ndi,CoServices,0x00010000,"value" ;Označuje servis, jenž je s filtrem spojený

HKR, Ndi,FilterType,0x00010001,0x00000002 ;Označení, že filtr bude monitorující

HKR, Ndi\Interfaces,LowerRange,,"nolower"

HKR, Ndi\Interfaces, FilterMediaTypes,,"ethernet" ;Říká k čemu je filtr připojený

HKR, Ndi,FilterRunType, 0x00010001, 1 ;filtr se musí spustit dříve, než služby jakýchkoliv jiných protokolů

nutné jsou zde další definice parametrů filtru, například:

HKR, FilterDriverParams\DriverParam,ParamDesc, , "Driverparam for lwf"

HKR, FilterDriverParams\DriverParam, default, , "5"

HKR, FilterDriverParams\DriverParam, type, , "int"

HKR, FilterAdapterParams\AdapterParam, ParamDesc, , "Adapterparam for lwf"

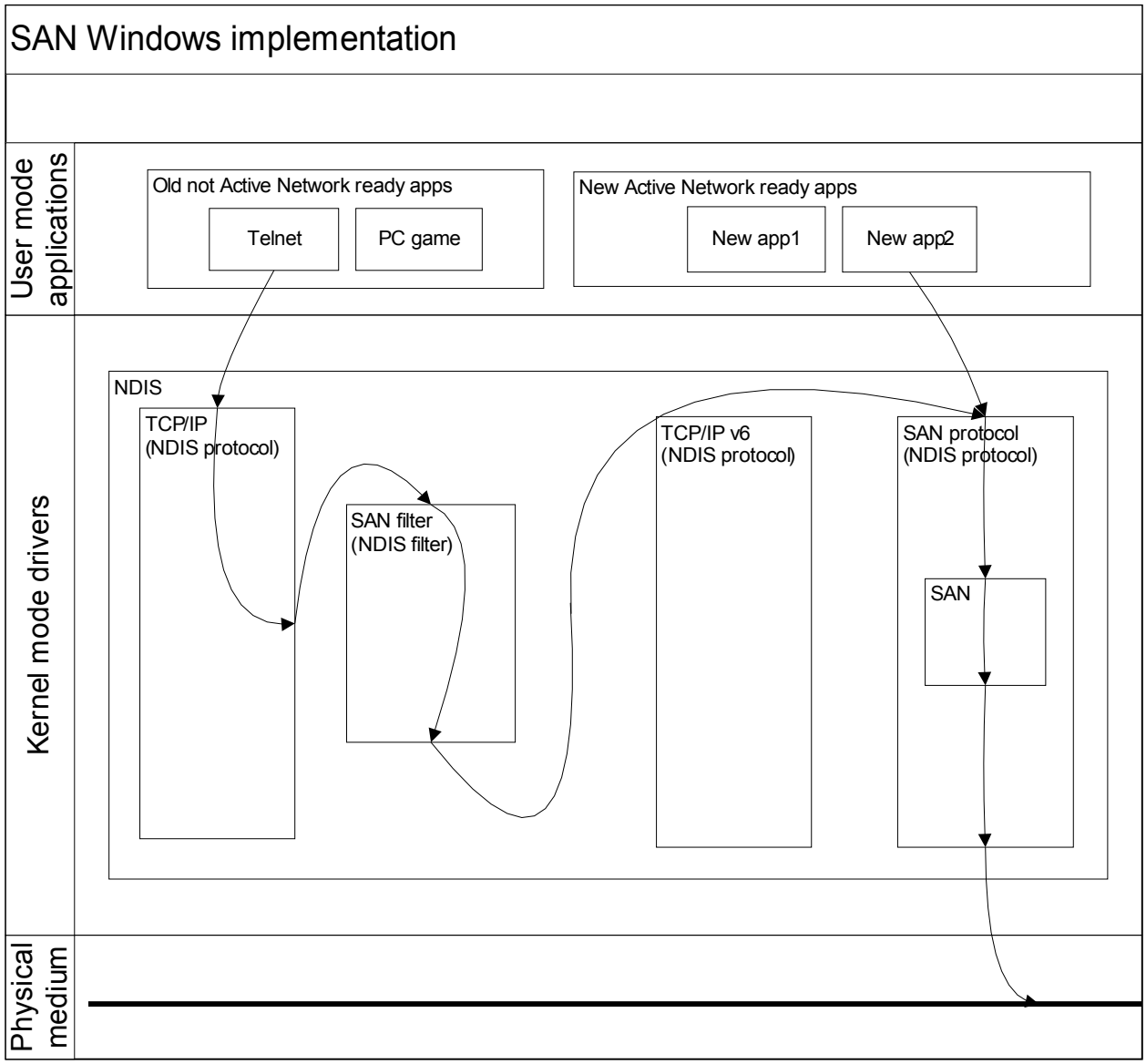
HKR, FilterAdapterParams\AdapterParam, default, , "10"

HKR, FilterAdapterParams\AdapterParam, type, , "int"

2.4 Kombinované řešení

2.4.1 Princip řešení

Řešení je vhodné pro finální nasazení v síti, kde hardware i všechny koncové stanice podporují aktivní sítě. Jde o kombinaci předchozích řešení tak, aby všechny data pro aktivní sítě byla zabalena v *SAN* capsulích a ne v *TCP/IP* paketech. Obrázek znázorňuje, jak probíhá posílání dat pomocí aktivních sítí. Příjem dat je opět analogický. V obrázku není pro přehlednost zahrnuto, že program může samozřejmě posílat data pomocí *TCP/IP*, aniž by byly nutně přesměrovány do *SAN* protokolu. Nová aplikace, vyvinutá pro podporu aktivních sítí umožňuje přímou komunikaci pomocí *SAN* protokolu. Stávající aplikace, u níž chceme také využít výhod aktivních sítí, pošle data standardně do protokolu *TCP/IP*. Procházející data jsou odchycena *SAN* filtrem, který data přesměruje na *SAN* protokol, pomocí něhož jsou data zabalena a odeslána na síť.



2.5 Instalace programu

Samotná instalace programu je pro obě varianty stejná. Níže uvedený postup je krok za krokem v systému Windows Vista.

Otevřete *Síťová připojení* (*Start* → *Ovládací panely* → *Centrum sítí a sdílení* → v levém panelu *Spravovat síťová připojení*) a zde klikněte pravým tlačítkem myši na síťové připojení a zvolte *Vlastnosti*. Objeví se seznam položek, které připojení používá a sem je třeba přidat další. Dále klikněte na *Nainstalovat*. Nyní je na výběr zda chcete instalovat klienta, službu či protokol. Toto je také jediné místo kde se liší instalace našich dvou variant. V prvním případě zvolte *protokol*, v druhém, tedy pokud instalujete filtr, zvolte *službu* a klikněte na *Přidat*. V následujícím okně klikněte na *Z diskety* a nyní je potřeba najít a otevřít .inf soubor vztahující se k vybrané službě / protokolu. Po odkývání zbytek udělá systém.

3 Instalační soubory

Pro instalaci ovladače a servisu, který nad ním pracuje, je používán v prostředí Windows soubor s informacemi - *.inf*. *INF (Setup Information File)* je textový soubor, který obsahuje informace o ovladači či hardware a nese důležité informace pro instalaci do systému.
Strukturou *INF* souborů je pouze posloupnost sekcí. Tyto sekce nejsou explicitně uzavřeny, končí tam, kde začíná sekce nová, popřípadě končí soubor. Sekce je otevřena názvem v hranatých závorkách např.: [version]. Za začátkem každé sekce následují direktivy a další specifikace. Ty jsou většinou zapisovány jako atribut=hodnota. Na každé řádce je vždy pouze jedna direktiva či začátek sekce, není zde žádný ukončovací znak. Středník se používá jako začátek komentáře.
Každý *INF* musí obsahovat sekci [version]. V té je důležitá direktiva *Signature*, udávající pro kterou verzi Windows je ovladač přizpůsoben. Dále třída do které ovladač spadá a identifikátor této třídy. Také zde může být uveden například výrobce či verze.

Další důležitou sekcí je [Manufacturer]. Zde se uvádí, jaká zařízení budou instalována, jakým modelem budou instalována (viz. dále) a například pro které architektury systému proběhne instalace.

Z [Manufakturer] přímo vychází *Model*. Zde je uveden popisek zařízení, ID zařízení a především odkaz na instalační sekci. Pokud je v [Manufakturer] uvedeno více architektur, definuje se potom model pro každou architekturu a tím je možné nastavovat různé specifikace.

Např.:

```
[Manufacturer]
%CMPN%=MDL,NTx86,NTia64,NTamd64
;Modely
[MDL.NTx86]
%Popisek%=Install, PROTOCOL
[MDL.NTia64]
%Popisek%=Install64, PROTOCOL
[MDL.NTamd64]
%Popisek%=Install64, PROTOCOL
```

V samotné instalační sekci je důležité direktivou *AddReg* zajistit potřebné informace pro přidání klíčů do registru a direktivou *CopyFiles* zajistit informace odkud a kam se mají zkopírovat binární soubory ovladače. Popis klíče je dán formátem *reg-root, [subkey], [value-entry-name], [flags], [value],* kde:

reg-root je kořenem, kam je klíč přidán

subkey je použit v případě, není-li klíč přímo pod kořenem

value-entry-name je název klíče

flag jeho bližší specifikace

value hodnota klíče

Např.:

```
HKR,Ndi,Service,,"abcd"
```

Klíč je zařazen systémem (HKR), dále do „složky“ Ndi, jedná se o službu, nemá žádné konkrétní specifikace a jeho hodnota je abcd.

Po přidání klíčů je na řadě instalace služby. Skrze direktivu *AddService* je určena sekce, kde jsou bližší informace. Ty jsou například zobrazované jméno služby, způsob běhu (jádro, interaktivní proces), kdy má být spuštěn (při zavádění systému, na požadavek), jaké budou reakce na chyby (ignorovány, ukončen systém), cesta k binárním souborům či popis služby. Dále zde lze také pomocí *DelService* služby odstranit.

Poslední částí instalace nového ovladače je specifikace výchozích a cílových umístění za pomoci několika sekcí. Tyto informace jsou pak využity při použití sekce [CopyFiles]. Jakým způsobem jsou informace o umístění souborů zadány bude nejlepší ukázat na příkladě.

Nejprve je třeba označit disk, kde se binární soubor nachází (zde 1) a k němu přidělit popisek, či další parametry jsou-li použity soubory *tag* (ověření správnosti disku) či *cab*.

```
[SourceDisksNames]
1 = %DiskDescription%,"",,
```

Poté na kterém disku (popřípadě kde na něm za pomocí druhého atributu) se binární soubor nachází.

```
[SourceDisksFiles]
protocol.sys = 1
```

A nakonec cílový adresář, kam se bude soubor kopírovat (12 značí předdefinovanou cestu systémem - *%windir%\system32\drivers*).

```
[DestinationDirs]
CpyFiles_Sys = 12
```

[CpyFiles_Sys] je jméno sekce, kde je určeno, které soubory budou zkopírovány. Nejprve je zde jméno zkopírovaného souboru, pokud se liší od původního, uvede se původní jako druhý atribut. Třetí je pro verze Windows 95, 98, Me a jde o dočasné jméno souboru během kopírování. 2 na konci je flag říkající, že kopírování nesmí být přerušeno uživatelem.

```
[CpyFiles_Sys]
protocol.sys,,,2
```

Poslední zmíněnou sekcí je [Strings], zde jsou uvedeny texty, řetězce, co jsou příliš dlouhé na pohodlné používání v průběhu psaní *INF* (například popisek servisu). V takovém případě je text zastoupen něčím jako makrem. To je pak na místě použití uzavřeno do procent a v sekci [Strings] je její plné znění.

Např.:

```
[version]
Provider = %ZCU%
[Strings]
ZCU="Západočeská univerzita v Plzni – Katedra Informatiky a výpočetní techniky"
```

Toto jsou potřebné sekce, pro zavedení nového protokolu. Pro ucelenější představu je listing vzorového *.inf* souboru součástí přílohy. Pro zavedení protokolu *SAN* do protokolového zásobníku je třeba vzorový *.inf* upravit, úpravy se ale týkají jen přepsání názvů souborů či jejich cest. Není třeba přidávat další sekce nebo direktivy.

4 Softwarové vybavení pro vývoj

4.1 Windows Driver Kits

Pro vývoj ovladačů ve Windows je nutné vývojové prostředí WDK, které obsahuje kompiler, nutné hlavičkové soubory a také vzorové zdrojové kódy ovladačů, ze kterých vycházíme

v této dokumentaci. Pro stažení vývojového prostředí *WDK* je nejprve nutné se zaregistrovat na serveru *connect.microsoft.com* (zdarma). Po přihlášení je možné stahovat z adresy https://connect.microsoft.com/Downloads/Downloads.aspx?SiteID=148 (cca 650MB). Po nainstalování se v menu start objeví zástupci k mnoha konzolím (např.: *Launch Windows Vista and Windows Server 2008 x86 Checked Build Environment*). Jednotlivé konzole se liší v cílové platformě, pro které má být výsledný ovladač nasazen a také slovem *Checked* či *Free*. Konzole s označením *Checked* kompiluje i s přidanými debugovacími informacemi, přikompilovává i kusy kódu uvedené v bloku preprocesoru `#if DBG … #endif`. Naopak stejný kód překompilovaný v konzoli s označením *Free* je bez všech těchto nadbytečných informací a ovladač je pak připravený k ostrému nasazení.

Pokud je *WDK* nainstalován ve standardním adresáři, nachází se veškeré vzorové zdrojové kódy ve složce *c:\WinDDK\xxxx.yyyyy\src*, kde *xxxx.yyyyy* označuje verzi vývojového prostředí *WDK*. Nás při vývoji síťových ovladačů budou zajímat ovladače v podsložce *network\ndis* a to konkrétně ovladač *ndisprot* a *filter*. Zdrojový kód *ndisprot* demonstruje přidání nového ovladače protokolu a *filter* umožňuje odchytávat pakety odcházející a přicházející ze sítě.

4.2 WinDbg

Nástroj sloužící k debugování driverů. Jde opět o produkt firmy *Microsoft*. Lze ho stáhnout v balíčku *Debugging Tools for Windows* z adresy http://www.microsoft.com/whdc/devtools/debugging/default.mspx. Debugování driverů má však omezení v tom, že se prakticky jedná o debugování jádra. A aby bylo možné debugovat proces běžící v režimu jádra, je zapotřebí dvou počítačů – jeden je debugován a na druhém je možné zadávat brakepointy a číst obsahy proměnných. Jednodušší možností debugování se zabývá kapitola DbgView. Možnosti propojení jsou pomocí seriové linky, USB a nebo FireWire kabelu. Spojení při debugování se ve WinDbg nastavuje v nabídce *File -> Kernel Debug...* . V nabídce je možné vybrat i debugování lokálního počítače, ovšem není možné nastavit brakepointy a lze jen sledovat debugovací výpisy z ovladače a na to použijeme jiný vhodnější nástroj DbgView. Ještě by bylo dobré se zmínit, že je nutné nastavit cestu symbolům ve *File -> Symbol File Path…*, což je cesta k souboru *.pdb*, který se vytvořil při kompilaci ovladače a také pokud je potřeba i k dalším souborům se symboly systému. Výsledek může vypadat například takto:

```
c:\WinDDK\6001.18001\src\network\ndis\filter\objchk_wlh_x86\i386\;c:\Program Files\Debugging Tools for Windows (x86)\sym\;SRV*C:\WebSymb*http://msdl.microsoft.com/download/symbols
```

Do *C:\WebSymb* se budou stahovat potřebné soubory se symboly systému a tahle složka

může narůst do řádu 10 až 100MB. Do *File -> Source File Path…* je vhodné zadat cestu ke zdrojovým souborů ovladače. Po otevření některého ze zdrojových souborů stačí zadat brakepoint a debugovat.

Pokud je na debugovaném stroji nainstalován operační systém Vista, je nutné upravit registry systému tak, aby byl systém schopen debugování. Úprava registrů je nutná kvůli jiné bezpečností politice Vist. Jedná se o přidání tří hodnot do registru a přesný postup o které se jedná je popsán v dokumentaci k *WinDbg v Enabling NDIS Debug Tracing By Setting Registry Values*.

4.3 DbgView

Pro jednodušší ‚debugování' ovladačů je možné použít program *DbgView*. Jde o produkt firmy Microsoft a lze ho stáhnout na adrese http://technet.microsoft.com/en-us/sysinternals/bb896647.aspx. Nejedná se o debugování jako takové. Program *DbgView* pouze odchytává debugovací výpisy a zobrazuje je na obrazovku. Pomocí tohoto jednoduchého nástroje můžeme sledovat, kterými metodami ovladač prochází, popřípadě i jaká data přijal atd. Debugovací výpis se v příkladech na WDK provádí pomocí makra DEBUGP, které ale pouze jednoduše volá systémovou metodu *dbgPrint()* (http://msdn.microsoft.com/en-us/library/ms792790.aspx). Pro to, aby byl program schopen zprávy od *dbgPrint()* odchytávat je nutné na operačním systému Windows Vista přidat do klíče *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Debug Print Filter* v registru přidat *DWORD* jménem ‚*DEFAULT*' s hodnotou 0x8. Poté je nutné systém restartovat. U předchozích operačních systémů změna registrů není nutná. Aby program *DbgView* zobrazoval informace z ovladače, musí mít nastavenou položku *Capture -> Capture Kernel*. Zachytávání zpráv se dá pozastavit/spustit položkou *Capture->Capture Win32*.

5 Řešení potíží

V průběhu vývoje ovladače se může stát, že ovladač shodí celý systém tak, že zůstane pouze modrá obrazovka s informacemi o chybě a možností pouze vypnout počítač. Pokud již máme nainstalovaný ovladač tak, že se spouští služba nad ním se startem Windows, tak nepomůže ani restart k nápravě, jelikož ještě než se uživatel stačí přihlásit, tak systém zavede ovladač a systém skončí stejnou modrou obrazovkou. Uvedu zde jednoduchý postup, jak vrátit systém do funkčního stavu.

1. Systém je nutné zavést v nouzovém režimu (vybrat možnost bez podpory sítě!)
2. Přihlásit se v nouzovém režimu na účet s administrátorskými právy

3. Pomocí *regedit.exe* upravit v klíči *HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Ndislwf* hodnotu *Start* z 0x1 na 0x3. (Kde *Ndislwf* je nutné nahradit názvem služby nad vámi upravovaným ovladačem; 0x1 znamená automatický start služby; 0x3 manuální start služby)
4. Restartovat počítač a zavést systém běžným způsobem.
5. Odstranit z protokolového zásobníku vadný ovladač
6. Zjistit kde se stala chyba, aby se znovu neobjevila modrá obrazovka

6 Závěr

Práce byla výzkumného charakteru, čili jsme nedocílili konkrétního softwarového produktu, ovšem přišli jsme na způsoby, jak tuto problematiku řešit. Řešení pouze pomocí protokolu *SAN* neumožňuje posílání dat pomocí stávajících aplikací, které nejsou požadavkům aktivních sítí přizpůsobené. Proto jsme navrhli další dvě řešení, které tento nedostatek odstraňují. Hlavní myšlenkou je použití filtru, jenž umožní nasazení aktivních sítí i na aplikace k tomu implicitně nepřizpůsobené. Dále jsme navrhli kombinované řešení filtru a nového *SAN* protokolu, které je vhodné pro nasazení v síti, kde všechny prvky včetně hardwaru (například přepínače, směrovače atd.) jsou přizpůsobeny na příjem *SAN* capsulí. Kombinované řešení je možné při vývoji *SAN* testovat na omezených sítích bez hardwarových směrovačů a přepínačů.

Použité materiály

[1] Michal Rejda, Smart Active Node, Diplomová práce, ZČU KIV 2008

[2] MSDN - NDIS Send and Receive Interfaces - http://msdn.microsoft.com/en-us/library/ms875352.aspx

[3] MSDN - NDIS Protocol Driver Reference - http://msdn.microsoft.com/en-us/library/bb314465.aspx

[4] MSDN - NDIS Filter Driver Reference - http://msdn.microsoft.com/en-us/library/bb961715.aspx

[5] MSDN - Introduction to Network Drivers - http://msdn.microsoft.com/en-us/library/aa938296.aspx

[6] MSDN - NDIS Protocol Drivers - http://msdn.microsoft.com/en-us/library/aa938323.aspx

[7] MSDN - NDIS Filter Drivers - http://msdn.microsoft.com/en-us/library/ms795077.aspx

[8] MSDN - IRP - http://msdn.microsoft.com/en-us/library/aa491631.aspx

[9] MSDN - NDIS MDL interface - http://msdn.microsoft.com/en-us/library/bb648495.aspx

[10] MSDN - DbgPrint - http://msdn.microsoft.com/en-us/library/ms792790.aspx

[11] TCP/IP - model, encapsulace, paket vs. rámec - http://www.samuraj-cz.com/clanek/tcpip-model-encapsulace-paketu-vs-ramec/

[12] NDIS - http://www.ndis.com/

[13] NDIS - Reading and Filtering Debugging Messages - http://msdn.microsoft.com/en-us/library/ms792789.aspx

Příloha

```
;-----------------------------------------------------------------------
; Vzorový INF pro přidání protokolu do protokolového zásobníku
;
; Václav Papež
; Petr Jaroš
;-----------------------------------------------------------------------
;-----------------------------------------------------------------------
; Informace o verzi
;-----------------------------------------------------------------------
[version]
;Ovladač je určený pro systém Windows založený na NT (NT, Server, XP, Vista)
Signature = "$Windows NT$"
;Třída do které ovladač spadá, NetTrans - komunikační protokoly
Class = NetTrans
;ID třídy - viz. http://msdn.microsoft.com/en-us/library/ms791134.aspx
ClassGUID = {4d36e975-e325-11ce-bfc1-08002be10318}
;Poskytovatel, výrobce
Provider = %ZCU%
;Verze instalovaného ovladače, ve formátu mm/dd/yyyy[,w.x.y.z]
DriverVer = 11/27/2008,0.0.0000.0
;-----------------------------------------------------------------------
; Sekce pro určení instalovaných zařízení
;-----------------------------------------------------------------------
[Manufacturer]
;%Výrobce%=ID instalačního modelu,[architektura systému] viz. http://msdn.microsoft.com/en-
us/library/ms794359.aspx
%ZCU% = SAN
;-----------------------------------------------------------------------
; Model instalace
;-----------------------------------------------------------------------
[SAN]
;%Popis zařízení%=jméno_instalační_sekce, ID zařízení
%SAN_Desc% = Install, KIV_SAN
;-----------------------------------------------------------------------
; Informace k instalaci
;-----------------------------------------------------------------------
[Install]
;Jméno sekce s informacemi pro přidání zařízení do registru
AddReg = Inst_Ndi
Characteristics = 0x0
;Jméno sekce s informacemi ke kopírování souborů
CopyFiles = CpyFiles_Sys
```

```
;-------------------------------------------------------------------------

; Klíče pro přidání do registru systému

;-------------------------------------------------------------------------

[Inst_Ndi]

;Ve tvaru reg-root, [subkey], [value-entry-name], [flags], [value]

;reg-root je kořenem, HKR nechává systém zařadit si klíče automaticky (zde je z kontextu bere jako
softwarové)

;subkey je "podadresář" ukládaného kliče

;value-entry-name je označení hodnoty (jméno registru)viz. http://msdn.microsoft.com/en-
us/library/ms794514.aspx

;flags jsou hexa číslo specifikujícíc další vlastnosti viz. http://msdn.microsoft.com/en-us/library/
ms794514.aspx

;value je hodnota registru

;Jedná se o servis

HKR,Ndi,Service,,"Ndisprot"

;Text pro nápovědu

HKR,Ndi,HelpText,,%SAN_HelpText%

;Spojení s nadřazeným rozhraním

HKR,Ndi\Interfaces, UpperRange,, noupper

;Spojení s rozhraním o úroveň níže

HKR,"Ndi\Interfaces","LowerRange",,"ndis5,ndis4,ndis5_prot"

;-------------------------------------------------------------------------

; Informace k instalaci servisu nad ovladačem

;-------------------------------------------------------------------------

[Install.Services]

;Informace k přidání servisu, AddService=jméno-servisu,[flag],jméno-instalační-sekce-servisu viz.
http://msdn.microsoft.com/en-us/library/ms794559.aspx

AddService = Ndisprot,,NDISPROT_Service_Inst


[NDISPROT_Service_Inst]

;Jméno servisu

DisplayName = %SAN_Desc%

;Typ servisu SERVICE_KERNEL_DRIVER (běží v jádře) viz. http://msdn.microsoft.com/en-
us/library/ms794559.aspx

ServiceType = 1

;Typ spouštění SERVICE_SYSTEM_START (při startu systému) viz. http://msdn.microsoft.com/en-
us/library/ms794559.aspx

StartType = 1

;Reakce na chyby SERVICE_ERROR_NORMAL (Servis spadne, ale systém nastartuje a uživatele upozorní)
viz. http://msdn.microsoft.com/en-us/library/ms794559.aspx

ErrorControl = 1

;Cesta k binární podobě servisu %12% značí %windir%\system32\drivers, jedná se o předdefinovanou
cestu viz. http://msdn.microsoft.com/en-us/library/ms790174.aspx

ServiceBinary = %12%\ndisprot.sys

LoadOrderGroup = NDIS

;Popis servisu

Description = %SAN_Desc%
```

```
[Install.Remove.Services]

;Smaže již dříve instalovaný servis, DelService=jméno,[flag], 0x00000200(SPSVCINST_STOPSERVICE) -
zastavit servis před smazáním viz. http://msdn.microsoft.com/en-us/library/ms794352.aspx

DelService = Ndisprot,0x200

;-------------------------------------------------------------------------

; Nastavení cílových adresářů pro práci s binárními soubory

;-------------------------------------------------------------------------

[SourceDisksNames]

;Disk ID = popis-disku[, další parametry při použití tag-or-cab-file] viz.
http://msdn.microsoft.com/en-us/library/ms794354.aspx

1 = %DiskDescription%,"",,


[SourceDisksFiles]

;Umístění instalovaného souboru jméno=disk ID[,podadresář][,velikost souboru]

ndisprot.sys = 1


[DestinationDirs]

;Cílový adresář, kam se budou soubory kopírovat, %12% značí %windir%\system32\drivers, jedná se o
předdefinovanou cestu viz. http://msdn.microsoft.com/en-us/library/ms790174.aspx

CpyFiles_Sys = 12

;-------------------------------------------------------------------------

; Kopírované soubory

;-------------------------------------------------------------------------

[CpyFiles_Sys]

;cílové-jméno-souboru[,zdrojové-jméno-souboru][,přechodné-jméno-souboru(pro Windows 95,98,ME][,flag
(2 značí COPYFLG_NOSKIP, kopírování nemůže být přerušeno uživatelem)] viz.
http://msdn.microsoft.com/en-us/library/ms794560.aspx

Ndisprot.sys,,,2

;-------------------------------------------------------------------------

; Deklarace řetězců používaných výše

;-------------------------------------------------------------------------

[Strings]

ZCU = "Západočeská univerzita - KIV"

SAN = "Smart Active Node"

DiskDescription = "SAN Protokol Driver Disk"

SAN_Desc = "Smart Active Node Protokol"

SAN_HelpText = "Ovladač protokolu aktivních sítí SAN"
```