

Standardy pro přenos e-mailových zpráv

29. května 2005

1 Model komunikace

Při běžné komunikaci pomocí e-mailu s použitím protokolů pop3 a smtp je situace následující (viz obrázek 1):

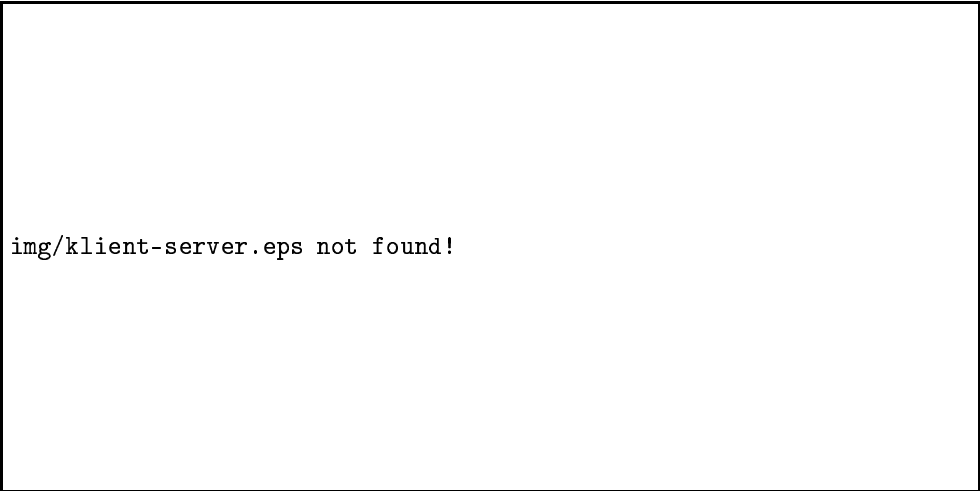
Pokud chce klient odeslat zprávu, musí ji předat pomocí protokolu smtp serveru s informacemi o příjemcích a odesílateli. Ten zjistí, které servery se starají o schránky příjemců. Pomocí protokolu smtp jim pak zprávu předá stejným způsobem, jako ji dostal (je v tuto chvíli klientem). Takto si předávají servery zprávu do té doby, dokud nedorazí na cílový server, ten ji vloží do příslušné schránky.

Zprávy pak klient získává od serveru pomocí protokolu pop3. V průběhu spojení se serverem většinou klient nejdříve zjistí, jestli nepřišly nové zprávy, poté si tyto případně od serveru vyžádá. Dále může určovat zprávy určené k smazání, ty se odstraní až při skončení komunikace. Během komunikace je stav schránky „zamčen“, nezobrazují se tedy ani nové zprávy.

2 Požadavky standardů

2.1 Pop3 (Post office protocol v.3)

Protokol Pop3 Je určen pro jednoduchý přístup k e-mailovým schránkám. Protokolu pop3 [11] je přiřazen port 110, funguje nad tcp [12], je textově orientovaný. Text by měl sestávat pouze z tisknutelných znaků znakové sady us-ascii. Řádky jsou oddělovány sekvencí CRLF (us-ascii [1] znaky 13 a 10). Komunikace probíhá pomocí příkazů a odpovědí, příkaz se obecně sestává z kódového slova (3-4 znaky, které jsou brány bez rozlišení velikosti), případně následovaného argumenty oddělenými mezerou (SP - znak 32), maximální délka jednoho argumentu je 40 znaků. Odpovědi sestávají z indikátoru stavu (+OK pro úspěch, -ERR pro selhání, opět nezáleží na velikosti písmen), a případných argumentů oddělených SP. Řádka odpovědi může být maximálně 512 znaků dlouhá. U některých příkazů mohou být odpovědi víceřádkové, u nich je to jasně dáno z kontextu. Zde po první řádce indikující stav následují další informativní řádky, konec je indikován sekvencí CRLF „.“ CRLF (znaky 13,10,46,13,10). Pokud by se v textu víceřádkové odpovědi vyskytoval znak „“ (desítkově 46) na začátku



img/klient-server.eps not found!

Obrázek 1: komunikace se serverem

řádky, je na straně serveru nahrazen za „.“, interpretace na straně klienta pak vypadá tak, že klient nejprve zkontroluje, zda řádka nezačíná „.“. Pokud ano a následuje ještě jednou „.“, první znak „.“ odstraní a pokračuje ve čtení, pokud však po „.“ následuje CRLF, považuje toto za konec odpovědi.

Pop3 server se během transakce může dostat do čtyř různých stavů, a to *greeting*, *auth*, *trans* a *update*. Po navázání spojení se klientu server ohlásí, je ve stavu *greeting*. Do stavu *auth* přejde po požadavku o přihlášení uživatele, zde čeká na zadání přístupových informací. Po úspěšné identifikaci uživatele přechází do stavu *trans*, zamkne informace o stavu účtu, přiřadí zprávám čísla 1...n a určí jejich velikost v byte. Během transakce se data nemění (tedy ani nepřidávají nové zprávy). Po přijetí příkazu QUIT přechází do stavu *update*, kde maže zprávy k mazání určené a ruší dočasná data.

Server se může při delší nečinnosti ze strany klienta odpojit, v tomto případě neprochází stavem *update*, nesmí to však udělat dříve než po deseti minutách.

V základní definici protokolu (bez rozšíření) jsou definovány dvě metody pro přihlášení:

- pomocí příkazů `USER name PASS password`
- a příkazu APOP (používá md5 digest)

V části *trans* pak lze používat tyto příkazy:

STAT *(bez parametrů)*. Vrátí počet zpráv, odpověď je +OK SP počet_zpráv
SP velikost_schránky [SP další informace] CRLF

LIST *[SP číslo_zprávy] CRLF*. Udává velikost jednotlivých zpráv. V případě, že server dostane číslo zprávy, odpoví pouze jednou řádkou

- +OK SP číslo_zprávy SP velikost CRLF*. Pokud nedostane, je odpovědí více řádek, první ve tvaru *+OK SP počet_zpráv CRLF*, další ve tvaru *číslo_zprávy SP velikost CRLF*, poslední „CRLF“.
- RETR *SP číslo_zprávy CRLF*. První řádka *+OK CRLF*. Na dalších vrátí ve víceřádkové podobě zprávu s daným číslem.
- DELE *SP číslo_zprávy CRLF*. Označí danou zprávu ke smazání, při dalším odkazování na ni hlásí -ERR, maže se však až v update.
- NOOP *CRLF*. Nic nedělá, pouze resetuje časovač pro automatické odpojení.
- RSET *CRLF*. Odstraní příznaky mazání ze zpráv.
- QUIT *CRLF*. Pokud je v jiném stavu než *auth*, přejde do *update*, vrátí +OK, když smaže všechny označené zprávy, -ERR, pokud se některé nepovedlo smazat. Vždy však odblokuje mailbox a uzavře spojení.

Dále jsou příkazy, které nemusí server podporovat, aby splňoval normu, jejich implementace je však doporučená:

- TOP *SP číslo_zprávy SP počet_řádků*. Odpoví víceřádkově, první řádka *+OK CRLF*, další jsou prvních počet_řádků řádků zprávy (případně méně, pokud je kratší), zakončené „CRLF“.
- UIDL *[SP číslo_zprávy] CRLF*. Vrátí identifikátor zprávy pro mailbox (měl by být neměnný a jedinečný) a to při udání čísla jednořádkově ve tvaru *+OK SP číslo_zprávy SP uidl CRLF*, nebo víceřádkově pro všechny zprávy neoznačené ke smazání. Tvar je pak u první řádky *+OK CRLF*, u dalších pak *číslo_zprávy SP uidl CRLF*. Zakončené opět „CRLF“.

2.2 SmtP (Simple mail transfer protocol)

Protokol SMTP [13, 8] je určený k přenosu zpráv, poskytuje prostředky pro vložení zpráv do systému a předávání zpráv mezi servery až k cílovému. Funguje na základě protokolu tcp [12], má přidělen port 25. Je textově orientovaný, založený na principu dotaz - odpověď, používá znakovou sadu us-ascii. Bez rozšíření [9] podporuje pouze dolních 7 bitů z byte. Příkazy a odpovědi sestávají z řádků oddělených sekvencí CRLF (us-ascii [1] znaky 13 a 10). Příkazy jsou alfabeticke řetězce, za nimi případně následují argumenty oddělené SP (us-ascii 32), zakončené CRLF. Odpověď serveru sestává z tří číslic (indikátoru stavu) a případného textu. Text je od číslic oddělen mezerou (SP). Každá odpověď může být víceřádková, poslední řádka musí mít tvar jednořádkové odpovědi, řádky předtím mají místo SP použit k oddělení indikátoru a textu znak „-“ (us-ascii 95). Tedy víceřádková odpověď může vypadat např. tak, že prvních n řádků je *250-keyword SP args* (indikátor „-“ text) a poslední řádek *250 SP keyword SP args* (tedy indikátor SP text).

Zde bude tento protokol popsán z hlediska mailového klienta, informace o předávání zpráv (relaying) a podobné budou vynechány.

Indikátory stavu se dělí do skupin podle jednotlivých číslic, první je nejvýznamnější, poslední nejméně.

Významové třídy jsou pak následující:

1xx	příkaz byl přijat, čeká na potvrzení informací z této odpovědi (pokračovat/zrušit), vrací jen extenze
2xx	akce byla přijata, v pořádku dokončena, může začít novou
3xx	částečné přijetí, server čeká na další informace (data) od klienta
4xx	dočasné odmítnutí = tento příkaz (dotaz) nelze teď splnit, všeobecně však není špatně a klient by to měl zkusit později
5xx	trvalé odmítnutí. nelze provést z trvalých důvodů (syntaktická chyba, špatná data,...) znovu by se mělo zkoušet až po opravě
x0x	syntaktické chyby, neimplementované či špatné příkazy
x1x	žádost o informace (např. help)
x2x	odpovědi informující o spojení (změně, připojen, odpojen,...)
x3x,x4x	nespecifikováno
x5x	odpověď indikuje stav systému (změnu)

Příkazy protokolu smtp jsou textové, bez rozlišení velikosti písmen, s jednoznačným významem. Standard pro protokol smtp povoluje registraci rozšíření a určuje způsob, kterým o jejich podpoře informuje server klienta.

V základním znění pak specifikuje následující příkazy a jejich významy:

EHLO *domain name CRLF*. Identifikace klienta, musí být použit před jakoukoliv další komunikací, *domain name* musí být plně specifikované doménové jméno, pokud existuje, nebo IP adresa klienta. Starší verze protokolu specifikovala příkaz HELO, syntaxe je stejná, některé starší servery stále ještě nemusí EHLO přijímat, stejně tak servery přijímající EHLO musí přijímat i HELO. Přijetím příkazu EHLO dává server na srozuměnou, že podporuje rozšíření, odpověď může být víceřádková, v tom případě jsou řádky po první ve formátu *250-keyword [SP args] CRLF*, v opačném pouze jedna řádka *250 domain [SP greetings] CRLF*. Keyword je klíčové slovo (bez rozlišení velikosti písmen) specifikující rozšíření s případnými argumenty.

- MAIL** *FROM: identifikace CRLF.* Uvozuje předání zprávy serveru, identifikace (většinou adresa odesílatele) je používána pro hlášení chyb v přenosu zprávy, v případě, že by hrozilo zacyklení (při posílání hlášení chyby), zadává se identifikace prázdná (<>), nemusí mít žádnou souvislost s hlavičkou *FROM* (viz 2.3).
- RCPT** *TO: identifikace CRLF.* Přidání cílové adresy ke zprávě, identifikace může být jak specifikace mailboxu, tak speciální adresy (Postmaster a pod). Server musí být ve stavu přijímání zprávy před přijetím tohoto příkazu, tedy mezi přijetím *MAIL FROM:* a *DATA*. Opět není definován žádný vztah ke hlavičkám *TO*, *CC* a *BCC* (více 2.3), tedy jediný způsob, jak skrýt adresy *BCC* je odeslat všem ostatním adresátům zprávu bez nich i bez uvedení ve zprávě a každému z adresátů *BCC* pak poslat zprávu a z *BCC* zkopírovat pouze jeho adresu.
- DATA** *CRLF.* Uvození samotné zprávy. Server by měl odpovědět indikátorem 354, klient pak předá zprávu ve standardním formátu, ukončenou *CRLF.CRLF* (13,10,46,13,10). Poslední 3 byty se do obsahu zprávy nepočítají, vyskytují-li se ve zprávě kdekoli v „“ na začátku řádku, přidá před ní klient „“, server toto opět odstraní stejným způsobem jako u protokolu pop3 [11] (viz 2.1).
- RSET** *CRLF.* Zruší předávání aktuální zprávy, server přejde do stavu jako po přijetí *EHLO*.
- QUIT** *CRLF.* Ukončuje sezení, server odpoví a rozpojí spojení.
- NOOP** *CRLF.* Resetuje čítač pro automatické odpojení.
- VERFY** *name CRLF.* Testuje platnost jména uživatele, jeho použití je dost nepraktické, protože není jednoznačně dáno, zda má server potvrzovat i uživatele z cizích serverů čistě ze syntaktického hlediska, či zamítat, může dávat na výběr při nejednoznačnosti, ale nedoporučuje se to z bezpečnostního hlediska, takže informace tímto příkazem získaná nemá výraznější vypovídací hodnotu.

Nepovinně může ještě smtp server podporovat příkaz *EXPN* pro kontrolu členství mail-listů.

Sezení probíhá následovně: Po navázání spojení pošle server klientu (klientem je ten, kdo spojení inicializoval) pozdrav, svojí identifikaci (např 220 ok, sendmail server ready) a čeká na identifikaci klienta. Ten se identifikuje pomocí příkazu *EHLO*, nebo případně staršího *HELO*. Při zdárném průběhu pak následuje pro každou zprávu, předávanou klientem serveru sekvence *MAIL FROM:* pro identifikaci odesílatele, *RCPT TO:* pro identifikaci každého adresáta a *DATA* pro předání samotné zprávy. Na konci je dán příkaz *QUIT* pro uzavření sezení.

Historicky se používaly explicitní cesty pro doručování e-mailových zpráv, například <@hosta.int,@jkl.org:userc@d.bar.org>, jejich používání se nedoporučuje, pokud smtp server takovou adresu přijme, měl by cestu odstranit a ignorovat. Při předávání zprávy na server by měl klient uvést v datové části i plnou hlavičku, ne však adresy neviditelných kopií (BCC). Není dán žádný vztah mezi hlavičkou a údaji danými v RCPT TO a MAIL FROM. Pro každou Bcc adresu by měl zprávu předat samostatně pouze s ní. Sntp může odmítnout zprávu s větším počtem příjemců, proto se nedoporučuje posílat zprávy s více jak sto příjemci.

Jsou dány následující limity pro délky dat s možností zvětšení prostřednictvím rozšíření:

- local part - maximálně 64 znaků
- domain - maximálně 255 znaků
- řádka příkazu - maximálně 512 znaků
- odpovědní řádka - maximálně 512 znaků
- textová řádka - maximálně 1000 znaků
- obsah zprávy - maximálně 65 KB (rozšíření pomocí extenze SIZE)

a časové limity pro odpovědi:

- úvodní uvítání - 5 minut
- příkaz MAIL - 5 min
- příkaz RCPT - 5 min
- uvození DATA - 2 min
- potvrzení dat - 10 min

2.3 Internet message

V dokumentu [14] je definován standardní formát pro textové zprávy přenášené prostřednictvím internetu, tzv. e-mailové zprávy. Zde popíšeme pouze hlavní rysy a globální formát zpráv, přesný syntaktický popis a gramatiky lze nalézt v [2, 14].

Standard neurčuje direktivně kódování znaků (sémantický význam) pro zobrazení, ani reprezentaci pro ukládání a přenášení (na rozdíl od smtp [8, 13] viz 2.2), zpráva by však měla sestávat pouze ze znaků hodnoty 1 - 127. Je definována jako sekvence znaků, dělená do řádek, konce řádek jsou reprezentovány sekvencí znaků CRLF (hodnoty us-ascii 13,10). Dále pak se dělí na hlavičku a tělo zprávy, přičemž hlavička zprávy je část po první sekvenci dvou CRLF jdoucích přímo za sebou.

Tělo zprávy obsahuje samotnou informaci, jejím prostřednictvím přenášenou, v těle zprávy může být v podstatě libovolný text, omezený pouze rozsahem hodnot 1 - 127 a délkou řádky, která by neměla být delší než 78 znaků a nesmí být delší než 998 znaků. CR a LF by se v textu nemělo vyskytovat jinde, než společně jako oddělovač řádek.

Hlavička zprávy obsahuje informace pro přenos zprávy a její doručení, tyto jsou reprezentovány jednotlivými položkami (např. FROM, TO, DATE a dalšími). Položka hlavičky sestává z jména a obsahu odděleného znakem „:“ (us-ascii 58). Základní formát je jednořádkový, tedy na začátku je jméno položky „:“ a obsah, s maximální délkou řádky 998 znaků a doporučenou délkou do 74 znaků, norma však povoluje rozdělit hlavičky delší než maximální délka řádky do více. Toto dělení může být provedeno pouze před tzv. bílým znakem (SP, TAB). Jednoduše řečeno, řádka hlavičky začínající jedním z těchto znaků se syntakticky považuje za pokračování řádky minulé, CRLF mezi nimi se ignoruje. Jednotlivé hlavičky mají svoji specifickou syntaxi, která někdy nepovoluje použití určitých znaků mimo předem definovaná místa, tyto se pak musí pro syntaktickou analýzu skrývat zapouzdřením. Zapouzdření jednoho znaku se provádí *quotováním* (to jest tak, že před znak se vloží „\“, US-ASCII [1] znak 92, znak následující je ignorován), víceznakové sekvence se pak uzavírají mezi “” (us-ascii znak 34).

Hlavička by pak měla obsahovat následující položky:

FROM adresa odesílatele zprávy, starší specifikace používá políčko SENDER. Zpráva musí obsahovat tuto položku právě jednou.

DATE datum vytvoření zprávy, mělo by být ve zprávě právě jednou.

MESSAGE-ID není povinné, jedná se o světově jedinečnou identifikaci zprávy.

Dále se doporučuje používat další položky:

REPLY-TO adresa, na kterou má být poslána případná odpověď na zprávu

REFERENCES odkaz na zprávu, ke které se vztahuje (její MESSAGE-ID)

IN-REPLY-TO odkaz na zprávu, na kterou je odpovědí.

SUBJECT věc, které se zpráva vztahuje, text pro čtenáře, v podstatě nadpis.

TO seznam přímých adresátů zprávy

CC seznam adresátů, kterým má být doručena kopie zprávy

BCC seznam adresátů, kterým má být doručena skrytá kopie (tzn. ostatní adresáti neví že jim byla kopie doručena, v jejich zprávě tato hlavička nebude).

2.4 Mime Message

Kolekce dokumentů [6, 5, 10, 3, 4] nazvaná MIME rozšiřuje formát internet message [2, 14], specifikuje metody pro vkládání jiných než textových dat do zpráv, použití jiných znakových sad než US-ASCII [1] jak v těle zprávy, tak v hlavičkách a jejich reprezentaci. Umožňuje také přenos zpráv sestávajících z více rozdílných částí, u takovýchto zpráv pak určuje způsob oddělení částí od sebe. Každá z těchto částí má vlastní hlavičku, specifikující její typ.

Pro přenos informací o formátu dodefinovává Mime následující hlavičky zprávy či její části (bez rozlišení velikosti písmen):

MIME-VERSION zpráva splňující standard z RFC 2045 by měla mít tuto položku s hodnotou 1.0 (tedy celý řádek vypadá *MIME-VERSION: 1.0*).

CONTENT-ID obdoba MESSAGE-ID z internet message

CONTENT-DESCRIPTION obsahuje popis dat zprávy

CONTENT-TYPE určuje typ dat obsažených v těle zprávy, podle typu pak obsahuje i atributy. Zjednodušeně je formát tohoto políčka *typ/subtyp *[:attribute=value]*, přičemž typ, subtyp a attribute je brána bez rozlišení velikosti písmen, value s rozlišením, pokud není k danému atributu specifikováno jinak. Pokud chybí, nebo není rozpoznán, považuje se jako kdyby byl přítomen *text/plain;charset=us-ascii*

CONTENT-TRANSFER-ENCODING určuje metodu transformace dat z běžné reprezentace do 7bitových dat. Pokud není přítomen, předpokládá se 7bit.

Metody transformace dat používané mime musí být deterministické a bezztrátové, výstup dekódování musí být identický vstupu kódování a algoritmus nesmí záviset na žádných externích informacích. Použitá transformační metoda neříká nic o typu dat, platí pro část, pro kterou je specifikována. Pokud je typ zprávy *multipart*, musí být kódování 7bit, 8bit nebo binary. Předpokládá se, že data jsou dělena na byte s řazením bitů Big endian a jakákoliv jiná data musí být na takováto převedena a doplněna na celé byte nulami, pro případné doplnění je specifikován parametr padding. Prostřednictvím IANA je možné registrovat další metody transformace. Při nerozpoznání Content-Transfer-Encoding se s částí musí zacházet, jako kdyby Content-type byl *application/octet-stream*.

Základní metody transformace jsou:

7bit data nebyla nijak transformována, jsou v původní podobě a tato využívá pouze dolních 7 bitů z byte.

8bit v podstatě to samé jako 7bit, pouze povoluje i použití osmého bitu.

- binary data nebyla nijak transformována, pouze byla případně doplněna nulami na celé byte, počet přidaných bitů specifikuje parametr hlavičky Content-type *padding*.
- Quoted-Printable Používá se u dat z větší části v US-ASCII [1], převádí pouze znaky mimo US-ASCII a formát řádek. Doporučuje se ho použít i u pouze US-ASCII dat pro zajištění správného průchodu přes *character-translating* nebo *line wrapping gateway*.
- Base64 Je vhodné pro kódování bytů, které nemusí být čitelné či netextových dat. Používá 65 znakovou podmnožinu US-ASCII, která je stejná ve všech národních verzích znakové sady ISO646 [1] i v EBCDIC [7]. Převádí vždy 3 byte (24bitů) na 4 výstupní znaky.

Postup kódování Base64 Spojíme 3 znaky jak jdou za sebou, tedy nejvyšší bit prvního znaku bude první a nejnižší bit třetího na konci, vznikne 24 bitový řetězec, který bereme jako skupiny po 6 bitech, které převedeny do desítkové soustavy dávají index znaku v převodní tabulce (1), který bude na výstupu. berou se v pořadí od nejvyššího k nejnižšímu. Tímto způsobem postupně převádíme celou zprávu, při kódování konce, když není počet bytů dělitelný 3 je postup následující: pokud jsou na konci 2 byty, spojí se, doplní na konci nulami a na výstup jdou místo 4 pouze 3 znaky a znak '=', pokud na konci přebývá 1 byte, doplní se na konci nulami, na výstup jdou 2 znaky a sekvence „=“.

Do výstupu jsou vloženy CRLF tak, aby se nevyskytovaly řádky delší než 76 znaků, dekodování je operací reverzní, znaky různé od base64 ignorovány (platí i pro přidané CRLF).

Hodnota	Znak	Hodnota	Znak	Hodnota	Znak	Hodnota	Znak
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	19	d	46	u	63	/
13	N	30	e	47	v	(pad)	=
14	O	31	f	48	w		
15	P	32	g	49	x		
16	Q	33	h	50	y		

Tabulka 1: Převodní tabulka Base64

Postup pro kódování quoted-printable

- (1) jakýkoliv oktet mimo CR a LF jako zakončení řádku se zakóduje jako =XY, kde XY je hodnota znaku v hexadecimálním tvaru s použitím velkých písmen (tedy 0123456789 ABCDEF), pokud nelze použít jiný bod.
- (2) byte s desítkovou hodnotou v rozmezích 33-60, 62-126 včetně je kódován jako US-ASCII [1] znak této hodnoty, doporučuje se ještě vynechat znaky „!\"#\$%&[]{} \^_|~“ a tyto kódovat podle (1) protože mohou být špatně přenášeny některými prostředími a „~“ protože je používána jako rozdělovací znak částí multipart message.
- (3) TAB a SPACE (9 a 32) se kódují jako tyto v US-ASCII, pokud jsou za nimi na řádce nějaké jiné znaky, tedy pokud řádka končí TAB nebo SPACE musí být tento kódován podle pravidla (1). Toto je potřeba kvůli některým serverům, které odstraňují prázdné řádky a konce řádek.
- (4) CRLF se nesmí kódovat podle pravidla (1), ale vložit jak je (hard line break)
- (5) pokud by se v kódovaných datech měly objevit řádky delší než 76 znaků, je vložena sekvence „=CRLF“ (soft line break), tyto jsou při dekódování odstraněny. CRLF se do délky řádky nepočítá.

Řešení chyb při dekódování quoted printable

- Pokud je v datech „=“ a za ním není hexadecimální hodnota, vloží se tak jak je a upozorní se na chybu.
- Pokud je v datech „=“ a za ním 16 hodnota v malých znacích, kóduje se, jako by byly velké.
- Pokud je v datech jiný kontrolní znak než TAB či CR, nebo LF jako CRLF, je vynechána a ohlásí se chyba.
- Pokud je v datech řádka delší než 76 znaků, nedekóduje se a ohlásí se chyba.

Základní typy zprávy [5] se dělí na typy „diskrétní“, jsou jimi

Text: Běžný text, zobrazitelný bez speciálního software. Může obsahovat formátovací příkazy, ne však takové, které nejsou smysluplně zobrazitelné (tedy např. html tag obsahovat může).

Základní subtyp je *plain*, ten nesmí obsahovat žádné formátovací znaky, kromě odělování řádků. V kanonické formě (tedy formě před převodem do transportní podoby) musí být oddělovače řádků CRLF, znaky CR a LF se nesmí objevovat mimo ně. Jsou-li v textu příliš dlouhé řádky, musí se tento textkódovat.

Pro tento typ by měl být specifikován parametr *charset*, udávající znakovou sadu textu, hodnota musí být registrována u IANA. Podle základní verze jsou registrovány všechny národní verze ISO-8859-X [6], tedy i US-ASCII.

Image: Tělo zprávy obsahuje obrázek, subtyp indikuje formát, rozlišuje se velikost písmen subtypu. Základní formát je jpeg [5] - obrázek standardu jfif encoding. Nerozpoznané podtypy se berou jako *application/octet-stream*. Klient se může rozhodnout je předat k zobrazení robustním aplikacím na prohlížení obrázků.

Audio: Tělo obsahuje zvukový záznam, subtyp určuje formát, základní je basic, je to jednokanálové audio kódované 8bit ISDN mu-law PCM (pulse code modulation) [5], sample rate 8000Hz. Nerozpoznané podtypy se opět berou jako *application/octet-stream* a klient je může předat robustním aplikacím pro přehrávání audia.

Video: Tělo obsahuje video, subtyp určuje formát. Nerozpoznané podtypy se opět berou jako *application/octet-stream* a klient je může předat robustním aplikacím pro přehrávání videa.

Application: Typ je určen pro diskrétní data, která nespádají do žádné jiné a kategorie a data, která musí být zpracována nějakým typem aplikace. Využití se také očekává pro tabulky a data pro agendy komunikující emailem, data pro distribuované výpočty.

U tohoto typu musí být brán zvláštní zřetel na bezpečnost, tedy e-mailový klient by neměl spouštět automaticky přiřazenou aplikaci, může však generovat události a data na základě dialogu s uživatelem.

Definovány jsou všeobecný subtype *octet-stream* a subtype *PostScript* [5]. Subtype *octet-stream* je určen pro binární či nerozpoznané data, doporučuje se používat parametr *type*, což je informace o typu pro lidskou interpretaci, spíše než automatizované zpracování, u binárních dat pak parametr *padding*, informaci, kolik bitů bylo přidáno pro zarovnání na byte. U podtypu *PostScript* jsou povoleny *level 1* a *level 2 PostScript language*, a vzhledem k tomu, že tyto mohou obsahovat nebezpečné operace, je třeba brát zvláštní zřetel na bezpečnost.

a typy „složené“, což jsou

Multipart: Typ pro přenášení zpráv složených z více rozdílných částí, tyto jsou v těle odděleny hraniční dělicí řádkou a každá část by měla mít vlastní mime hlavičku.

Dělicí element je řádka sestávající z dvou znaků „“ (us-ascii 45) a dělicího klíče, předaného v parametru *boundary*, který je povinný. Klíč může obsahovat pouze 7bit us-ascii znaky. Tento řádek je před první částí, mezi všemi částmi a za poslední částí je s připojením dalších dvou znaků „“ na konci. Vše, co je před první a za poslední částí se ignoruje (většinou se před první část dává zpráva pro klienty nepodporující mime). Tento klíč s dvěma „“ na začátku se nesmí vyskytovat jako předpona žádného řádku v částech, které odděluje. S částmi zprávy se nakládá jako se zprávou mime podle jejích hlaviček, vícenásobné zapouzdření (část multipart message typu multipart) je povoleno. V případě chybného zkrácení zprávy je třeba rozpoznat na libovolné úrovni i dělicí řádky vnějších úrovní.

Pro tento typ musí být *Content-transfer-encoding* 7bit, 8bit nebo binary a *charset* us-ascii. Kódování, pokud je potřeba, se provádí na nejnižší úrovni zanoření. Hlavičky částí jsou od těla odděleny stejně jako v případě zprávy samotné, význam mají pouze položky *Content*-, ostatní se ignorují.

Základní subtypy jsou *mixed*, obsahuje části bez definování vzájemného vztahu, *alternative*, kde všechny části mají stejný obsah v různých formátech, seřazených podle úrovně zachování informací od nejhoršímu k nejlepšímu (například text-plain, enriched text, Microsoft Word doc). Subtype *digest* je totožný s *mixed* s tím rozdílem že implicitní typ je *message/rfc822*. Subtype *parallel* se používá pro zprávy, kde všechny části by měly být zobrazeny současně. Nerozpoznané subtypy se berou jako subtype *mixed*.

Message: Tělo zprávy či části takto označené obsahuje zapouzdřenou zprávu.

Subtype *RFC822* se používá pro zapouzdření e-mailové zprávy, u této nemusí být uveden adresát, musí však obsahovat minimálně jeden z *from*, *subject*, či

date. Nemusí být striktně podle RFC822 [2, 14], může být i MIME či jiná. Kódování musí být *7bit, 8bit nebo binary*.

Subtyp *partial* se používá k posílání objemově velkých zpráv, velká zpráva je pro přenos rozdělena do více menších částí, každá z nich je pak zapouzdřena do zprávy typu *multipart/partial*. Content-transfer-encoding musí být 7bit, díly samotné mohou být dále rozděleny. Pro tento subtyp jsou povinné tři parametry, a to *id* - světově jedinečný identifikátor rozdělené zprávy, *number* - pořadové číslo části a *total* - počet fragmentů, který musí být jen v posledním fragmentu. Parametry mohou být v libovolném pořadí. Při fragmentaci je povoleno zprávu dělit pouze na rozhraní řádek, je vhodné generovat položku hlavičky references odkazující na předchozí část. Při skládání musí být spojeny hlavičky první vnitřní zprávy s hlavičkami zprávy obalové, z vnitřní části se berou pouze položky *Content-, subject, message-id, encrypted a mime-version*.

Subtyp *external-body* je typ, kde data nejsou v těle přítomna, v těle je pouze odkaz na ně a parametry specifikují mechanismus přístupu. Parametr *access-type* určuje přístupový mechanismus, základní možnosti jsou FTP, ANON-FTP, TFTP, LOCAL-FILE, MAIL-SERVER [5]. Další parametry jsou např. *expiration, size, permissions*, pro FTP pak *name, site, directory, mode*, pro MAIL-SERVER pak *server a subject*.

Vkládání znaků mimo US-ASCII do hlaviček [10] Podle mime lze na určitá místa do hlaviček vkládat speciálním způsobem zakódovaná data, která se skládají ze znaků mimo US-ASCII [1]. Kromě použití národních abeced ve hlavičkách se touto metodou doporučuje kódovat znaky, o kterých se ví, že dělají problémy některým e-mailovým serverům, vyskytujícím se na internetu.

Formát kódového slova je „=?charset?encoding?encoded_text?“ kde *charset* je jednoslovné označení znakové sady textu, *encoding* je *q* pro quoted-printable a *b* pro base64. *Encoded_text* je pak zakódovaný text, metoda převodu je identická jako u těl zpráv. Kódové slovo by nemělo být delší než 75 znaků včetně ohraničení, mělo by být od zbytku hlavičky odděleno mezerou (SP nebo TAB), pokud je potřeba kódovat delší část, rozdělí se tato na dvě a kódová slova se oddělí CRLF SP (viz folding 2.3).

Pravidla pro vkládání a rozpoznávání encoded-word jsou:

- může nahradit v políčkách subject nebo comment *text*, nebo body part **text*.
- v komentářích mezi '(' a ')' v *ctext*
- *word* v *phrase* např. ve from, to cc...
- nesmí se objevit v *addr-spec, quoted string*, hlavičce received, jako parametr content-type, nebo content-disposition, nebo strukturovaném body mimo *comment a phrase*

- encoded text nesmí být rozdělen do dvou částí, tedy část base64 bude 4*x char, a v q budou za „=“ dvě čísla. V každém bude zakódován celý počet oktetů / znaků. Ani více oktetový znak (například u 16bit unicode) nesmí být rozdělen do dvou encoded word

Program vytvářející zprávu mime musí zaručit, že se ve hlavičce neobjeví text ve formátu kódového slova, které jím není. Příjemce musí zaručit, aby jiný než běžný materiál tímto způsobem zakódovaný a vložený do hlavičky při zobrazení neměl nežádané vedlejší efekty. Tento způsob není definován pro vkládání netextového obsahu do hlaviček, nedoporučuje se také kódovat řetězce čistě s us-ascii znaků.

U *quoted-printable* transformace se pravidla pro převádění liší od standardního v následujícím:

- v encoded_word není vždy dovolen výskyt SP, TAB a problémy mohou činit řídicí znaky - tyto by měly být taky kódovány
- lze převádět znak SP na „_“ (znak 95).

Minimální kritéria pro splnění standardů mime [4] V posledním dokumentu standardu mime jsou uvedeny minimální požadavky na programy MIME implementující.

Pro klienta to znamená, že musí

- vždy generovat hlavičku Mime-version 1.0,
- rozpoznat hlavičku content-transfer-encoding a být schopen dekodovat přijatá data pomocí metod quoted-printable a base64, musí také rozpoznávat 7bit, 8bit a binary
- při posílání musí data obsahující znaky mimo 7bit rozsah řádně označit content-transfer-encoding 8bit, pokud přenosové médium tyto nepodporuje, musí data řádně zakódovat
- při nerozpoznání content-transfer-encoding musí brát obsah jako application/octet-stream s přihlédnutím k content-type
- rozpoznat a interpretovat hlavičku content-type, nezobrazovat neinterpretovaná data jiných než textových částí
- musí být schopen odeslat alespoň čistý text v us-ascii, nebo specifikovat jeho charset
- ignorovat content-type parametry, které nerozpozná
- nerozpoznané content-type brát jako application/octet-stream bez parametrů a sub-parametrů, nabídnout uložení do souboru

- pokud podporuje nestandardní ne-mime zprávy v jiném než us-ascii, musí tak činit jen na vstupu, neposílat takové zprávy
- musí správně označit parametrem *charset* data, pokud jsou v jiné znakové sadě než us-ascii
- musí zajistit, aby se ve hlavičce vyskytovala v formě začínající =? a končící ?= jen správná kódová slova
- musí být schopen rozpoznat encoded word ve hlavičce. musí podporovat b i q *encoding* pro všechny znakové sady, které podporuje. Musí být schopen zobrazit nekódovaný obsah pro alespoň us-ascii, a pro ISO-8859-* jejich znaky s us-ascii shodné

Explicitně pracovat s následujícími media typy:

- text - rozpoznat a zobrazit text v us-ascii, rozpoznat ostatní znakové sady a minimálně informovat o typu. Rozpoznat ISO-8859-* a zobrazit všechny znaky shodné s us-ascii, text s neznámou znakovou sadou brát jako application/octet-stream
- Application - minimálně musí umět dekodovat a uložit do souboru
- Multipart - rozpoznat mixed type, zobrazit všechny relevantní informace na úrovni zprávy i částí a zobrazit, či nabídnout k zobrazení každou část zvlášť. Rozpoznat subtype alternative a vyhnout se zobrazení nadbytečných částí. Rozpoznat subtype digest a zobrazit obsah jako message/rfc822. Nerozpoznané typy brát jako mixed.
- Message - rozpoznat a zobrazit minimálně RFC822 message, zachovávat strukturu zapouzdření a zobrazit data podle jejich typu. Nerozpoznané subtypy brát jako application/octet-stream

Pokud klient toto splňuje, může být nazván mime-conformant (splňující MIME).

Návod pro posílání mail-data Existuje mnoho serverů, přes které zpráva může projít na cestě od odesílatele k příjemci, které neodpovídají mime. proto je třeba brát ohled na následující:

- base64 odpovídá všem požadavkům, quoted-printable nemusí projít přes některé servery pracující se znakovou sadou EBCDIC [7].
- někdy se při přechodu musí změnit encoding a při tom i sekvence CRLF
- některé systémy mohou převést CRLF na svoje lokální konvence. Izolované CR a LF nejsou všeobecně tolerované
- NULL je při transportu problematický znak
- tab může být někdy převeden na sekvenci znaků SP

- řádky delší než 76 znaků mohou být rozděleny
- prázdné znaky na konci řádky mohou být někdy odstraněny
- pouze u alfabetských znaků, číslic a „(),+’-./:=?“ je zaručen bezproblémový přenos
- některé systémy chybně interpretují např. „“ samostatně se vyskytující na řádce a řádky začínající „From“ (doporučuje se kódovat *quoted-printable* a místo těchto používat =2E a =46rom)

Základní model pro postup kódování

- nejdřív se vytvoří lokální formát podle lokálních zvyklostí
- pak se převedou všechny informace (i atributy souborů, velikost a podobné) do univerzální základní podoby. zde se např. převádí znaková sada, převádí se audio, komprimuje...
- poté se zakóduje pro přenos (base64, quoted-printable)
- dále se vloží do mime entity společně s adekvátními hlavičkami v těle a hlavičkami zprávy
- dekódování se provede obrácením pořadí těchto kroků, výsledek nemusí být identický díky rozdílným lokálním konvencím (např. konce řádků)
- kroky lze libovolně prohazovat, ale výsledek musí být identický výsledku bez prohození.

• Reference

- [1] Coded character set – 7bit american standart code for informational interchange, 1986.
- [2] D. Crocker. Standart for the format of arpa internet text messages, 1982. <http://www.faqs.org/rfcs/rfc822.html>.
- [3] Klensin J. Freed, N. and J. Postel. Multipurpose internet mail extensions (mime) part four: Mime registration procedures, 1996. <http://www.faqs.org/rfcs/rfc2048.html>.
- [4] N. Freed and N Borenstein. Multipurpose internet mail extensions (mime) part five: Conformance criteria and examples, 1996. <http://www.faqs.org/rfcs/rfc2049.html>.
- [5] N. Freed and N Borenstein. Multipurpose internet mail extensions (mime) part two: Media types, 1996. <http://www.faqs.org/rfcs/rfc2046.html>.
- [6] Ned Freed. Mime part one: Format of internet message bodies, 1996. <http://www.faqs.org/rfcs/rfc2045.html>.

- [7] IBM. Extended binary coded decimal interchange code.
- [8] J. Klensin. Simple mail transfer protocol, 2001. <http://www.faqs.org/rfcs/rfc2821.html>.
- [9] J. Klensin, N. Freed, M. Rose, E. Stefferud, and D Crocker. Smtп service extensions, 1995. <http://www.faqs.org/rfcs/rfc1869.html>.
- [10] K. Moore. Multipurpose internet mail extensions (mime) part three: Representation of non-ascii text in internet message header, 1996. <http://www.faqs.org/rfcs/rfc2047.html>.
- [11] J. Myers, Carnegie Mellon, and M. rose. Post office protocol - version 3, 1996. <http://www.faqs.org/rfcs/rfc1939.html>.
- [12] Informational Sciences Institute University of Southern California. Transmission control protocol, 1981. <http://www.faqs.org/rfcs/rfc793.html>.
- [13] J.B Postel. Simple mail transfer protocol, 1982. <http://www.faqs.org/rfcs/rfc821.html>.
- [14] P. Resnick. Internet message format, 2001. <http://www.faqs.org/rfcs/rfc2822.html>.