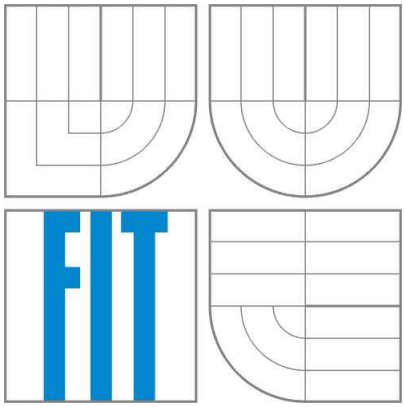
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



Dokumentácia k projektu do predmetu ISA

Detekcia maximálneho MTU po ceste

22. apríla 2016 Pavol Loffay

Obsah

1 Úvod

Tento dokument postupne popisuje teóriu, návrh riešenia a implementáciu zisťovania maximálnej prenosovej jednotky, ďalej *PMTU*[1] z jedného hostiteľa na druhého. Navrhnutý program pracuje ako konzolová aplikácia pre systém Linux.

Dokument sa skladá z viacerých časti. V kapitole 2 sa venuje analýze problému a popisuje techniky, ako sa dá zistiť maximálne *PMTU*.Kapitola 3 sa zoberá návrhom algoritmov. Ďalej nasleduje kapitola 4, kde popisujeme ovládanie programu a vlastnú implementáciu.

2 Analýza problému a princíp jeho riešenia

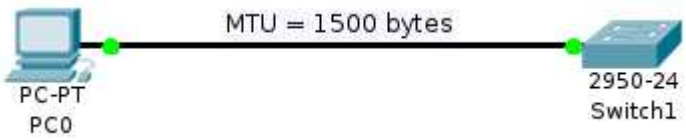
2.1 Popis zadania

Úlohou tohto projektu bolo implementovať zistenie maximálnej prenosovej jednotky na ceste – *PMTU*. V prípade použitia *IPv4* sme museli použiť bit zákaz fragmentácie, skrátene – *DF*[2]. Pre implementáciu *IPv6* neboli bližšie špecifikácie. Pre oba protokoly bolo určené použitie protokolov *ICMP*[3] pre *IPv4* a analogicky *ICMPv6*[4] pre *IPv6*.

Program mal pracovať ako konzolová aplikácia, pre operačne systémy založené na Unixu. Bolo špecifikovane že máme použiť BSD schránky a knižnice z `netinet/*` a pre preklad doménového mena `resolv.h` a `netdb.h`.

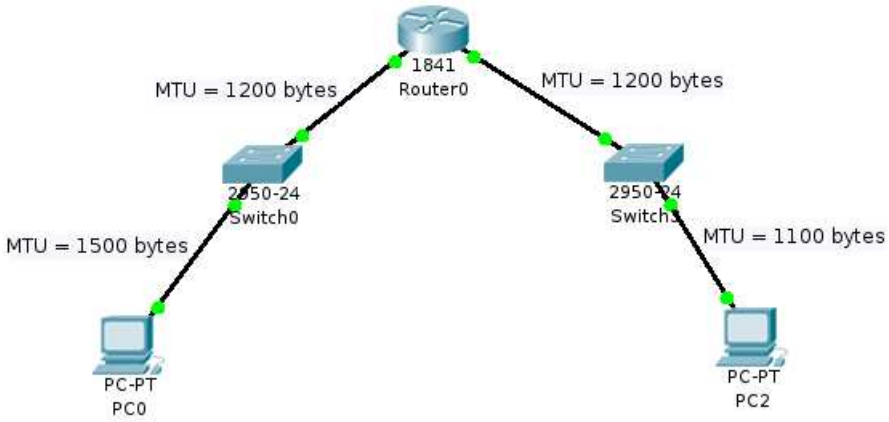
2.2 MTU

Zisťovanie maximálnej prenosovej jednotky je veľmi dôležite. Pretože, ak by zdroj vysielal pakety ktoré sú väčšie ako *PMTU*, bude ich musieť niektorý zo sieťových uzlov fragmentovať. Tým pádom spracovanie jedného paketu bude náročnejšie a bude trvať dlhšiu dobu. Naproti tomu ak by zdroj vysielal príliš male pakety, taktiež by dochádzalo k plytvaniu sieťových prostriedkov. Ešte musíme zmieniť, že pri fragmentácii, alebo vysielaní príliš malých paketov je väčšia pravdepodobnosť, že sa pakety stratia pretože je ich väčšie množstvo. Preto je dôležité vhodne zvoliť čím najväčšiu prenosovú jednotku.



Obr. 1: *MTU* – maximálna veľkosť prenosovej jednotky na jednom fragmente siete.

[1] Path maximum transmission unit
[2] Don't fragment
[3] Internet Control Message Protocol
[4] Internet Control Message Protocol Version 6



Obr. 2: *PMTU* – maximálna veľkosť prenosovej jednotky na celej ceste.

2.2.1 Ujasnenie pojmov MTU a PMTU

MTU[5] charakterizuje maximálnu veľkosť prenosovej jednotky medzi dvoma zariadeniami, napríklad medzi počítacom a rozbočovačom. *PMTU* – maximálna veľkosť prenosovej jednotky medzi zdrojom a cieľom po celej ceste. Čiže paket prejde viacerými segmentmi siete bez fragmentácie. *PMTU* je znázornený na obrázku 2 medzi počítačmi *PC0* a *PC2*, jeho veľkosť je *1100 bajtov*.

2.3 PMTU pre IPv4

Protokol *IPv4* je postavený tak, že smerovače ak je veľkosť paketu väčšia ako *MTU* na odchádzajúcej linke, paket fragmentujú. Tým pádom musí byť tomu prispôsobenia aj hlavička *IPv4*, kde je potom určené či je paket fragmentový a následne poradie fragmentov.

Pre zistenie maximálneho *MTU* pre *IPv4* je nutné použiť protokol *ICMP* a korektne nastaviť hlavičku *IPv4*, konkrétne bit zákazu fragmentácie – *DF* bit viz [7]. Protokol *IPv4* používa na zisťovanie stavu zariadení v sieti protokol *ICMP*. Ten je veľmi dôležitý a typicky sa používa napríklad v aplikáciach `ping` a `traceroute`.

Naša aplikácia bude posielať *ICMP* správy *Echo Request* s rôznou veľkostnou, pričom v hlavičke IP paketu, musí byť nastavený bit zákaz fragmentácie. Následne bude program čakať na prijatie *ICMP* správy. Ak príde *Echo Reply* znamená to, že odoslaný paket bol korektne prijatý vzdialenou stanicou a môžeme poslať väčší paket. Ak prijme správu *ICMP Destination Unreachable* s kódom *4*[6], znamená to že náš odoslaný paket sa nedostal k cieľovému zariadeniu pretože bola jeho velkosť väčšia ako niektoré *MTU*, takže musíme zmenšiť objem posielaných dat. Týmto spôsobom veľmi jednoducho zistíme *PMTU*.

2.3.1 Protokol ICMPv4

Tento protokol patri medzi najželezitejšie protokoly internetu. Využívajú ho operačne systémy sieťových uzlov na posielanie rôznych informačných a chybových správ viz [8]. Napríklad o vypršaní *TTL* alebo nedostupnosti služby. V *OSI* modele ho môžme zaradiť k Sieťovej vrstve. *ICMP* správa je priamo obsiahnutá v IP datagrame. V tabuľke 1 môžeme vidieť, vybrané správy tohto protokolu.

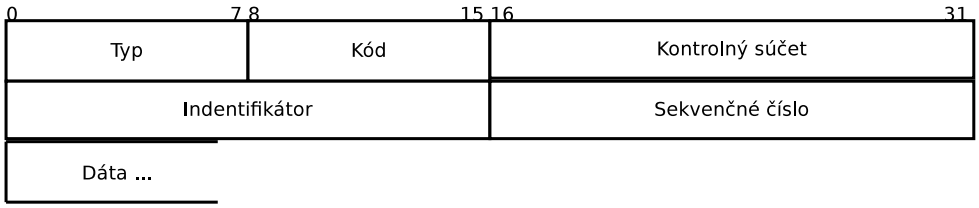
[5] Maximum transmission unit

[6] fragmentation need and DF set

Nazov	**Cislo typu**
Echo Reply	0
Destination Unreachable	3
Redirect	5
Echo Request	8
Time Exceed	11
Parameter Problem	12

Tabuľka 1: Tabuľka vybranych *ICMP* sprav.

Bližšie sa zameriame na popis *ICMP* správy typu *Echo Reply* a *Echo Request*. Tieto sa používajú na zistenie či je cieľový počítač dosiahnuteľný. Tento proces funguje následovne: zdrojový počítač pošle správu *Echo Request*, ak túto správu obdrží zariadenie s cieľovou adresou, tak odošle *Echo Reply*. Tým pádom pôvodne zdrojové zariadenie zistí, že cieľová stanica je dostupná.



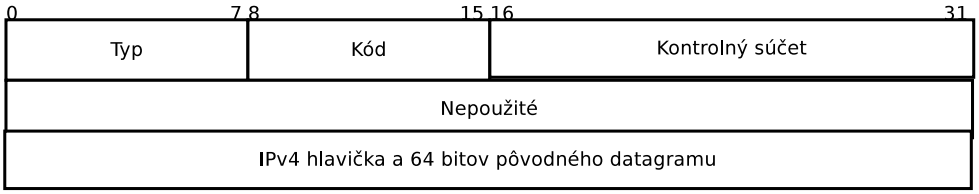
Obr. 3: Struktura *ICMP* správy *Echo Request* a *Echo Reply*

Štruktúra správ Echo Request a Echo Reply

- Typ: *Echo Request 8* a *0* pre *Echo Reply*.
- Kód: hodnota 0.
- Kontrolný súčet: *16* bitový jednotkový doplnok súčtu jednotkového doplnku celej *ICMP* správy.
- Identifikátor: Nastavuje ho odosielateľ, prijímajúci uzol vytvorí odpoveď s identifikátorom odpovedajúcim k prijatej správe. Podľa toho prijímateľ rozlíši odpovede k rôznym odoslaným správam.
- Sekvenčne číslo: každá vytvorená správa s rovnakým identifikátorom musí mať iné sekvenčné číslo. Uzol ktorý odoslal *Echo Request*, spozná odpoveď ak sa bude zhodovať identifikátor a sekvenčné číslo správy.

Štruktúra Destination Unreachable

- Túto správu s kódom *4* používajú smerovače na oznamovanie že *MTU* odchádzajúcej linky je menšie ako veľkosť paketu, avšak iba tých ktoré majú nastavený *DF* bit.
- Typ: nastavená hodnota na *3*
- Kód: obsahuje hodnotu *4*.



Obr. 4: Struktura *ICMP* správy *Destination Unreachable*

- Posledná položka: obsahuje pôvodne odoslanú *IPv4* hlavičku a *64* bitov pôvodného datagramu, čiže nami odoslanú *ICMP* správu. Podla jej položiek identifikátor a sekvenčne číslo zistíme či prijatá správa prislúcha k nami odoslanej.

2.4 PMTU pre IPv6

V prípade použitia *IPv6* musia koncové stanice zaistiť správnu veľkosť odosielaného paketu pretože na ceste nedochádza k jeho fragmentovaniu. Aby sme mohli zistiť *PMTU* pre *IPv6* stačí použiť protokol *ICMPv6*. Na rozdiel od *IPv4* nemusíme upravovať hlavičku *IPv6* paketu, aby nedochádzalo k fragmentácii.

Na zistenie *PMTU* pre *IPv6*, budeme postupovať analogickým spôsobom ako pri *IPv4*. Aplikácia bude posielať *ICMPv6 Echo Request* správy s rôznou veľkosťou. Ak prijmeme *Echo Reply* prislúchajúcu k nami odoslanému *Echo Request* je zrejmé, že môžme zväčšiť veľkosť odosielaného paketu, pretože *PMTU* je menšie ako náš odoslaný paket. Ak niektoré *MTU* po ceste je menšie ako odoslaná správa, sieťový uzol vygeneruje *ICMPv6* správu *Packet Too Big* v ktorej je veľkosť *MTU* linky kde mal byť paket poslaný ďalej viz [6].

2.4.1 Protokol ICMPv6

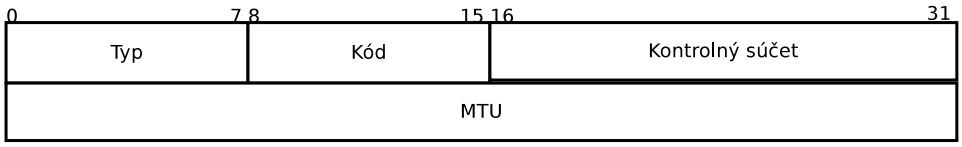
Jeho účel a použitie je zhodný s protokolom *ICMPv4* avšak sa používa pre *IPv6*. Principálne obsahuje tie iste správy, ale kvôli miernym odlišnostiam *IPv4* od *IPv6* definuje nové správy, aby bola zaručená funkčnosť *IPv6* viz [2].

Tabuľka 2 ukazuje vybrané typy správ, môžme si všimnúť že rovnaké správy pre *ICMPv4* a verziu *6* majú iné hodnoty typu.

Štruktúra správ *ICMPv6* je pomerne zhodná s *ICMPv4*. Napríklad *Echo Requst* a *Echo Reply* majú zhodnú štruktúru.

Nazov	**Číslo typu**
Echo Reply	129
Destination Unreachable	1
Redirect	137
Echo Request	128
Time Exceed	3
Parameter Problem	4
Packet Too Big	2

Tabuľka 2: Tabuľka vybraných *ICMPv6* správ.



Obr. 5: Štruktúra *ICMPv6* správy *Packet Too Big*

Popis správy Packet Too Big

- Tato správa je generovaná smerovačom na odpoveď paketu, ktorý nemôže byť poslaný pretože je jeho veľkosť vačšia ako *MTU* na odchádzajúcej linke.
- MTU: veľkosť *MTU* na odchádzajúcej linke v bajtoch.

2.5 Problémy pri PMTU

Zisťovanie maximálnej prenosovej jednotky po celej ceste môže byť obtiažna úloha, pretože nie všetky sieťové zariadenia vysielajú korektné *ICMP* správy, alebo ich nevysielajú vôbec. Môže to byť spôsobené zlou konfiguráciou, alebo chybou operačného systému smerovača. Bežnou chybou môže byť, že smerovače neposielajú *Destination Unreachable* s kódom *4*, čo znamená že na odchádzajúcom rozhraní je menšie *MTU* ako veľkosť paketu. Naša aplikácia by sa stým mala vyrovnať, tak že ak nepríde odpoveď do určitého času, zníži veľkosť paketu. Niekedy taktiež môže dochádzať k filtrovaniu paketov pomocou firewallu viz [5].

2.6 Binárne vyhľadávanie

Pre voľbu veľkosti posielaného *ICMP Echo Request* paketu sme zvolili binárne vyhľadávanie, alebo inakšie nazývanú metódu rozpoľovania intervalu. Je to vyhľadávací algoritmus na nájdenie zadanej hodnoty v usporiadanom zozname viz [4]. V každom kroku skráti zoznam o polovicu. Na základe výsledku porovnania sa rozhodne v hornej alebo dolnej časti zoznamu a rekurzívne pokračuje od začiatku.

3 Návrh riešenia problému

Po analýze problému, kde sme vysvetlili základne princípy a teóriu ohľadne zisťovania maximálnej prenosovej jednotky, nasleduje abstraktný popis algoritmov. Taktiež tu ukážeme voľbu hodnôt veľkosti paketu.

3.1 Minimálna veľkosť paketu pre IPv4

Každý internetový prvok musí byť schopný poslať datagram o veľkosti *68* bajtov bez ďalšej fragmentácie viz [1].

3.2 Popis algoritmu pre IPv4

Predpokladajme tri premenné *minimálna*, *maximálna* a *aktuálna* veľkosť paketu.

1. Nastav *minimálnu* veľkosť *MTU* na *68* bajtov.
2. Nastav *maximálnu* hodnotu na *1500* bajtov, alebo podľa zadaného parametru `-m <mtu>`.
3. Nastav *aktuálnu* dĺžku paketu `(minimálna veľkosť + maximálna veľkosť) / 2`.
4. Vytvor *IPv4* hlavičku s nastaveným *DF* bitom. K hlavičke pripoj *ICMP Echo Reguest* správu. Celková veľkosť musí byť rovná *aktuálnej* veľkosti.
5. Pošli vytvorený paket a zapni časovač.
6. Ak sa nepodarilo poslanie nastav *maximálnu* veľkosť na *aktuálnu* a choď na bod 3.
7. Prijímaj pakety pokiaľ nepríde *Echo Reply*, alebo *Destination Unreachable* s kódom *4*. Oba odpovedajúce na náš paket.
8. Ak prišiel *Echo Reply* nastav *minimálnu* veľkosť na *aktuálnu* veľkosť a choď na bod 3.
9. Ak prišiel *Destination Unreachable* s kódom *4*, nastav *maximálnu* veľkosť na *aktuálnu* a choď na bod 3.
10. Ak vypršal časovač a neprišiel *ICMP* paket, tak nastav *aktuálnu* veľkosť na *maximálnu* veľkosť a choď na bod 3.
11. Opakuj body 3 až 10 pokiaľ nie je *maximálna* veľkosť zhodná s *minimálnou*.
12. *Aktuálna* veľkosť paketu udáva veľkosť *PMTU*.

3.3 Minimálna veľkosť paketu pre IPv6

Podla [3] musí každý sieťový prvok s *IPv6* protokolom pripojený na linku s *MTU 1280* bajtov. Avšak pre experimentálne účely sme zvolili minimálnu veľkosť na *150* bajtov.

3.4 Popis algoritmu pre IPv6

Predpokladajme premenné *minimálna*, *maximálna* a *aktuálnu* veľkosť, ďalej premennú indikujúcu, že bola prijatá *ICMP Packet Too Big* správa s názvom *too_big*.

1. Nastav *minimálnu* veľkosť *MTU* na *150* bajtov.
2. Nastav maximálnu hodnotu na *1500*, alebo podľa zadaného parametru `-m <mtu>`.
3. Nastav *aktuálnu* veľkosť paketu na `(minimálna veľkosť + maximálna veľkosť) / 2`.
4. Vytvor *Echo Reguest* s a*aktuálnou* veľkosťou.
5. Pošli vytvorený paket a zapni časovač.
6. Ak sa nepodarilo poslanie nastav *maximálnu* veľkosť na *aktuálnu* a chod na bod 3.
7. Prijímaj pakety pokiaľ nepríde *Echo Reply* alebo *Packet Too Big*. Oba odpovedajúce na náš paket.

8. Ak prišiel *Echo Reply* a je nastavená premenná *too_big* tak choď na bod 12, inak nastav *minimálnu* veľkosť na *aktuálnu* a pokračuj bodom 3.

9. Ak prišiel *Packet Too Big*, nastav *aktuálnu* veľkosť na položku *MTU* z prijatej správy, ale odpočítaj z nej *40* bajtov. *Maximálnu* veľkosť nastav na *aktuálnu*. Ďalej nastav premennú *too_big*. Choď na bod 4.

10. Ak vypršal časovač a nepríšiel *ICMPv6* paket, tak nastav *aktuálnu* veľkosť na *maximálnu* veľkosť a choď na bod 3.

11. Opakuj body 3 až 10 pokiaľ nie je *maximálna* veľkosť zhodná s *minimálnou*.

12. Ak je nastavená premenná *too_big* tak k *aktuálnej* veľkosti pripočítaj *40*. *Aktuálna* veľkosť udáva *PMTU* v bajtoch.

3.5 Preklad doménového mena

Aplikácia z príkazového riadka preberá adresu, alebo doménové meno. Pri použití adresy nie je problém. Rozozná sa verzia IP protokolu a tá sa následne použije. Pri zadaní doménového mena aplikácia urobí preklad doménového mena adresy, pričom ak po preklade získame viacej adries, tak pre každú sa vypočíta *PMTU* a vyberie sa najväčšia hodnota.

4 Popis riešenia

Pri implementácii som vychádzal z urobenej analýzy problému a návrhu riešenia. Aplikácia je implementovaná v jazyku *C++*.

4.1 Ovládanie programu

Program funguje ako konzolová aplikácia. Pri spustení bez parametrov, alebo s chybnými vypíše nápovedu s použitím a ukonči sa s chybovým návratový kódom. Obrázok 6, ukazuje ovládanie programu.

Povinný parameter je IP adresa alebo doménové meno. Aplikácia príma jeden voliteľný parameter `-m`, ktorý špecifikuje hornú hranicu testovaného *PMTU* v bajtoch.

```
./mypmtud [-m max] adresa
```

Obr. 6: Ukážka ovládania programu.

Ak program prebehne v poriadku, vypíše na štandardný výstup maximálne *PMTU* v bajtoch a vráti normálny návratový kód[7]. Obrázok 7 ukazuje, výstup programu.

Ak dôjde k chybe počas programu, tak sa ukonči s chybovým návratovým kódom[8] a na štandardný chybový výstup vypíše popis chyby.

[7] V jazyky C konštantu `EXIT_SUCCESS`
[8] V jazyku C konštanta `EXIT_FAILURE`

```
resume: 1400 bytes
```

Obr. 7: Ukážka výstupu programu.

4.2 Popis implementácie

O spracovanie parametrov sa stará trieda `params`, jej objekt je definovaný na začiatku funkcie `main`. Potom sa zavolá metóda `parse`, ktorá spracuje parametre a korektne náplni položky objektu.

Následne sa volá funkcia `max_mtu` do ktorej sa predajú získane parametre. V tejto funkcii prebehne preklad doménového mena na adresu, ak bola zadaná IP adresa, prekladá sa tiež. Preklad prebieha pomocou volania funkcie `getaddrinfo`, ktorá vráti zoznam nájdených adries. Pre každú adresu sa zistí *PMTU*. Podľa typu adresy sa vytvorí príslušný objekt. Pre *IPv4* je to `mtu_ipv4` a pre *IPv6* `mtu_ipv6`, kde následne prebieha zistenie samotného *PMTU* v metóde `calculate` pre obe triedy. Funkcia `max_mtu` nakoniec vráti najväčšie *PMTU*, ktoré bolo nájdene.

Ak sa preklad pomocou `getaddrinfo` nepodaril a bola zadaná IP adresa, tak sa použije bez prekladu.

5 Záver

Program sa dá úspešne použiť na zistenie *PMTU* pre protokoly *IPv4* a *IPv6*. Poukázal by som na fakt, že aplikácia zisti z doménového mena všetky jeho adresy a pre všetky sa zistí *PMTU* a následne program vypíše to najväčšie.

Program bol úspešne testovaný na systémoch *GNU/Linux*. Taktiež aplikácia striktne dodržuje formát vystupujúcich dát, čiže môže byť použitá v iných programoch alebo skriptoch.

Ako rozšírenie programu by mohli byť implementované prepínače, ktorými by sa určilo či sa ma po preklade doménového mena použiť *IPv4* alebo *IPv6* protokol.

Literatúra

[1] *RFC791: Internet Protocol Darpa Internet Program Protocol Specification* [online]. Září 1981 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc791>.

[2] CONTA, A., DEERING, S. a GUPTA, M. *RFC4443: Internet Control Message Protocol (ICMPv6)* [online]. Březen 2006 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc4443>.

[3] DEERING, S. a HINDEN, K. *RFC2460: Internet Protocol Version 6 (IPv6) Specification* [online]. Prosinec 1981 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc2460>.

[4] FAJMON, B. a RUZICKOVA, I. *Matematika 3*. Brno: Vysoké uční technické, 2005. 255 s. Studijni opora. [online], [cit. 2012-11-10].

[5] LAHLEY, K. *RFC2923: TCP Problems with Path MTU Discovery* [online]. Září 2000 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc2923>.

[6] MCCANN, J., DEERING, S. a MOGUL, J. *RFC1981: Path MTU Discovery for IP version 6* [online]. Srpen 1996 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc1981>.

[7] MOGUL, J. a DEERING, S. *RFC1191: Path MTU Discovery* [online]. Listopad 1990 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc1191>.

[8] POSTEL, J. *RFC2792: Internet Control Message Protocol* [online]. Září 1981 [cit. 2012-11-17]. Dostupné na: <http://tools.ietf.org/html/rfc792>.

A Metriky kódu

Počet súborov: 16 súborov

Počet riadkov zdrojového kódu: 1753 riadkov

Veľkosť spustiteľného súboru: 40kB (systém GNU/Linux, 64 bitová architektúra, pri preklade bez ladiacich informácií)