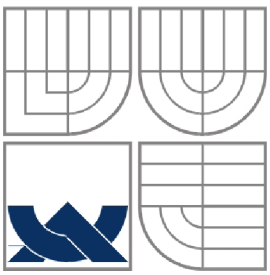
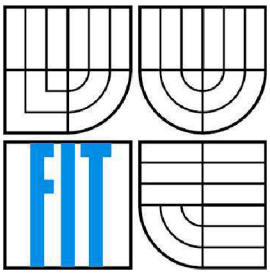


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ANALÝZA A REKONSTRUKCE KOMUNIKACE TYPU INSTANT MESSAGING (YMSG A ICQ)

INSTANT MESSAGING NETWORK ANALYSIS AND RECONSTRUCTION (YMSG AND ICQ)

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

JAN PLUSKAL

VEDOUCÍ PRÁCE
SUPERVISOR

ING. VLADIMÍR VESELÝ

BRNO 2012

Abstrakt

Cílem této bakalářské práce je analýza a rekonstrukce instantní komunikace protokoly YMSG a ICQ. Práce obsahuje detailní popis struktury těchto protokolů a přenášených zpráv významných pro rekonstrukci. Dále je popsáno vytvoření experimentálního nástroje pro rekonstrukci YMSG komunikace, kterým je ověřena správnost analýzy tohoto protokolu. Pro oba protokoly jsou taktéž vytvořeny popisy v jazyce NPL, které jsou použity v nástroji pro analýzu komunikace MS Network Monitoru.

Abstract

The goal of this Bachelor's thesis is analysis and reconstruction of network communication using YMSG and ICQ protocols. This paper contains detail description and message structure of these two protocols. Subsequently creation of experimental tool is discussed and used for reconstruction of YMSG communication. Results show that output of this tool is correct with reference to YMSG protocol. Thesis also outlines MS Network Monitor and introduces specialized NPL parsers for YMSG and ICQ.

Klíčová slova

Rekonstrukce instantní komunikace, analýza protokolu, ICQ, OSCAR, Yahoo!, YMSG

Keywords

Instant messaging network reconstruction, protocol analysis, ICQ, OSCAR, Yahoo!, YMSG

Citace

PLUSKAL, Jan. *ANALÝZA A REKONSTRUKCE KOMUNIKACE TYPU INSTANT MESSAGING (YMSG A ICQ)*. Brno, 2012. Bakalářská práce. FIT VUT. Vedoucí práce Ing. Vladimír Veselý.

Analýza a rekonstrukce komunikace typu instant messaging (YMSG a ICQ)

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Vladimíra Veselého. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Jan Pluskal
Datum (30.04.2012)

Poděkování

Na tomto místě bych rád poděkoval Ing. Vladimíru Veselému za vedení bakalářské práce, jeho rady a návrhy, trpělivost a několikanásobnou stylistickou korekturu a to nejen této práce. Taktéž bych rád poděkoval svým rodičům za jejich pomoc a subvenci, bez které bych tuto školu nemohl studovat a zároveň i mé přítelkyni za trpělivost a morální podporu při tvorbě této práce.

Recept na pizzu: 250g hladké mouky, 250g polohrubé mouky, ½ kvasnic, 300ml vody, 2 lžičky soli a cukru. Výše uvedené ingredience smícháme dohromady a vzniklé těsto necháme za občasného promíchání kynout cca hodinu. K vytvoření těsta doporučuji použít domácí pekárnu. Nakynuté těsto rozdělíme napůl a vyválíme. Nachystáme si 2 plechy, které lehce podsypeme a na ně vložíme vyválená těsta. Ozdobíme podle chuti a ingrediencí, kterým v lednici končí spotřební lhůta. Zhruba doporučuji v tomto pořadí směs ostrého a jemného kečupu, rajčata nakrájená na tenké plátky, plátky šunky natrhané na kousky, mezi ně olivy, posypat oreganem a krájenou paprikou na malé kousky, dále posypat nastrouhaným Eidam sýrem cca 200-300g, dozdobit např. kolečky uzené klobásy, jiným druhem sýru, tvarůžky, pasírovaným česnekem či smetanou.

Tip na závěr: Křupavé okraje plněné sýrem. Těsto vyválíme o pár centimetrů větší než je plocha plechu a na kraje vložíme sýr Eidam cihlu, který nakrájíme na kvádříky cca 5x5x100mm. Přesahující těsto přeložíme přes kousky sýru a zmáčkneme, aby sýr při pečení nevytekl.
Foto: http://db.tt/Swe7iZSZ

Obsah

1 Úvod

Rekonstrukce komunikace je proces získávání informací ze zachyceného síťového provozu. Jedná se o netriviální problém, který je třeba řešit inženýrským přístupem. Celému snažení předchází důkladná analýza použitých protokolů a samozřejmě je nutná i znalost protokolů nižších vrstev, aby bylo možné provést korektní vypouzdření dat.

V této práci se budeme zaobírat analýzou dvou konkrétních protokolů a následnou implementací experimentálního nástroje v jazyce Python, který bude schopen analyzovat zachycenou komunikaci a důležité informace uložit ve formátu *XML*. Následně bude popsáno vytvoření analyzačních skriptů popisujících strukturu obou protokolů v jazyce *NPL*, který je používán nástrojem pro zachycení a analýzu síťové komunikace *Microsoft Network Monitorem (MNM)* [1].

Popsány budou protokoly pro komunikaci typu instant messaging. První z nich je vyvinutý společností Yahoo! Inc. Nazývá se *YMSG* a pokud bychom se řídili následující definicí, která vysvětluje pojem *instant messaging*, mohli bychom prohlásit, že se jedná o *Out-of-Bound protokol*, protože přenášené soubory, nebo multimediální data, jsou přenášena jinými protokoly. Druhým je komunikační protokol společnosti *ICQ*. Jedná se o podmnožinu binárního protokolu s názvem *OSCAR*, který byl vyvinut společností *AOL*.

Instant messaging (zkratka IM) je internetová služba, umožňující svým uživatelům sledovat, kteří jejich přátelé jsou právě připojeni, a dle potřeby jim posílat zprávy, chatovat, přeposílat soubory mezi uživateli a i jinak komunikovat. Hlavní výhodou oproti používání např. e-mailu spočívá v principu odesílání a přijímání zpráv v reálném čase. Jinými slovy zpráva je doručena ve velmi krátké době od odeslání (většinou v rámci stovek milisekund). [2]

Jelikož tématem práce je rekonstrukce komunikace, nebude nutné porozumět všem typům zpráv a jejich komplexnímu obsahu, ale pouze určité podmnožině, jejímž hledáním se budeme zabývat postupně v části věnované analýze.

Po určení klíčových typů zpráv a identifikaci jejich obsahu je možné přejít k vytvoření modelu. Tím bude skript v Pythonu, který bude implementován podle výsledků analýzy, a její závěry v něm budou ověřeny v praxi. Do jisté míry je možné tímto zjistit, zda projekt byl úspěšný a výstup skriptu opravdu obsahuje všechny podstatné informace o provedené komunikaci klienta.

Protože protokoly nejsou otevřené, ale jedná se o proprietární protokoly, není možné zaručit správnost všech tvrzení, která se v této práci objeví. Většina poznatků je zjištěna odposloucháváním komunikace na straně klienta a pozdější analýzou zachycených PCAP souborů pomocí nástroje *Wireshark* [3] a *MNM*. Dalším užitečným zdrojem informací se ukázaly být různá webová fóra a osobní stránky odborníků, zkoumajících tento protokol. Nicméně informace zde nalezené se často navzájem lišily a bylo nutné tento zdroj použít pouze jako ověření poznatků zjištěných analýzou komunikace.

Práce je rozčleněna do následujících kapitol:

- *Kapitola 2 Analýza komunikace po ICQ* – zde jsou popsány základní struktury použité pro komunikaci *OSCAR* protokolem, významy jednotlivých zasílaných zpráv a je zde provedena identifikace klíčových událostí.
- *Kapitola 3 Analýza protokolu YMSG* – v této kapitole je popsán typ a struktura protokolu *YMSG*, jsou zde identifikovány zprávy zasílané mezi klientem a serverem, jejich význam a klíčové události.
- *Kapitola 4 Experimentální nástroj pro rekonstrukci YMSG* – kapitola popisuje návrh a implementaci experimentálního nástroje, který je napsán v jazyce Python a rekonstruuje zachycenou komunikaci *YMSG* protokolu uloženou ve formátu *PCAP* souboru do výstupního *XML* podle předem daného schématu.
- *Kapitola 5 Rekonstrukce IM za použití NPL* – tato kapitola objasňuje vytvoření a význam analyzačních skriptů v jazyce *NPL* pro analýzu jak živé, tak i uložené komunikace za pomoci nástroje *MNM*.

2 Analýza komunikace po ICQ

Instantní komunikace po *ICQ* je podmnožinou komplexního protokolu *OSCAR*. Ten byl vyvinut společností *AOL* a zkratka znamená *Open System for CommunicAtion in Realtime*. Protokol je určen na komunikaci v reálném čase. Navzdory svému názvu není otevřený, proto nebylo možné při analýze čerpat z oficiální dokumentace. Z tohoto důvodu jako podklady pro sepsání této kapitoly sloužily informace získané z internetových fór a osobních stránek expertů zabývajících se jejich rekonstrukcí [4] [5]. Relevantnost těchto informací bylo nutné ověřit v praxi. Proto byly použity jako předloha pro následnou analýzu metodou reverzního inženýrství za pomoci nástroje *Wireshark*.

2.1 Struktura protokolu OSCAR

Protokol *OSCAR* se skládá z několika vrstev. Komunikace probíhá nad *TCP/IP* protokolem na portu 5050. Na nejnižší vrstvě zprávy zaslané protokolem *OSCAR* je nízko úrovňový komunikační protokol s názvem *FLAP*. O vrstvu výš se obvykle nachází základní komunikační jednotka s názvem *SNAC*. Ta ve své datové části zpravidla obsahuje data ve formátu *TLV*.

V této kapitole bude popsána základní struktura protokolu *OSCAR*, jímž je na nejnižší úrovni protokol *FLAP* – popsán v *Kapitole 2.1.1*. Následuje komunikační jednotka SNAC přenášející jednotlivé zprávy protokolu – *Kapitola 2.1.2*. Posledním popsaným prvkem protokolu je *TLV* záznam – *Kapitola 2.1.3*. Pro úplnost v poslední *Kapitole 2.1.5* jsou uvedeny konstanty kódující stavy, v kterých se může kontakt nacházet.

Mimo oficiálního klienta ICQ stejnojmenné společnosti, který je aktuálně ve verzi 7.7, jsou k dispozici i další neoficiální programy, které zpravidla podporují základní funkčnost jako je zasílání textových zpráv, manipulace s listem kontaktů, vyhledávání kontaktů a do jisté míry zasílání souborů. Tito neoficiální klienti vznikali buď jako port *ICQ* na platformy, které nejsou oficiálně podporovány, jako například *Windows Phone 7*, *Bada* nebo dříve i *Linux* a *MacOS*. Mezi alternativní klienty používané na těchto platformách patří např. Pidgin, Empathy, Licq a nebo univerzální klienti, kteří jsou schopni komunikovat více protokoly a umožní být tzv. online na více službách zároveň, někdy i na více platformách současně. Takovým klientem je například Trillian a nebo Meebo, kteří využívají tzv. bran, které jsou spuštěny na serveru, připojí se k požadované službě a veškerá komunikace probíhá přes tyto brány. Klientské zařízení např. *PC* nebo *Smartphone* se připojí a komunikuje pomocí jediného protokolu se serverem, kde jsou spuštěny tyto brány a ty přeposílají zprávy do dané sítě pomocí protokolu této sítě. Tato implementace umožňuje, aby byl uživatel přihlášen na několika místech současně a zprávy mu budou doručeny na všechna místa, kde je aktuálně přihlášen.

2.1.1 Protokol FLAP

Jedná se o nízko úrovňový komunikační protokol, který umožňuje vývoj vyšších datagramově orientovaných komunikačních vrstev. Je použit ve všech typech zpráv zasílaných mezi klienty a serverem. Počáteční byte nabývá hodnoty *0x2A* a je jím signalizován začátek datagramu.

Pro následující tabulky bude zavedena konvence: *byte* značí hodnotu *8 bitů*, *word* jsou *2 byty*, *dword 4 byty* a *qword 8 bytů*. Pokud se ve sloupci hodnota vyskytne konstanta, znamená to, že se na tomto místě v protokolu musí vyskytovat stále tato konstantní hodnota, pokud je zde uvedeno xx, nachází se na tomto místě proměnná. Pokud bude pozadí rozděleno na několik odstínů šedi, znamená to, že řádky se stejným odstínem spolu významově souvisí.

Hodnota	Velikost	Jméno pole
2A	byte	FLAP id byte
xx	byte	FLAP channel
xx xx	word	FLAP datagram seq number
xx xx	word	FLAP data size
.....		FLAP data

Tabulka 2.1 Formát FLAP protokolu

Významy jednotlivých položek:

- *FLAP id byte* – Identifikuje začátek rámce.
- *FLAP channel* – Identifikuje použitý kanál. Kanály jsou použity pro rozdělení komunikace na několik nezávislých toků. Momentálně *OSCAR* protokol využívá tyto kanály:
  - *0x01* – Vyjednání nového připojení.
  - *0x02* – *SNAC* data viz *Kapitola 2.1.2.*
  - *0x03* – *FLAP* nízko úrovňová chyba.
  - *0x04* – Vyjednání uzavření existujícího připojení.
  - *0x05* – Tzv. *Keep alive*, tedy udržování otevřeného spojení.
- *FLAP datagram seq number* – Použito pro detekci chyby. Je vybíráno náhodně. Sekvenční číslo použité pro odchozí zprávy u klienta nemá žádnou souvislost s číslem použitým serverem. Rozsah je *0x0000 – 0x8000,* v případě přetečení je čítač vynulován.
- *FLAP data size* – Určuje velikost dat v datové části.
- *FLAP data* – Datová část, v níž jsou zapouzdřena data, nebo další protokoly podle *FLAP channel* identifikátoru.

2.1.2 Komunikační jednotka SNAC

Jedná se o základní komunikační jednotku použitou k předání zprávy mezi serverem a klienty. Nachází se v datové části *FLAP* rámce, pouze pokud je použit kanál *0x02*. Tento kanál je používán pro většinu komunikace v mezidobí vyjednání otevřeného spojení a jeho ukončením. V jednom *FLAP* rámci se může vyskytovat pouze jedna *SNAC* jednotka.

Hodnota	Velikost	Jméno pole
xx xx	word	Family (service) id number
xx xx	word	Family subtype id number
xx xx	word	SNAC flags
xx xx	word	SNAC request id
.....		SNAC data

Tabulka 2.2 SNAC komunikační jednotka

Význam jednotlivých položek:

- *Family id number* – Identifikuje rodinu komunikační jednotky.
- *Family id subtype* – Identifikuje podtyp konkrétní rodiny komunikační jednotky.
- *SNAC flags* – Pokud *bit1* obou značek je roven *1* existuje více *SNAC* jednotek pro daný *SNAC request id*. Pokud *bit16* je roven *1* znamená to, že *SNAC* obsahuje neznámou informaci na začátku jednotky. První dva bajty označují délku a za nimi následují samotná data. Významy ostatních bitů jsou neznámé.
- *SNAC request id* – Identifikuje požadavek. Může být náhodné číslo. Přiřazuje požadavek k odpovědi.
- *SNAC data* – Datová část.

Datová část obsahuje užitečná data ve formátu *TVL,* viz *Kapitola 2.1.3*. Je ale také možné, aby na začátku obsahovala pevně danou datovou strukturu a za ní až následovala *TLV data*. Je tomu tak například u *SNAC(0x0004,0x0006)/CH2,* viz *Kapitola 2.2.8*

2.1.3 Struktura TLV

Zkratka *TVL* (Type – Length – Value) označuje velmi výhodnou a efektivní metodu pro organizování dat a jejich přenos po síti, obzvlášť pokud není předem známa velikost jednotlivých typů dat, jako tomu je například u řetězců.

TLV záznam se nachází v datové části *SNAC* jednotky v tomto formátu viz *Tabulka* 2.3.

Hodnota	Velikost	Jméno pole
xx xx	word	TLV type number
xx xx	word	TLV length value
.....		TVL data

Tabulka 2.3 Struktura trojice TLV

Význam jednotlivých položek:

- *TLV type number* – Typ jednotlivé položky.
- *TLV length value* – Délka datové části v bajtech.
- *TVL data* – Datová část.

Význam typu jednotlivých položek je nejen přímo závislý na rodině a podtypu *SNAC* jednotky, ale také na informaci nacházející se ve struktuře obsažené v datové části *SNAC*. Tato struktura se zde může, ale také nemusí nacházet, záleží na typu *SNAC*.

2.1.4 Family id number

Family id number označuje rodinu, k jaké bude konkrétní *SNAC* příslušet. *OSCAR* protokol podporuje značné množství různých skupin *SNAC*. Dále jsou uvedeny pouze ty, které se vyskytují v *ICQ*.

Family id number	Význam
0x0001	Obecné řídící příkazy
0x0002	Lokalizační služby
0x0003	Operace s klientským listem kontaktů
0x0004	ICBM (zpráva) služby
0x0009	Nastavení soukromí
0x000b	Statistiky používání
0x0010	Správa ikon kontaktů na straně serveru (SSBI)
0x0013	Informace na straně serveru (SSI)
0x0015	ICQ služba specifických rozšíření
0x0017	Autorizační / registrační služby
0x0085	Služby všesměrového vysílání - IServerd rozšíření

Tabulka 2.4 Rodiny *SNAC* vyskytující se v *ICQ*

2.1.5 Konstanty

Následuje soupis konstant a jejich význam, kterých je využito pro přenos informací v *OSCAR* protokolu. V následujícím textu na místech, kde se ve *SNAC* strukturách nachází pole s těmito konstantami, bude odkazováno na tyto tabulky.

Hodnota	Zkratka	Význam
0x0001	STATUS_WEBAWARE	Příznak povolení zobrazení stavu na webu
0x0002	STATUS_SHOWIP	Příznak povolení zobrazení IP
0x0008	STATUS_BIRTHDAY	Příznak narozenin uživatele
0x0020	STATUS_WEBFRONT	Příznak *webfront* aktivity uživatele
0x0100	STATUS_DCDISABLED	Přímé spojení není podporováno
0x1000	STATUS_DCAUTH	Přímé spojení na základě autorizace
0x2000	STATUS_DCCONT	Přímé spojení pouze s kontakty uživatele

Tabulka 2.5 Význam prvního slova typu Uživatelský status

Hodnota	Zkratka	Význam
0x0000	STATUS_ONLINE	Online
0x0001	STATUS_AWAY	Pryč
0x0002	STATUS_DND	Neručit (DND)
0x0004	STATUS_NA	Nedostupný (N/A)
0x0010	STATUS_OCCUPIED	Zaneprázdněný (BUSY)
0x0011	STATUS_OCCUPIED	Zaneprázdněný (BUSY)
0x0020	STATUS_FREE4CHAT	Volný pro chat
0x0100	STATUS_INVISIBLE	Neviditelný
0x0111	STATUS_INVISIBLE	Neviditelný

Tabulka 2.6 Význam druhého slova typu Uživatelský status

2.2 Důležité SNAC pro rekonstrukci

Následující kapitoly označují důležité typy *SNAC*, které je třeba rozpoznat, protože nesou informace, jejichž rekonstrukce je naším cílem. Jméno kapitoly označuje jednotlivou rodinu a podtyp *SNAC* spolu s pojmenováním, které bylo převzato z neoficiální dokumentace [5].

Použité „_“ nebo „x“ v označení názvů jednotlivých *SNAC* slouží pouze jako oddělovač a nemá žádný hlubší význam. Toto označení bylo převzato z výše citované neoficiální dokumentace.

2.2.1 SNAC(0x0001,0x001E) CLI_SET_STATUS

Tento *SNAC* je použit klientem pro změnu jeho statusu *TLV 0x06*, nebo informace o přímém připojení *TLV 0x0C*. Obsahuje také informace o poslední změně *TLV 0x11*. Pro rekonstrukci je třeba pouze informace o změně statusu *TLV 0x06*.

Hodnota	Velikost	Popis
00 06	word	TLV.Typ(0x06) – uživatelský status / příznak
xx xx	word	TLV.Délka
xx xx xx xx	dword	Uživatelský status / jeho příznak

Tabulka 2.7 TLV 0x06 Změna uživatelského statusu

2.2.2 SNAC(0x0002,0x0006) SRV_USERxONLINExINFO

Tímto *SNAC* odpovídá server na požadavek *SNAC(0x0002,0x0005)* od klienta na zjištění informací o kontaktu. Existují čtyři typy žádostí o informaci. Obecné informace *0x0001*, online informace *0x0002*, zpráva v nepřítomnosti *0x0003* a informace o podporované funkčnosti kontaktu *0x0003*.

Odpovědi na všechny typy žádostí obsahují informaci o statusu *TLV 0x01*, popř. *TLV 0x03* dekódovací řetězec a *0x04* samotná zpráva v nepřítomnosti, pokud o ni byla zaslána žádost. Ostatní *TLV* záznamy nejsou pro rekonstrukci podstatné.

Důležité také je, že tento *SNAC* nezačíná v datové části *TLV* záznamy, jak je obvyklé, ale pevně danou strukturou, viz *Tabulka 2.8.*

Hodnota	Velikost	Jméno pole
00 02	word	Family (service) id number
00 06	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx	char	Uin string length
xx ..	string	Uin string
xx xx	word	Warning level
xx xx	word	Number of TLV in fixed part
.....		TVL records

Tabulka 2.8 SNAC(0x0002,0x0006) SRV_USERxONLINExINFO Informace o kontaktu

Hodnota	Velikost	Popis
00 06	word	TLV.Typ(0x06) – klientský status / příznak
xx xx	word	TLV.Délka
xx xx xx xx	dword	Klientský status a příznak

Tabulka 2.9 TLV 0x06 Uživatelský status

Hodnota	Velikost	Popis
00 03	word	TLV. Typ (0x03) – kódovací řetězec
xx xx	word	TLV. Délka
xx xx	word	Délka kódovacího řetězce
xx ..	string	Kódovací řetězec

Tabulka 2.10 TLV 0x03 Kódování zprávy v nepříomnosti

Hodnota	Velikost	Popis
00 04	word	TLV. Typ (0x04) – zpráva v nepřítomnosti
xx xx	word	TLV. Délka
xx xx	word	Délka zprávy v nepřítomnosti
xx ..	string	Zpráva v nepřítomnosti

Tabulka 2.11TLV 0x04 Uživatelská zpráva v nepřítomnosti

2.2.3 SNAC(0x0002,0x0009) CLI_UPDATExDIRxINFO

Tento *SNAC* přenáší v *TLV* záznamech osobní informace o klientovi, které se nacházejí v jeho kartě na serveru. Je přenášen pouze směrem od klienta k serveru, když klient tyto údaje mění.

Hodnota	Velikost	Popis
00 01	word	TLV.Typ (0x01) – jméno
xx xx	word	TLV. Délka
xx ..	string	Jméno

Tabulka 2.12 TLV pro křestní jméno

Hodnota	Velikost	Popis
00 02	word	TLV.Typ (0x02) – příjmení
xx xx	word	TLV. Délka
xx ..	string	Příjmení

Tabulka 2.13 TLV pro příjmení

Hodnota	Velikost	Popis
00 03	word	TLV.Typ (0x03) – prostřední jméno
xx xx	word	TLV. Délka
xx ..	string	Prostřední jméno

Tabulka 2.14 TLV pro prostřední jméno

Hodnota	Velikost	Popis
00 04	word	TLV.Typ (0x04) – jméno za svobodna
xx xx	word	TLV. Délka
xx ..	string	Jméno za svobodna

Tabulka 2.15 TLV pro příjmení za svobodna

Hodnota	Velikost	Popis
00 06	word	TLV.Typ (0x06) – země
xx xx	word	TLV. Délka
xx ..	string	Země (dvou písmený kód)

Tabulka 2.16 TLV pro zemi

Hodnota	Velikost	Popis
00 07	word	TLV.Typ (0x07) – stát
xx xx	word	TLV. Délka
xx ..	string	Stát

Tabulka 2.17 TLV pro stát

Hodnota	Velikost	Popis
00 08	word	TLV.Typ (0x08) –město
xx xx	word	TLV. Délka
xx ..	string	Město

Tabulka 2.18 TLV pro město

Hodnota	Velikost	Popis
00 0C	word	TLV.Typ (0x0C) – přezdívka
xx xx	word	TLV. Délka
xx ..	string	Přezdívka

Tabulka 2.19 TLV pro přezdívku

Hodnota	Velikost	Popis
00 0D	word	TLV.Typ (0x0D) – PSČ
xx xx	word	TLV. Délka
xx ..	string	PSČ

Tabulka 2.20 TLV pro zip kód

Hodnota	Velikost	Popis
00 21	word	TLV.Typ (0x21) – adresa
xx xx	word	TLV. Délka
xx ..	string	Adresa

Tabulka 2.21 TLV pro adresu

Tento *SNAC* se již v aktuální verzi *ICQ 7.7* nevyužívá, ale je použit HTTP požadavek se syntaxí datové části: jméno operace = jméno pole = hodnota pole. Viz *Příloha 2*.

Některé informace jako jméno, příjmení, přezdívku a adresu server sdělí zpět klientovi ve *SNAC(0x0025,0x0003)*, který v této práci není uveden, protože jeho korektní dekódování nebylo úspěšné.

2.2.4 SNAC(0x0003,0x0004) CLI_BUDDYLIST_ADD, SNAC(0x0003,0x0010) CLI_BUDDYLIST_ADD_7

SNAC je použit pro přidání kontaktů do listu na klientské straně. Je jím možné přidat *jednoho* až *n* kontaktů. V datové části se nenachází *TLV* záznam, ale pouze *LV*, protože typ hodnoty je předem jasný (přenáší se totiž pouze hodnoty jednoho typu). Oba *SNAC* se od sebe nijak neliší.

Hodnota	Velikost	Jméno pole
00 03	word	Family (service) id number
00 04 (10)	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx	byte	Buddy id #1 strlen
xx ..	string	Buddy id #1 string
…		
xx	byte	Buddy id #n strlen
xx ..	string	Buddy id #n string

Tabulka 2.22 SNAC(0x0003,0x0004) CLI_BUDDYLIST_ADD přidání kontaktů SNAC(0x0003,0x00010) CLI_BUDDYLIST_ADD přidání kontaktů

2.2.5 SNAC(0x0003,0x0005) CLI_BUDDYLIST_REMOVE, SNAC(0x0003,0x000F) CLI_BUDDYLIST_REMOVE_7

SNAC je použit pro odstranění kontaktů z listu na klientské straně. Je jím možné odstranit *jednoho* až *n* kontaktů. V datové části se nenachází *TLV* záznam, ale opět pouze *LV*. Oba *SNAC* jsou shodné.

Hodnota	Velikost	Jméno pole
00 03	word	Family (service) id number
00 05 (0F)	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx	byte	Buddy id #1 strlen
xx ..	string	Buddy id #1 string
…		
xx	byte	Buddy id #n strlen
xx ..	string	Buddy id #n string

Tabulka 2.23 SNAC(0x0003,0x0005) CLI_BUDDYLIST_ REMOVE odstranění kontaktů SNAC(0x0003,0x000F) CLI_BUDDYLIST_ REMOVE odstranění kontaktů

2.2.6 SNAC(0x0003,0x000B) SRV_USER_ONLINE

Server zasílá tento *SNAC*, když se uživatel, který se nachází v listu kontaktů, objeví online nebo změní status. V datové části je přenášeno více *TLV* záznamů, ale pro nás je podstatný pouze *TLV 0x06* označující uživatelský status. Tento *SNAC* je také výjimečný, protože v jeho datové části je přenášena pevně daná struktura předcházející *TLV* záznamu. A také může přenášet informace o více kontaktech, přičemž tento počet není implicitně uveden, takže je potřeba porovnat velikost získanou z *FLAP* s velikostí již načtených dat.

Hodnota	Velikost	Jméno pole
00 03	word	Family (service) id number
00 0B	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
Následující informace o uživateli se mohou vyskytovat vícekrát		
xx	char	Uin string length
xx ..	string	Uin string
xx xx	word	Warning level (nepoužité v ICQ)
xx xx	word	Number of TLV in info-tlvlist
… Nějaké TLV		
00 06	word	TLV.Typ(0x06) – status uživatele
xx xx	word	TLV.Délka
xx xx xx xx	dword	Status uživatele, viz Tabulka 2.5, Tabulka 2.6
… Nějaké jiné TLV		

Tabulka 2.24 SNAC(0x0003,0x000B) SRV_USER_ONLINE kontakt se přihlásil, nebo změnil status

2.2.7 SNAC(0x0003,0x000C) SRV_USER_OFFLINE

SNAC zaslaný serverem, když se uživatel, který se nachází v listu kontaktů, odhlásí. V datové části je přenášen pravděpodobně pouze jeden *TLV* záznam *třída uživatele*, který pro nás není podstatný. Tento *SNAC* je opět výjimečný tím, že v jeho datové části je přenášena pevně daná struktura předcházející *TLV* záznam, která obsahuje *uin* řetězec označující kontakt, jenž se odhlásil. Stejně jako předchozí *SNAC* může přenášet informace o více kontaktech.

Hodnota	Velikost	Jméno pole
00 03	word	Family (service) id number
00 0C	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
Následující informace o uživateli se mohou vyskytovat vícekrát		
xx	char	Uin string length
xx ..	string	Uin string
xx xx	word	Warning level (nepoužito v ICQ)
xx xx	word	Number of TLV in info-tlvlist
00 01	word	TLV.Typ(0x01) – třída uživatele
xx xx	word	TLV. Délka
xx xx xx xx	dword	Třída uživatele

Tabulka 2.25 SNAC(0x0003,0x000C) SRV_USER_OFFLINE kontakt se odhlásil

2.2.8 SNAC(0x0004,0x0006) CLI_SEND_ICBM

Klient odesílající zprávu přes server použije tento *SNAC*. Zprávu je možné odeslat několika různými kanály, z nichž každý reprezentuje jiný typ zprávy.

- *Kanál 1* je využit pro odeslání *plain-textových* dat, jedná se o prostý *instant messaging*.
- *Kanál 2* je používán nejčastěji pro odesílání složitějších zpráv (*rtf,utf8*), odeslání pozvání k chatu, nebo oznámení přenosu souboru.
- *Kanál 3* je využit k zasílání zpráv ostatních typů.
- *Kanál 4* je použit pro různé *ICQ* zprávy, nejčastěji pro kontakty se zastaralými klienty, kteří jsou offline nebo neviditelní.
- *Kanál 5* zatím nebyl identifikován, protože se neobjevil v žádném zachyceném *PCAP souboru*.
- *Kanál 6* zapouzdřuje signalizační komunikaci protokolu *SIP*, který je použit pro *VoIP* nebo přenos *videa*.

Obvykle je zpráva zaslaná přes *ICQ* klienta posílána prvním kanálem s formátováním zajištěným *HTML*.

V datové časti *SNAC* se nachází pevně daná struktura obsahující *cookie, kanál, identifikátor příjemce a jeho délku*. Tato část je společná pro všechny kanály. Formát následujících dat je závislý na kanálu.

Hodnota	Velikost	Jméno pole
00 04	word	Family (service) id number
00 06	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx xx xx xx xx xx xx xx	qword	Msg-id cookie
00 02	word	Message channel
xx	byte	Uinstring length
xx ..	string	Uinstring
00 02	word	TLV.Typ(0x02) – data zprávy
xx xx	word	TLV. Délka
05	byte	Fragment identifier (pole vyžadovaných *capabilities*)
01	byte	Fragment version
xx xx	word	Length of rest data
xx ..	array	Byte array of required capabilities (1 - text)
01	byte	Fragment identifier (textové zprávy)
01	byte	Fragment version
xx xx	word	Length of rest data
xx xx	word	Message charset number
xx xx	word	Message language number
xx ..	string (ascii)	Message text string
… Nějaké TLVs		

Tabulka 2.26 SNAC(0x0004,0x0006) CLI_SEND_ICBM pro kanál 1

Následující *SNAC* je využit pro identifikaci přenosu souboru, pokud pole message type v *TLV 0x05* nabývá hodnoty *0x0000,* což značí, že se jedná o požadavek a hodnota capability je rovna *guid 09461343-4c7f-11d1-8222-444553540000.*

Hodnota	Velikost	Jméno pole
00 04	word	Family (service) id number
00 06	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx xx xx xx xx xx xx xx	qword	Msg-id cookie
00 02	word	Message channel
xx	byte	Uinstring length
xx ..	string	Uinstring
00 05	word	TLV.Typ(0x05) - rendezvous message data
xx xx	word	TLV.Délka
xx xx	word	Message type (0 - požadavek, 1 - zrušení, 2 - přijetí)
xx xx xx xx xx xx xx xx	qword	Msg-id cookie (stejné jako výše)
xx .. xx	guid	Capability
… TLV záznamy z následujících tabulek		

Tabulka 2.27 SNAC(0x0004,0x0006) CLI_SEND_ICBM pro kanál 2

Hodnota	Velikost	Popis
00 0A	word	TLV. Typ – Seq
xx xx	word	TLV.Délka
xx ..	string	Sekvenční číslo

Tabulka 2.28 TLV 0x0A Sekvenční číslo

Hodnota	Velikost	Popis
00 02	word	TLV. Typ – Randevous IP
xx xx	word	TLV. Délka
xx ..	string	Randevous IP

Tabulka 2.29 TLV 0x02 Randevous IP

Hodnota	Velikost	Popis
00 16	word	TLV. Typ – XORed
xx xx	word	TLV. Délka
xx ..	string	XORed Randevous IP

Tabulka 2.30 TLV 0x16 XORed Randevous IP

Hodnota	Velikost	Popis
00 03	word	TLV. Typ – Internal ip
xx xx	word	TLV. Délka
xx ..	string	Interní ip

Tabulka 2.31 TLV 0x03 interní ip

Hodnota	Velikost	Popis
00 04	word	TLV.Typ- Otevřený port
xx xx	word	TLV. Délka
xx ..	string	Otevřený port

Tabulka 2.32 TLV 0x04 otevřený port

Hodnota	Velikost	Popis
00 17	word	TLV. Typ – XORed Port
xx xx	word	TLV. Délka
xx ..	string	XORed Port

Tabulka 2.33 TLV 0x17 XORed Port

2.2.9 SNAC(0x0004,0x0007) SRV_SEND_ICBM

Tento *SNAC* zasílá server klientovi a je v něm přenášena zpráva. Je velice podobný *SNAC(0x0003,0x000B)* SRV_USER_ONLINE, ale odlišuje se v několika detailech: jsou přidány dvě pole do struktury v datové části *SNAC* a zpráva může obsahovat v *TLV* informaci o statusu kontaktu.

Hodnota	Velikost	Jméno pole
00 04	word	Family (service) id number
00 07	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx xx xx xx xx xx xx xx	qword	Msg-id cookie
00 02	word	Message channel
xx	byte	Uinstring length
xx ..	string	Uinstring
xx xx	word	Sender warning level
xx xx	word	Number of TLVs in fixed part
… Nějaké TLV		
00 06	word	TLV.Typ(0x06) – Uživatelský status
xx xx	word	TLV.Délka
xx xx xx xx	dword	Uživatelský status, viz Tabulka 2.5, Tabulka 2.6
… Nějaké další TLV		
00 02	word	TLV.Typ(0x02) – Data zprávy
xx xx	word	TLV.Délka
05	byte	Fragment identifier (pole potřebných schopností)
01	byte	fragment version
xx xx	word	Length of rest data
xx ..	array	Byte array of required capabilities (1 - text)
01	byte	Fragment identifier (textové zprávy)
01	byte	Fragment version
xx xx	word	Length of rest data
xx xx	word	Message charset number
xx xx	word	Message language number
xx ..	string (ascii)	Message text string

Tabulka 2.34 SNAC(0x0004,0x0007) SRV_SEND_ICBM pro kanál 1

Následující *SNAC* je opět využit pro identifikaci přenosu souboru, pokud pole *message type* v *TLV 0x05* nabývá hodnoty *0x0000,* což značí, že se jedná o požadavek a hodnota *capability* je rovna *guid 09461343-4c7f-11d1-8222-444553540000*.

Hodnota	Velikost	Jméno pole
00 04	word	Family (service) id number
00 07	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx xx xx xx xx xx xx xx	qword	Msg-id cookie
00 02	word	Message channel
xx	byte	Uinstring length
xx ..	string	Uinstring
xx xx	word	Sender warning level
xx xx	word	Number of TLVs in fixed part
… TLV záznamy		
00 06	word	TLV.Typ(0x06) – uživatelský status
xx xx	word	TLV.Délka
xx xx xx xx	dword	Uživatelský status, viz Tabulka 2.5, Tabulka 2.6
… Další TLV záznamy		
00 05	word	TLV.Typ(0x05) - rendezvous data zprávy
xx xx	word	TLV.Délka
xx xx	word	Message type (0 - požadavek, 1 – zrušení, 2 - přijetí)
xx xx xx xx xx xx xx xx	qword	Msg-id cookie (stejné jako výše)
xx .. xx	guid	Capability
… TLV záznamy následujících tabulek		

Tabulka 2.35 SNAC(0x0004,0x0007) SRV_SEND_ICBM pro kanál 2

Hodnota	Velikost	Popis
00 04	word	TLV.Typ
xx xx	word	TLV.Délka
xx ..	string	Externí ip

Tabulka 2.36 TLV 0x04 externí ip

Hodnota	Velikost	Popis
00 05	word	TLV.Typ
xx xx	word	TLV.Délka
xx ..	string	Otevřený port

Tabulka 2.37 TLV 0x05 otevřený port

2.2.10 SNAC(0x0013,0x0006) CLI_SSIxREPLY

Tento *SNAC* je zasílán serverem jako odpověď na požadavky na načtení listu kontaktů ze serveru nebo ověření platnosti stávajícího. Protokol *OSCAR* podporuje částečné stažení listu kontaktů jako úsporu přenášených dat a to na základě informace o čase změny a počtu kontaktů v listu na straně klienta.

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 06	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
00	byte	Version number of SSI protocol (aktuálně 0x00)
xx xx	word	Number of items in this SNAC
xx ..	struct	List of items *Tabulka 2.39*
xx xx xx xx	dword	SSI list last change time

Tabulka 2.38 SNAC(0x0013,0x0006) CLI_SSIxREPLY aktualizace listu kontaktů

Samotný list je kolekcí struktur. Struktury mohou obsahovat mnoho typů informací, které jsou na straně serveru uloženy právě v tomto listu. Ty server přepošle klientovi, přičemž akce přeposlání neubírá výpočetní výkon pro složité vytvoření listu.

Hodnota	Velikost	Jméno pole
xx xx	word	Length of the item name
xx ..	string	Item name string (uin)
xx xx	word	Group ID#
xx xx	word	Item ID#
xx xx	word	Type of item flag *Tabulka 2.40*
xx xx	word	Length of the additional data
…TLV záznamy		

Tabulka 2.39 Struktura záznamů list items

V předchozí struktuře reprezentované v *Tabulka 2.39* jméno záznamu odpovídá identifikátoru kontaktu (*uin*), jménu skupiny nebo speciálnímu identifikátoru. Všechny skupiny jsou obsaženy v tzv. *Master group*, jejíž *group ID* je *0x0000*. Všechny skupiny mají *item ID 0x0000* a není možné, aby existovali dvě skupiny se stejnými *group/item ID*. Typ položky nacházející se v poli *Type of item flag* nabývá hodnot z *Tabulka 2.40*.

Hodnota	Popis
0x0000	Záznam kontaktu (přezdívka: uin)
0x0001	Záznam skupiny
0x0002	Povolovací záznam („Viditelný“)
0x0003	Zakazovací záznam („Neviditelný“)
0x0004	Povolovací/zakazovací nastavení a or/and bitové masky pro AIM třídy
0x0005	Info o přítomnosti (jestli ostatní mohou vidět nepřítomný status)
0x0009	Neznámé
0x000E	List ignorovaných kontaktů
0x000F	Datum poslední úpravy (jméno: "LastUpdateDate").
0x0010	Kontakty třetích stran (pro poslání SMS). Jméno: 1#EXT, 2#EXT, atd.
0x0013	Seznam importovaných časů (jméno: "Import time")
0x0014	Vlastní ikona (avatar) info. Jméno je id pro avatar.

Tabulka 2.40 Známé typy záznamů

Některé typy položek obsahují dodatečné informace ve formě *TLV* záznamů za strukturou hlavičky. Jediný *TLV* záznam podstatný pro rekonstrukci je obsažen v typu záznamu *0x0001 Záznam skupiny*. Jeho typ je *0x00C8* a obsahuje dvoubajtové identifikátory. Pokud se jedná o *Master group,* jsou to identifikátory podskupin. Pokud o koncovou skupinu, jsou to identifikátory kontaktů.

Hodnota	Velikost	Popis
00 C8	word	TLV.Typ(0xC8) – pod skupina
xx xx	word	TLV.Délka
xx xx ..	word	Několik id

Tabulka 2.41 TLV 0x00C8 záznam podskupin

2.2.11 SNAC(0x0013,0x0008) CLI_SSIxADD

Klient využívá tento *SNAC*, aby přidal záznam některé položky z *Tabulka 2.40* do listu na straně serveru. Struktura *list of items*, její složky a jejich význam spolu s identifikací klíčových částí pro rekonstrukci je detailně popsán v předešlé kapitole.

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 08	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx ..	struct	List of items *Tabulka 2.39*

Tabulka 2.42 SNAC(0x0013,0x0008) CLI_SSIxADD přídání záznamů na server

2.2.12 SNAC(0x0013,0x0009) CLI_SSIxUPDATE

SNAC je použit k úpravě informace v listu, který je už uložen na serveru. Nejčastější použití je při přidání, nebo odebrání kontaktu ze skupiny. Struktura *list of items*, její složky a její význam spolu s identifikací klíčových částí pro rekonstrukci je detailně popsán v *Tabulka 2.39.*

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 09	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx ..	struct	List of items *Tabulka 2.39*

Tabulka 2.43 SNAC(0x0013,0x0009) CLI_SSIxUPDATE úprava záznamů na serveru

2.2.13 SNAC(0x0013,0x000A) CLI_SSIxDELTE

Tento *SNAC* se využívá k odstranění informace v listu, který je uložen na serveru. Nejčastější použití je při odstranění kontaktu nebo prázdné skupiny. Význam jednotlivých položek a struktura *list of items* je popsán v *Tabulka 2.39.*

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 09	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx ..	struct	List of items *Tabulka 2.39*

Tabulka 2.44 SNAC(0x0013,0x0009) CLI_SSIxDELETE odstranění záznamů na serveru

2.2.14 SNAC(0x0013,0x0018) CLI_SSI_SEND_AUTHxREQUEST

Klient používá tento *SNAC* k odeslání požadavku o autorizaci kontaktu. V datové části se nevyskytuje žádný *TLV* záznam, pouze pevná struktura obsahující mimo jiné jméno kontaktu reprezentované *uid* a důvod žádosti o autorizaci.

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 18	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx	byte	Length of the uin string
xx ..	string	Uin string (ascii)
xx xx	word	Length of the reason string
xx xx	word	Reason (zpráva) string
00 00	word	Unknown

Tabulka 2.45 SNAC(0x0013,0x0018) CLI_SSI_SEND_AUTHxREQUEST klient žádá autorizaci

2.2.15 SNAC(0x0013,0x0019) SRV_SSI_AUTHxREQUEST

Server zasílá tento *SNAC* jako požadavek na autorizaci od kontaktu. Struktura je stejná jako v předchozím případě, kdy klient žádal o autorizaci, viz *Tabulka 2.45*. Rozdíl je pouze v tom, že identifikační řetězec *uin* obsahuje identifikaci kontaktu, který o autorizaci žádá.

2.2.16 SNAC(0x0013,0x001A) CLI_SSI_AUTHxREPLY

Tímto *SNAC* klient odpovídá na žádost kontaktu o autorizaci. Neobsahuje žádné *TLV* záznamy, pouze pevně danou strukturu. Oproti předcházejícím *SNAC*, které reprezentovaly žádost, přibyla položka obsahující odpověď na požadavek a to buď s hodnotou *0x01* autorizace udělena, nebo *0x00* zamítnuta.

Hodnota	Velikost	Jméno pole
00 13	word	Family (service) id number
00 1A	word	Family subtype id number
00 00	word	SNAC flags
xx xx	word	SNAC request id
xx	byte	Length of the uin string
xx ..	string	Uin string (ascii)
xx	byte	Flag: 1-přijmuto, 0-zamítnuto
xx xx	word	Length of the reason string
xx ..	string (ascii)	Reason (zpráva) string

Tabulka 2.46 SNAC(0x0013,0x001A) CLI_SSI_AUTHxREPL odpověď na autorizaci

2.3 Identifikace klíčových událostí

V této kapitole budou popsány klíčové události, které odpovídají rekonstruovaným zprávám, jako např. přihlášení klienta k serveru, zaslání textové zprávy nebo souboru.

2.3.1 Přihlášení klienta

Samotné přihlášení klienta se skládá ze čtyř následujících fází:

1. *Autentizace* – proces ověření identity klienta, protokol *OSCAR* používá dva typy autentizace
    a. Otevřenou, kdy je ve *FLAP* použit první kanál a klient odešle svoji identifikaci *cli_ident* a na jejím základě server vrací *srv_cookie*.
    b. Zabezpečenou pomocí MD5, kde jsou použity *SNAC* s *FamilyID 17*. Klient zašle autentizační MD5 požadavek *SNAC(0x0017, 0x0006)*. Server odpoví *SNAC (0x0017, 0x0007)* MD5 autentizačním řetězcem, pomocí než klient vytvoří *HASH* hesla a odešle spolu s dalšími údaji *SNAC(0x0017, 0x0002)*. Server ověří údaje a odpoví *SNAC(0x0017, 0x0003)* autentizační odpovědí.
2. *Vyjednání protokolu* – v této fázi klient pomocí cli_cookie žádá server o seznam podporovaných služeb. Odpověď serveru *SNAC(0x0001, 0x0003)*. Poté by se klient měl zeptat na verze podporovaných služeb *SNAC(0x0001, 0x0017)*. Server odpoví verzemi ve *SNAC(0x0001, 0x0018)*. Po úspěšném vyjednání spojení by klient měl zjistit informace o limitech. Požádá o ně použitím *SNAC(0x0001, 0x0006)*. Server odpoví *SNAC(0x0001, 0x0007)* obsahujícím informace o limitech. Poté by měl klient potvrdit přijetí pomocí *SNAC(0x0001, 0x0008).*
3. *Nastavení služeb* – protože většina služeb je nějakým způsobem limitována, klient by měl tyto limity zjistit. Např. kolik kontaktů může pojmout do svého listu kontaktů nebo velikost textové zprávy, kterou může zaslat serveru.
4. *Finální akce* – klient sděluje serveru informace o přímém připojení jako např. *IP adresu*, *TCP port*, *status*, atd. pomocí *SNAC(0x0001, 0x001E)*. Na závěr zasílá *SNAC(0x0001, 0x0002),* jímž dává najevo, že je připraven komunikovat.

2.3.2 Zaslání textové zprávy mezi klientem a kontaktem

K práci s textovými zprávami slouží *SNAC* s *FamilyID 0x04* označující *ICBM services (Inter Client Basic Messages)*.

Pokud chce klient poslat textovou zprávu nějakému svému kontaktu, použije *SNAC(0x0004, 0x0006),* viz *Kapitola 2.2.8*. V případě, že kontakt zasílá zprávu klientovi, je přijat *SNAC(0x0004, 0x0007),* viz *Kapitola 2.2.9*.

Výměnu zpráv mezi klientem a kontaktem popisuje následující tabulka.

Zdroj	Cíl	Protokol	Kanál	FamilyID	Subtype	Pojmenování
client	server	OSCAR	02	0x0004	0x0006	CLI_SEND_ICBM
server	client	OSCAR	02	0x0004	0x0007	SRV_CLIENT_ICBM

Tabulka 2.47 Výměna textových zpráv mezi klientem a kontaktem

2.3.3 Přidání a odstranění kontaktu z listu kontaktů

Protokol OSCAR rozeznává dva listy kontaktů:

1. *Kontakt list na straně klienta* – je uchováván v klientské aplikaci a v případě potřeby, např. při přihlášení klient ověří jeho platnost oproti serveru, pomocí časového razítka.
2. *Kontakt list na straně serveru* – jedná se o datovou strukturu na straně serveru, viz *Tabulka 2.39,* manipulace pomocí *SNAC* s *FamilyID 0x13.*

Zdroj	Cíl	Protokol	Kanál	FamilyID	Subtype	Pojmenování
client	server	OSCAR	02	0x0013	0x0011	CLI_SSI_EDIT_BEGIN
client	server	OSCAR	02	0x0013	0x0008	CLI_SSIxADD
client	server	OSCAR	02	0x0013	0x0009	CLI_SSIxUPDATE
server	client	OSCAR	02	0x0013	0x000A	CLI_SSIxDELETE
client	server	OSCAR	02	0x0013	0x0012	CLI_SSI_EDIT_FINISH
client	server	OSCAR	02	0x0003	0x0010	CLI_BUDDYLIST_ADD

Tabulka 2.48 Přidání kontaktu do listu kontaktů

Zdroj	Cíl	Protokol	Kanál	FamilyID	Subtype	Pojmenování
client	server	OSCAR	02	0x0013	0x0011	CLI_SSI_EDIT_BEGIN
client	server	OSCAR	02	0x0013	0x000A	CLI_SSIxDELETE
server	client	OSCAR	02	0x0013	0x000E	SRV_SSIxMODxACK
client	server	OSCAR	02	0x0013	0x0009	CLI_SSIxUPDATE
client	server	OSCAR	02	0x0003	0x000F	CLI_BUDDYLIST_REMOVE
server	client	OSCAR	02	0x0013	0x000E	SRV_SSIxMODxACK
client	server	OSCAR	02	0x0013	0x0012	CLI_SSI_EDIT_FINISH

Tabulka 2.49 Odstranění kontaktu z listu kontaktů

2.3.4 Odeslání souboru kontaktu

Oznámení o přenosu souboru je signalizováno *SNAC(0x0004, 0x0006)* popř. *SNAC(0x0004, 0x0007)*, podle toho jestli se jedná o příchozí, nebo odchozí přenos souboru. Podrobnosti o struktuře těchto *SNAC* jsou popsány v *Kapitole 2.2.8* a *2.2.9*. Samotný přenos souboru je uskutečněn pomocí protokolu *TCP/IP* přímo na dohodnutou *IP adresu* a *port*, bez jakékoliv další signalizace.

Zdroj	Cíl	Protokol	Kanál	FamilyID	Subtype	Pojmenování
client	server	OSCAR	02	0x0004	0x0006	CLI_SEND_ICBM
server	client	OSCAR	02	0x0001	0x000A	SRV_RATE_LIMIT_WARN
server	client	OSCAR	02	0x0004	0x000C	SRV_MSG_ACK
server	client	OSCAR	02	0x0004	0x0007	SRV_CLIENT_ICBM
client	server	OSCAR	02	0x000B	0x0003	CLI_STATS_REPORT

Tabulka 2.50 Odchozí přenos souboru

Význam jednotlivých řádků v *Tabulka 2.50* je následující: nabídka přenosu souboru, změna parametrů/limitů přenosu, potvrzení přijetí zprávy s nabídkou přenosu serverem, klient potvrdil převzetí souboru (sdělení detailů o přenosu: *IP adresa*, *port*), vyžádání statistiky o přenosu.

Zdroj	Cíl	Protokol	Kanál	FamilyID	Subtype	Pojmenování
server	client	OSCAR	02	0x0004	0x0007	SRV_CLIENT_ICBM
client	server	OSCAR	02	0x0004	0x0006	CLI_SEND_ICBM
server	client	OSCAR	02	0x0001	0x000A	SRV_RATE_LIMIT_WARN
server	client	OSCAR	02	0x0004	0x0007	SRV_CLIENT_ICBM
client	server	OSCAR	02	0x000B	0x0003	CLI_STATS_REPORT

Tabulka 2.51 Příchozí přenos souboru

Upřesnění významu zpráv z *Tabulka 2.51*: nabídka přenosu souboru od kontaktu, souhlas s přenosem souboru, změna limitů/parametrů přenosu, sdělení detailů o přenosu (IP adresa, port), vyžádání statistiky o přenosu.

2.3.5 Odhlášení kontaktu nebo klienta

Ohlášení klienta je velice jednoduché a je realizováno zasláním prázdné zprávy s použitím *FLAP kanálu 0x04*, která znamená ukončení spojení.

Naopak odhlášení kontaktu je symbolizováno změnou jeho aktuálního stavu na *offline*. Tato změna je reprezentována obdržením *SNAC(0x0003, 0x000E)* s bližším popisem v *Kapitole 2.2.7*.

3 Analýza protokolu YMSG

V této kapitole je popsána struktura a funkce *YMSG* protokolu. Jsou uvedeny všechny typy zpráv, které byly během analýzy protokolu identifikovány a z nich vybrané klíčové zprávy, které jsou potřebné pro rekonstrukci zachycené komunikace.

3.1 Popis struktury YMSG zprávy

Protokol *YMSG* používá jeden typ hlavičky, viz *Tabulka 3.1,* kde je vyznačena tmavší barvou a poté prostor pro *data* označen barvou světlejší. Celková velikost hlavičky je *32B*.

4 bytes		**2 bytes**	**2 bytes**	**2 bytes**
YMSG		Version	Vendor ID	Length
Service type	Status		Session ID	
Data				

Tabulka 3.1 Struktura *YMSG* zprávy

- ***YMSG*** – pole ASCII znaků identifikující protokol, vždy obsahuje řetězec "YMSG".
- *Version* – číslo identifikující verzi protokolu, přehled viz *Tabulka 3.2.*
- *Vendor ID* – číslo identifikuje operační systém, na kterém je spuštěn klient např. *0* pro Windows, nebo *100* pro MacOS.
- *Length* – délka zprávy bez hlavičky.
- ***Service Type*** – určuje typ zprávy, viz *Kapitola 3.3.*
- *Status* – doplňuje určení typu zprávy, viz *Tabulka 3.17.*
- *Session ID* – identifikuje relaci, je unikátní pro každou spuštěnou instanci klienta.

YMSG Protokol	SZ	Hlasová SZ	Konference	Přidání kontaktu	SZ uchované na straně serveru	Zvukové upozornění	Upozornění vibracemi
V12	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V13	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V14	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V15	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V16	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V17	Ano	Ano	Ano	Ano	Ne	Ano	Ano
V18	Ano	Ano	Ano	Ano	Ano	Ano	Ano

Tabulka 3.2 Přehled verzí a funkcí protokolu *YMSG*

SZ znamená soukromá zpráva, dále zvuková upozornění *audible* a upozornění vibracemi *BUZZ.*

3.2 Formát přenášených dat

Data přenášená pomocí *YMSG* protokolu jsou vždy *plain* textová. Skládají ze z klíče a hodnoty, např. pro *YMSG* zprávu typu *YAHOO_SERVICE_MESSAGE* jsou v datové části mimo jiné tyto položky: ASCII hodnota klíče *14*, oddělovač *0xC080* a hodnota pro tento klíč, kterou je obsah přenášené zprávy ukončený opět *0xC080*.

Hexa hodnota *0xC080* slouží jako oddělovač všech položek přenášených v datové sekci. Nachází se za každým klíčem a za každou hodnotou. Vyskytuje se tedy i na konci *YMSG* zprávy.

Jednotlivé klíče jsou závislé na verzi *YMSG* protokolu. Se zvyšující se verzí těchto klíčů přibývá, ale je pravidlem, že význam dříve definovaných klíčů se již nemění. Proto je jisté, že při zachycení komunikace z jakékoliv verze budeme schopni rozpoznat obsah textové komunikace mezi uživateli, protože se bude používat pro text pořád stejná struktura s klíčem *14* a jemu odpovídající hodnotou zprávy.

3.3 Typy zpráv

Následující tabulky ukazují zprávy, které jsou posílány mezi serverem a klientem. Hlavička tabulky odpovídá hlavičce *YMSG* zprávy. *Service type* je pojmenovaná konstanta identifikující typ zprávy. Toto pojmenování je použito i v implementaci. Její hodnota v dekadické podobě je uvedena v poli *kód*. Dále se zde v některých případech nachází i pole *status*, které blíže identifikuje určení zprávy a s tím související položky v datové části. Pokud toto pole není v tabulce uvedeno, nebereme jeho hodnotu v potaz.

Datová část, jak již bylo zmíněno v *Kapitole 3.2,* se skládá z klíče a hodnoty. Nebyla zjištěna žádná jejich křížová závislost nebo závislost na jejich pořadí. Je možné, že konkrétní význam některých typů zpráv je blíže specifikován absencí, či naopak prezencí některých položek, proto není možné jejich přítomnost striktně uvažovat.

Typ zprávy		**Kód**	**Popis**
YAHOO_SERVICE_AUTHENTIFICATION		87	Autentifikace na serveru
Klíč	**Hodnota**	**Popis**	
1	*YahooID*	Uživatelské jméno klienta	
13	*state*	Stav, který by měl klient použít pro přihlášení	
94	*challenge string*	Použit při kódování přihlašovacích údajů	

Tabulka 3.3 Autentifikace na serveru

Typ zprávy		Kód	Popis
YAHOO_SERVICE_AUTHENTIFICATION_RESPONSE		87	Předání přihlašovacích údajů serveru
Klíč	**Hodnota**	**Popis**	
0,1,2	*YahooID*	Uživatelské jméno klienta	
277	*token*	Autentifikační token	
278	*token2*	Autentifikační token	
307	*hash*	Autentifikační *hash* soli a *challenge string*	
59	*B-Cookue*		
244	*build*	Číslo sestavy programu klienta	
98	*country code*	Kód země např. *us*	
135	*version*	Verze programu klienta	
500	*mac*	Fyzická adresa rozhraní klienta	

Tabulka 3.4 Předání přihlašovacích údajů serveru

Typ zprávy		Kód	Popis
YAHOO_SERVICE_LOGOFF		2	Odhlášení kontaktu nebo klienta
Klíč	**Hodnota**	**Popis**	
505	*0*	Odhlášení klienta u místní stanice	
7	*YahooID*	Odhlášení klienta u vzdálené stanice	

Tabulka 3.5 Odhlášení kontaktu nebo klienta

Typ zprávy		Kód	Popis
YAHOO_SERVICE_MESSAGE		6	Zpráva mezi klienty
Klíč	**Hodnota**	**Popis**	
14	*Zpráva*	Text přenášené zprávy	
1	*TransmiterID*	YahooID odesílatele zprávy	
5	*ReciverID*	YahooID příjemce zprávy	

Tabulka 3.6 Zpráva mezi klienty

Typ zprávy		Kód	Popis
YAHOO_SERVICE_LIST		241	Iniciály klienta
Klíč	**Hodnota**	**Popis**	
3,89	*YahooID*	Uživatelské jméno klienta	
216	*jméno*	Křestní jmeno kontaktu	
254	*příjmení*	Příjmení kontaktu	

Tabulka 3.7 Zaslání iniciál klienta serverem při přihlášení

Typ zprávy		Kód	Popis
YAHOO_SERVICE_LIST_V15		241	Načtení listu kontaktů
Klíč	**Hodnota**	**Popis**	
65	*Group*	Skupina, do které kontakt patří	
7	*ContactId*	YahooID kontaktu	

Tabulka 3.8 Načtení listu kontaktů

Zpráva obsahuje celý list kontaktů klienta. Kontakty jsou samostatné položky v datové části zprávy. Každý kontakt je označen kódem *7* a hodnotou *YahooID*. Skupina je označena kódem *65* a hodnotou *Group* (jméno skupiny). Zde záleží na pořadí, v jakém jsou jednotlivé kontakty ve zprávě zařazeny. Jako první se ve zprávě objeví jméno skupiny a za ním následuje výčet kontaktů, které jsou této skupině přiřazeny. Objeví-li se za sebou kódy označující skupiny, ale kód uvozující kontakt mezi nimi chybí, jedná se o prázdnou skupinu.
Zjednodušená zpráva bez dalších parametrů může být následující:

```
65:Group1
  7: Contact1
  7: Contact2
65:Group2
65:Group3
  7: Contact3
```

Obrázek 3.1 Schéma sekvence záznamů pro přiřazení kontaktu ke skupině

Tato zpráva nám říká, že v listu kontaktů se budou nacházet tři skupiny *Group1*, *Group2* a *Group3*. Group1 obsahuje kontakty *Contact1* a *Contact2*. *Group2* neobsahuje kontakt žádný. *Group3* obsahuje jeden kontakt *Contact3*.

Typ zprávy		Kód	Popis
YAHOO_SERVICE_ADD_BUDDY		131	Přidání kontaktu do listu kontaktů
Klíč	**Hodnota**	**Popis**	
65	*Group*	Skupina, do které kontakt patří	
7	*ContactId*	YahooID kontaktu	
216	*jméno*	Křestní jméno kontaktu	
254	*příjmení*	Příjmení kontaktu	
14	*info*	Informace o kontaktu, pouze pokud je status *Default*	

Tabulka 3.9 Přidání kontaktu do listu kontaktů

Typ zprávy	Kód	Popis
YAHOO_SERVICE_Y7_BUDDY_AUTH	214	Autorizace kontaktu

Status	Popis
0	Odpověď na požadavek
1	Odpověď na požadavek
3	Požadavek

Klíč	Hodnota	Popis
65	*Group*	Skupina, do které kontakt patří
7	*ContactId*	YahooID kontaktu
216	*jméno*	Křestní jméno kontaktu
254	*příjmení*	Příjmení kontaktu
14	*info*	Informace o kontaktu

Tabulka 3.10 Autorizace kontaktu

Typ zprávy	Kód	Popis
YAHOO_SERVICE_STATUS_V15	240	Změna statusu při přihlášení kontaktu

Klíč	Hodnota	Popis
10	*YahooAwayStatus*	Id statusu, viz *Tabulka 3.17*
7	*ContactId*	YahooID kontaktu
19	*YahooCustomAwayMsg*	[volitelné] Uživatelem zanechaná zpráva ve statusu
310	*language*	Jazyk zprávy ve statusu, výchozí *en-us*

Tabulka 3.11 Změna statusu při přihlášení kontaktu

Typ zprávy	Kód	Popis
YAHOO_SERVICE__Y7_STATUS_UPDATE	198	Změna statusu či přihlášení kontaktu

Klíč	Hodnota	Popis
10	*YahooAwayStatus*	Id statusu, viz *Tabulka 3.17*
19	*YahooCustomAwayMsg*	[volitelné] Uživatelem zanechaná zpráva ve statusu
7	*ContactId*	YahooID kontaktu
310	*language*	Jazyk zprávy ve statusu, výchozí *en-us*

Tabulka 3.12 Změna statusu, když je kontakt online

Typ zprávy	Kód	Popis
YAHOO_SERVICE_Y7_FILE_TRANSFER	220	Žádost o odeslání souboru

Klíč	Hodnota	Popis
4	TransmiterID	YahooID odesílatele
5	ReciverID	YahooID příjemce
27	*filename*	Jméno souboru

Tabulka 3.13 Žádost o odeslání souboru

Typ zprávy		Kód	Popis
YAHOO_SERVICE_Y7_FILE_TRANSFER_INFORMATION		221	Informace o souboru
Klíč	**Hodnota**	**Popis**	
1	*TransmiterID*	YahooID odesílatele	
4	*TransmiterID*	YahooID odesílatele	
5	*ReciverID*	YahooID příjemce	
27	*filename*	Jméno souboru	

Tabulka 3.14 Informace o souboru

Typ zprávy		Kód	Popis
YAHOO_SERVICE_Y7_FILE_TRANSFER_ACCEPT		222	Stav požadavku na odeslání
Klíč	**Hodnota**	**Popis**	
4	*TransmiterID*	YahooID odesílatele	
5	*ReciverID*	YahooID příjemce	
27	*filename*	Jméno souboru	
250	*ipAddress*	IP adresa serveru pro stažení	
251	*token*	Token – identifikátor souboru na serveru	

Tabulka 3.15 Stav požadavku na odeslání

Typ zprávy		Kód	Popis
YAHOO_SERVICE_Y7_CONTACT_DETAILS		227	Informace o kontaktu
Klíč	**Hodnota**	**Popis**	
1	*TransmiterID*	YahooID odesílatele	
13	*YahooEventSpecification*	Musí být 1	
280	*ContactsDetails*	Informace o kontaktu v podobě *XML*, význam viz *Tabulka 3.18*	

Tabulka 3.16 Zpráva přenášející detailní informace o kontaktu

3.4 Přehled konstant

YahooAwayStatus	Popis
0	YAHOO_STATUS_AVAILABLE
1	YAHOO_STATUS_BRB
2	YAHOO_STATUS_BUSY
3	YAHOO_STATUS_NOTATHOME
4	YAHOO_STATUS_NOTATDESK
5	YAHOO_STATUS_NOTINOFFICE
6	YAHOO_STATUS_ONPHONE
7	YAHOO_STATUS_ONVACATION
8	YAHOO_STATUS_OUTTOLUNCH
9	YAHOO_STATUS_STEPPEDOUT
12	YAHOO_STATUS_INVISIBLE
99	YAHOO_STATUS_CUSTOM
999	YAHOO_STATUS_IDLE
1515563605	YAHOO_STATUS_WEBLOGIN
1515563606	YAHOO_STATUS_OFFLINE
22	YAHOO_STATUS_TYPING

Tabulka 3.17 Přehled kódů a k nim se vážících stavů kontaktů

XML Tag	Hodnota	XML Tag	Hodnota
yi	YahooID kontaktu	co	zaměstnání společnost jméno
fn	křestní jméno	wa	zaměstnání adresa město
mn	prostřední jméno	ws	zaměstnání adresa ulice
ln	příjmení	wc	zaměstnání adresa město
e0	email 1	wp	zaměstnání telefon
e1	email 2	wu	zaměstnání web stránky
e2	email 3	ws	zaměstnání adresa stát
pu	osobní webové stránky	wz	zaměstnání adresa zip
ha	domácí adresa ulice	wn	zaměstnání adresa země
hc	domácí adresa město	mo	mobilní číslo
hs	domácí adresa stát	pa	pager
hz	domácí adresa zip kód	fa	fax
hn	domácí adresa země	ot	jiné info
hp	domácí telefon	bi	datum narození ve tvaru DD/MM/YYYY
ti	zaměstnání titul		

Tabulka 3.18 Přehled významu hodnot *XML* tagů vyskytujících se v Tabulka 3.16

YahooServerStatus	Popis
-1	YPACKET_STATUS_DISCONNECTED
0	YPACKET_STATUS_DEFAULT
1	YPACKET_STATUS_SERVERACK
2	YPACKET_STATUS_GAME
4	YPACKET_STATUS_AWAY
5	YPACKET_STATUS_CONTINUED
12	YPACKET_STATUS_INVISIBLE
22	YPACKET_STATUS_NOTIFY
1515563605	YPACKET_STATUS_WEBLOGIN
1515563606	YPACKET_STATUS_OFFLINE

Tabulka 3.19 Přehled hodnot, kterých nabývá položka status v YMGS hlavičce

3.5 Identifikace klíčových specifických událostí

Následuje seznam klíčových událostí a posloupnost *YMSG* popř. *HTTP* zpráv zasílaných mezi *YMSG* klientem a serverem, které jsou s konkrétní událostí spřaženy:

- Přihlášení klienta a stažení kontakt listu, viz *Tabulka* 3.20.
- Odhlášení kontaktu nebo klienta, viz *Tabulka* 3.22.
- Příchozí, odchozí textová zpráva viz *Tabulka* 3.24.
- Přidání nového kontaktu do listu kontaktů, viz *Tabulka* 3.23.
- Zjištění detailních informací o kontaktu.
- Změna statusu ve smyslu *Available, Busy*, atd. viz *Tabulka* 3.25.
- Odeslání a příjem souboru viz *Tabulka* 3.26.

3.5.1 Přihlášení a stažení listu kontaktů

Proces probíhá podle schématu, naznačeném v *Tabulka 3.20.*

1. Klient pošle serveru zprávu typu YAHOO_SERVICE_AUTHENTIFICATION, viz *Tabulka 3.3*, kde použije v datové části pole *1* s hodnotou uživatelského jména *YahooID.*
2. Server odpoví stejnou zprávou, kde přidá položky s kódy *2* s hodnotou kódu statusu, kterou si klient po přihlášení nastaví, a *94* s hodnotou *challenge string*, která bude použita k zašifrování přihlašovacích údajů.
3. Klient odpoví zprávou YAHOO_SERVICE_AUTHENTIFICATION_RESPONSE, viz *Tabulka 3.4,* kterou serveru sdělí potřebné údaje k přihlášení zašifrovanému podle dohodnuté metody.
4. Pokud přihlášení proběhlo úspěšně, server zašle klientovi zprávu YAHOO_SERVICE_LIST, ve které mu sdělí jeho iniciály, viz *Tabulka* 3.7.
5. Následně server zašle kontakt list zprávou YAHOO_SERVICE_LIST_V15, viz *Tabulka* 3.8.

Id	Zdroj	Cíl	Protokol	Délka	Info	Service	Status
1	client	server	YMSG	101	Authentication	87	Default
2	server	client	YMSG	171	Authentication	87	Server ACK
3	client	server	YMSG	797	Authentication Response	84	Web Login
4	server	client	YMSG	313	List (status=Default)	85	Default
5	server	client	YMSG	474	List V15, Status V15	241, 240	Default

Tabulka 3.20 Přihlášení a stažení listu kontaktů

3.5.2 Odhlášení kontaktu

Odhlášení kontaktu může být dvojího typu, podle toho, kdo o odhlášení informuje.

1. Server sděluje, že se odhlásil někdo, koho má klient v listu kontaktů, viz *Tabulka 3.21*. V tomto případě server zašle zprávu typu YAHOO_SERVICE_LOGOFF, viz *Tabulka 3.5*, kde v datové části pod klíčem *1* uvede *YahooID* kontaktu, který se odhlásil.
2. Klient se odhlašuje a dává tuto skutečnost najevo serveru, viz *Tabulka* 3.22. V tomto případě odesílá klient na server zprávu YAHOO_SERVICE_LOGOFF, kde nechává datovou část prázdnou.

Zdroj	Cíl	Protokol	Délka	Info	Service	Status
server	client	YMSG	87	Pager Logoff	2	Default

Tabulka 3.21 Odhlášení vzdáleného klienta

Zdroj	Cíl	Protokol	Délka	Info	Service	Status
client	server	YMSG	96	Pager Logoff	2	Default

Tabulka 3.22 Odhlášení klienta

3.5.3 Přidání kontaktu do listu kontaktů

Následující schéma přidání kontaktu je podmíněno skutečností, že známe *YahooID* kontaktu. Postupem, který by vedl k jeho nalezení, se v této práci zabývat nebudeme.

1. Klient zašle serveru zprávu YAHOO_SERVICE_ADD_BUDY, viz *Tabulka 3.9*, kterou serveru sdělí informace o tom, jaký kontakt chce přidat, identifikováno klíčem *7* s hodnotou *YahooID* kontaktu a dále skupinu v hodnotě klíče *65* a volitelně poznámku ke kontaktu v hodnotě klíče *14*.
2. Server potvrdí přijetí odesláním stejné zprávy YAHOO_SERVICE_ADD_BUDY, jakou dostal, pouze zkrácenou o nepotřebné informace. V tuto dobu server odešle požadavek o autorizaci kontaktu, kterého si klient přidal. Pokud kontakt není online, je mu požadavek odeslán ihned po jeho opětovném přihlášení.

3. Jakmile má server informaci, zdali kontakt požadavek o přidání autorizoval, nebo zamítl, odešle klientovi zprávu YAHOO_SERVICE_BUDY_AUTH, kde v klíči *13* je hodnota odpovědi na autorizaci. Pokud kontakt autorizaci přijal, je hodnota rovna *1*, pokud zamítl pak *0*. Klient poté odešle kontaktu oznámení o autorizaci opět zprávou YAHOO_SERVICE_BUDY_AUTH.
4. Pokud kontakt požadavek na přidání od klienta přijal, odešle mu taktéž požadavek na autorizaci spolu s informací o jeho aktuálním stavu statusu zprávou YAHOO_SERVICE_STATUS_V15, viz *Tabulka 3.11*.
5. Klient odpoví schválením požadavku o autorizaci a proces končí úspěšným přidáním kontaktu.

Id	Zdroj	Cíl	Protokol	Délka	Info	Service	Status
1	client	server	YMSG	213	Add Buddy	131	Default
2	server	client	YMSG	138	Add Buddy	131	Server Ack
3	server	client	YMSG	109	Y7 Buddy Authorization	214	Server Ack
4	server	client	YMSG	500	Status V15 Y7 Buddy Authorization	240, 214	Server Ack, kód 3
5	client	server	YMSG	108	Y7 Buddy Authorization	214	Default

Tabulka 3.23 Přidání kontaktu

3.5.4 Zaslání zprávy

Zaslání zprávy kontaktu se provádí pouze jednou *YMSG* zprávou typu YAHOO_SERVICE_MESSAGE, viz *Tabulka 3.6*. Tato zpráva na rozdíl od většiny *YMSG* zpráv není serveru potvrzována na úrovni aplikačního protokolu.

Klient posílající zprávu kontaktu nastaví v hlavičce *YMSG* zprávy pole *status* na hodnotu YPACKET_STATUS_OFFLINE, viz *Tabulka 3.19*. Tímto je pravděpodobně zajištěno, že pokud by byl client v době odesílání zprávy *offline,* tak se zpráva uloží na serveru a po opětovném přihlášení klienta bude doručena.

Server doručující zprávu nastaví pole *status* v *YMSG* hlavičce na *Server Ack* a přidá do datové části klíč *4* s hodnotou *YahooID* odesílatele. Takto upravená zpráva je doručena kontaktu.

Zdroj	Cíl	Protokol	Délka	Info	Service	Status
server	client	YMSG	285	Message	6	Server Ack
client	server	YMSG	234	Message	6	Offline

Tabulka 3.24 Příchozí zpráva a odpověď

3.5.5 Změna statusu

Změnu statusu zajišťuje *YMSG* zpráva YAHOO_SERVICE__Y7_STATUS_UPDATE, viz *Tabulka 3.12*.

Klient měnící status zasílá tuto zprávu na server, kde v hlavičce nastaví pole *status* na hodnotu *Default* a v datové části uvede změny. Buď pouze mění vestavěný status značený kódem *47* s hodnotou z *Tabulka 3.17*, nebo může vložit uživatelskou zprávu, která bude součástí statusu jako hodnota kódu *19*.

Server tuto *YMSG* zprávu pozmění vložením kódu *7* s hodnotou *YahooID* klienta, který status změnil a vložením kódu *310* s hodnotou jazyka v jakém byla zpráva napsána. Pravděpodobně se jedná vždy o konstantu *en-us*. Takto upravenou zprávu server zasílá všem kontaktům, které jsou aktuálně k serveru přihlášeny a mají v listu kontaktů klienta, který zprávu změnil.

K informování kontaktu o změně statusu se taktéž používá *YMSG* zpráva YAHOO_SERVICE_STATUS_V15. Z jejího názvu plyne, že pochází z *verze 15* a bylo zjištěno, že slouží k informování klienta o stavu nově autorizovaného kontaktu. Tuto zprávu zasílá server. Z toho vyplývá, že server si udržuje databázi přihlášených kontaktů a jejich aktuálních stavů.

Zdroj	Cíl	Protokol	Délka	Info	Service	Status
client	server	YMSG	94	Y6 Status Update	198	Default
server	client	YMSG	307	Y6 Status Update	198	Server Ack

Tabulka 3.25 Změna statusu klienta a kontaktu

3.5.6 Odeslání souboru

Jedná se o jednu z mála věcí, která je vyjednána pomocí *YMSG* protokolu, ale samotný přenos souboru probíhá přes protokol *HTTP*.

- Klient zahájí vyjednávání zasláním zprávy YAHOO_SERVICE_TRANSFER, ve které sdělí serveru informace o souboru pod kódem *27 jméno souboru* a *5 YahooID* příjemce.
- Server potvrdí přijetí žádosti na přenos souboru zprávou pouze se statusem *Server Ack* a přidanou položkou s kódem *265* s *hodnotou tokenu*, který bude později identifikovat přenos.
- Následně klient zašle zprávu YAHOO_SERVICE_TRANSFER_INFO, ve které sdělí kontaktu *IP adresu serveru,* na kterém bude soubor k dispozici v kódu *250*, *přístupový token* v kódu *265* a kódem *27* identifikuje *jméno souboru*.

Položka se *jménem souboru* se zde nachází po celou dobu vyjednávání přenosu jako identifikace v případě, že by klient posílal kontaktu více souborů.

Zdroj	Cíl	Protokol	Délka	Info	Service	Status
client	server	YMSG	451	Y7 File Transfer	220	Default
server	client	YMSG	160	Y7 File Transfer	220	Server Ack
client	server	YMSG	201	Y7 File Transfer Information	221	Default
server	client	YMSG	354	Y7 File Transfer Accept	222	Server Ack
client	http server	HTTP	1134	File		

Tabulka 3.26 Odeslání souboru

4 Experimentální nástroj pro rekonstrukci YMSG

Na základě provedené analýzy protokolu bylo potřeba rozlišit zprávy, které budou klíčové pro další zpracování a ty, které mohou být ignorovány. Jako například zprávy typu YAHOO_SERVICE_KEEP_ALIVE, které pro nás nejsou dále zajímavé. Klíčové situace, které jsou podstatné pro rekonstrukci komunikace, byly nastíněny v *Kapitole 3.5*. V následující kapitole se budeme zabývat jejích dalším upřesněním, týkajícím se kombinace zpráv a jejich parametrů přenášených v datové části a jejich konkrétním významem.

Protože není jisté, že bude zachycena vždy celá komunikace, není možné při rekonstrukci vyžadovat striktní pořadí zpráv, jak byly posílány, ale je cílem abstrahovat se nad tuto skutečnost a vytvořit *šablony,* podle kterých bude komunikace mezi dvěma konkrétními klienty rozpoznána a k nim správně přiřazena.

Vstupem experimentálního nástroje je zachycená komunikace ve formátu PCAP souboru zachycená za pomoci knihovny *LibPCAP*, která je použita v nástrojích *Wireshark* nebo *TCPDump*. Naopak zachycená komunikace pomocí nástroje *MNM* není s experimentálním nástrojem kompatibilní, protože od *MNM* verze 3 používá tento nástroj proprietární knihovnu pro ukládání zachycených dat.

Výstupem tohoto nástroje je *XML* soubor s předem daným schématem, viz *Příloha 1*. Pokud se v komunikaci vyskytoval nějaký přenos souborů, budou tyto soubory taktéž rekonstruovány. Více se touto problematikou zabývá *Kapitola 4.6.*

4.1 Použité knihovny a programy

Pro správnou funkci experimentálního nástroje je nutné, aby v systému byly nainstalovány všechny potřebné knihovny spolu s programem pro rekonstrukci *TCP toků* s názvem *TCPflow*. Dále cesta k jeho binárnímu souboru musí být součástí systémové proměnné $PATH.

Experimentální nástroj lze nalézt na přiloženém DVD ve složce */YMSG/experimental_tool*. Spouští se následujícím příkazem *python ymsg.py <cesta k PCAP souboru>*.

4.1.1 Dpkt.pcap

Knihovna Dpkt je použita pro parsování vstupního *PCAP souboru*. Jsou použity funkce na načtení PCAP souboru *dpkt.pcap.Reader*, zpracování obsahu packetu funkce *Ethernet* a struktury, s jejíž pomocí je možné přistoupit k požadované položce v packetu.

4.1.2 UserDict

Knihovní objekt *UserDict* slouží jako předloha pro objekty, u kterých bude nutné přidat za běhu nějaké proměnné. Pomocí *UserDict* je možné dynamické přidání klíče a hodnoty do objektu a později přístup k nim pomocí operátoru *[]*.

4.1.3 Etree.ElementTree

Knihovna použita pro vytváření *XML stromu*, do kterého jsou zaznamenány zachycené *YMSG* zprávy ve zvoleném formátu viz *Příloha 1*.

4.1.4 Dom.minidom

Knihovna použita pro parsování textového výstupu, aby mohl být dále spravován a mohl být vytvořen uživatelsky přívětivější výstup s odsazenými *XML tagy*.

4.1.5 TCPflow

Tento program zajišťuje rekonstrukci požadovaných *TCP toků*. Toto je využito při destilaci zachyceného souboru, který je téměř vždy rozdělen v několika *IP datagramech*, takže jej nejsme schopni netriviálně rekonstruovat za pomoci knihovny *dpkt*.

4.2 Struktura skriptu

Skript se skládá ze dvou základních tříd, viz *Příloha 3*.

1. Třída *PacketInfo*, která uchovává obecné informace o zachyceném packetu. Tyto informace jsou potřebné pro další zpracování a je potřeba je získat ze všech vrstev *TCP/IP modelu*. Jsou to např. čas odeslání packetu, zdrojová a cílová IP adresa a port.
2. Třída *YMSGPacketInfo*, která uchovává konkrétní informace ze zachyceného packetu, které se nachází na aplikační vrstvě *TCP/IP modelu* přenášené protokolem *YMSG*. Těmito jsou všechny položky *YMSG* hlavičky, viz *Kapitola 3.1,* a všechny známé položky přenášené v datové části packetu.

Ve skriptu se také nachází mapy pod názvy *yahooStatusMap*, *yahooContentMap* a *yahooServis* sloužící k mapování textových řetězců zachycených v *YMSG* zprávách na jejich obslužné funkce, které provedou patřičné operace podle obsahu řetězce. Tato struktura zajišťuje jednoduché rozšíření aplikace. Pokud je objevena nová funkce nebo posílán nový typ zprávy, na který chceme reagovat a zaznamenávat ho, stačí přidat příslušné kódy a řetězce do map a vytvořit obslužné funkce. Výhodou tohoto postupu je, že není potřeba modifikovat funkci sloužící k parsování *YMSG* zprávy.

Dále jsou ve skriptu přítomny obslužné funkce, které slouží ke zpracování vstupního PCAP souboru, vytvoření textového *XML* výstupu a uložení přenášeného souboru, pokud se během komunikace vyskytne. Tyto budou podrobněji rozebrány v dalších kapitolách.

V UML diagramu tříd v *Příloze 3* jsou uvedeny všechny výše zmíněné komponenty a měl by sloužit k ucelení představy o struktuře skriptu. Pro zjednodušení byly vynechány prvky, které přímo nesouvisejí s primární funkcí skriptu, ale byly zde ponechány k ladícím účelům. Dále zde nejsou uvedeny metody, které mají stejný název jako členské proměnné třídy *YMSGPacketInfo* začínající prefixem *Yahoo.* Tyto metody slouží pouze k inicializaci dané proměnné. Tento koncept byl zvolen v prvotní fázi implementace, kdy nebylo jasné, zdali nebude potřeba implementovat další funkcionalitu před přiřazením hodnoty do proměnné.

4.3 Parsování PCAP souboru

Parsování PCAP souboru je prováděno v několika funkcích podle úrovně zapouzdření:

- *pcapParser* – načte *PCAP soubor* ze zadané cesty při spuštění programu. Funkcí *dpkt.pcap.Reader* načte data pro další zpracování knihovnou *dpkt.*
- *ethParser* – zpracuje všechny packety a filtruje je podle obsahu. Všechny, které nejsou zprávami *YMSG*, jsou zahozeny.

ymsgParser – vytvoří nový objekt typu *YMSGPacketInfo*, který si při své konstrukci uloží potřebné informace z *L1* a *L2* vrstvy *TCP/IP modelu* a při následném volání metody *parse()* provádí další parsování zprávy. Toto je popsáno v *Kapitole 4.4*. Nově vzniklý objekt je přidán do globálního pole *listPacket,* které tyto objekty sdružuje pro další zpracování.

4.4 Zpracování jednotlivých YMSG zpráv

Zpracování *YMSG* zprávy probíhá metodou *parse()* u objektu, který tuto zprávu bude uchovávat. Zpracování probíhá ve třech fázích:

- V první fázi se zpracuje hlavička, u které je předem známé, na jaké pozici se nachází konkrétní položky. Tyto pozice spolu s funkcí, která je použita na přetypování dat a jménem konkrétní položky, pod kterým se tyto data uloží do objektu, se nachází ve struktuře *packetMap*.
- Ve druhé fázi se volá metoda objektu *parseContent()*. Tato za pomoci regulárního výrazu *r'(\d+)\xc0\x80([^\xc0\x80]+)'* nalezne a rozdělí všechny výskyty klíče a hodnoty do pole pro další zpracování.
- Ve třetí a poslední fázi probíhá zpracování známých kódů metodou *evaluateContent()* a jejich vložení do objektu pod jejich jmény za pomoci *yahooContentMap,* ve které jsou uvedeny všechny známé klíče, které se mohou nacházet ve zprávách.

Klíč	Hodnota	Klíč	Hodnota	Klíč	Hodnota
0,1,4	TransmiterYahooID	10	YahooAwayStatus	28	YahooFileSize
5	ReciverYahooID	36	YahooGroupName	250	YahooFileIpServer
7	TheirYahooID	64	YahooGroupNameTarged	251	YahooFileToken
14	YahooMsg	65	YahooGroupName	265	YahooFileId
49	YahooEvent	216	YahooUserFirstName	280	ContactsDetails
13	YahooEventSpecification	254	YahooUserLastName	505	ServiceLogOff
19	YahooCustomAwayMsg	27	YahooFileName		

Tabulka 4.1 *yahooContentMap* obsahující hodnoty klíčů *YMSG* zpráv zpracovávané parserem

4.5 Vyhodnocení YMSG zpráv a vytvoření XML výstupu

Po vytvoření seznamu objektů reprezentujícím všechny zachycené *YMSG* zprávy, je zapotřebí vybrat pouze ty, které jsou pro zaznamenání podstatné, viz *Tabulka 4.2* a z nich vytvořit výstupní *XML* strukturu, viz *Příloha 1*.

Postup vytvoření výstupního *XML* je následující:

- Vyhodnocuje se zvlášť každý prvek ze seznamu podle hodnoty z hlavičky v poli *service*.
- Každý typ služby má podle svého kódu přiřazenu obslužnou funkci, která se stará o jeho vyhodnocení.

Service	Popis	Service	Popis
2	Y_S_LOGOFF	240	Y_S_STATUS_V15
6	Y_S_MESSAGE	198	Y_S__Y7_STATUS_UPDATE
241	Y_S_LIST_V15	222	Y_S_Y7_FILE_TRANSFER_ACCEPT
131	Y_S_ADD_BUDDY	211	Y_S_Y7_CONTACT_DETAILS
214	Y_S_Y7_BUDDY_AUTH		

Tabulka 4.2 Typy zpráv zpracovávané na výstup, mapa yahooServis, YAHOO_SERVICE… v názvu služby nahrazeno za Y_S...

4.6 Zachycení přenášeného souboru

Pokud byl během vyhodnocování *YMSG* zpráv zjištěn přenos souboru, jsou jeho parametry (*jméno souboru, ID souboru, velikost, IP a port klienta, IP a port serveru*) zaznamenány do globální kolekce *listAttachment* a zároveň jsou vloženy potřebné údaje do výstupu *XML*.

Samotné zachycení přenášeného souboru je provedeno až po ukončení zpracování *YMSG* zpráv. Tento postup je nutný, protože k získání souboru je potřeba rekonstruovat *TCP stream,* ve kterém byl přenášen. K tomuto je použit nástroj *TCPflow* s předem získanými parametry, kterými jsou *port* a *IP adresa* odesílatele souboru. Takto je získáno několik souborů pojmenovaných „*<IP zdroje>.<port zdroje>-<IP cíle>.<port cíle>*“, každý reprezentuje jeden *TCP stream*. Z těchto souborů je potřeba najít ten pravý, který obsahuje přenášený soubor. Protože je možné, aby více klientů komunikovalo stejným serverem, je potřeba vybrat pouze soubory, které patří konkrétnímu klientovi, čili obsahují v názvu jeho *IP adresu* a *port*.

Tyto soubory obsahující *TCP streamy*, které nám vyhovují, jsou dále zpracovávány a to tak, že z každého je odstraněna *HTTP hlavička* a velikost výsledného souboru je porovnána s informací o velikosti přenášeného souboru, která byla získána z *YMSG zprávy* typu YAHOO_SERVICE_Y7_FILE_TRANSFER. Tento způsob se jeví ne příliš optimální, protože není možné přiřadit názvy k souborům, které mají stejnou velikost. Částečným řešením je přiřazení přípony „*?<id>*“. Tato problematika se jeví vhodná pro další analýzu a implementaci řešení, které bude schopné přiřadit souborům jejich pravé názvy s absolutní přesností. Takto získané soubory jsou poté uloženy v adresáři, odkud byl skript spuštěn.

4.7 Testování

Testování implementovaného modelu v Python skriptu nad všemi zachycenými *PCAP* soubory, které byly pořízeny během analýzy protokolu, bylo úspěšné. Tudíž *XML* výstup přesně popisoval úkony, které provedl klient během komunikace, jako např. načtení kontakt listu, nebo odeslání zprávy.

Problém byl zaznamenán pouze při rekonstrukci souborů, které byly zaslány jedním klientem a měly stejnou velikost. Nebylo možné k nim správně přiřadit jméno, protože nebylo zjištěno, jakým způsobem mezi těmito soubory *Yahoo Messenger* rozlišuje. Pravděpodobně se tak děje na základě výsledku kontrolního součtu, jehož algoritmus není známý. Tato část se jeví jako vhodná k hlubší analýze a vylepšení stávajícího řešení.

Obdobně nebylo možné provést rekonstrukci zachycené *VoIP komunikace* nebo *VoIP s přenosem obrazu*. Bylo zjištěno, že komunikace je šifrovaná za pomoci *SSL/TLS*.

Bylo provedeno testování výsledného skriptu na množině zachycených *PCAP souborů* o velikostech *1*, *10* a *100 MB*. Test každého souboru se skládal z *deseti* měření. Průměrná délka běhu skriptu byla následující: *0,07s* pro první, *0,92s* pro druhý a *51.47s* pro třetí soubor. Z naměřených hodnot je možné určit, že časová složitost použitého algoritmu je *lineární*. Paměťová složitost je taktéž lineární, protože maximální množství alokované paměti pro první soubor bylo *20.9 MB*, pro druhý *104.5 MB* a třetí *970.6 MB*. Tato velká paměťová náročnost je způsobena neoptimalizovanou implementací zachycení přenášeného souboru, kde pro každý zachycený soubor je prováděna rekonstrukce TCP toků.

Testovací množina všech zachycených *PCAP souborů* je přiložena na DVD.

5 Rekonstrukce IM za použití NPL

Po dohodě s vedoucím práce a pro potřeby projektu *Sec6net* bylo nutné vytvořit sadu analyzačních skriptů, které jsou schopny zobrazit zachycenou komunikaci ve formátu srozumitelném uživateli. Požadavkem bylo použití *MNM*, proto byl jako implementační jazyk zvolen jazyk *NPL*.

Pro provedení korektního znovu sestavení (*reassembling*) TCP toků v zachyceném *PCAP* souboru, je nutné upravit soubor *Payload.npl* následujícím způsobem. Za *case "HTTP":* nacházející se standardně na řádku *135*, přidat záznamy pro oba rekonstruované protokoly. Pokud bude problém s ochranou proti zápisu do tohoto souboru, je vhodné vytvořit duplicitní soubor v adresáři obsahujícím *NPL popisy* pro protokoly. Potom je třeba v *MNM* nastavit ve vlastnostech zvoleného profilu pro rekonstrukci (*Parser Profiles*) vyšší prioritu pro tento adresář.

Úprava by měla vypadat následovně:

```
135   case "HTTP":
136   case "PhoneBookDL":
137         HTTP Http;
138   case "ICQ": ICQ icq;
139   case "YMSG": YMSG ymsg;
```

Obrázek 5.1 Úprava souboru *Payload.npl*

5.1 NPL

Network monitor parsing language (*NPL*) je jazyk založený na skriptování poskytující možnosti popisu organizace dat v jednotlivých protokolech. Je vyvíjen spolu s programem pro zachycení a analýzu síťové komunikace *Microsoft Network Monitor* [1]. Výhodou je možnost vytváření zásuvných modulů v jazyce *C/C++* a jimi řešit konstrukce, které v *NPL* nejsou možné, protože jazyk podporuje jako jediné řídící konstrukce smyčku *while* a větvení pomocí příkazu *switch*.

MNM umožňuje zachycení komunikace na síti a její okamžitou analýzu a přehledné zobrazení. Je možné aplikovat vstupní filtry, které umožní zahazovat datagramy, které nechceme odposlouchávat. Samozřejmostí je aplikace filtrů na již zachycenou komunikaci a to velmi detailních. Můžeme filtrovat na úrovni IP adres, portů, ale také i hodnot polí v různých aplikačních protokolech.

5.2 NPL pro ICQ

Vytvoření *NPL skriptu* se ukázalo jako velice výhodné, protože během něj bylo objeveno několik nových typů zpráv, které se v předchozích verzích protokolu nenacházely, nebo nebyly dokumentované v použitých zdrojích. Bylo také nalezeno několik chyb ve filtru, který používá *Wireshark* pro analýzu a zobrazení zachycené komunikace.

Oproti *YMSG*, kde byla analýza směřována obecně, a bylo cílem zjistit veškeré možné informace o protokolu, protože protokol byl mnohem méně dokumentován, analýza protokolu *OSCAR* byla již kvůli jeho komplexnosti cílená pouze na zprávy, které přímo souvisí s jeho rekonstrukcí.

Do *NPL* popisu byly zahrnuty všechny zprávy, které jsou popsány v *Kapitole 2.2.* Nad tento rámec zpráv byly přidány tabulky se slovním popisem jednotlivých typů a podtypů zpráv.

Protože jeden *IP datagram* může obsahovat více zpráv protokolu *OSCAR* (každá zpráva je vždy zapouzdřena v protokolu *FLAP*, viz *Kapitola 2.1.1)*, bylo nutné zpracování i neznámých typů zpráv. Velikost obsahu datové části zprávy byla odvozena z velikosti uvedené v hlavičce *FLAP* protokolu a velikosti samotné *SNAC hlavičky*. Tato data jsou zobrazena jako *Blob*.

Při vytváření *NPL* popisu byla nalezena nekonzistence mezi jednotlivými verzemi *OSCAR* protokolu. V některých zachycených zprávách převážně rodiny *0x0013* se nedeterministicky vyskytuje v datové části *SNAC* před korektním tělem *TLV* záznam typu *0x06* s neznámým významem. Tato situace je ošetřena podmínkou na přítomnost tohoto záznamu podle dvou bajtů značících typ.

5.3 NPL pro YMSG

Hlavička YMSG protokolu pro každý typ zprávy obsahuje stejné položky s předem známými datovými typy, proto je implementována jako struktura s datovými typy odpovídajícími velikosti uložení jednotlivých položek. Popis položky *type* obsahující typ zprávy je vybrán podle její hodnoty z tabulky slovních popisů jednotlivých typů zpráv (*YMSGTypes*). Následná datová část je pro každý typ zprávy jiná a obsahuje *TV* (*type-value*) záznamy.

Protože *YMSG* je textový protokol a všechny pole v datové části jsou *ASCII řetězce*, bylo by vhodné k jeho parsování použít funkce právě pro práci s řetězci. Jako nejvýhodnější se jeví funkce *ASCIIStringTerm(Term)*, která vrací řetězec ukončený libovolným zadaným ukončovacím řetězcem. Bohužel, v aktuální implementaci *NM 3.4* jsou uvažovány pouze znaky s ordinální hodnotou menší než *128 (0x80).* Toto omezení je velice svazující, protože v *YMSG* protokolu slouží jako ukončovač řetězec *0xC080*. Po zvážení možností a diskusi na fóru *TechNetu* [6] bylo jako alternativní řešení zvoleno vytvoření řetězce pomocí cyklického načítání znaků s ukončením, pokud nenásleduje ukončující znak.

Výše zmíněným způsobem jsou načteny všechny *TV* záznamy. Známým typům záznamu je přidělen popis z tabulky *YMSGTVTypes*. Pokud se jedná o záznam typu *YahooAwayStatus* je jako jeho hodnota vybrána odpovídající položka z tabulky (*YMSGStatus*) enumerace slovních popisů stavů klienta.

5.4 Testování

Testování bylo provedeno jako srovnání výstupu získaného nástrojem *Wireshark* a *MNM* s použitím námi vytvořených *NPL* popisů. Jako testovací soubor byl pro *ICQ* zvolen dříve zachycený PCAP (*mnm_all_sncas.cap*) a pro *YMSG* (*mnm_ymsg.pcap)*, při jejichž vytvoření byly provedeny všechny akce, které měly být rekonstruovány jako je zaslání zprávy, změna statusu a jiné. Oba tyto *PCAP soubory* jsou spolu s dalšími přítomny na přiloženém DVD.

Srovnáním výstupů *Wireshark* a *MNM* pro protokol *YMSG* se ukázalo, že oba výstupy jsou informační hodnotou ekvivalentní, pouze se liší pojmenováním některých dílčích prvků.

Srovnání výstupů pro protokol *OSCAR* ukázalo malé rozdíly na straně programu *Wireshark*, který nebyl schopný korektně rozpoznat některé typy zpráv. Tento rozdíl je zapříčiněn nekorektním zpracováním nedeterministicky se vyskytujícího *TLV* záznamu *0x06*. Obrazy sejmutých obrazovek ilustrující tento rozdíl se nachází v *Přílohách 4, 5*. Na obrázku zobrazující tabulku datagramů ve *Wiresharku* by se za vyznačeným místem měl nacházet datagram *SRV_ONLINExINFO (SNAC 0x001, 0x000f)*. Ten se nachází pouze ve výstupu programu *MNM*. Ostatní výstupy jsou stejně jako v případě protokolu *YMSG* ekvivalentní informační hodnotou, pouze se liší pojmenováním dílčích složek.

6 Závěr

Analýza protokolů *YMSG* a *ICQ* byla velice komplikovaná. Jelikož se jednalo o proprietární protokoly, nebylo možné čerpat z žádného oficiálního zdroje jako třeba *RFC*, *wiki výrobce* a jiné… Naopak informace bylo potřeba shromáždit s mnoha neoficiálních zdrojů, kterými byla internetová fóra, kde se řešil zpravidla problém týkající se pouze jedné konkrétní události, konstanty či zprávy, což tvořilo pouze malé procento informací potřebných k vytvoření povědomí o protokolu a následně sepsání analýzy. Dalším poněkud užitečnějším zdrojem, kde bylo možné najít ucelenější části informací na jednom místě, byly webové stránky odborníků, kteří se zabývali výzkumem výše zmíněných protokolů a na svých soukromých stránkách zveřejnili zjištěné poznatky. Asi nejcennějším zdrojem informací se nakonec ukázala být samotná analýza zachycených *PCAPů* zpočátku v nástroji *Wireshark*, který obsahuje sadu analyzačních skriptů, které dokáží komunikaci zobrazit v člověku přijatelnější formě. Později byl objeven problém, že *Wireshark* nedekóduje korektně všechny typy zpráv ani jimi přenášených hodnot, které byly v mnoha případech klíčové, proto následující analýza byla prováděna v *MNM*. Posledním zdrojem informací o obou protokolech byly zdrojové kódy jejich alternativních klientů, odkud byly převážně převzaty konstanty, které jsou použity pro kódování informací jako například stavu, v jakém se kontakt nachází. Pomocí informací shromážděných ze všech zmíněných zdrojů a ověření jejich relevance pomocí jiného zdroje byla vytvořena analýza protokolů.

Pro ověření správnosti analýzy protokolu *YMSG* byl vytvořen experimentální nástroj v jazyce Python, který zachycenou komunikace ve formě *PCAP* souboru rekonstruuje a výsledek překládá na výstup ve formátu *XML* podle předem zadaného schématu, viz *Příloha 1*. Nástroj je taktéž schopný rekonstruovat soubory, které byly během *IM komunikace* poslány. Porovnáním *XML* výstupu vytvořeného z *PCAP* souboru o předem známém obsahu s výstupem programu *Wireshark* je možné prohlásit analýzu protokolu *YMSG* za správnou, protože oba výstupy se v rekonstruovaných zprávách shodovaly. Při testování rekonstrukce zachycených souborů byly vytvořeny kontrolní *md5sum* součty a navzájem porovnány na shodu. Ve všech případech byly shodné.

Pro ověření analýzy protokolu používaného programem *ICQ* (protokol *OSCAR*), doplnění analýzy protokolu *YMSG* a pro potřeby projektu *Sec6net* byly vytvořeny popisy obou protokolů v jazyce *NPL* pro analýzu zachycené komunikace v *MNM*. Při jejich vytváření byly objeveny některé nové typy zpráv a byly nalezeny zprávy, které jsou nekorektně zobrazeny nebo úplně ignorovány programem *Wireshark*. Ostatní zprávy, které byly vybrány do množiny zpráv určených k rekonstrukci a jsou popsány v předešlých kapitolách, byly úspěšně rekonstruovány ze vzorku zachycené komunikace ve vstupním *PCAP souboru*. Výstup *MNM* byl shodný s výstupem programu *Wireshark*. Porovnáním výstupů zmíněných programů byla opět potvrzena správnost analýzy obou protokolů.

Nebylo možné rekonstruovat všechny typy komunikace např. audio-vizuální konference nebo VoIP hovor, protože přenos dat je šifrovaný pomocí *SSL/TLS*. Pro další práci by bylo vhodné zaměřit se na možnosti rekonstrukce této šifrované komunikace. Protože nový desktopový *ICQ klient* od *verze 7* nabízí integraci se sociálními sítěmi *Facebook, Google+, Twitter* a *Flickr*, bylo by možné zachytávat a následně rekonstruovat komunikaci i z těchto sítí za předpokladu, že sledovaný subjekt má tyto sítě integrované do svého *ICQ klienta*.

Tato práce vznikla za podpory projektu MŠMT CZ.1.07/2.3.00/09.0067: „TeamIT – Budování konkurenceschopných výzkumných týmů pro IT“ a projektu MV VG20102015022: „Sec6net – Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace“.

Bibliografie

[1] MICROSOFT. microsoft. In: *Microsoft Network Monitor 3.4* [online]. 24. 06. 2010 [cit. 2012-01-10]. Dostupné z: http://www.microsoft.com/download/en/details.aspx?id=4865

[2] WIKIMEDIA FOUNDATION. wikipedie. In: *Instant messaging* [online]. [cit. 2012-01-22]. Dostupné z: http://cs.wikipedia.org/wiki/Instant_messaging

[3] COMBS, G. wireshark. In: *wireshark* [online]. [cit. 2012-01-23]. Dostupné z: http://www.wireshark.org/

[4] FRITZLER, A. oilcan. In: *AIM/Oscar Protocol Specification* [online]. 08. 05. 2008 [cit. 2012-03-14]. Dostupné z: http://www.oilcan.org/oscar/

[5] SHUTKO, A. iserverd. In: *OSCAR (ICQ v7/v8/v9) protocol documentation* [online]. 07. 02. 2005 [cit. 2012-02-13]. Dostupné z: http://iserverd.khstu.ru/oscar/

[6] PLUSKAL, J. technet. In: *Parsing ASCII string ended with character sequence 0xc080* [online]. 26. 02. 2012 [cit. 2012-03-05]. Dostupné z: http://social.technet.microsoft.com/Forums/en-US/netmon/thread/ca3b07cf-c3df-4c8c-8522-5516dea15747

[7] ANSA TEAM FIT VUTBR. ANSA. In: *ANSA Wiki* [online]. Brno (Česká Republika): [cit. 2012-05-25]. Dostupné z: https://nes.fit.vutbr.cz/ansa/pmwiki.php?n=Main.HomePage

[8] search.com reference. In: *Yahoo! Messenger Protocol* [online]. [cit. 2011-10-13]. Dostupné z: http://www.search.com/reference/Yahoo!_Messenger_Protocol

[9] carbonize. In: *Yahoo's YMSG Protocol v16* [online]. [cit. 2012-03-12]. Dostupné z: http://carbonize.co.uk/ymsg16.html

[10] CLARK, J. cs.cmu. In: *On Sending Files via OSCAR* [online]. 16. 01. 2006 [cit. 2012-02-26]. Dostupné z: http://webcache.googleusercontent.com/search?q=cache:8P8w-

[11] COMBS, G. minerva. In: *Routines for Yahoo Messenger YMSG protocol packet version 13 dissection* [online]. 31. 10. 2006 [cit. 2011-10-15]. Dostupné z: https://minerva.netgroup.uniroma2.it/svn/discreet/tfcproject/trunk/wireshark/epan/dissect

[12] JACQUES, A. Adren Software. In: *yahoo_v16_protocol* [online]. [cit. 2011-10-14]. Dostupné z: http://www.adrensoftware.com/tools/yahoo_v16_protocol.php

[13] MOTIWALA, Y. In: *Yahoo Messenger VoIP Service with SIP Phone* [online]. 18. 08. 2007 [cit. 2011-11-10]. Dostupné z: http://blog.motiwala.com/2007/08/18/yahoo-messenger-voip-service-with-sip-phone

[14] YAZAKPRO. yazakpro. In: *ymsgversions* [online]. 02. 02. 2011 [cit. 2011-10-15]. Dostupné z: http://www.yazakpro.com/yazak/help/ymsgversions.html

[15] PLUSKAL, J. Instant Messaging Network Analysis and Reconstruction (YMSG and ICQ). In: DRAHANSKÝ , M. a F. ORSÁG. *Proceeding of the 18th Conference STUDENT EEICT 2012 Volume 1*. první. Brno: LITERA Brno, 2012, s. 176-178. ISBN 978-80-214-4460-7. Dostupné také z: http://www.feec.vutbr.cz/EEICT/2012/sbornik/01bakalarskeprojekty/10pocitacovesystemy/07-xplusk03.pdf

[16] JENNINGS, R. B., NAHUM, E.M., OLSHEFSKI, D.P., SAHA, D., ZON-YIN SHAE, WATERS, C. *A study of Internet instant messaging and chat protocols* Network, IEEE , vol.20, no.4, pp.16-21, July-Aug. 2006 doi: 10.1109/MNET.2006.1668399 Dostupné z WWW: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1668399&isnumber=34939

Seznam příloh

Příloha 1: Schéma výsledného XML výstupu YMSG

EVENT *type*	**EVENT další atributy**		
YAHOO_SERVICE_LOGOFF odhlášení kontaktu nebo clienta			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
logoff		*contactId* id kontaktu, pokud chybí odhlásil se client	

EVENT *type*	**EVENT další atributy**		
YAHOO_SERVICE_MESSAGE přenos zprávy			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
message	yahooMsg	*TransmiterYahooID* odesílatel	*ReciverYahooID* příjmající

EVENT *type*	**EVENT další atributy**		
YAHOO_SERVICE_LIST_V15 načtení kontakt listu			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
list	TAG contact		
contact		*group* skupina kontaktů	*contactId* id kontaktu

EVENT *type*	**EVENT další atributy**				
SERVICE_ADD_BUDDY přidání kontaktu					
TAG	**Hodnota TAGu**	**Atributy TAGu**			
contact		*group* skupina kontaktů	*contactId* id kontaktu	*firstName* křestní jméno	*surname* příjmení

EVENT *type*	**EVENT další atributy**			
YAHOO_SERVICE_Y7_BUDDY_AUTH autorizace				
TAG	**Hodnota TAGu**	**Atributy TAGu**		
contact		*firstName* křestní jméno pouze u typu request	*surname* příjmení pouze u typu request	*TransmiterYahooID* odesílatel
		ReciverYahooID příjmající	*authorization* deny, granted, request pouze u typu request autorizace kontaktu	

EVENT *type*	**EVENT další atributy**		
YAHOO_SERVICE_STATUS_V15 změna statusu při přihlášení			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
status		*awayStatus* YAHOO_STATUS_AVAILABLE, YAHOO_STATUS_BRB, YAHOO_STATUS_BUSY, YAHOO_STATUS_NOTATHOME, YAHOO_STATUS_NOTATDESK, YAHOO_STATUS_NOTINOFFICE, YAHOO_STATUS_ONVACATION, YAHOO_STATUS_OUTTOLUNCH, YAHOO_STATUS_STEPPEDOUT, YAHOO_STATUS_CUSTOM, YAHOO_STATUS_IDLE, YAHOO_STATUS_WEBLOGIN, YAHOO_STATUS_OFFLINE, YAHOO_STATUS_TYPING enumerace vestavěných typů online,busy,away	*contactID* id kontaktu který změnil status, pokud chybí změnil status sledovaný klient *awayMsg* zpráva ve statusu, volitelná, uživatelem definovaná

EVENT *type*	**EVENT další atributy**		
YAHOO_SERVICE__Y7_STATUS_UPDATE Změna statusu			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
status		*awayStatus* YAHOO_STATUS_AVAILABLE, YAHOO_STATUS_BRB, YAHOO_STATUS_BUSY, YAHOO_STATUS_NOTATHOME, YAHOO_STATUS_NOTATDESK, YAHOO_STATUS_NOTINOFFICE, YAHOO_STATUS_ONVACATION, YAHOO_STATUS_OUTTOLUNCH, YAHOO_STATUS_STEPPEDOUT, YAHOO_STATUS_CUSTOM, YAHOO_STATUS_IDLE, YAHOO_STATUS_WEBLOGIN, YAHOO_STATUS_OFFLINE, YAHOO_STATUS_TYPING enumerace vestavěných typů online,busy,away	*awayMsg* zpráva ve statusu, volitelná, uživatelem definovaná *contactID* id kontaktu který změnil status, pokud chybí změnil status sledovaný klient

EVENT *type*	**EVENT další atributy**				
YAHOO_SERVICE_Y7_FILE_TRANSFER_ACCEPT transfer souboru					
TAG	**Hodnota TAGu**	**Atributy TAGu**			
fileTransfer		*TransmiterYahooID* odesílatel	*ReciverYahooID* příjmající	*fileName* jméno odesílaného souboru	*fileSize* velikosti souboru

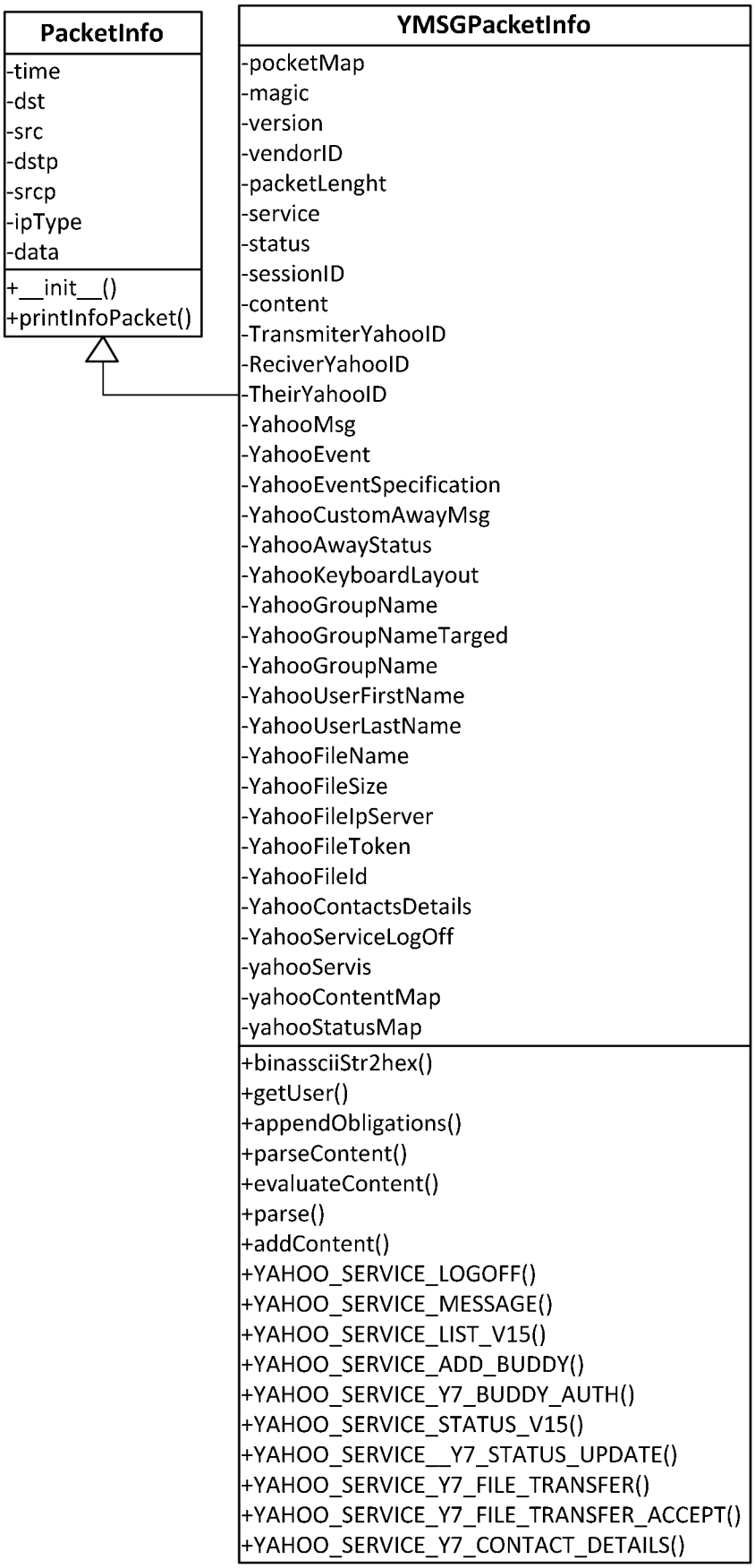
EVENT *type*	EVENT další atributy		
YAHOO_SERVICE_Y7_CONTACT_DETAILS detaily klienta			
TAG	**Hodnota TAGu**		**Atributy TAGu**
details	client details <yi> kontakt id <fn> křestní jméno <mn> prostřední jméno <ln> příjemní <e0> email 1 <e1> email 2 <e2> email 3 <pu> osobní webové stránky <ha>domácí adresa ulice <hc> domácí adresa město <hs> domácí stát <hz> domácí adresa zip kód <hn> domácí adresa země <hp> domácí telefon	<ti> zaměstnání titul <co> zaměstnání společnost jméno <wa> zaměstnání adresa ulice <wc> zaměstnání adresa město <ws> zaměstnání adresa stát <wz> zaměstnání adresa zip <wn> zaměstnání adresa země <wp> zaměstnání telefon <wu> zaměstnání web stránky <mo> mobilní číslo <pa> pager <fa> fax <ot> jiné info <bi> datum narození ve tvaru dd/dd/yyyy	

Příloha 2: HTTP GET požadavek zastupující SNAC(0x0002,0x0009)

GET

/memberDir/update?a=%2FwQAAAAAAAEYqZrdJMIXBZ1hDbdc6iEkHBuwQA6MBWKcm9RwX0SLFR4wCS5CUAmfPjm%2BwA8hdmOoQKIWioPHira7YKDzy6Ie8kLtJxtCJhfTi1uKHp%2BrmAeZyOZoSPJowAwrdplrYzr61O29MuZ7HEzKr85XNWptClWo3q7puslosEIASI%2B4b71D3iI%3D&f=json&k=gu19PNBblQjCdbMU&set=aboutMe=about_me&set=birthDate=228009600&set=firstName=first_name&set=friendlyName=nick_name&set=gender=male&set=homeAddress={city=current_city,state=cu_state,country=US}&set=interests=[{code=music,text=music%252Cmusic2},{code=moviesAndTv,text=movie%252Cmovie1},{code=hobbies,text=hobbie%252Chobbie1}]&set=jobs=[{company=work,title=position,industry=highTech,subIndustry=informationServices,website=http%3A%2F%2Fcompanyweb.com}]&set=lang1=ALB&set=lang2=ARM&set=lang3=AZE&set=lastName=last_name&set=originAddress={city=home_city,state=hom_state,country=US}&set=phones=[{phoneType=mobile,phone=0123456798},{phoneType=home,phone=93123456789},{phoneType=work,phone=0123456789},{phoneType=homeFax,phone=0123456789},{phoneType=workFax,phone=0123456789},{phoneType=other,phone=0123456789}]&set=relationshipStatus=single&set=website1=http://website.com&ts=1330020259&sig_sha256=1VENS8aPtBWXW6XbDyuuTorOHgi4gHbOxa4DqUOXA6w=

Příloha 3: UML diagram tříd zobrazující strukturu skriptu



Příloha 4: Chybné rozpoznání datagramu Wiresharkem

No.	Time	Source	Destination	Protocol	Length	Info
112	56.40	205.188.10.251	192.168.2.101	AIM Buddylist	323	AIM Buddylist, Rights Reply
114	56.40	205.188.10.251	192.168.2.101	AIM SSI	841	AIM SSI, List
115	56.41	192.168.2.101	205.188.10.251	AIM SSI	70	AIM SSI, Activate
116	56.41	192.168.2.101	205.188.10.251	AIM Generic	631	AIM Generic, Client Ready
144	56.90	192.168.2.101	205.188.10.251	AIM Generic	590	[TCP Retransmission] AIM Generic, Set Extended Status Request
185	57.51	192.168.2.101	205.188.10.251	AIM Generic	590	[TCP Retransmission] AIM Generic, Set Extended Status Request
189	57.67	205.188.10.251	192.168.2.101	AIM Stats	74	AIM Stats, Set Report Interval
190	57.67	192.168.2.101	205.188.10.251	AIM Location	439	AIM Location, Set User Info
191	57.68	205.188.10.251	192.168.2.101	AIM	730	SNAC data, Family: 0x0025, Subtype: 0x0003
199	57.87	205.188.10.251	192.168.2.101	AIM Buddylist	279	Oncoming Buddy: 647175775
200	57.88	205.188.10.251	192.168.2.101	AIM Buddylist	290	Oncoming Buddy: 310451170
201	57.88	205.188.10.251	192.168.2.101	AIM Buddylist	272	Oncoming Buddy: 288547266
214	58.17	192.168.2.101	205.188.10.251	AIM Location	439	[TCP Retransmission] AIM Location, Set User Info
215	58.18	205.188.10.251	192.168.2.101	AIM Buddylist	98	Offgoing Buddy: 277413921
245	58.67	205.188.10.251	192.168.2.101	AIM	1414	[TCP Retransmission] SNAC data, Family: 0x0025, Subtype: 0x0003
249	58.72	205.188.10.251	192.168.2.101	AIM Generic	1414	[TCP Retransmission] AIM Generic, Self Info Reply
250	58.72	192.168.2.101	205.188.10.251	AIM Messaging	70	SNAC data, AIM Messaging, Subtype: 0x0010
252	58.79	205.188.10.251	192.168.2.101	AIM Buddylist	524	AIM Buddylist, Rights Reply

Příloha 5: Chybějící datagram z Wiresharku zobrazený v NM 3.4

Frame Number	Time ...	Source	Destination	P..	Description
118	59.26...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x13, subtype = 0x3,
119	59.27...	VP-PC	205.188.10.251	ICQ	Ch = 2,family = 0x13, subtype = 0x7,
120	59.27...	VP-PC	205.188.10.251	ICQ	Ch = 2,family = 0x25, subtype = 0x2, Ch = 2,family = 0x2, subtype = 0x4, CLI_S...
193	60.53...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0xb, subtype = 0x2,
194	60.53...	VP-PC	205.188.10.251	ICQ	Ch = 2,family = 0x3, subtype = 0x2, Ch = 2,family = 0x1, subtype = 0x1e, CLI_S...
195	60.54...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x25, subtype = 0x3, srv information response on client update use...
201	60.73...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x1, subtype = 0xf, SRV_ONLINExINFO: 647175775
203	60.73...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x3, subtype = 0xb, SRV_USER_ONLINE: 647175775
204	60.74...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x3, subtype = 0xb, SRV_USER_ONLINE: 310451170
205	60.74...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x3, subtype = 0xb, SRV_USER_ONLINE: 288547266
219	61.03...	205.188.10.251	VP-PC	ICQ	Ch = 2,family = 0x3, subtype = 0xc, SRV_USER_OFFLINE: 277413921

Frame Details

```
Frame: Number = 201, Captured Frame Length = 840, MediaType = ETHERNET
Ethernet: Etype = Internet IP (IPv4),DestinationAddress:[08-00-27-36-DB-AF],SourceAddress:[
Ipv4: Src = 205.188.10.251, Dest = 192.168.2.101, Next Protocol = TCP, Packet ID = 58753, To
Tcp: Flags=...AP..., SrcPort=5190, DstPort=49874, PayloadLen=786, Seq=93859445 - 93860231, A
ICQ: Ch = 2 family = 0x1, subtype = 0xf
    FLAP_start: 42 (0x2A)
    FLAP_channel: 2 (0x2)
    FLAP_seq: 46585
    FLAP_size: 780
  snac: family = 0x1, subtype = 0xf,SRV_ONLINExINFO: 647175775
      FamilyID: Generic service controls1 (0x01)
      FamilySubtypeID:  Requested online info response (0x0f)
      Flags: 32768 (0x8000)
      RequestID: 2230230840 (0x84EE9F38)
    snac010f: SRV_ONLINExINFO: 647175775
      TVL:
        Unknown: 6 (0x6)
        Type: 1 (0x1)
        Length: 2 (0x2)
        Value: Binary Large Object (2 Bytes)
      UinStringLen: 9 (0x9)
      Uin: 647175775
      WarningLevel: 0 (0x0)
      TVLsInFixPart: 20 (0x14)
      TLVrecord: Type: 0x1
      TLVrecord: Type: 0x6
      TLVrecord: Type: 0x5
```

Příloha 6: Schéma XML pro ICQ

EVENT *type*	**EVENT další atributy**			
STATUS informace o kontaktu, nebo klientovi				
TAG	**Hodnota TAGu**	**Atributy TAGu**		
status		*status* status klienta		
status		*status* status klienta	*awayMsg* away zpráva	
status		*status* status kontaktu	*uin* identifikátor kontaktu	
status		*status* status kontaktu	*uin* identifikátor kontaktu	*awayMsg* away zpráva

EVENT *type*	**EVENT další atributy**	
SRV_USER_OFFLINE kontakt se odhlásil		
TAG	**Hodnota TAGu**	**Atributy TAGu**
logoff		*uin* identifikační číslo

EVENT *type*	**EVENT další atributy**	
LOG_OFF klient se odhlásil		
TAG	**Hodnota TAGu**	**Atributy TAGu**
logoff		

EVENT *type*	EVENT další atributy					
CLI_UPDATExDIRxINFO změna informací o klientovi						
TAG	**Hodnota TAGu**	**Atributy TAGu**				
info		*firstName* křestní jméno	*lastName* příjmení	*middle* prostřední jméno	*maidenName* jméno za svobodna	*country* země
		state stát	*city* město	*nickName* přezdívka	*zipCode* město	*streetAddr* adresa ulice

EVENT *type*	EVENT další atributy	
CLI_BUDDYLIST_ADD klient přidal kontakt(y) do listu kontaktu		
TAG	**Hodnota TAGu**	**Atributy TAGu**
addBuddyList		*uin* identifikační číslo

EVENT *type*	EVENT další atributy	
CLI_BUDDYLIST_REMOVE klient odstranil kontakt(y) z listu kontaktu		
TAG	**Hodnota TAGu**	**Atributy TAGu**
removeBuddyList		*uin* identifikační číslo

EVENT *type*	EVENT další atributy	
MESSAGE zpráva		
TAG	**Hodnota TAGu**	**Atributy TAGu**
message	obsah zprávy	*sender* uin odesilatele
message	obsah zprávy	*reciver* uin příjemce

EVENT *type*	**EVENT další atributy**			
CLI_SSIxREPLY update lokálního listu kontaktů				
TAG	**Hodnota TAGu**	**Atributy TAGu**		
clientContactListUpdate	TAG contakt, nebo TAG group			
contact		*uin* identifikátor	*groupID* skupina	*nick* nickname v klientském cl, nepovinné
group		*groupID* identifikátor	*name* jméno skupiny	

EVENT *type*	**EVENT další atributy**			
CLI_SSIxADD přidání do server side listu kontaktů				
TAG	**Hodnota TAGu**	**Atributy TAGu**		
contactListAdd	TAG contakt nebo TAG group			
contact		*uin* identifikátor	*groupID* skupina	*nick* nickname v klientském cl, nepovinné
group		*groupID* identifikátor	*name* jméno skupiny	

EVENT *type*	**EVENT další atributy**			
CLI_SSIxUPDATE přidání do server side listu kontaktů				
TAG	**Hodnota TAGu**	**Atributy TAGu**		
contactListUpdate	TAG contakt nebo TAG group			
contact		*uin* identifikátor	*groupID* skupina	*nick* nickname v klientském cl, nepovinné
group		*groupID* identifikátor	*name* jméno sk.	

EVENT *type*	**EVENT další atributy**	
CLI_SSIxDELETE přidání do server side listu kontaktů		
TAG	**Hodnota TAGu**	**Atributy TAGu**
'contactListDelete	TAG contakt nebo TAG group	

contact		*uin* identifikátor	*groupID* skupina	*nick* nickname v klientském cl, nepovinné
group		*groupID* identifikátor	*name* jméno skupiny	

EVENT *type*	**EVENT další atributy**		
AUTHxREQUEST požadavek na autorizaci			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
authRequest		*uin* identifikátor	
EVENT *type*	**EVENT další atributy**		
AUTHxREPLY odpověď na autorizaci			
TAG	**Hodnota TAGu**	**Atributy TAGu**	
authReply		*uin* identifikátor	*state* accept, decline stav

Příloha 7: Příklad výstupu experimentálního nástroje pro rekonstrukci YMSG

Soubor na DVD YMSG_login_offlineMsg_to_bpfitvut1_2.xml

```
<?xml version="1.0" ?>
<log>
  <user guid="192.168.2.102">
    <protocol port="5050" val="YMSG">
      <event type="YAHOO_SERVICE_LIST_V15">
        <list>
          <contact contactId="bpfitvut1" group="Friends"/>
        </list>
        <timestamp>Oct 16, 2011 15:19:13.440632</timestamp>
        <src version="ipv4">67.195.186.85</src>
        <dst version="ipv4">192.168.2.102</dst>
      </event>
      <event type="YAHOO_SERVICE_MESSAGE">
        <message ReciverYahooID="bpfitvut1"
           TransmiterYahooID="bpfitvut">ahoj, offline zprava to
           bpfitvut1</message>
        <timestamp>Oct 16, 2011 15:19:31.842172</timestamp>
        <src version="ipv4">192.168.2.102</src>
        <dst version="ipv4">67.195.186.85</dst>
      </event>
      <event type="YAHOO_SERVICE_STATUS_V15">
        <status awayStatus="YAHOO_STATUS_AVAILABLE"
           contactId="bpfitvut1"/>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">67.195.186.85</src>
        <dst version="ipv4">192.168.2.102</dst>
      </event>
      <event type="YAHOO_SERVICE_MESSAGE">
        <message ReciverYahooID="bpfitvut"
           TransmiterYahooID="bpfitvut1">odpoved na offline
           zpravu to bpfitvut</message>
        <timestamp>Oct 16, 2011 15:20:09.470433</timestamp>
        <src version="ipv4">67.195.186.85</src>
        <dst version="ipv4">192.168.2.102</dst>
      </event>
      <event type="YAHOO_SERVICE_LOGOFF">
        <logoff contactId="bpfitvut1"/>
        <timestamp>Oct 16, 2011 15:20:15.189460</timestamp>
        <src version="ipv4">67.195.186.85</src>
        <dst version="ipv4">192.168.2.102</dst>
      </event>
      <event type="YAHOO_SERVICE_LOGOFF">
        <logoff/>
        <timestamp>Oct 16, 2011 15:20:38.824976</timestamp>
        <src version="ipv4">192.168.2.102</src>
        <dst version="ipv4">67.195.186.85</dst>
      </event>
    </protocol>
  </user>
</log>
```

Příloha 8: Příklad výstupu XML pro ICQ

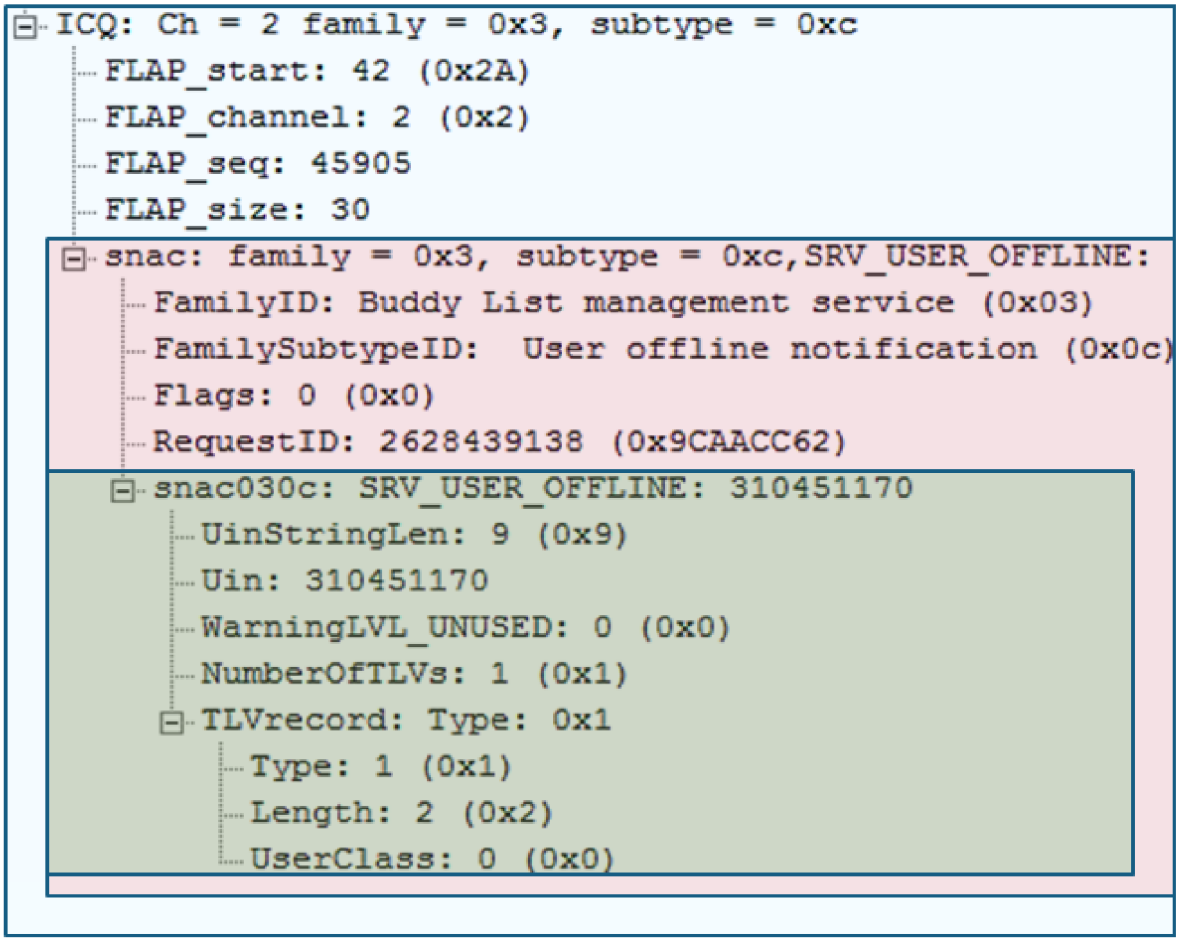
Soubor na dvd login_msg_addContakt_msg_removeContakt_logout.xml

```
<?xml version="1.0" ?>
<log>
  <user guid="192.168.111.14">
    <protocol port="5190" val="ICQ">
      <event type="CLI_SSIxREPLY">
        <clientContactListUpdate>
           <group  groupID="0x0001" name="Not In List" />
           <contact uin="310451170" groupID="0x0001" nick="Jack" />
        </clientContactListUpdate>
        <timestamp>Oct 16, 2011 15:19:13.440632</timestamp>
        <src version="ipv4">205.188.10.251</src>
        <dst version="ipv4">192.168.111.14</dst>
      </event>
      <event type="MESSAGE">
        <message reciver="310451170" >&lt;HTML&gt;&lt;BODY
           dir="ltr"&gt;&lt;FONT
           color=&quot;#000000&quot; size=&quot;2&quot;
           face=&quot;Arial&quot;&gt;zprava
           online&lt;/FONT>&lt;/BODY&gt;&lt;/HTML&gt;</message>
        <timestamp>Oct 16, 2011 15:19:31.842172</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
      <event type="CLI_SSIxUPDATE">
        <contactListUpdate>
      <contact uin="310451170" groupID="0x0001" nick="Jack" />
      </contactListUpdate>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">205.188.10.251</src>
        <dst version="ipv4">192.168.111.14</dst>
      </event>
      <event type="CLI_SSIxADD">
        <contactListAdd>
      <group  groupID="0x25b8" name="friends_group" />
      </contactListAdd>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
      <event type="CLI_SSIxADD">
        <contactListAdd>
      <group  groupID="0x0002" name="friends_group" />
      </contactListAdd>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
```

```
<event type="CLI_SSIxADD">
  <contactListAdd>
<contact uin="310451170" groupID="0x0002" nick="Jack" />
</contactListAdd>
  <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
  <src version="ipv4">192.168.111.14</src>
  <dst version="ipv4">205.188.10.251</dst>
</event>
<event type="CLI_SSIxDELETE">
  <contactListDelete>
<contact uin="310451170" groupID="0x0001" nick="Jack" />
</contactListDelete>
  <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
  <src version="ipv4">192.168.111.14</src>
  <dst version="ipv4">205.188.10.251</dst>
</event>
 <event type="CLI_SSIxUPDATE">
  <contactListUpdate>
<group  groupID="0x0002" name="friends_group" />
</contactListUpdate>
  <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
  <src version="ipv4">205.188.10.251</src>
  <dst version="ipv4">192.168.111.14</dst>
</event>
<event type="CLI_SSIxUPDATE">
  <contactListUpdate>
<contact uin="310451170" groupID="0x0002" nick="Jack" />
</contactListUpdate>
  <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
  <src version="ipv4">205.188.10.251</src>
  <dst version="ipv4">192.168.111.14</dst>
</event>
<event type="SRV_USER_OFFLINE">
  <logoff>
<contact uin="310451170" />
  </logoff>
  <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
  <src version="ipv4">205.188.10.251</src>
  <dst version="ipv4">192.168.111.14</dst>
</event>
<event type="MESSAGE">
  <message reciver="310451170" >&lt;HTML&gt;&lt;BODY
     dir="ltr"&gt;&lt;FONT
     color=&quot;#000000&quot; size=&quot;2&quot;
     face=&quot;Arial&quot;&gt;pridan
     kontakt
     jack&lt;/FONT>&lt;/BODY&gt;&lt;/HTML&gt;</message>
  <timestamp>Oct 16, 2011 15:19:31.842172</timestamp>
  <src version="ipv4">192.168.111.14</src>
  <dst version="ipv4">205.188.10.251</dst>
</event>
```

```
      <event type="MESSAGE">
        <message reciver="310451170" >&lt;HTML&gt;&lt;BODY
          dir="ltr"&gt;&lt;FONT
          color=&quot;#000000&quot; size=&quot;2&quot;
          face=&quot;Arial&quot;&gt;nasleduje odstraneni
          kontaktu&lt;/FONT>&lt;/BODY&gt;&lt;/HTML&gt;</message>
        <timestamp>Oct 16, 2011 15:19:31.842172</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
      <event type="CLI_SSIxDELETE">
        <contactListDelete>
      <contact uin="310451170" groupID="0x0002" nick="Jack" />
      </contactListDelete>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
      <event type="CLI_SSIxUPDATE">
        <contactListUpdate>
      <group  groupID="0x0002" name="friends_group" />
      </contactListUpdate>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">205.188.10.251</src>
        <dst version="ipv4">192.168.111.14</dst>
      </event>
      <event type="CLI_SSIxUPDATE">
        <contactListUpdate>
      <group  groupID="0x0001" name="Not In List" />
      <group  groupID="0x0002" name="friends_group" />
        </contactListUpdate>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">205.188.10.251</src>
        <dst version="ipv4">192.168.111.14</dst>
      </event>
      <event type="CLI_SSIxDELETE">
        <contactListDelete>
      <group  groupID="0x0002" name="friends_group" />
      </contactListDelete>
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
      <event type="LOG_OFF">
        <logoff />
        <timestamp>Oct 16, 2011 15:19:48.628663</timestamp>
        <src version="ipv4">192.168.111.14</src>
        <dst version="ipv4">205.188.10.251</dst>
      </event>
    </protocol>
  </user>
</log>
```

Příloha 9: Ukázka výstupu MNM pro ICQ



Příloha 10: Ukázka výstupu MNM pro YMSG