Multicast to unicast

ISA projekt, 2014

FIT VUT, Brno

Autor: Ján Spišiak (xspisi03@stud.fit.vutbr.cz)

Datum: 30. 11. 2014

Obsah

Abstrakt
Cieľom aplikácie je preposielať multicastovú komunikáciu na dané unicastové zariadenie. Na to musí aplikácia byť schopná sa registrovať do multicastovej skupiny a komunikovat s unicastovým zariadením, a to pomocou použitia BSD socketov.

Úvod
Existuje viacero typov komunikácií, medzi ne patrí: anycast, broadcast, multicast, unicast. Každá ma svoj účel, pre nás budú podstatné len dve. Prvou je unicast, ide o *one-to-one* komunikáciu, tj. medzi dvoma zariadeniami. Druhou je multicast, teda *one-to-unique-many* no vo zvláštnych prípadoch môže byť zdrojov aj viacej. Najväčšou výhodou multicastu je zníženie objemu dát tečúcich cez sieť, avšak za následok si musí sieť pamätať ktoré zariadenia su pripojené k multicastovej skupine. To je však veľmi náročné na smerovacie zariadenia vo veľkých sietiach, a teda multicastová komunikácia nie je možná kdekoľvek. Teda v niektorých prípadoch je nutné posielať dáta konkrétnemu adresátovi a to nám umožní unicast.

Návrh aplikácie
Aplikácia je pomerne jednoduchá, po nastavení spojenií stačí v sľučke preposielať prijaté dáta, a vyskočiť zo sľučky pokiaľ dostane program signály SIGINT, SIGTERM alebo SIGHUP. Pri ukončení stačí zavrieť sockety a operačný systém sa postará o dokončenie operácií, a zároveň aj odhlásenie z multicastovej skupiny, pokiaľ iný program nie je prihľásený na tom istom sockete [1].

Implementácia

Spracovanie argumentov
Použitím funkcie *getopt()* vieme jednoducho spracovať jedno písmenkové argumenty. Následne treba vyextrahovať adresu a port, čo robí funkcia *parseUrl()*.

Spracovanie signálov
Náš program môže byť kedykoľvek ukončení signálmi SIGINT, SIGTERM alebo SIGHUP [2] a preto pomocou *sigaction()* nastavíme callback pre spracovanie signálov *signal_handler()* ktorý nastaví príznak *shouldExit*.

Inicializácia spojení

Následne pomocou funkcie *getaddrinfo()* zvalidujeme adresy, a dostaneme štruktúry potrebné na vytvorenie spojenie s adresátmi. Vytvoríme sockety, môžeme nastaviť *SOCK_REUSABLE* aby bolo možné použiť ten istý socket viacerými programmi, prípadne aby sme nemuseli pri reštartovaní programu čakať kým operačný systém uvolní socket pre použitie.

Po vytvorení môžeme „pripojiť“ unicastový socket. Keďže UDP je bezstavová komunikácia, funkcia *connect()* si len zapamätá dáta potrebné pre posielanie packetov, čím sa vyhneme použitiu funkcie *sendto()*.

Na základe verzie IP ktorú sme dostali od *getaddrinfo()* pripravíme vhodné štruktúry: *ip_mreqn* pre IPv4 a *ipv6_mreq* pre IPv6 [3]. Pomocou funkcie *setsockopt()*, IP_ADD_MEMBERSHIP pre IPv4 a IPV6_JOIN_GROUP pre IPv6 sa pripojíme k multicastovej skupine.

Beh programu

Beh beží v sľučke kontrlujúcej *shouldExit*. Každé dáta načítane *recv()* sú následné preposlané cez *send()* na unicastového adresáta.

Ukončenie

Pri ukončení sa zavrú sockety a uvolní pamäť alokovaná *getaddrinfo()*. Pri uzavretí socketov, sa operačný systém postará o dokončenie operácií, a zároveň aj odhlásenie z multicastovej skupiny, pokiaľ iný program nie je prihľásený na tom istom sockete [1].

Použitie

Viz. README.

Informácie

Veľkosť spustitelného súboru: 12kB

Riadkov kódu: 290

Referencie

- http://www.tldp.org/HOWTO/Multicast-HOWTO-6.html
- http://www.gnu.org/software/libc/manual/html_node/Sigaction-Function-Example.html
- http://linux.die.net/man/7/ipv6