Analýza a implementácia podpory pre WPA firmware pre at76c503

(Bakalárska práca)

Milan Plžík

Bratislava, 2008

Katedra Informatiky
Fakulta Matematiky, Fyziky a Informatiky
Univerzita Komenského, Bratislava

Analýza a implementácia podpory pre WPA firmware pre at76c503

(Bakalárska práca)

Milan Plžík

Odbor: 9.2.1 Informatika
Vedúci: RNDr. Jaroslav Janáček

Bratislava, 2008

Čestne prehlasujem, že som túto diplomovú prácu vypracoval(a) samostatne s použitím citovaných zdrojov.

.....

Poďakovanie

Chcel by som vysloviť poďakovanie všetkým, ktorí ma v tejto práci podporovali. Ich zoznam isto nebude úplný, avšak verím, že sa mi podarí zachytiť aspoň podstatnú časť.

Vďaka patrí v prvom rade mojim rodičom a starým rodičom, ktorí ma podporovali vždy, keď som to potreboval. Ďalej RNDr. Jaroslavovi Janáčkovi za to, že bol ochotný viesť mnou navrhovanú tému bakalárskej práce, ako aj za jeho rady, vďaka ktorým táto práca dosiahla oveľa vyššiu úroveň, Mgr. Vladimírovi Koutnému za jeho neoceniteľné rady odhaľovaní tajomstiev mac80211 a nekonečnú trpezlivosť pri odpovedaní na moje otázky, Mgr. Michalovi Foríškovi za jeho LaTeX-ový template na (nielen) bakalársku prácu, Danke Babiarovej za jej trpezlivosť, podporu a ochotu počúvať reči o téme, ktorá jej nie je až tak blízka.

Ďalej moja vďaka patrí všetkým, ktorí mi pred niekoľkými rokmi pomáhali robiť prvé kroky na handhelds.org – Paul Sokolovsky, Erik Hovland, Florian Bohr, Anton Vorontsov, ale aj veľa ďalších. Bez ohľadu na ich poradie boli pre mňa neoceniteľným zdrojom informácií a zároveň inšpirácie. Taktiež ďakujem Jörgovi Albertovi za zapožičanie testovacieho zariadenia.

Na záver patrí vďaka všetkým, ktorí mi držali palce a napriek môjmu vlastnému pesimizmu verili, že to dokážem. Ešte raz ďakujem.

Abstrakt

Autor:	Milan Plžík
Názov práce:	Analýza a implementácia podpory pre WPA firmware pre at76c503
Škola:	Univerzita Komenského
Fakulta:	Fakulta matematiky, fyziky a informatiky
Katerda:	Katedra informatiky
Školiteľ:	RNDr. Jaroslav Janáček
Miesto:	Bratislava
Rok:	2008
Kľúčové slová	mac80211, wifi, USB, at76, WPA, WEP, linux, wireless, IEEE 802.11

Práca rozoberá detaily komunikácie bezdrôtového sieťového adaptéru s chipsetom at76c503 s firmvérom verzie 1.103.0-175, pričom sa podrobne venuje porovnaniu s komunikáciu pri použití staršieho firmvéru. Ďalej sa práca venuje úpravám ovládača at76_usb potrebným na dosiahnutie korektného fungovania s firmvérom verzie 1.103.0-175.

Predhovor

Prvotnou motiváciou na výber právej tejto témy bakalárskej práce bola skúsenosť s linuxom na PDA ho iPaq rady h5000. Tieto zariadenia obsahujú bezdrôtový sieťový adaptér s chipsetom spomínaným v téme tejto práce. Cieľom práce je pridať podporu pre novšie verzie firmvéru do (v súčasnosti jediného udržiavaného) ovládača pre tieto adaptéry. Informácie pritom budú získavané prevažne analýzou pôvodného a v súčasnosti už neudržiavaného ovládača, ako tiež z dostupnej dokumentácie. Úspešným zakončením tejto práce budú tieto zariadenia využiteľné do maximálnej miery umožnenej firmvérom a informáciami z oficiálneho ovládača.

Obsah

Zoznam tabuliek

Zoznam skratiek

- *AES* – Advanced Encryption Standard
- *API* – Application Programming Interface
- *BSSID* – Basic Service Set Identifier
- *CCMP* – Counter Mode with Cipher Block Chaining Message Authentication Code Protocol
- *CKIP* – Cisco Key Integrity Protocol
- *ESSID* – Extended Service Set Identifier
- *PCMCIA* – Personal Computer Memory Card International Association (tiež People Can't Memoize Computer Industry Acronyms)
- *IEEE* – Institute of Electrical and Electronics Engineers
- *LLC* – Logical Link Control
- *MAC* – Media Access Control
- *MIC* – Message Integrity Code
- *NDIS* – Network Driver Interface Specification
- *PCI* – Peripheral Component Interconnect
- *TKIP* – Temporal Key Integrity Protocol
- *USB* – Universal Serial Bus
- *WEP* – Wired Equivalent Privacy
- *WPA* – Wi-Fi Protected Access

Kapitola 1

Úvod

1.1 Typografické konvencie

V tejto práci používam nasledovné typografické konvencie:

- `Hrubým písmom` sú označné názvy súborov, cesty, časti kódu, názvy štruktúr a konštánt.
- *Zvýraznené kurzívou* budú dôležité, prvý raz spomínané pojmy, dôležité názvy, názvy funkcií.

Použitá terminológia zahŕňa základné pojmy definované v štandarde USB verzie 2.0 a IEEE 802.11 .

1.2 Základné informácie

Bezdrôtové sieťové karty na báze chipsetu *at76c503* vyrábala dávnejšie firma *Atmel*. Jedná sa o inteligentné sieťové karty s vlastným procesorom, ktorý komunikuje s rádiovou časťou a pomocou USB1.1 rozhrania komunikuje s USB hostom. Tieto zariadenia podporujú komunikáciu podľa štandardu *IEEE 802.11b*. Čo sa týka bezpečnosti, schopnosti týchto zariadení závisia od použitého firmvéru. Verzie firmvéru 0.90.x (*starší firmvér*) podporujú iba otvorené siete a *WEP*, zatiaľ čo firmvér verzie 1.103.x (*novší firmvér*) už podporuje aj *WPA*, resp. *WPA2*.

Tento chipset bol používaný jednak priamo na výrobu bezdrôtových sieťových kariet, ale zároveň býval aj zabudovaný do niektorých zariadení, ako napríklad niektoré zariadenia HP iPaq rady h5000. Pokiaľ je na týchto pôvodný operačný systém nahradený linuxom, používateľ je limitovaný na použitie slabšieho zabezpečenia bezdrôtovej siete.

V čase, keď boli tieto zariadenia aktívne udržiavané, pracovníci firmy Atmel udržiavali ovládač atmelwlandriver, ktorý podporoval aj novšie verzie firmvéru. Keďže už ale tieto zariadenia ďalej nie sú podporované, vývoj ovládača sa zastavil.

Paralelne s oficiálnym ovládačom sa vyvíjal aj neoficiálny ovládač *at76_usb* (predtým *at76c503a*), ktorý je ešte stále udržiavaný, navyše je snaha o jeho začlenenie do hlavnej vetvy linuxu. Tento ovládač ale podporuje iba staršie verzie firmvéru, teda jediný spôsob, ktorý umožňuje aspoň čiastočne využivať WPA je robiť potrebné výpočty v softvérovom 802.11 stacku na strane hosta, čo je ale neefektívne a zbytočne ho zaťažuje, teda môže spôsobovať problémy najmä pri vyššie spomenutých zariadeniach.

Cieľom tejto práce je analyzovať rozdiely v spôsobe, ako atmelwlandriver komunikoval so starším a ako s novším firmvérom a tieto rozdiely implementovať do at76_usb, pričom výsledkom má byť schopnosť používať at76_usb na komunikáciu vo WPA, resp. WPA2 sieťach.

Kapitola 2

Bezdrôtové siete a ovládače

2.1 Všeobecná situácia

Bezdrôtové siete za posledných niekoľko rokov prešli zo štádia, kedy s nimi mala možnosť pracovať iba úzka skupina ľudí do stavu masového rozšírenia, kedy pre jednotlivca nie je problém vytvoriť a spravovať vlastnú takúto sieť – zvýšila sa dostupnosť hardvéru, zjednodušila sa konfigurácia, pohodlie poskytované bezdrôtovými sieťami začalo byť vyhľadávanejšie.

Na pohodlné využívanie všetkých výhod je potrebné, aby všetky komponenty použité na vytvorenie siete korektne spolupracovali. Pri samotnom hardvéri táto časť nie je problematická, keďže prešiel WiFi certifikáciou a teda je zaručená funkčnosť. Problémom ale zostáva spolupráca hardvéru so softvérom. Výrobcovia zvyčajne odmietajú sprístupniť dokumentáciu ku zariadeniam, ktoré pri nesprávnych nastaveniach umožňujú porušovať telekomunikačné zákony. S WiFi adaptérom sa preto zvyčajne dodávajú binárne ovládače spĺňajúce NDIS 5.x (popr. 6.0, ale tie v súčasnosti iba zriedkavo). Tieto však majú niektoré dosť významné obmedzenia – zvyčajne sú dostupné iba pre architektúry založené na x86 (popr. x86_64), v drvivej väčšine vyladené pre Microsoft Windows.

Pre iné operačné systémy exisistuje čiastočné riešenie vo forme rôznych *wrapperov*, ktoré umožňujú používať NDIS ovládace napríklad v Linuxe (*ndiswrapper*) alebo vo FreeBSD (*NDISulator*). Nie vždy však fungujú úplne korektne, navyše API NDIS 5.x umožňuje pracovať iba s rámcami IEEE 802.3 (*wired ethernet*), čo síce postačuje na bežné používanie adaptéra, avšak zabraňuje využitiu pokročilejších funkcií. NDIS 6 tento problém rieši tým, že umožňuje používať režim *monitor*, v ktorom sa pracuje s IEEE 802.11 rámcami, avšak tieto ovládače sú v rámci Windows podporované iba vo Windows Vista, o iných operačných

systémoch nehovoriac.

Ďalším problémom, vo svojej podstate nemenej významným, je vlastná uzavretosť ovládačov. Vo svete slobodného softvéru je ovládač v čisto binárnej forme považovaný prinajmenšom za „nutné zlo“, ktoré je v dlhodobom horizonte nepoužiteľné (príliš veľká naviazanosť na súčasné API, závislosť na konkrétnej architektúre).

Logickým dôsledkom všetkých vyššie uvedených faktov aplikovaných na slobodný softvér je snaha o vlastné ovládače, ktoré spĺňajú kritériá otvorenosti a zároveň umožňujú plné využitie schopností konkrétneho zariadenia. V ďalších častiach bude podrobnejšie rozobratý vlastný vývoj ovládačov na bezdrôtové sieťové adaptéry pod Linuxom.

2.2 Vzťah klasických a bezdrôtových sietí (OSI model)

Podľa štandardu [4, IEEE Std 802.11 is required to appear to higher layers (logical link control (LLC)) as a wired IEEE 802 LAN.] (časť 5.1.1.4) platí, že od LLC vrstvy OSI modelu už nerozlišujeme medzi klasickými a bezdrôtovými sieťami, čo umožňuje napríklad bezproblémové premostenie takýchto sietí. Ďalším, a pre túto prácu najzaujímavejším dôsledkom je samotný fakt, že od LLC vrstvy je už možné využívať existujúci *networking stack*.

Vrstvy nachádzajúce sa pod LLC (teda MAC podvrstva a fyzická vrstva) sú zodpovedné za všetky špecifiká bezdrôtových sietí, ako sú napríklad:

- šifrovanie (nešifrované siete, WEP, WPA, WPA2) a použité šifrovacie algoritmy (RC4/TKIP, AES/CCMP)
- rôzne typy bezdrôtových sietí, napríklad IBSS, BSS, ESS, mesh (IEEE 802.11s) a rolí, ktoré v nich zariadenia môžu mať (prístupový bod, stanica, ...)
- vyhľadávanie, asociácia do bezdrôtovej siete, disasociácia, ...

Túto časť komunikácie už musí ošetrovať softvér mimo všeobecného sieťového stacku a navyše, vďaka vlastnostiam IEEE 802.11 môže implementácia požadovaných vlastností byť celkom komplikovaná.

Sieťové adaptéry zvyčajne implementujú fyzickú vrstvu priamo v hardvéri. Pri podvrstve MAC je situácia už o čosi komplikovanejšia – v závislosti od hardvéru môže byť niektorá jej funkcionalita implementovaná priamo v ňom, v inom prípade sa môže spoliehať na ovládač, ktorý danú funkcionalitu zabezpečí.

Ranné verzie ovládačov na bezdrôtové adaptéry zvyčajne implementovali potrebné časti MAC vo vlastnej réžii. Niektoré adaptéry podporovali značnú časť MAC priamo v hardvéri (resp. firmvéri), takže nebolo nutné zabezpečovať veľa funkcionality. Postupom času sa ale začali objavovať adaptéry, ktoré implementovali iba veľmi malú časť MAC a väčšinu práce nechávali na hostiteľskom zariadení. Pre tieto už bola implementácia MAC menej priamočiara, navyše pre každé zariadenie sa svojím spôsobom istá časť kódu duplikovala.

Táto duplicita kódu napokon viedla ku myšlienke všeobecnej implementácie podvrstvy MAC, ktorá by mohla byť zdieľaná medzi ovládačmi. Ovládače by potom ošetrovali iba hardvérové špecifiká adaptéru a zvyšok práce, ktorú doteraz robili, by vykonával zdieľaný kód. Navyše by tento kód kód, vďaka jeho „nešitiu na mieru“ konkrétnemu ovládaču mohol byť viac odladený a obsahovať podporu pre komplikovanejšie, ťažšie implementovateľné črty .

2.3 *mac80211* a jeho využitie

Počas existencie linuxového jadra bolo pokusov o zjednotenie kódu pre bezdrôtové adaptéry viacero, v súčasnosti je výsledkom tohto snaženia *mac80211*. Ako už bolo vyššie spomenuté, jedná sa o implementáciu IEEE 802.11 podvrstvy MAC, pričom hlavným cieľom je, aby bola dostatočne flexibilná pre akýkoľvek sieťový adaptér.

atmelwlandriver, ktorý je analyzovaný v tejto práci, implementoval podvrstvu MAC vo vlastnej réžii, navyše na konfiguráciu WPA využíval vlastné rozhranie špecifické iba pre tento ovládač. Ovládač bol písaný v dobe, keď *mac80211* ešte neexistoval, takýto prístup umožňoval udržať si relatívne veľkú nezávislosť na rôznych API jadra. Na druhej strane, daňou je komplikovanosť a množstvo kódu, ktoré ovládač obsahuje.

Ovládač at76_usb pôvodne taktiež implementoval veľa vecí MAC podvrstvy sám, avšak neskôr bol upravený tak, aby *mac80211* využíval. Týmto sa znížilo množstvo kódu, keďže teraz sa ovládač stará iba o vlastné ovládanie adaptéra a navyše je možné využívať štandardné *Wireless Extensions*.

Prechod na *mac80211* má v tomto prípade aj svoje nevýhody. Keďže on samotný je vo vývoji a API sa mení, ovládač si bude vyžadovať častejšie úpravy, až kým sa API nestabilizuje. Navyše, v momentálnom stave chýba funkčný *ad-hoc* režim, ktorého podpora v ovládači zatiaľ chýba.

Kapitola 3

Analýza

Pôvodný atmelwlandriver je relatívne rozsiahly ovládač, obsahujúci ovládače pre celú radu at76c50x zariadení pripojených nielen cez USB, ale aj pomocou zbernice PCI, resp. PCMCIA. Taktiež obsahuje vlastn[implementáciu častí IEEE 802.11 MAC podvrstvy. V tejto práci bude rozoberaná iba podpora pre USB zariadenia. Najpodstatnejšie súbory, o ktoré sa budem opierať, sú v adresároch `src/usb/` a `src/includes/usb`.

3.1 Komunikačný protokol

Ako už bolo spomenuté v úvode, vlastná komunikácia prebieha po USB. Zariadenie po jeho zapnutí môže byť v závislosti od konkrétnej konfigurácie v stave, kedy očakáva primárny/sekundárny firmvér. Pre účely tejto práce je podstatný režim s už nahratým a funkčným firmvérom. V tomto režime má zariadenie dva *endpointy* – *endpoint zero*, ktorý je vyžadovaný USB štandardom (používa sa na konfiguráciu zariadenia) ďalej endpoint pre *bulk* prenosy, ktorý sa používa na prenos vlastných dát.

3.2 Komunikácia cez *endpoint zero*

3.2.1 Príkazy

Endpoint zero je komunikačný bod, ktorého existenciu a základnú funkcionalitu vyžaduje priamo štandard USB. V našom prípade okrem štandardnej funkcionality vyžadovanej USB štandardom poskytuje aj funkcie implementované výrobcom, pomocou ktorých ovládač riadi sieťovú kartu. Stručný a nekompletný zoznam príkazov nájdete v tabuľke 3.1, ktorá bol vytvorená na základe [1], do-

Typ	Request	Value	Idx	Payload	Význam
OUT	0	1000	0	–	DFU detach
OUT	0x1	blknum	0	buffer	Write FW block
IN	0x3	0	0	6B status	Get DFU status
IN	0x5	0	0	1B state	Get DFU state
OUT	0xA	0	0	–	Remap
OUT	0xE	0	0	0xB, 0, ...	Start-up device
OUT	0xE	0x1	0	1B mode	Set mode of operation
OUT	0xE	0	0	0x1, 0, ...	Set MIB values
OUT	0xE	0	0	0x3, 0, ...	Scan
OUT	0xE	0	0	0x4, 0, ...	Join
OUT	0xE	0	0	0x5, 0, ...	Start IBSS
OUT	0xE	0x802	blk	...	Write ext. FW block
OUT	0xE	0xid02 [1]	0	...	Set HW configuration
IN	0x22	CmdID	0	...	Get command status
IN	0x33	0x1	0	1B data	Get mode of operation
IN	0x33	0xty00 [2]	offs.	...	Get MIB values
IN	0x33	0xA02	0	...	Get HW configuration

Tabuľka 3.1: Príkazy implementované výrobcom

Typ	Request	Value	Idx	Payload	Význam
OUT	0xE	0	0	0x6,0,0,0	SetRadioOn
OUT	0xE	0	0	0x7,0,0,0	SetRadioOff

Tabuľka 3.2: Príkazy implementované v [2]

kumentujúceho starší firmvér. Pri [2] bol pre analýzu relevantný obsah súboru `src/usb/command.c`.

V zdrojových kódoch [2] bolo nájdených niekoľko ďalsíších príkazov, ktoré v [1] neboli dokumentované, tieto nájdete v tabuľke 3.2, navyše tieto príkazy používa aj at76_usb, z čoho vyplýva, že zrejme boli uverejnené v neskorších revíziách dokumentácie.

Záverom tejto časti je, že čo sa týka samotných príkazov, nenastala žiadna zmena. Pokiaľ nastala nejaká zmena, musí sa nutne jednať o zmenu dátových štruktúr, ktoré sa s príkazmi predávajú firmvéru. Porovnaniu týchto štruktúr sa venuje nasledujúca časť.

[1] *id* je ID konfigurácie
[2] *ty* je typ MIB

Štruktúra	Parameter pre *Get/Set MIB values*
LOCAL_MIB	0x01
MAC_ADDRESS_MIB	0x02
MAC_MIB	0x03
MAC_MGMT_MIB	0x05
MAC_WEP_MIB	0x06
PHY_MIB	0x07
FW_VERSION_MIB	0x08
MDOMAIN_MIB	0x09

Tabuľka 3.3: Zoznam MIB definovaných v at76_usb

3.2.2 Dátové štruktúry príkazov

Niektoré príkazy spomenuté v predchádzajúcej časti komunikujú s firmvérom pomocou odosielania predom definovaných dátových štruktúr. V tejto časti sa zameriam na porovnanie týchto štruktúr v [1] a [2], ktoré sú deklarované v súbore `src/includes/usb/command.h`.

V [2] sú definované štruktúry `SCAN`, `JOIN`, `START_BSS` a tiež `USB_CARD_CONFIG`, ako tiež štruktúry pre príkazy manipulujúce s MIB hodnotami `SET_MIB`, avšak pri ich porovnaní s [1] ani [3] neboli zistené žiadne rozdiely. Aj táto časť teda zachováva kompatibilitu medzi starším a novším firmvérom.

3.2.3 MIB premenné

MIB premenné slúžia na nastavovanie rozličných parametrov zariadenia. Ich modifikácia prebieha pomocou už spomínaných príkazov *Set MIB values* a *Get MIB values*. V tejto časti sa pokúsim nájsť a zdokumentovať rozdiely medzi medzi starším a novším firmvérom z tohto pohľadu. V oboch ovládačoch sú definované zhodné MIB štruktúry, ktoré nájdete v tabuľke 3.3.

Porovnávaním štruktúr v zdrojových kódoch [2] a [3] boli zistené nasledovné fakty:

- `LOCAL_MIB`, `MAC_ADDRESS_MIB`, `MAC_MGMT_MIB` a `MDOMAIN_MIB` sú definované zhodne.
- `MAC_MIB` a `FW_VERSION_MIB` majú v [2] jednu z ich hodnôt označenú ako *Reserved*, kým [3] jej priraďuje konkrétne využitie
- `MAC_WEP_MIB` majú známe hodnoty definované zhodne, štruktúra v [2] však na konci navyše obsahuje 3 bajty vyhradených hodnôt. Toto by však ne-

malo mať vplyv na celkové fungovanie ovládača, keďže podľa [1] funkcie na čítanie/zápis MIB nemusia pracovať s celou štruktúrou.

- `PHY_MIB` má v [2] navyše na konci bajt označený ako *CurrentTxPowerLevel*, ktorý udáva vysielací výkon v desiatkach dBm. Toto v [3] zatiaľ nie je implementované.

V `src/includes/usb/mib.h` je definovaná štruktúra `MAC_ENCRYPTION_MIB`, ktorá ale v tomto súbore nemá, na rozdiel od statných MIB, definované žiadne číslo. Pohľad do `src/usb/command.c` odhalí, že príkazy manipulujúce s MIB používajú okrem hodnôt z tabuľky 3.3 aj `Mac_Encryption_Mib_Type`, ktorá v `mib.h`, tak isto, ako v žiadnom inom hlavičkovom súbore, nie je definovaná – nájsť sa však dá v `src/usb/command.c`. Podľa definície na manipuláciu s touto štruktúrou používame pri príkazoch *Get/Set MIB values* číslo 0x6, ktoré je už priradené pre `MAC_WEP_MIB`.

V [2] sa používajú obe štruktúry súčasne, pričom pri častiach, ktoré sa starajú iba o nastavenia WEP, sa používa `MAC_WEP_MIB`. `MAC_ENCRYPTION_MIB` je použitý vo funkciách *SetEncryptionStatus*, *SetEncryptionKeyRsc*, *SetEncryptionKeyS* a *GetEncryptionMIB*.

Pre porovnanie uvediem obe štruktúry, pričom 3.2 je upravený tak, aby neobsahoval príkazy makroprocesora.

Listing 3.1: Štruktúra MAC_WEP_MIB

```
typedef struct __MAC_WEP_MIB {
        BOOLEAN    PrivacyInvoked;
        UCHAR      WEPDefaultKeyID;
        UCHAR      WEPKeyMappingLength;
        BOOLEAN    ExcludeUnencrypted;
        ULONG      WEPICErrorCount;
        ULONG      WEPExcludedCount;
        UCHAR      WEPDefaultKeyValue[4][13];
        UCHAR      EncryptionLevel;
        UCHAR      Reserved[3];
} MAC_WEP_MIB;
```

Listing 3.2: Štruktúra MAC_ENCRYPTION_MIB

```
typedef struct __MAC_ENCRYPTION_MIB {
        UCHAR      CipherDefaultKeyValue[4][40];
        UCHAR      TKIPBSSID[6];
```

```
        BOOLEAN     PrivacyInvoked;
        UCHAR       CipherDefaultKeyID; //0..3
        UCHAR       CipherDefaultGroupKeyID;
        BOOLEAN     ExcludeUnencrypted;
        UCHAR       WEPEncryptionType;
        BOOLEAN     CkipKeyPermutation;
        ULONG       WEPICVErrorCount;
        ULONG       WEPExcludedCount;
        UCHAR       KeyRSC[4][8];
} MAC_ENCRYPTION_MIB;
```

Pozornému čitateľovi istotne neuniklo, že listing 3.2 je viac-menej nadmnožinou 3.1. Otázne teraz je, ako ovládač môže súčasne používať obe štruktúry s jednou verziou firmvéru, pokiaľ ich skutočne používa. Tomuto problému sa budem venovať neskôr.

V `command.c` sú uvedené všetky funkcie, ktoré priamo komunikujú s adaptérom, medzi iným aj všetky funkcie pracujúce s MIB. Pokiaľ sa zameriame na funkcie meniace MIB, zistíme, že `MAC_WEP_MIB` mení funkcia s názvom *SetWEPvalue* (ostatné funkcie sú ňou používané) a `MAC_ENCRYPTION_MIB` menia funkcie *SetEncryptionStatus* a ako callback aj *SetEncryptionKeyS*. Mimo pôvodného súboru sú tieto funkcie používané iba na málo miestach, a to konkrétne:

- `callbacks.c`: *GetCmd_callback*
- `vnet.c`: Funkcia *RemoveKey*, ktorá odstraňuje šifovacie kľuče.
- `vnetusba.c`: *atmel_wpa_ioctl* volá WPA funkcie ovládača, *usb_vnet_ioctl*, *vnet_set_encode* používajú WEP funkcie.

Z vyššie uvedených údajov sa zameriame hlavne na posledný súbor. Funkcia *vnet_set_encode* je priamo využívaná rozhraním *Wireless Extensions*. *atmel_wpa_ioctl* sa používa jedine vnútri funkcie *usb_vnet_ioctl*, pričom ju bez akýchkoľvek obmedzení zadaním spravnych parametrov možno vyvolať. Z toho vyplýva, že ovládač žiadnym spôsobom neošetruje rôzne verzie firmvéru s rôzne implementovanou podporou pre šifrovanie a používateľ je schopný vyvolať odoslanie ľubovoľnej z týchto dvoch štruktúr do adaptéru. Pokiaľ teda adaptér má reagovať korektne na príkazy odoslané z ovládača, musí nejakým spôsobom získať informáciu o tom, o akú štruktúru sa jedná.

Pravdepodobné vysvetlenie tohto problému je, že pri použití staršej verzie firmvéru je nutné použiť na nastavovanie kľúča rozhranie *wireless-extensions*,

Význam	Hodnota
CIPHER_SUITE_NONE	0
CIPHER_SUITE_WEP_64	1
CIPHER_SUITE_TKIP	2
CIPHER_SUITE_AES	3
CIPHER_SUITE_CCX	4
CIPHER_SUITE_WEP_128	5

Tabuľka 3.4: Možné hodnoty pre *CipherDefaultKeyValue[38]*, čerpané z [2]

Význam	Bit
Neznámy význam (vždy 1)	1
Párový *pairwise* kľúč	2
Kľúč používaný na vysielanie	4

Tabuľka 3.5: Možné hodnoty príznakových bitov *CipherDefaultKeyValue[39]*, čerpané z [2]

zatiaľ čo pri novšej verzii je možné používať jedine špeciálne rozhranie exportované ovládačom pre *wpa_supplicant*. Pri nedodržaní týchto podmienok sa do MIB zapíšu nesprávne hodnoty, čo spôsobí nekorektné fungovatnie adaptéra.

3.2.4 Štruktúra *MAC_ENCRYPTION_MIB*

V tejto časti sa zameriam na analýzu práce s *MAC_ENCRYPTION_MIB*, pričom ako podklad budem brať primárne funkcie *SetEncryptionStatus*, *SetEncryptionKeyS* z `command.c`, ako tiež *atmel_wpa_ioctl* z `vnetusba.c`.

Manipulácia so šifrovacími kľúčmi prebieha pomocou úprav *CipherDefaultKeyValue*, pričom každá položka v tomto poli určuje jeden kľúč, s ktorým sa bude pracovať. Každá položka má veľkosť 40 bajtov, pričom posledné dva bajty sa majú špeciálny význam:

- *CipherDefaultKeyValue[38]* určuje, o aký typ kľúča sa jedná, pričom možnosti sú popísané v tabuľke 3.4.
- *CipherDefaultKeyValue[39]* je použité ako pole príznakových bitov, pričom významy sú popísane v tabuľke 3.5.

Okrem vlastného nastavenia kľúča potrebuje zariadenie určiť, ktorý kľúč má používať na komunikáciu so zariadením, na čo sa používajú *CipherDefaultKeyID* a *CipherDefaultGroupKeyID*, ktoré určujú štandardný párový a skupinový kľúč. Ďalšou hodnotou v štruktúre je *ExcludeUnencrypted*, ktorú ovládač štandardne

necháva nastavenú na 1. Hodnota *WEPEncryptionType* zostáva nastavená na 0 a hodnota *CkipKeyPermutation* sa v ovládači nastavuje buď na 0, alebo na hodnotu rovnomennej premennej z privátnej štruktúry ovládača, ktorú ale nič nenastavuje. Implementácia podpory pre CKIP však presahuje rámec tejto práce, keďže sa nejedná o súčasť WPA.

3.3 Komunikácia cez *bulk endpoint*

Po tomto koncovom bode prebieha odosielanie a príjem rámcov z bezdrôtovej siete. Navyše, každý z nich má na začiatok pridanú hlavičku, ktorou sa prenášajú niektoré informácie o okolnostiach prijatia alebo parametroch odosielania. V pôvodnom firmvéri mali podľa [1] tvar, ako je vypísané v listingu 3.3 a 3.4.

Listing 3.3: Formát prijímaných dát

```
typedef struct __RX_BUFFER {
        USHORT WLength;
        USHORT RxRate;
        UINT8 NewBSS;
        UINT8 Reserved;
        UCHAR RSSI;
        UCHAR Reserved1[6];
        UCHAR WirelessPacket[MAX_WIRELESS_SIZE];
}
```

Listing 3.4: Formát odosielaných dát

```
typedef struct __TX_BUFFER {
        USHORT WLength;
        UCHAR TxRate;
        UCHAR PaddingBytes;
        UCHAR Reserved[4];
        UCHAR WirelessPacket[MAX_WIRELESS_SIZE];
}
```

V [2] sa namiesto týchto štruktúr už používajú ich mierne pozmenené varianty (súbor `src/includes/usb/usbvneta.h`), a to 3.5 a 3.6.

Listing 3.5: Nový formát prijímaných dát

```
typedef struct __RX_BUFFER {
        USHORT WLength;
```

```
        UCHAR RxRate;
        UCHAR newbss;
        UCHAR KeyIndex;
        char RSSI;
        UCHAR LinkQuality;
        UCHAR CipherType;
        ULONG Reserved;
        UCHAR WirelessPacket[MAX_WIRELESS_SIZE];
}
```

Listing 3.6: Nový formát odosielaných dát

```
typedef struct __TX_BUFFER {
        USHORT WLength;
        UCHAR TxRate;
        UCHAR PaddingBytes;
        UCHAR KeyID;
        UCHAR CipherType;
        UCHAR CipherLength;
        UCHAR Reserved;
        UCHAR WirelessPacket[MAX_WIRELESS_SIZE];
}
```

Tieto štruktúry sú už kompatibilné do miery, kedy je možné, aby pri použití ľubovoľnej verzie firmvéru bol ovládač schopný pracovať s ním

3.4 Záver

Analýzou rozdielov bolo zistené, že bola zmenená štruktúra jedného MIB, ako aj definovaný význam pre niektoré predtým vyhradené premenné pre prijímací a odosielací buffer. Tieto rozdiely bude potrebné implementovať do at76_usb.

Kapitola 4

Implementácia

Ako už bolo v úvode spomenuté, táto práca si kladie za úlohu rozšíriť už existujúci ovládač, ktorý je dostupný vo vetve linuxového jadra *wireless-testing* – vetvy jadra určenej primárne na vývoj bezdrôtového subsystému a ovládačov na bezdrôtové adaptéry. Podrobnejšie informácie o použitých súboroch sú uvedené v zozname literatúry, položka [3].

V tejto verzii tento ovládač taktiež využíva *mac80211* stack, ktorý zabezpečuje sa stará o komunikáciu po bezdrôtovej sieti v prípadoch, ktoré neošetruje priamo hardvér. Sústrediť sa budem na funkcionalitu poskytovaný novým firmvérom. Samotná implementácia bola rozdelená do niekoľkých fáz, z ktorých každá nutne vyžaduje dokončenie predchádzajúcich:

1. Kontrola minimálnej funkcnionality ovládača a oprava prípadných zistených nedostatkov.
2. Implementácia podpory pre WEP64 a WEP128 (splnením tohto cieľa sa ovládač dosiahne úroveň funkcionality, akú mal so starým firmvérom).
3. Implementácia podpory pre WPA.

V pôvodnom implementačnom pláne, ktorého sa mala táto práca držať, chýbala prvá časť, bohužiaľ sa pri pokuse o začiatok druhou časťou ukázala ako nevyhnutná.

Na vývoj a ladenie bolo okrem výpočtovej techniky k použité nasledovné vybavenie:

- USB adaptér at76c503a (testovacie zariadenie)
- Orinoco Gold card s firmvérom verzie 6.16 – IEEE 802.11b PCMCIA karta použitá na monitorovanie bezdrôtovej siete.

- Asus WL-500g Deluxe, Asus WL-500g Premium a Linksys WRT54GS – bezdrôtové prístupové body použité na vytvorenie bezdrôtovej siete a testovanie komunikácie pomocou at76c503a USB adaptéra.

V nasledujúcich častiach postupne popíšem jednotlivé implementačné fázy, postupne vstupný stav ovládača pred úpravami, potrebné kroky a stav ovládača po vykonaní potrebných úprav.

4.1 Minimálna funkčnosť

Vstupný stav: pôvodný stav ovládača nachádzajúceho sa vo *wireless-testing*.

Počas tejto fázy bolo potrebné spraviť niekoľko základných úprav, pričom testovanie prebiehalo v otvorenej sieti bez použitia akéhokoľvek šifrovania. Postupne boli zistené nasledovné problémy.

- Po vykonaní vyhľadávania sietí pomocou príkazu `iwlist wlan0 scan` už sú vždy výsledky vyhľadávania prázdne. Tento problém spôsobovalo dvojité volanie `mutex_lock`, zapríčiňujúce deadlock.
- Vrstva *mac80211* nikdy nevolá odosielaciu funkciu ovládača. Tento problém bol spôsobený chýbajúcim volaním funkcií *ieee80211_start_queues*.
- Systém z karty prijíma nekorektné dáta. Po dlhšom testovaní bolo zistené, že prijímané dáta majú odseknuté posledné 4 bajty. Nasledovným porovnaním s [2] bolo zistené, že zariadenia s označením *RFMD*, *R505* a *R505_2958* odstraňujú z príjatých rámcov *FCS*. Nastavením správneho príznaku pre *mac80211* sa tento problém vyriešil.
- Pri použití príkazu `ping` postupne narastá latencia o násobky základného intervalu odosielania paketov. Toto bolo spôsobené chýbajúcim volaním *ieee80211_wake_queues*, vďaka ktorému sa fronta odosielaných rámcov vyprázdňovala iba sporadicky.

Výstupný stav: Ovládač je schopný pripojiť sa do nešifrovanej siete, odosielať a prijímať dáta.

4.2 Podpora pre WEP

Vstupný stav: Ovládač schopný komunikácie po nešifrovanej sieti.

V tejto časti bola hlavnou úlohou zmena funkcie *at76_set_key* tak, aby dokázala namiesto starej štruktúry `MIB_MAC_WEP` pracovať s novšou `MIB_MAC_ENCRYPTION`. Na základe informácií z predchádzajúcej kapitoly bola vytvorená

funkcia ukladajúca WEP kľúče na korektnú pozíciu v MIB a zároveň meniaca hodnoty východzích skupinových a párových kľúčov. Ostatné hodnoty tejto MIB boli nastavené na hodnoty použité v [2] – *privacy_invoked* a *exclude_unencrypted* na 1, zvyšné hodnoty na 0. Ďalej bolo implementované odstraňovanie kľúčov z adaptéra, ako aj nastavovanie indexov východzích kľúčov.

Po týchto úpravách a nastavení správneho WEP kľúča bol ovládač schopný pripojť sa do siete zabezpečenej pomocou WEP so 64- aj 128-bitovým kľúčom. Navyše, počas úprav bola snaha o zachovanie spätnej kompatibility so zariadeniami so starším firmvérom, takže v ideálnom prípade by ovládač mal byť schpný fungovať s oboma verziami firmvéru bez väčších problémov.

Výstupný stav: Ovládač má implementovanú základnú podporu pre nový firmvér, ako aj podporu pre WEP.

4.3 Podpora pre WPA

Vstupný stav: Ovládač podporujúci základné funkcie nového firmvéru a WEP

V tomto prípade vďaka už implementovanej funkcii pre nastavovanie kľúčov stačilo jej rozšírenie pre kľúče TKIP a AES/CCMP. Opäť sa ale vyskytlo nieľko problémov.

- Nemožnosť korektne sa pripojiť do bezdrôtovej siete. Toto bolo spôsobené nenastavovaním *TKIPBSSID* v `MIB_MAC_ENCRYPTION`.
- Po odoslaní nieľkokých rámcov bolo zariadenie deautentifikované s odôvodnením 14. Podľa tento kód znamená *Message integrity code (MIC) failure*. Na vyriešenie tohto problému bolo nutné WPA kľúčom nastaviť príznak, vďaka ktorému bude *mac80211* generovať *Message Integrity Code (MIC)*. Po tejto úprave komunikácia prebiehala bez väčších problémov.

Bohužiaľ z funkčnosti bolo možné otestovať jedine podporu pre TKIP, keďže AES/CCMP nebolo s adaptérom, ktoré bolo k dispozícii, podporované. Podľa podporujú tento druh šifrovania iba niektoré USB adaptéry, a to konkrétne *at76c505a* a *at76c505amx*.

Výstupný stav: Zariadenie je schopné úspešne komunikovať pomocou TKIP.

4.4 Spätná kompatibilita

Súčasťou úprav v predchádzajúcich častiach je tiež kód, ktorý kontroluje prítomnú verziu firmvéru a podľa toho určuje, či sa použije stará alebo nová implementácia. Hoci v princípe nemusí byť nutné zachovávať spätnú kompatibilitu,

v niektorých prípadoch môže byť výhodnejšie stále používať starší firmvér. Typickým príkladom môže byť zariadenie, ktoré má firmvér uložený v EEPROM pamäti a ktorého aktualizácia je minimálne nepohodlná, v praxi však často aj spojená s rizikom trvalého poškodenia zariadenia, spôsobeného nekorektnou aktualizáciu firmvéru.

4.5 Záver

V troch implementačných fázach boli postupne do ovládača pridané všetky rozdiely nevyhnutné na korektné fungovanie s novým firmvérom. Každá z častí predstavuje logický celok, ktorý potrebuje predchádzajúce časti, no nie je závislý na nasledujúcich.

Reálna implementácia pozostáva zo sady *patchov*, pričom dôraz bol kladený hlavne na oddelenie opráv, ktoré opravovali samotný pôvodný at76_usb ovládač a úprav, ktoré pridávali podporu pre nový firmvér. Tento prístup by mal umožniť pohodlnejšiu integráciu úprav do jadra, kedy opravy môžu byť aplikované okamžite, pretože väčšinou zavádzajú iba menšie množstvo kódu. Väčšie úpravy môžu naopak byť najskôr otestované a až neskôr začlenené do jadra.

Kapitola 5

Záver

5.1 Zhrnutie

Cieľ stanovený na začiatku tejto bakalárskej práce je úpravami popísanými v predchádzajúcich častiach splnený, teda adaptér má implementovanú podporu pre WPA v klientskom (STA) režime. Toto posúva zariadenia tejto triedy do kategórie, ktorú je dnes opäť možné bezpečne použiť na bezdrôtovú komunikáciu.

Toto vylepšenie ale nie je jediné, ktoré ovládač at76_usb potrebuje. Na podporu čo najširšieho spektra sietí mu chýba podpora pre ad-hoc režim, ďalej nie sú využité niektoré možnosti pre správu napájania. Tieto témy môžu byť námetom pre ďalšie bakalárske práce.

Literatúra

[1] Atmel Corporation, *Preliminary Driver-Firmware Interface Specification of AT76C503A-RFMD*. Atmel Corporation, 2000, nie je voľne dostupné

[2] Atmel Corporation, *atmelwlandriver 3.4.1.1,* `http://atmelwlandriver.sourceforge.net`, 2004.

[3] Kurth O., Albert J., Alex, Jones N., Seeber B., Roskin P., Guenther G., Valo K., *at76_usb snapshot*, `http://git.kernel.org/?p=linux/kernel/git/linville/wireless-testing.git;a=blob;f=drivers/net/wireless/at76_usb.c;hb=b3e9bfcaef5014b3f908906b05cbf0e12aaabe30`, `http://git.kernel.org/?p=linux/kernel/git/linville/wireless-testing.git;a=blob;f=drivers/net/wireless/at76_usb.h;hb=7a6f03419571c0a01222059454f9e60abb48c40d`

2008

[4] IEEE Computer Society, *IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007*

Prílohy

Ku textovej verzii bakalaárskej práce je priložené CD s nasledovným obsahom:

- adresár `at76_usb` obsahuje zdrojové súbory pôvodného ovládača v stave, na aký sa odkazuje [3]
- adresár `patches` obsahuje súbory so zmenami ktoré boli spomínané v implementačnej časti.
- adresár `result` obsahuje modifikované zdrojové súbory po aplikovaní potrebných úprav.
- súbor `readme.txt` obsahuje informácie o obsahu CD.