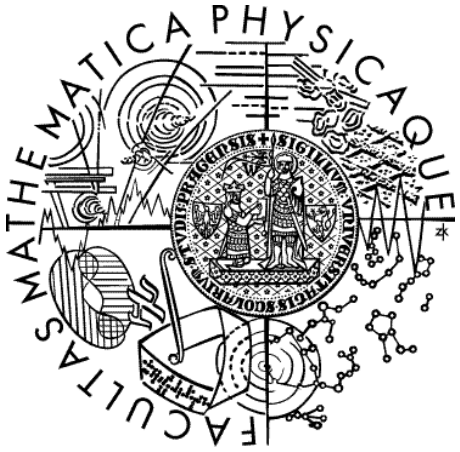
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Miroslav Novotný

Podpora mobility v bezdrátových sitích a Internetu

Katedra softwarového inženýrství.
Vedoucí diplomové práce: Doc. Ing. Jan Janeček, CSc
Studijní program: Informatika, architektura a principy systémového prostředí.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze dne: 12.4.2007 Miroslav Novotný

Obsah

Název práce: Podpora mobility v bezdrátových sítích a Internetu.
Autor: Miroslav Novotný
Katedra: Katedra softwarového inženýrství.
Vedoucí diplomové práce: Doc. Ing. Jan Janeček, CSc.
e-mail vedoucího: janecek@fel.cvut.cz
Abstrakt:
Důležitou součástí komunikačních systémů je jejich schopnost podpořit mobilitu jednotlivých koncových zařízení, případně programových a datových komponent. Tato diplomová práce se zabývá metodami řešení mobility v Internetu a bezdrátových sítí. V kapitole týkající se Internetu popisuje metody rozšíření IP protokolu o mobilní prvky a metody řešící mobilitu pomocí doménových jmen. Shrnuje klady a zápory těchto přístupů a navrhuje nový přístup založený na distribuovaných hašovacích tabulkách. Součástí práce je také simulační model navrhovaného řešení. V části o bezdrátových sítí shrnuje existující bezdrátové technologie a porovnává jejich přístup k mobilitě. Na závěr uvádí představu plně mobilního uzlu, který využívá maximum z diskutovaných metod a ilustruje možnosti, které mobilita nabízí.

Klíčová slova:
Mobilita, Distribuované hašovací tabulky, Bezdrátové sítě, Internet.

Title: Mobility support in Internet and Wireless Network
Author: Miroslav Novotný
Department: Department of Software Engineering
Supervisor: Doc. Ing. Jan Janeček, CSc.
Supervisor's e-mail address: janecek@fel.cvut.cz
Abstract:
Important part of communication systems is their capability to support mobility of user's devices, eventually of program and data components. This thesis deals with methods of mobility solution in the Internet and Wireless networks. In the chapter relating to the Internet it describes of mobile extensions of IP protocol and methods which use domain names. It summarizes positives and negatives both approaches and suggests a new approach based on distributed hash tables. A part of thesis is also a simulation model of the suggested approach. In the part about Wireless networks it summarizes existing wireless technologies and compares their access to mobility. In the conclusion it states a concept of a full mobile node that uses maximum from the methods discussed above and illustrates the possibilities that the mobility offers.

Keyword:
Mobility, Distributed Hash Tables, Wireless network, Internet.

1.Úvod

Celosvětový rozvoj komunikačních technologií v posledním desetiletí přinesl i nové požadavky na jejich schopnosti. V návaznosti na masivnější rozšíření mobilních zařízení jako jsou PDA, laptopy nebo mobilní telefony, vznikla potřeba podpořit mobilitu i na úrovni komunikačních protokolů. Větší mobilita obyvatelstva a jeho závislost na komunikačních technologiích tuto potřebu ještě podtrhuje.

V dnešním globalizované světě je přístup na Internet odkudkoliv již téměř samozřejmostí. Existuje bezpočet technologií, které nám zprostředkují přístup do Internetu při cestě autem, vlakem nebo letadlem, z domova i z kanceláře, ale také na obchodní cestě na druhém konci světa. Chceme využívat Internet, když se pohybujeme rychlostí 260 km/h při cestě rychlovlakem, i když sedíme v teple svého domova.

Za různých okolností využíváme jinou technologii připojení a pravděpodobně i jiného poskytovatele. Naše mobilní zařízení se musí mezi všemi dostupnými technologiemi a poskytovateli umět orientovat a pokud je potřeba, přepnout na jinou technologii. Pro nás by ale tato změna měla být naprosto neviditelná. Změna by se nijak neměla dotknout našeho šifrovaného spojení na firemní aplikační server ani našeho VoIP hovoru se šéfem. Ať už jsme doma nebo právě na cestě letadlem nad Atlantikem, chceme, aby služby, které naše mobilní zařízení nabízí na Internetu (např. sdílení souborů), byly stále stejně dostupné.

Cílem této práce je shrnout existující přístupy k řešení mobility v různých komunikačních systémech, především pak v Internetu, bezdrátových sítí a ad-hoc sítích. Dále pak pokusit se navrhnout přístupy nové a rozebrat jejich vlastnosti, výhody a nevýhody vzhledem k již existujícím nebo navrženým systémům.

2.Vymezení cílů studie

Cílem studie je u vybraných komunikačních technologií porovnat jejich přístup k zajištění mobility. Mobilita je v tomto kontextu chápaná jako schopnost zařízení změnit své fyzické umístění aniž by tak byla ohrožena jeho identifikace v rámci dané komunikační sítě, a to ani v době, kdy probíhá přesun. Tedy aby zařízení mohlo cestovat mezi přípojnými body do sítě a bylo stále pro ostatní zařízení v síti dosažitelné pod svoji jednoznačnou identifikací. Uzel s touto schopností budeme nazývat mobilní uzel. Naproti tomu uzel, který tuto schopnost nemá a je odkázán na jedno „statické“ umístění, nazveme statický uzel.

Identifikátor může být v různých sítích chápán odlišně, nebo dokonce odlišně i v rámci jedné sítě. Je pak na místě mluvit o mobilitě v kontextu jednotlivých identifikátorů nebo síťových vrstev.

Ve studii rozeberu některé běžně používané komunikační technologie. Podrobně popíši jejich komunikační protokoly, které zajišťují mobilitu. Zvážím jejich použitelnost a vztah k ostatním přístupům. V hodnocení jednotlivých existujících technik se zaměřím na následující kriteria:

- Zátěž adresačního mechanismu.
- Zabezpečení.
- Požadavky na přenosovou síť.
- Přínos mobility v dané síti.
- Rozšíření a stav implementace.

Tam, kde to je možné, se pokusím navrhnout odlišný přístup k řešení mobility a porovnat ho s již existujícími řešeními.

2.1 Zátěž adresačního mechanismu.

Adresačním mechanismem rozumíme způsob, jakým je uzel v rámci sítě identifikován a jak je na základě této identifikace realizován přenos dat mezi dvěma uzly. Ve většině případů je identifikace uzlu těsně spjata s topologií sítě, což je pro účely mobility naprosto nevyhovující. Z tohoto pohledu můžeme komunikační sítě rozdělit do dvou kategorií:

- Sítě, jejichž adresační mechanismus počítá s podporou mobility a ta je jeho nativní součástí.
- Sítě, jejichž adresační mechanismus s podporou mobility nepočítá a je třeba budovat nástavby, jenž nějakým způsobem adresační mechanismus obchází.

Podpora mobility by také neměla způsobit dramatický nárůst zátěže mezilehlých uzlů oproti stavu, kdy mobilita podporována nebyla. Mezilehlými

uzly rozumíme uzly, jenž slouží jako prostředníci při komunikaci dvou vzdálených uzlů.

Další důležitou otázkou je, jak rychle dokáže síť reagovat na změnu lokace mobilního uzlu.

2.2 Zabezpečení.

Zabezpečení je dosti široká problematika o které by se toho dalo napsat skutečně hodně, ale není to prioritním cílem této práce. Přesto se však musíme zmínit o některých bezpečnostních aspektech, které podpora mobility může přinést.

- Identifikace mobilního uzlu - mobilní uzel se může do sítě připojit v podstatě kdekoliv nebo nemusí být připojený vůbec. Po připojení uzlu je potřeba rozhodnout, jestli uzel je skutečně tím, za koho se vydává (autentifikace), a jestli má právo přístupu do sítě (autorizace).
- Zabezpečení mobilního uzlu - mobilní uzel je z principu více zranitelný než uzel statický. Musí počítat s tím, že se bude připojovat do sítě v místě, které není chráněné. Bezpečnostní rizika známá z běžných „statických“ sítích jsou více pravděpodobná. Bezpečnostní mechanismy, které jsou ve „statických“ sítích volitelné, by měly být proto v mobilní implementaci povinné. Navíc přibývají některé nové bezpečnostní otázky:
    - Zneužití identity – někdo se vydává za mobilní uzel.
    - Zajištění soukromí – fyzické umístění mobilního uzlu by mělo být plně transparentní. Tedy ostatní uzly v síti by neměly mít možnost toto fyzické umístění nějakým způsobem zjistit.

2.3 Požadavky na přenosovou síť.

Mobilní uzel by v ideálním případě neměl mít vyšší požadavky na komunikační síť než jeho statický ekvivalent. Některé přenosové charakteristiky se však mohou s podporou mobility změnit. V této souvislosti sledujeme tyto hodnoty:

- Zvýšení režijních nákladů přenosu – o kolik se zvýšila režie přenosu oproti statickému uzlu. Kolik je režie na zajištění mobility.
- Snížení doby odezvy – tedy o kolik je komunikace s mobilním uzlem pomalejší než se statickým. Rychlost komunikace samozřejmě závisí na rychlosti sítě v místě připojení mobilního uzlu, to ale není pro toto kriterium podstatné. Podstatné je zdržení, které je přímým důsledkem mechanismu mobility. Např. zbytečně dlouhá cesta datových rámců od odesilatele k příjemci.

2.4 Přínos mobility v dané síti.

Někdy vzniká otázka, jestli je mobilita v konkrétní komunikační síti skutečně přínosná. Jestli výhody, které nabízí, jsou skutečně natolik důležité, aby

vyvážily zvětšené režijní náklady na přenos a odůvodnily investice do nových zařízení. Zda by nebylo efektivnější implementovat mobilitu na jiné úrovni nebo ji neimplementovat vůbec.

Bez poptávky na straně koncových uživatelů nemá technologie skutečné šance na výraznější rozšíření.

2.5 Rozšíření a stav implementace.

Toto kriterium je v podstatě odrazem předchozího, zkoumáme v něm jak se technologie skutečně prosazuje v praxi. Pokusím se v něm shrnout a porovnat implementace v nejběžnějších operačních systémech a síťových zařízeních.

3. Mobilita v Internetu

První komunikační sítí, o kterou se budeme zajímat, je Internet. Je to vůbec nejpoužívanější komunikační síť ve světe počítačů. Jejím základem je IP protokol a jedno z chápaní mobility je na úrovni IP adres. Dnešní Internet znamená především protokol IP verze 4 (IPv4), i když se pomalu začíná prosazovat i protokol IP verze 6 (IPv6). Přístup k mobilitě je u obou verzí podobný, IPv6 lépe řeší problémy, které se objevily v předchozí verzi protokolu.

3.1 Internet Protokol verze 4

Každý uzel v Internetu má přiřazenu IP adresu, která ho jednoznačně identifikuje. Na jejím základě jsou uzlu doručovány IP datagramy. Adresa uzlu musí být jedna z adres, které jsou přiřazeny síti, v které se uzel fyzicky nachází. Jinak by směrovací mechanismus nedokázal uzel v Internetu lokalizovat.

Problém nastane v případě, že se uzel připojí do jiné sítě a nechce ztratit možnost komunikace s ostatními uzly. Pokud chceme tento problém řešit bez nějakého rozšíření IP protokolu, nabízejí se dvě možnosti:

1. Uzel změní svoji IP adresu podle sítě, v které je nově připojen.
2. Uzel si ponechá svoji IP adresu a propaguje změnu svého umístění do směrovacích tabulek všech směrovačů.

První možnost mění identifikaci uzlu v rámci komunikační sítě a nesplňuje tak definici mobility na úrovni IP adres. Pro vyšší vrstvy je nemožné pokračovat v již navázaných spojeních a pro vzdálené uzly, které chtějí zahájit komunikaci, je takovýto uzel nedostupný. Problém mobility pak můžeme řešit na jiné úrovni, v kontextu doménových jmen, jak je popsáno v části 3.3.

Druhá možnost už sice splňuje požadavky na mobilitu, ale cena, která se za to musí platit, je příliš vysoká. Znamenala by zásah do směrovacích tabulek velkého množství směrovačů a při počtech mobilních zařízení, které se dnes připojují do Internetu, by znamenala neúměrnou zátěž.

Je tedy zřejmé, že bez nějakého specializovaného rozšíření protokolu se podpora mobility v Internetu neobejde.

3.1.1 Popis protokolu pro mobilitu v IPv4

Při návrhu protokolu pro mobilitu byl kladen důraz na transparentnost. Změny by se měly týkat pouze síťové vrstvy a vyšší vrstvy by neměly být ovlivněny. Také je vhodné, aby převážná část implementace protokolu byla na straně mobilního uzlu a bylo tak možné mobilně komunikovat s uzly, které mobilitu neznají. Snaha o dosažení těchto cílů vyvrcholila zatím poslední verzí RFC 3344 [1] z roku 2002.

Pro účely mobility v IPv4 síti musíme nejprve definovat některé nové pojmy:

- **Mobilní uzel –** uzel, pro který chceme zajistit mobilitu. Tedy uzel (Laptop, PDA, ...), který se může připojit do Internetu v různých místech. Pro ostatní uzly je viditelný pod svoji **domovskou adresou**, která odpovídá adrese, která mu byla přidělena v jeho domovské síti. Pod touto adresou je také veden v DNS.
- **Domovský agent** – je uzel v domovské síti (nejlépe směrovač), který zajišťuje mobilitu pro mobilní uzel. Jeho úkolem je přesměrovat komunikaci pro mobilní uzel na jeho nové umístění. Respektive na cizího agenta sítě, v niž se mobilní uzel právě nachází.
- **Cizí agent** – je uzel v cizí síti (opět nejlépe směrovač), který zajišťuje mobilitu pro mobilní uzel. Jeho úkolem je převzít data od domovského agenta a předat je mobilnímu uzlu. Navíc hraje důležitou roli v procesu registrace mobilního uzlu.
- **Korespondující uzel**[1] – takto budeme označovat uzel, který právě komunikuje s mobilním uzlem.

Základní koncept popisuje obrázek 1. Následující body popisují komunikaci na obrázku:

1. Korespondující uzel vyšle IP datagram, který je adresován na domovskou adresu mobilního uzlu.
2. Domovský agent tento IP datagram zachytí. Ví, že mobilní uzel, kterému je datagram určen, se nachází v cizí síti a pomocí IP tunelu [2] ho přepošle na adresu příslušného cizího agentova.
3. Cizí agent datagram rozbalí a po lokání síti ho doručí mobilnímu uzlu.
4. Pro odpovědi mobilního uzlu se již používá standardní cesta.

Komunikace se děje po pomyslném trojúhelníku (1,2,4) a odtud se vžilo označení triangle routing.

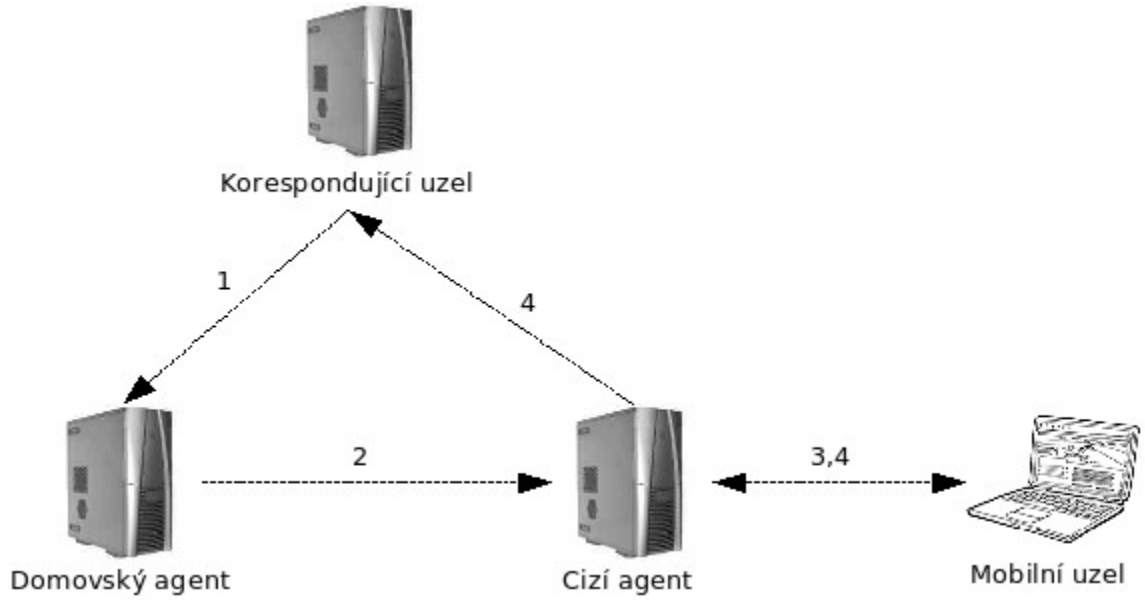
Rozeberme teď celý mechanismus podrobněji a rozdělme ho do několika fází.

a) Objevování agentů

Po připojení mobilního uzlu je jeho prvním úkolem zjistit informace o připojené síti. Především to, jestli se jedná o jeho domovskou nebo cizí sít a jestli je dostupný příslušný agent. K tomuto účelu se používá rozšíření ICMP zpráv „router discovery“ [3]. Tyto zprávy jsou běžně používány k informování uzlu o směrovačích v dané síti, jejich rozšíření obsahuje informace o schopnostech cizího agenta a o adrese, kterou lze přidělit pro obsluhu mobilního uzlu.

- **Obslužná adresa**[2] – tak budeme nazývat adresu cizího agenta, kterou může mobilní uzel využít k přesměrování svých datagramů. Je to IP adresa, která určuje druhý konec IP tunelu. Prostřednictvím tohoto tunelu domovský agent přeposílá zprávy cizímu agentovy. Jednu obslužnou adresu může simultánně využívat více mobilních uzlů.

1 Vycházíme z anglické terminologie: correspondent node.
2 V originále je to „care-of address“.

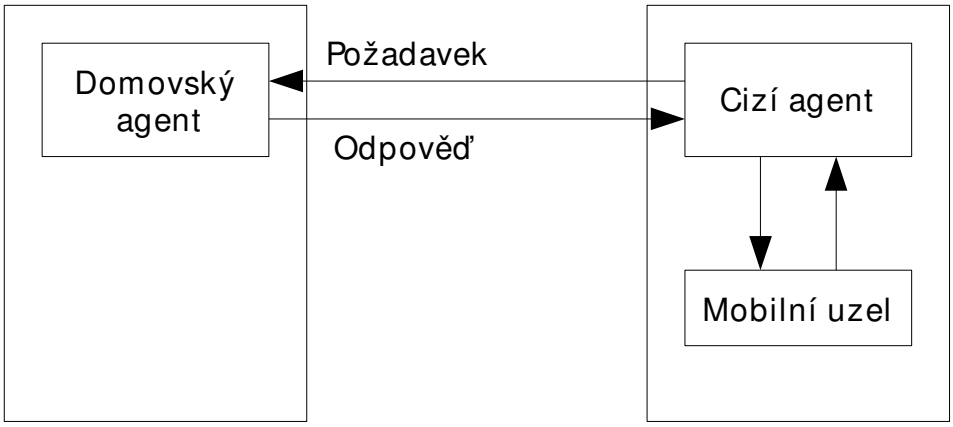


Obrázek 1: Základní koncept mobility v IPv4 síti.

b) Registrace

Poté, co mobilní uzel získá obslužnou adresu, musí ji zaregistrovat u svého domovského agenta. Požadavek o registraci zašle cizímu agentovi a ten ji přepošle na domovského agenta. Požadavek musí obsahovat domovskou adresu mobilního uzlu, adresu domovského agenta, obslužnou adresu, dobu platnosti registrace a další parametry. Pro účely registrace se používají UDP datagramy na portu 434.

Domovský agent po obdržení požadavku o registraci nejprve rozhodne o oprávněnosti požadavku. Pokud je oprávněný, zapíše do svých tabulek příslušnou vazbu a odešle kladnou odpověď na adresu cizího agenta. Ten ji předá mobilnímu uzlu. Pokud je registrace zamítnuta, je o tom mobilní uzel prostřednictvím cizího agenta také informován. Proces registrace zachycuje obrázek 2.



Obrázek 2: Registrace mobilního uzlu

c) Komunikace

Po úspěšné registraci může komunikace s mobilním uzlem probíhat tak, jak je uvedeno na obrázku 1.

d) Přesun do jiné sítě

Pokud je mobilní uzel náhle odpojen od sítě, nelze o této skutečnosti informovat domovského agenta. Datagramy jsou i nadále přesměrovávány na cizího agenta, který je předává do lokální sítě, kde jsou zahazovány. To se děje až do té doby, než vyprší platnost registrace nebo dokud se mobilní uzel neregistruje z jiné sítě. Pokud je tento výpadek krátkodobý, dokáží se s ním protokoly vyšší vrstvy vyrovnat a nepřerušit navázané spojení.

Když se mobilní uzel přesune do své domovské sítě, posílá žádost o registraci s dobou platnosti nastavenou na nulu. To způsobí zrušení dříve registrovaného přesměrovaní.

3.1.2 Bezpečnostní otázky mobilního IPv4

Bezpečnost je stěžejní problém mobility v IPv4. Je zřejmé, že přesunem do cizí sítě se mobilní uzel stává více zranitelnější než uzel statický. Bezpečnost komunikace mobilního uzlu je ovlivněna úrovní zabezpečení cizí sítě. Navíc díky mechanismu mobility se útočníkovi otevírají nové možnosti napadení uzlu.

Největší bezpečnostní hrozbu představuje registrační proces. Pokud

nezajistíme jeho bezpečnost, mohl by útočník způsobit nedostupnost mobilního uzlu nebo dokonce přes sebe přesměrovat veškerá data určená pro mobilní uzel.

V každém registračním požadavku je proto požadována autentizace mobilního uzlu. Také musíme zajistit integritu požadavku (jestli nebyl cestou změněn) a zajistit ochranu proti replay útokům.

Mezi mobilním uzlem a domovským agentem musí byt ustavena mobilní bezpečnostní asociace. Protokol neřeší, jak je tato asociace ustanovena, předpokládá, že se tak stalo už dřív nezávisle na mobilitě. V mnoha implementacích ji musíme nastavit manuálně.

- **Mobilní bezpečnostní asociace (MSA)** – je vázána na dva komunikační partnery a obsahuje kolekci bezpečnostních kontextů.
- **Bezpečnostní kontext** – určuje autentizační algoritmus, sdílené tajemství (sdílené klíče pro symetrické šifry nebo veřejné a soukromé klíče pro asymetrické šifry) a algoritmus pro zajištění ochrany proti replay útokům. V rámci MSA je kontext identifikován pomocí „security parametr index (SPI)“.

a) Autentizace a integrita.

Protokol definuje rozšíření registračního požadavku „Mobile-Home Authentication“, které musí být přítomno v každé žádosti o registraci. Obsahuje autentizační hodnotu a SPI v rámci MSA mezi domovským agentem a mobilním uzlem. Autentizační hodnota je hash závislý na použitém algoritmu specifikovaném v rámci SPI. Standard předepisuje zahrnout do počítání hash hodnoty užitečný náklad UDP datagramu, všechna dřívější rozšíření a také délku, typ a SPI autentizačního rozšíření. Pro hašovací funkci se použije sdílený klíč definovaný v SPI. Tímto způsobem zajistíme i integritu zprávy.

Implicitní algoritmus je HMAC-MD5, který používá hašovací funkcí MD5 rozšířenou o použití sdíleného klíče metodou prefix+sufix.

Rozšíření „Mobile-Foreign Autentication“ je obdobou předchozího a slouží k autentizaci vůči cizímu agentovi. Toto rozšíření není povinné a je použito v případě, kdy je potřeba autentizovat mobilní uzel i u cizího agenta. Například v komerční sféře, kde se platí za přístup do Internetu. Tento přístup vyžaduje, aby měl mobilní uzel ustavenu MSA s každým cizím agentem.

Další nepovinné rozšíření je „Foreign-Home Autentication“, které slouží pro autentizaci cizího agenta vůči domovskému.

b) Ochrana proti replay útokům.

Ochrana proti replay útokům využívá 64bitové identifikační číslo v hlavičce požadavku. To má zajistit, aby útočník nemohl zopakovat registraci na základě odchycených paketů. Standard popisuje dva algoritmy, které lze použít. Jeden pracuje na základě časových razítek, druhý na základě náhodných čísel. První jmenovaný je povinný v každé implementaci.

Ochrana proti replay útokům s použitím časových razítek:

Tato ochrana je postavena na aktuálním čase, který odesílatel vloží do identifikátoru každého paketu. Aktuální čas je formátován stejným způsobem jako v paketech Network Time Protocol (NTP) [4]. Příjemce kontroluje, jestli časové razítko je „blízko“ jeho aktuálního času. Konkretní „blízkost“ pak určuje použité SPI, nemělo by to však být méně než tři sekundy.

Pro tuto metodu je vyžadováno, aby oba komunikační partneři měli synchronizované hodiny, to lze zajistit například pomocí NTP.

Ochrana proti replay útokům s použitím náhodných čísel:

Myšlenka je založena na použití náhodného čísla, které odesílatel vloží do odesílaného paketu. V odpovědi pak očekává stejné náhodné číslo a také náhodné číslo protějšku, které může použít v další zprávě. Tímto zajistíme, že každá zpráva bude použitelná pouze jednou. Tato metoda vyžaduje dobrý generátor náhodných čísel.

c) Další bezpečnostní otázky

Základní standard [1] nechává některé bezpečnostní otázky otevřené. Ochrana proti odposlouchávaní na zranitelné cizí síti musí být řešena na jiné úrovni. Integrita zpráv je zajištěna pouze pro registrační zprávy. Cizí agent je považován za důvěryhodný, neimplementuje se tedy ochrana proti analýze nebo odposlouchávání přenosu, které by mohl provádět právě cizí agent. V komerčním prostředí je potřeba implementovat dokonalejší metody pro autentizaci, autorizaci a podporovat účtování. Tyto požadavky jsou řešeny v RFC 2977 [5].

3.1.3 Problémy mobilního IPv4 a jejich řešení

V konceptu Mobility v IP, tak jak byl popsán výše, se objevilo několik problémů. Následující text popisuje problémy tak, jak jsou uvedeny v článku [6], a nastiňuje i možnosti řešení.

a) Neexistence cizího agenta v síti.

V případě, že v cizí síti není k dispozici žádný cizí agent, je mobilnímu uzlu povoleno nahradit jeho služby vlastními silami. Od cizí sítě očekává pouze přidělení veřejné směrovatelné IP adresy. V terminologii mobility v IP se nazývá lokalizovaná obslužná adresa.

- Lokalizovaná obslužná adresa[3] – je platná směrovatelná IP adresa, která je přidělena mobilnímu uzlu. Na této adrese mobilní uzel naslouchá a je schopný přijímat tunelované datagramy.

Registrační proces probíhá přímo mezi domovským agentem a mobilním uzlem. Jako koncový bod tunelu je nastavena přímo lokalizovaná obslužná adresa.

3 V originále „co-located care-of address“.

Tento přístup má několik problémů. Jednak na obsluhu mobility vyžaduje větší počet adres. Každý mobilní uzel totiž musí mít svoji jednoznačnou obslužnou adresu. Nelze použít jednu adresu pro všechny uzly tak, jak tomu bylo v případě existence cizího agenta. Také pokud chceme účtovat provoz pro mobilní uzly, je to o mnoho složitější.

b) Filtrování na základě adresy odesilatele.

Některé směrovače v Internetu jsou konfigurovány tak, že kontrolují i adresu odesilatele. Datagram, který přichází z určitého rozhraní, musí mít adresu odesilatele v rozsahu, který je specifikován pro dané rozhraní. Pokud toto nesplňuje je směrovačem zahozen. Cílem tohoto opatření je omezit možnosti používaní podvržených adres odesilatele. Takovéto kontroly doporučuje RFC 2827 [7].

Pokud se mobilní uzel připojí do sítě, kde je takovýto mechanismus implementován, nemůže komunikovat způsobem jenž ilustruje obrázek 1. Datagramy, které posílá korespondujícímu uzlu, mají právě takto podvrženou adresu odesilatele. Řešením je přeposílat odchozí datagramy na svého domovského agenta. K tomu se využívá stejný způsob jako pro příchozí datagramy, tedy IP tunel mezi domovským a cizím agentem. Mobilní uzel může o vytvoření reverzního tunelu požádat během registrace nebo i později. Celý tento mechanismus popisuje RFC 3024 [8].

c) Filtrování příchozích datagramů v domovské síti.

Na obdobný problém jako v části b) narazíme i v případě, že mobilní uzel chce komunikovat s uzlem ve své domovské síti. Například s internetovým serverem, který není veřejně přístupný. Firewall domovské sítě nepropustí datagramy od mobilního uzlu, protože přicházejí z Internetu. Navíc mají jako adresu odesilatele adresu z domovské sítě a tím jsou pro firewall velice podezřelé.

Tento problém se, stejně jako předchozí, řeší pomocí reverzního tunelu. Pak stačí, aby firewall byl nakonfigurovaný tak, aby propouštěl příchozí UDP pakety na portu 434 pro domovského agenta a umožnil tak registraci.

d) NAT.

Dalším problémem je použití NATu v cizí síti. Tunelované pakety totiž nemají šanci projít překladem adres. Při použití jednoduchého IP tunelu překladač nemá dostatek údajů pro vytvoření vazby mezi domovským a cizím agentem.

Řešením v tomto případě je použít tunel postavený na UDP datagramech tak, jak je popsáno v RFC 3519 [9].

e) Neoptimální cesta datagramů.

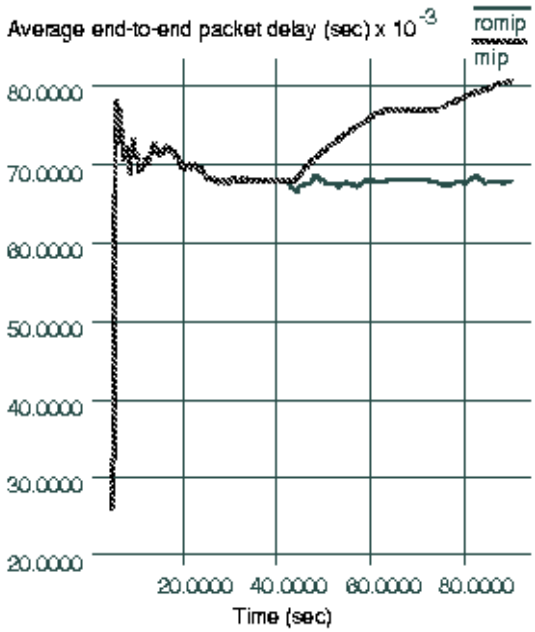
Přeposílání datagramů od domovského k cizímu agentovi rozhodně není optimální. Datagram musí urazit až dvojnásobnou vzdálenost než je skutečná

vzdálenost mezi mobilním a korespondujícím uzlem. Komunikace se tak podstatně zpomaluje. Pokud se použije i reverzní tunel, je situace ještě horši.

Jakékoliv řešení tohoto problému musí být postaveno na spolupráci korespondujícího uzlu. Ten musí vědět, že komunikuje s mobilním uzlem, který se nachází jinde než ve své domovské síti (znát jeho obslužnou adresu). Pak může posílat tunelované datagramy přímo cizímu agentovi. Technický popis obsahuje dokument [10].

Nevýhodou tohoto přístupu je, že vyžaduje zásah do síťové vrstvy i na straně korespondujícího uzlu. V případě, že korespondující uzel o mobilitě nikdy neslyšel, nebude fungovat.

Pro korespondující uzel už není umístění mobilního uzlu plně transparentní. Dokáže přesněji lokalizovat, kde se mobilní uzel nachází a to v některých případech může být nežádoucí.



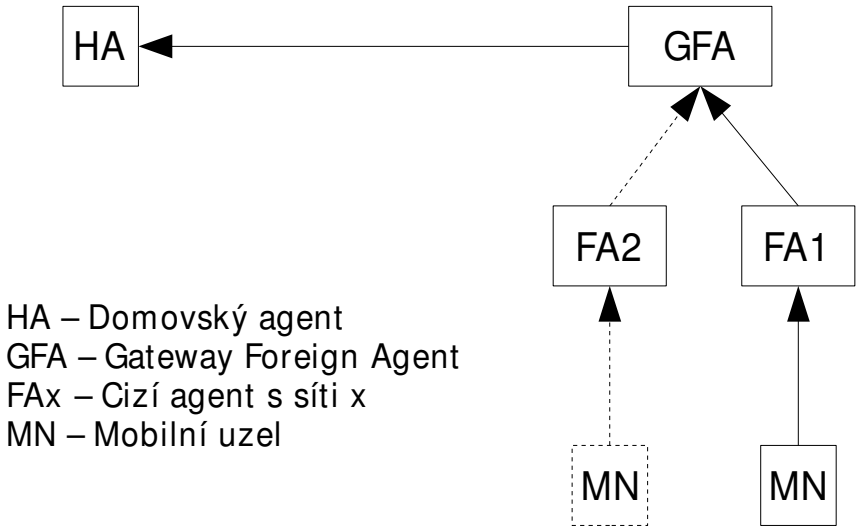
Obrázek 3: Průměrné zpoždění datagramů v mobilním IP s a bez optimalizace.

Přínosem této optimalizace by mělo být snížení zpoždění datagramů. Dosažení tohoto cíle ilustruje obrázek 3. Uvedený graf vychází ze simulace [11], která obsahuje mobilní a korespondující uzel, jednoho domovského agenta a dva cizí agenty. Zpoždění je vždy měřeno mezi korespondujícím a mobilním uzlem. Mobilní uzel začíná ve své domovské síti, v čase 35 sekund přechází do první cizí sítě a v čase 65 sekund přechází do druhé cizí sítě. Plná čára zachycuje zpoždění bez použití optimalizace, přerušovaná se zapnutou optimalizací.

f) Rychlost přechodu do jiné sítě.

Přechod do jiné sítě vyžaduje opětovnou registraci mobilního uzlu. Tento proces jistou dobu trvá a během této doby je vysoká pravděpodobnost ztráty paketů. Pokud netrvá příliš dlouho, jsou vyšší vrstvy schopny se s tím vyrovnat. V některých případech to ale může působit nepříjemnosti například ve voice-over-IP nebo v jiných real-time službách. Také při příliš rychlém pohybu mobilního uzlu a jeho častých přeregistracích může být komunikace úplně znemožněna. Tyto důvody vedly k vývoji mechanismů, které mají čas na přechod do jiné sítě (handover times) snížit na minimum. Jsou používány dva základní přístupy:

Regionální registrace[4] - vychází z hierarchie cizích agentů pro síťovou doménu (sítě pod stejnou správou). Na vrcholu hierarchie je "gateway foreign agent" (GFA), podřízení cizí agenti se nazývají regionální cizí agenti. Žádost o registraci od mobilního uzlu směřuje k regionálním cizím agentům a ty ji předávají nadřízenému uzlu v hierarchii. Teprve GFA kontaktuje domácího agenta. Datagramy jsou pak tunelovány stejnou cestou mezi uzly v hierarchii.



Obrázek 4: Přesun mobilního uzlu v prostředí s hierarchií cizích agentů

Při změně registrace se tato změna odehrává jen na dané úrovni hierarchie. Tedy nemusí se kontaktovat domovský agent. Tento přístup přináší zrychlení v případě, že domovský agent je hodně vzdálen od navštívené domény. Samozřejmě také za předpokladu, že mobilní uzel se pohybuje jen v rámci domény.

Obrázek 4 ilustruje přechod mobilního uzlu mezi dvěma sítěmi ve stejné doméně. Nejprve je uzel připojen do sítě s cizím agentem FA1 (plná čára).

4 V originále: 'regional registration'

Registrace a následný IP tunel probíhá po trase MN – FA1 – GFA – HA. Po přesunu uzlu do "blízké" sítě s agentem FA2 je registrační proces prováděn po trase MN – FA2 -GFA – HA. GFA, ale pozná, že tento uzel má již zaregistrovaný a pouze si přesměruje svůj tunel na FA2. Je potřeba tedy změnit jen první část tunelu (přerušovaná čára).

Dokument popisující mechanismy regionální registrace [12] uvažuje jen dvouúrovňovou hierarchii, ta ale může být snadno rozšířena.

Rychlé předávky[5] - tento mechanismus je použitelný pouze v případě, že mobilní uzel může být informován o IP adrese nového cizího agenta ještě před opuštěním původní sítě.. Návrh standardu je popsán v dokumentu [13]. Jsou možné dva základní přístupy a jejich kombinace.

- **Předregistrace** – mobilní uzel ještě před odpojením od stávající sítě zahájí registraci pro nové umístění. Kdykoliv během registračního procesu se pak mobilní uzel může přesunout do nové sítě. Potvrzení o registraci je doručeno na obě lokace.
- **Post registrace** – je založena a obousměrném tunelu, který je ustaven mezi starým a novým cizím agentem. Přes tento tunel jsou přenášeny pakety dokud není formálně provedena celá registrace.

3.1.4 Zhodnocení mobilního IPv4

a) Zátěž adresačního mechanismu.

Nad mobilitou do IP sítě se začalo přemýšlet až po rozšíření protokolu IP v podobě dnešního Internetu. Původní adresační mechanismus s ním nepočítal a celý IP protokol je vlastně volitelná nástavba. Přesto protokol v nejjednodušší podobě vystačí pouze se dvěma uzly podporujícími mobilitu (domovským agentem a mobilním uzlem). Tento nejjednodušší přístup má však řadu nevýhod, které ho činí nepoužitelný na dnešním IPv4 Internetu. Bylo tedy nutné přistoupit ke zvýšení počtu uzlů, které se na mobilitě podílejí (cizí agent, korespondující uzel nebo celá hierarchie cizích agentů). Za cenu značného zesložitění původně jednoduchého protokolu se podařilo odstranit většinu vážnějších problémů.

Stále platí, že mezilehlé směrovače o mobilitě nemusí nic vědět. Toho se dosáhlo tunelováním paketů. Jediný, kdo si drží aktuální vazbu mezi domovskou a obslužnou adresou mobilního uzlu, je domovský agent. Na něm také spočívá největší díl odpovědnosti za zajištění mobility.

b) Zabezpečení.

Z bezpečnostního hlediska mobilita v IPv4 spoléhá hlavně na ochranu registračního procesu pomocí předpřipravených bezpečnostních asociací. Zajištění těchto asociací v globálním měřítku je neproveditelný úkol. Musíme

5 V originále: 'fast handoffs"

se tedy buďto vzdát optimalizace cesty, anebo se domluvit se správcem vzdáleného uzlu a ustanovit bezpečnostní asociaci. V tomto vidím největší nedostatek celého protokolu.

c) Požadavky na přenosovou sít.

Při komunikaci mobilního uzlu s korespondujícím uzlem nedojde k nikterak dramatickému nárůstu požadavků na přenosovou síť.

Mírným snížením doby odezvy je vinna dlouhá cesta datagramů, které jsou směrovány přes domovského agenta. Počet těchto datagramů lze ale snížit použitím optimalizace cesty. Dostaneme se tak do stavu, kdy pouze úvodní datagram bude muset projít touto dlouhou cestou.

Nárůst požadavků na přenosovou kapacitu může způsobit vlastní tunelování datagramů. Tunelované datagramu jsou totiž větší o další IP hlavičku. To samo o sobě není nikterak velký problém, je to rozumná daň za výhody, které nám mobilita poskytuje. Může však dojít k problému s fragmentací, kdy vlivem pár byte navíc je každý tunelovaný datagram je fragmentován. A to je již nepříjemné.

d) Přínos mobility v dané síti.

Mobilní zařízení připojované do Internetu z různých lokálních sítí dostane vždy adresy z rozsahu této sítě. Pokud přejde do jiné sítě, dostane novou adresu a všechna dosavadní spojení budou zrušena a nově otevřena z nového umístění. To aplikacím jako prohlížení webu nebo odesílaní emailu nevadí, protože během spojení není ukládaný žádný důležitý stav. Problém se objeví až u aplikací, které vyžadují plně stavovou komunikaci. Jsou to například všechny aplikace, které vyžadují přihlášení uživatele (SSH, IRC, různé Instant Messengery,..). Po novém navázaní spojení musí být tento stav obnoven a uživatel se musí znovu přihlásit. Jde to sice dělat automaticky, aby uživatel o tom ani nevěděl, ale způsobí to nepříjemné zdržení a ztrátě některých stavových informací se stejně nevyhneme, např. pád SSH session.

Vlastní změna IP adresy také může působit problémy. Většina služeb je postavena na modelu klient/server. U serveru se nepředpokládá, že by byl mobilní a může tak mít neměnnou adresu. Klient, který server kontaktuje, pak pevnou adresu ani nepotřebuje. V poslední době se však od tohoto modelu upouští a rozdíly mezi klientem a serverem jsou stírány. Nové aplikace (sdílení souborů, instant messaging) už pracují v modelu peer-to-peer, kde jsou oba uzly rovnocenné. Změna adresy jednoho uzlu (nebo dokonce obou) může způsobit vážné komunikační problémy.

Existují i aplikace, kde se mobilní uzel může objevit i v roli serveru. Pokud je nějaký internetový uzel bude potřebovat kontaktovat. To by bez použití nějakého druhu mobility nebylo možné vůbec.

e) Rozšíření a stav implementace.

Existuje mnoho implementaci do různých operačních systémů (Linux,

Windows, Solaris, ...). Jedná se jak o firemní implementace, tak i o univerzitní. Mobilní IP je podporováno i na mnoha směrovačích a ostatních sítových prvcích. Mnohé z nich implementují i další rozšíření mobilní Ipv4 jednoduše proto, že bez nich není příliš použitelné.

f) Shrnutí.

Se všemi vylepšeními uvedenými v části 3.1.3 se mobilita v IPv4 stává skutečně použitelnou. Je třeba si ale uvědomit, že řada z těchto vylepšení je zatím ve fázi návrhu standardu a zřejmě i tak zůstanou. Prakticky neexistují zařízení, které by je implementovala úplně všechna. V době, kdy se do popředí dostává protokol IPv6 je logické, že výrobci síťových zařízení nechtějí investovat do technologie postavené na IPv4, která by se navíc prosazovala jen velmi pomalu.

Hlavními překážkami pro rozšíření mobility v IPv4:

- Malá poptávka ze strany koncových uživatelů i správců sítí.
- Složitost protokolu způsobená omezením současných IPv4 sítí.
- Náklady na straně cizí sítě, které nepřinášejí žádný zisk pro uživatele cizí sítě (cizí agent).
- Pro globální rozšíření mobilního IP je potřeba globální bezpečnostní infrastruktura.

Přesto, že se mobilita v IPv4 příliš neprosadila, je to důležitý článek při vývoji skutečně mobilního Internetu. Mnoho nápadů a získaných zkušeností bylo použito při návrhu protokolu IPv6, kde je podpora mobility zabudována přímo do jádra protokolu.

3.2 Internet Protokol verze 6

Mobilita v IPv6 vychází ze zkušeností získaných při vývoji mobility v IPv4 a z nových možností poskytovaných IPv6. Díky tomu, že podpora mobility byla vyvíjena téměř souběžně s vývojem vlastního síťového protokolu, se podařilo odstranit většinu nedostatků. Aktuální RFC, které popisuje mobilitu v IPv6 sítích, má označení 3775 [14].

Základní rozdíly mezi mobilitou v IPv4 a IPv6 shrneme do několika bodů:

- V cizí síti není potřeba žádný speciální směrovač (obdoba cizího agenta v IPv4). Mobilita funguje v každé cizí síti bez požadavků na místní router.
- Podpora optimalizace cesty musí být obsažena v každé mobilní implementaci. Není to volitelné rozšíření jako tomu bylo v IPv4.
- Optimalizace cesty pracuje bezpečně bez nutnosti předpřipravených bezpečnostních asociacích. To umožňuje, aby optimalizace byla nasazena v globálním měřítku mezi všemi mobilními a korespondujícími uzly.
- Mechanismus optimalizace cesty pracuje i v případě směrovačů

aplikující filtry na základě adresy odesilatele (ingress filtering).

- Pakety, posílané mobilnímu uzly v cizí síti, jsou posílané pomocí směrovacích hlaviček v záhlaví IPv6 paketu. Nepoužívá se tedy IP tunel jako v případě IPv4 a snižuje se tak režie přenosu.
- Protokol pro mobilitu není svázaný s konkretní linkovou vrstvou. Místo protokolu ARP používá mechanismus objevování sousedů z IPv6, což ho činní robustnějším.

3.2.1 Popis protokolu pro mobilní IPv6

Základní koncept je stejný jako v případě IPv4. Mobilní uzel v cizí síti dostane standardním IPv6 mechanismem přidělenu IP adresu s prefixem cizí sítě. To bude jeho obslužná adresa, s tou se registruje u svého domovského agenta. Na tuto adresu mu domovský agent bude přeposílat pakety, které byly adresovány na jeho domovskou adresu. Mobilní uzel může poskytnout informace o své současné lokaci i korespondujícímu uzlu. Komunikace pak probíhá jen mezi korespondujícím a mobilním uzlem.

a) Objevování agentů a prefix management.

Protokol také poskytuje podporu pro použití více domovských agentů a omezenou podporu pro rekonfiguraci domovské sítě. Mobilní uzel nemusí znát adresu svého domovského agenta (nemusí to být ani směrovač domovské sítě), ale může použít mechanismus označovaný jako "dynamické objevování adresy domovského agenta"[6]. Tímto způsobem může mobilní uzel získat adresu svého domovského agenta i v případě, že se nenachází ve své domovské síti.

Základem tohoto mechanismu jsou anycastové adresy s prefixem domovské sítě, které jsouNapříklad uzel nepřijme paket s volbou „home address option“, pokud nezná příslušnou vazbu přiřazené právě domovským agentům. Anycastové adresy v IPv6 označují skupinu uzlů. Paket adresovaný této skupině je doručen vždy právě jednomu (nejbližšímu) členu skupiny. Mobilní uzel vyšle na tuto anycastovou adresu ICMP zprávu s požadavkem o adresu domovského agenta. Uzel, kterému je zpráva doručena, na ní odpoví jinou ICMP zprávou a uvede v ní svoji unicástovou adresu jako adresu domovského agenta. Na tuto adresu pak může mobilní uzel provést svoji registraci.

Mobilní uzel může být také informován o změnách v prefixech přiřazených jeho domovské síti i v době, kdy k ní není připojen. Děje se to opět pomocí ICMP zpráv vyměňovaných mezi domovským agentem a mobilním uzlem.

b) Registrace

Proces registrace dělíme na dva typy:

- Domovská registrace – registrace mobilního uzlu u domovského agenta.
- Registrace u korespondujícího uzlu – za účelem optimalizace cesty může

6 V originále: "dynamic home agent address discovery"

mobilní uzel registrovat svoji obslužnou adresu u korespondujícího uzlu.

Typicky může mobilní uzel získat v cizí síti více obslužných adres, ale pouze s jednou se registruje u svého domovského agenta. Tuto adresu nazýváme primární obslužnou adresou. Proces domovské registrace je podobný registraci v IPv4 bez použití cizího agenta. Mobilní uzel vyšle domovskému agentu zprávu, kde požádá o aktualizaci vazby mezi svojí domovskou adresou a primární obslužnou adresou. Domovský agent nejprve ověří oprávněnost požadavku, zanese si vazbu do své tabulky a pošle potvrzení mobilnímu uzlu. Domovská registrace vyžaduje dopředu ustanovenou bezpečnostní asociaci mezi domovským agentem a mobilním uzlem. Může se ustanovit manuálně nebo může být použít automatický key management IKE [15]. Vlastní přenos požadavku pak musí být chráněn ESP hlavičkou.

Poté, co je mobilní uzel zaregistrován u svého domovského agenta, může se podobným postupem registrovat u korespondujícího uzlu. Zde není potřeba dopředu ustanovovat žádnou bezpečnostní asociaci. Korespondující uzel k ověření identity mobilního uzlu používá mechanismus zvaný "return routability procedure".

- **Return routability procedure (RRP)** - mechanismus, který iniciuje mobilní uzel tím, že vyšle dvě zprávy korespondujícímu uzlu. Jedna zpráva půjde přes domovského agenta (home test init) a druhá zpráva putuje přímo korespondujícímu uzlu (care-of test init). Každá zpráva obsahuje bitový řetězec (cookie), který mobilní uzel očekává v odpovědi. Korespondující uzel na každou zprávu odpoví, odpověď pošle stejnou cestou jako k němu dorazil požadavek (zprávy "home test" a "care-of test") a přidá k nim hodnoty odvozené od svého tajného klíče. Mobilní uzel z obou těchto zpráv odvodí klíč, který bude použit pro registraci u korespondujícího uzlu. Mobilní uzel tak prokáže schopnost přijímat pakety z obou směrů, tedy jak od domovského agenta tak přímo na obslužné adrese. Tento mechanismus v sobě zahrnuje ochranu proti replay útoku, ale není odolný proti útočníkům na cestě mezi domovským agentem a korespondujícím uzlem. Takoví útočníci ale dokáží ohrozit komunikaci i bez použití mobility.

c) Komunikace

Komunikace mezi mobilním a korespondujícím uzlem je možná ve dvou režimech. První režim "obousměrný tunel" používá jako prostředníka domovského agenta. Domovský agent musí na sebe přesměrovat pakety poslané na domovskou adresu mobilního uzlu. Využije k tomu mechanismus objevování sousedů, který definuje IPv6. Takto odchycené pakety pošle prostřednictvím IP tunelu na obslužnou adresu mobilního uzlu. Pakety v opačném směru používají stejnou cestu, tedy IP tunel na domovského agenta a odtud už klasicky až ke korespondujícímu uzlu. Tento režim se použije v případě, že korespondující uzel nepodporuje mobilitu a nebo mu mobilní uzel nechce odhalit svojí lokaci.

Druhý režim "optimalizace cesty" je šetrnější k síťovým zdrojům. Poté, co

mobilní uzel zaregistruje svoji obslužnou adresu u korespondujícího uzlu, může komunikace probíhat přímo. Korespondující uzel si zanese vazbu domovská adresa – obslužná adresa do své cache. Při odeslání paketu na domovskou adresu mobilního uzlu opatří paket speciálním typem IP hlavičky (tzv. směrovací hlavička typu 2). Ta specifikuje, že daný paket má být směrován nejprve na obslužnou adresu mobilního uzlu. Mobilní uzel tedy takový paket příjme a předá aplikaci jako by přišel na domovskou adresu. Pakety v opačném směru jsou odesílány podobným způsobem. Adresa odesílatele paketu je nastavena na obslužnou adresu a do pakety je přidán speciální položka „home address option“, která obsahuje domovskou adresu mobilního uzlu.

Na rozdíl od IPv4 nedochází v IPv6 k situaci, kdy pakety v jednom směru putují jinou cestou než pakety ve směru opačném (triangle routing). Ani k situaci, kdy pakety mají "falešnou" adresu odesilatele a hrozí nebezpečí, že budou zahozeny některými směrovači (ingress filtering).

d) Přesun do jiné sítě

Přesun do jiné sítě znamená pro mobilní uzel dva kroky. První krok je vlastní detekce přesunu. Pokud není informace dodána přímo linkovou vrstvou, může být detekce založena na mechanismu detekce nedostupnosti sousedů v IPv6. Mobilní uzel detekuje nedostupnost výchozího směrovače a spustí mechanismus k jeho objevení.

V druhém kroku provede registraci se svojí novou obslužnou adresou. Tím dojde k přepsání původního záznamu. Pokud mobilní uzel v prvním kroku zjistí, že je připojen ve své domovské síti, provede registraci s dobou životnosti nastavenou na nula. To způsobí vymazání záznamů.

Je třeba obnovit všechny registrace k domovskému agentovi i ke všem korespondujícím uzlům.

3.2.2 Bezpečnostní otázky mobilního IPv6

K bezpečnostním otázkám je v mobilní implementaci IPv6 přistupováno zodpovědněji než tomu bylo v IPv4. V návrhu standardu jsou diskutovány možné bezpečnostní hrozby:

- Hrozby vyplývající z procesu registrace u domovského agenta nebo u korespondujícího uzlu. Útočník by mohl tvrdit, že daný mobilní uzel se nachází jinde než tomu je ve skutečnosti. To otevírá cestu útokům proti důvěryhodnosti, integritě nebo dostupnosti mobilního uzlu. Tyto hrozby blíže diskutuje článek [16] nebo [17].
- Hrozby spojené s užitečným nákladem paketů. Útočník například může využít nových směrovacích hlaviček typu dvě k obejití ingress filtrů, tzv. reflection attacks. Podrobněji diskutováno v [18].
- Hrozby spojené s mechanismem objevování domovských agentů a prefixů domovské sítě, pomocí nichž může útočník zjistit senzitivní

informace o síti.

- Hrozby pro vlastní bezpečnostní mechanismy protokolu. V tomto případě útočník donutí cílový uzel počítat náročnou kryptografickou funkci nebo alokovat velké množství paměti. Postižený uzel poté nebude mít dostatek zdrojů pro obsluhu oprávněných požadavků.

Tyto hrozby se protokol snaží minimalizovat. Základní bezpečnostní rysy mobilního IPv6:

- Povinné použití zpětného tunelu.
- Ochrana procesu domovské registrace.
- Ochrana procesu registrace u korespondujícího uzlu.
- Ochrana proti reflection attacks.
- Ochrana tunelu mezi domovským agentem a mobilním uzlem.
- Ochrana proti zneužití směrovacích hlaviček.
- Minimalizace hrozeb typu DoS proti vlastním bezpečnostním mechanismům protokolu.

Použití zpětného tunelu je důležité například v případě, kdy komunikujeme s uzlem ve své domovské sítí. Odstranění triangle routingu celkově zvyšuje bezpečnost protokolu. Proces domovské registrace je chráněn pomocí ESP hlavičky. Registrace u korespondujícího uzlu je zas chráněna pomocí RRP. Cílem RRP není chránit proti útokům, které by byly proveditelné bez použití mobility, ale pouze nepřidávat další bezpečnostní rizika spojená s použití mobilního IP. Její výhodou je také to, že na korespondujícím uzlu není vytvářen žádný stav až do doby, dokud není proces registrace kompletně dokončen. To minimalizuje možnosti útoku typu DOS proti tomuto mechanismu.

Pro použití směrovacích hlaviček a volby „home address option“ jsou v protokolu stanovena jasná pravidla. Každý uzel by měl při přijetí paketu tyto pravidla ověřit a zabránit tak jejich zneužití. Například uzel nepřijme paket s volbou „home address option“, pokud nezná příslušnou vazbu.

Tunel mezi domovským agentem a mobilním uzlem je vhodné ochránit pomocí ESP hlavičky a omezit tak možnosti útočníků na cestě mezi domovskou sítí a mobilním uzlem. Bližší diskuzi o nasazení IPsec v signalizaci mezi domovským agentem a mobilním uzlem obsahuje RFC 3776 [19].

Výsledky srovnání mobilního IPv6 oproti běžné IPv6 komunikaci:

- Útočníci, kteří se nenacházejí na cestě mezi domovskou síti a korespondujícím uzlem, nemají více možností jak napadnout komunikaci dvou uzlů.
- Útočníci, kteří se nacházejí v domovské síti nebo na cestě mezi domovskou sítí a korespondujícím uzlem, mají přibližně stejné možnosti jak napadnout komunikaci dvou uzlů. Jediným rozdílem je to, že útočník nemusí být připojen na cestě paketů neustále, ale pouze na krátký čas,

aby zfalšoval proces registrace a pak může svůj útok dokončit odjinud.

Proti naposled zmíněné zranitelnosti se mobilní IP brání tak, že omezuje dobu životnosti registrace a dobu mezi registrací a posledním provedením RRP.

3.2.3 Problémy mobilního IPv6 a jejich řešení

Většina problémů, s kterými jsme se potýkali v IPv4, vzala v IPv6 za své. Cizí agent v IPv6 neexistuje a argument, že je to plýtvání IP adresami je v IPv6 neopodstatněný. Design protokolu také vylučuje problémy s filtry na základě adresy odesilatele a s filtry firewallů koncové sítě. S nasazením NATu v IPv6 sítích se také nepočítá. Vzniká sice soubor doporučení jak dosáhnout funkcionality podobné NATu v IPv6, ale ty nepředstavují pro mobilitu problém. Otázka optimalizace cesty je také uspokojivě řešena. Jediný problém, který zbývá ještě řešit, je rychlost přechodu do jiné sítě.

a) Rychlost přechodu do jiné sítě.

Otázka rychlosti přechodu do jiné sítě je v IPv6 o trochu naléhavější. Do času potřebného na přechod do jiné sítě a obnovení komunikace se vzdáleným uzlem je potřeba počítat: vlastní detekci přesunu, získání nové obslužné adresy, registrace u domovského agenta, provedení RRP a nakonec registraci u korespondujícího uzlu.

První RFC, které se zabývá tímto problémem, je ještě relativně čerstvé (RFC 4068 [20]) a v době psaní těchto řádků bylo označeno za experimentální. Popisuje mechanismus nazvaný „rychlé předávky“[7], který je postaven na spolupráci přístupových bodů cizí sítě, přičemž přístupovým bodem je zde myšlen přístupový uzel bezdrátové sítě. Návrh předpokládá, že přístupový bod bude mobilnímu uzlu zprostředkovávat informace o sousedních přístupových uzlech a urychlí tak případnou detekci přesunu a získaní nové obslužné adresy pro mobilní uzel. Navíc, díky možnosti tunelu mezi původním a novým přístupovým bodem, lze zajistit, že během přesunu nedojde ke ztrátě paketů.

Jiný přístup k řešení tohoto problému přináší RFC 4140 [21], které popisuje hierarchický management. Zavadí do mobility nový prvek, který nazývá mobilní záchytný bod[8] (MAP).

- **Mobilní záchytný bod (MAP)** – je směrovač umístěný v cizí síti. Pro mobilní uzel slouží jako lokální domovský agent.

Jeden MAP pokrývá několik hraničních směrovačů, ke kterým se připojují mobilní zařízení. Mobilní uzel získá od hraničního směrovače připojené sítě tzv. linkovou obslužnou adresu[9] a seznam uzlů MAP, které může využít. Linková obslužná adresa je v podstatě to, co jsme v původně nazývali jen obslužná adresa. Zde je použít jiný název, protože se tu objevuje i jiný druh obslužné adresy.

7 V originále „fast handovers“
8 V originále „mobility anchor point“
9 V originále „on-link care-of address“

Pokud mobilní uzel nechce využívat výhod hierarchického managementu, může se s linkovou obslužnou adresou registrovat u svého domovského agenta a vše funguje. V opačném případě se může registrovat u jednoho z uzlů MAP, kde získá regionální obslužnou adresu[10]. S regionální obslužnou adresou se pak registruje u svého domovského agenta i u všech korespondujících uzlů. Pakety mezi korespondujícími uzly a mobilním uzlem jsou pak tunelovány na MAP a odtud pak dalším tunelem na linkovou obslužnou adresu k mobilnímu uzlu. Výhodou tohoto přístupu je, že přesun k jinému směrovači neznamená změnu regionální obslužné adresy a nevyžaduje přeregistraci u všech korespondujících uzlů. Jediné, co musí mobilní uzel udělat, je aktualizovat vazbu mezi regionální a linkovou obslužnou adresou. Regionální obslužná adresa je vlastně cosi jako domovská adresa v cizí síti.

Tento mechanismus dovoluje značnou variabilitu. Mobilní uzel získá také informaci o vzdálenosti k jednotlivým uzlům MAP a může zvolit nejoptimálnější. Tvůrci návrhu také počítají s možností registrovat se u více MAP a použít různou regionální obslužnou adresu pro komunikaci s různými skupinami korespondujících uzlů. Například při komunikaci s uzlem na stejné síti jako mobilní uzel se použije nejbližší MAP uzel. Naopak pro komunikaci se vzdáleným uzlem je výhodnější použít MAP, který se nachází dál.

Použitím hierarchického managementu může mobilní uzel také částečně utajit svoji fyzickou pozici, aniž by se musel vzdát výhod optimalizace cesty.

Nevýhodou je samozřejmě potřeba dalšího prvku na straně cizí sítě, který musí podporovat mobilitu.

Nový mechanismus sebou nese i nová bezpečnostní rizika. Pro ochranu regionální registrace je potřeba ustanovit bezpečnostní asociaci mezi mobilním uzlem a MAP. Za prvé mobilní uzel potřebuje vědět, že MAP, ke kterému se chce registrovat, je důvěryhodný. Toho lze dosáhnout tím, že MAP se bude prokazovat certifikátem podepsaným důvěryhodnou certifikační autoritou.

Na straně MAP není potřeba nijak ověřovat první požadavek o regionální registraci od mobilního uzlu, regionální adresa je stejně jenom dočasná. Ověření je potřeba až v případě, kdy se mobilní uzel přesune jinam a chce aktualizovat svojí vazbu. MAP potřebuje vědět, že je to stejný uzel, který předtím tuto vazbu zavedl. Pro tento účel lze použít IPsec s automatickou distribucí klíčů (IKE).

Oba tyto mechanismy lze samozřejmě s výhodou kombinovat.

3.2.4 Zhodnocení mobilního IPv6

a) Zátěž adresačního mechanismu

Značnou výhodou IPv6 je skutečnost, že blok IP adres, který je přiřazen koncové síti je skutečně obrovský. Protokol pro mobilitu není omezen počtem

10 V originále „regional care-of address“

volných IP adres a může si tak dovolit vynechat cizího agenta. To zjednoduší návrh protokolu a odstraní prvek, který může být potencionálně nebezpečný. Pokud se ale snažíme nějak minimalizovat čas na přesun do jiné sítě, tak se nějakému spolupracujícímu prvku nevyhneme. Jeho úloha je však odlišná od úlohy cizího agenta v IPv4. Tuto službu zajišťuje jednak hraniční router, který umožňuje rychlé předávky. To by měla být standardní služba bezdrátových sítí. A jednak mobilní záchytný bod, na který se lze dívat jako na chytrý směrovač, který směruje mezi regionální a linkovou obslužnou adresou.

Díky zdokonalenému a již skutečně použitelnému mechanismu optimalizace cesty je režie spojená se správným směrováním paketů distribuována na korespondující uzly a na vlastní mobilní. Hlavní úkol domovského agenta je na začátku komunikace, kdy musí umožnit provedení RRP. Lze se tedy na něj dívat jen jako na prostředníka, který potvrdí identitu uzlu.

Naopak uzel, který slouží jako MAP, musí vykonávat hodně činností související se směrováním. Přes něj jdou veškeré pakety pro mobilní uzel, musí si udržovat vazbu mezi regionální a linkovou obslužnou adresou a zajistit správné směrování paketů na mobilní uzel. A to všechno pro potencionálně velké množství uzlů. Díky možné variabilitě při použití hierarchického managementu lze uvažovat o zpoplatnění služeb MAPu v komerčních sítích nebo jeho použití jen pro kritické aplikace typu Voice over IP.

Nutno ale podotknout, že hierarchický management je ve stádiu návrhu standardu, který ještě může doznat určitých změn.

b) Zabezpečení

Problém zabezpečení mobilního IPv6 je diskutován v mnoha článcích (viz. Sekce 3.2.2) i v samotném RFC. Design protokolu se nespoléhá pouze na použití bezpečnostních mechanismů běžných v IPv6 (IPsec, IKE), ale přináší vlastní vylepšení v podobě autentifikace na základě směrování, tedy RRP. Tato autentifikace je postavena na předpokladu, že pro útočníka je nemožné odchytit pakety, které neprochází přes jeho uzel. Aby mohl útočník ohrozit RRP musí tedy nejprve kompromitovat některý směrovač mezi komunikujícími uzly. V takovém případě je útočník nebezpečný pro veškerou komunikaci putující přes tento směrovač, tedy i pro klasické IPv6 konexe. Nespornou výhodou RRP je možnost autentifikace mobilního uzlu bez potřeby budování globální bezpečnostní infrastruktury.

Důležitým aspektem bezpečnosti protokolu je i jeho implementace. Zvlášť při použití některých nových hlaviček a voleb v IP paketech by mohl útočník obelstít některou nedokonalou implementaci. Proto už vlastní standard obsahuje podrobný popis toho, jak se má uzel zachovat v různých situacích a jaké všechny předpoklady musí být splněny aby byl přijatý paket zpracován.

c) Požadavky na přenosovou síť

Požadavky na přenosovou síť jsou ještě o něco menší než v případě IPv4. Za to můžou především nové směrovací hlavičky, které nahrazují tunel z IPv4. Díky

dokonalejší metodě optimalizace cesty a vypuštěním cizího agenta dochází i ke zmenšení zpoždění při přenosu paketů. Ta se může dostat v optimálním případě až na hodnotu stejnou jako bez použití mobility.

d) Přínos mobility v dané síti

Výhody mobility v Internetu jsme již rozebírali v části 3.1.4.d. Díky velkému adresnímu prostoru lze předpokládat, že se v IPv6 mobilní uzel dostane častěji do role serveru a mobilita IP se tak pro něj stane nepostradatelnou.

Ale i na straně klienta může mít pevná adresa své výhody. Tvůrci dnešních internetových aplikací nejsou příliš zvyklí na to, že se mohou spolehnout na pevnou adresu uzlů. Naopak, většina aplikací musí být navržena tak, aby byla schopna pracovat v sítích za NATem a s dynamickou adresou klienta. Pevná IP by návrh těchto aplikací zjednodušila.

V IPv6 s podporou mobility není problém, aby měl koncový uzel k dispozici více domovských adres s různými síťovými prefixy a vystupoval tak v internetu pod vícero identifikacemi. Mobilita by mohla v budoucnu nahradit i připojení pomocí VPN ke vzdálené síti. Jediné na co si je potřeba dát pozor, je nutnost použít šifrování při přístupu ke zdrojům ve vzdálené síti. Mobilita otázku šifrování vlastního přenosu neřeší a aplikace v tomto případě nepozná, jestli komunikuje pouze přes lokální síť nebo přes Internet.

e) Shrnutí

Mobilita je uváděna jako jedna z výhod IPv6 a je sním neodmyslitelně spjata. Je navržena tak, aby využívala potenciál, který IPv6 nabízí. Autoři si zřejmě uvědomovali nevýhody, které sebou nese mobilita v IPv4. Jejich snahou bylo se těmto nevýhodám vyhnout. To se jím do jisté míry podařilo a mobilita se stala jedním z lákadel pro přechod na IPv6.

3.3 Jmenné domény a End-to-End přístup k řešení mobility

Předchozí dva přístupy řešily mobilitu na úrovni síťové vrstvy. Za neměnný identifikátor mobilního uzlu považovaly jeho domovskou IP adresu, která se pro protokoly vyšší vrstvy (TCP,UDP,..) jevila transparentně.

Existuje ale i jiný přístup, který řeší mobilitu na koncových uzlech a nevyžaduje žádný zásah do IP infrastruktury (tzv. End-to-End přístup). Tento přístup dovoluje mobilním uzlům měnit IP adresu kdykoliv to vyžaduje změna jejich přípojného bodu do internetu. IP adresu už tedy nelze použít jako pevný identifikátor mobilního uzlu a je potřeba hledat nové metody jak identifikovat mobilní uzel. Přirozený kandidát pro tuto funkci je jméno zanesené v systému DNS. Toto jméno je snáze zapamatovatelné pro člověka a již dnes ho lze ve většině aplikací použít jako identifikátor internetového uzlu. Prvním úkolem by tedy mělo být zajistit, aby proces převodu doménového jména na IP adresu vždy vrátil aktuální adresu mobilního uzlu (lokalizování uzlu).

Druhým problémem je, co s navázaným spojení v případě, kdy jeden z

komunikujících uzlů změní svoji IP adresu. TCP konexe je identifikována čtveřicí <zdrojová adresa, zdrojový port, cílová adresa, cílový port>. Pokud se jeden z těchto údajů změní, spojení nemůže dále pokračovat. Tento problém řeší rozšíření TCP protokolu, které dovoluje uzlu převzít své předchozí navázané spojení z jiné IP adresy (migrace spojení). Obdobné rozšíření lze aplikovat i na jiné protokoly transportní vrstvy.

End-to-End přístup předpokládá, že aplikace o podpoře mobility vědí a samy se rozhodnou jestli ji chtějí využít. Existuje několik tříd mobilních aplikací s různými požadavky:

- Ostatní uzly se k ním připojují (mobilní web servery, mobilní telefony,..) - mají užitek z obojího: lokalizování uzlu i migrace spojení.
- Připojují se k jiným uzlům (poštovní klienti, webové browsery,..) – mají užitek převážně z migrace spojení.
- Provádějí krátké transakce, které mohou být v případě neúspěchu opakovány na aplikační vrstvě (DNS dotazy, …) - nevyužijí ani jednu metodu.

Mobilní IP, které bylo rozebíráno v části 3.1 a 3.2, poskytovalo mobilitu všem aplikacím bez ohledu na jejich potřeby. End-to-End přístup umožňuje aplikacím zvolit způsob řešení mobility, případně ji vůbec nepoužít a snížit tak režii komunikace.

Limity End-to-End přístupu

Jedním ze zajímavých požadavků na mobilitu je zajistit plynulé doručování. Výpadek v doručování paketů by neměl přesáhnout 50ms. V případě, že přenášíme zvukový stream, je 50ms výpadek hranice, kterou ještě není lidské ucho schopné rozpoznat. Je ale možné tento hodně tvrdý požadavek v End-to-End přístupu zajistit ?

Uvažme následující modelovou situaci: Uzel A komunikuje s uzlem B. Komunikace se odehrává po trase *P1* a round-trip-time trasy *P1* je t_{C1}. V určitém okamžiku se uzel B přesune na jinou pozici, novou komunikační trasu mezi uzlem A a B nazveme *P2* a její round-trip-time t_{C2}. Čas za jaký se uzel A dozví novou pozici uzlu B označíme T_P (čas propagace změny). Rozlišme čtyři situace:

1. Uzel B ztratí okamžikem přechodu možnost komunikovat po trase P1. Vlastní komunikace se odehrává nespojitě a nespolehlivě. (např. UDP).

   Uzel A vysílá pakety směrem k uzlu B po trase P1. Jakmile se dozví o nové pozici uzlu B, začne vysílat po trase P2. Tedy:

   - Doba, po kterou nejsou žádné pakety doručovány uzlu B: $T_V = T_P + T_{C2}$
   - Doba, po kterou jsou pakety ztráceny: $T_Z = T_P + T_{C1}$
2. Uzel B ztratí okamžikem přechodu možnost komunikovat po trase P2. Vlastní komunikace se odehrává spojitě a spolehlivě (TCP).

Uzel A vysílá pakety směrem k uzlu B po trase P1. Jakmile se dozví o nové pozici uzlu B, začne s migrací spojení a poté začne vysílat po trase P2. Dobu nutnou na migraci spojení označíme T_M. Tedy:

- Doba, po kterou nejsou žádné pakety doručovány uzlu B: $T_V = T_P + T_M + T_{C2}$
- Doba, po kterou je nutné ukládat pakety na A pro pozdější odeslání: $T_O = T_P + T_M + T_{C1}$

3. Uzel B má ještě po dobu T_D po přechodu možnost přijímat pakety cestou P1. Vlastní komunikace se odehrává nespojitě a nespolehlivě (UDP).

   Uzel A se chová stejně jako v případě 1.

   - Doba, po kterou nejsou žádné pakety doručovány uzlu B:

     $T_V = když(\ T_{C2} > T_{C1})\ tak\ T_{C2} - T_{C1}\ \ jinak\ 0$

   - Minimální doba T_D , aby nedošlo ke ztrátě paketů: $min(T_D) = T_P + T_{C1}$

4. Uzel B má ještě po dobu T_D po přechodu možnost přijímat pakety cestou P1. Vlastní komunikace se odehrává spojitě a spolehlivě.

   Uzel A se chová stejně jako v případě 2.

   - Doba, po kterou nejsou žádné pakety doručovány uzlu B:

     $T_V = když(\ T_{C2} > T_{C1})\ tak\ (T_{C2} - T_{C1}) + T_M\ \ jinak\ T_M$

   - Minimální doba T_D , aby nedošlo ke ztrátě paketů: $min(T_O) = T_P + T_{C1}$

   Předpokládáme mírnou úpravu migračního protokolu pro TCP popsaného výše tak, aby byl po migraci schopný přijímat zbylé pakety z minulé IP adresy.

Pro nás je kritický čas T_V, který by neměl přesáhnout zmiňovaných 50ms. Ve variantě 1 a 2 však tento čas závisí na round-trip-time jedné z cest internetem a ta může být obecně větší než 50ms. Požadavek na výpadek přenosu pod 50ms tedy nelze zajistit. Ve variantě 4 je minimální výpadek T_M a uvážíme-li, že TCP migrace vyžaduje přenést 3 pakety (tedy $T_M \sim 3\ T_{C2}$), nelze tento požadavek splnit ani ve variantě 4. Nejblíže splnění tohoto požadavku je varianta 3, kde výpadek závisí na rozdílu časů jednotlivých cest, ale ani zde nemůžeme mít stoprocentní jistotu. **Závěr je, že při End-to-End přístupu nelze požadavek plynulého doručování paketů stoprocentně zajistit.**

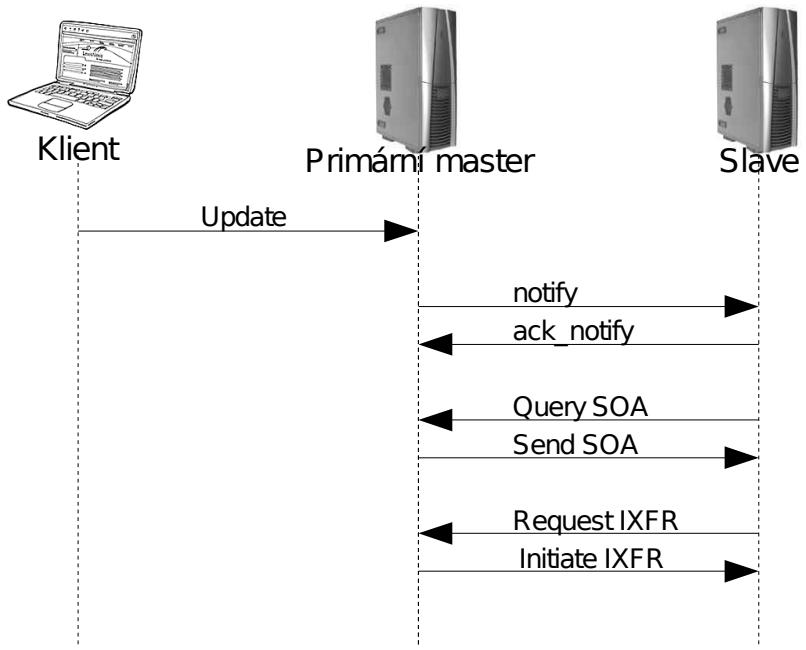
Čas na propagaci změny se na celkovém čase výpadku doručování paketů podílí jen ve variantě 1 a 2. Ve variantě 3 a 4 ovlivňuje tento čas technické parametry spojení (jak dlouho je třeba držet dvě linky).

V optimálním případě by čas T_P měl být rovný času T_{C2}. To je čas, za který může uzel A informovat uzel B o změně své polohy v případě přímé komunikace.

3.3.1 Lokalizování uzlu

Domain Name System (DNS [22]) je distribuovaná databáze, která zabezpečuje převod mezi hierarchickým jménem a IP adresou[11]. Pro každou část databáze (zóna) existuje skupina jmenných serverů, které udržují informace o této části (autoritativní jmenné servery). Jeden z autoritativních jmenných serverů je master a ostatní jsou slaves. Pouze master server má právo dělat změny v zóně a ty pak distribuuje na podřízené servery.

Tento mechanismus byl původně navržen jako statický, jakékoliv změny v zóně bylo potřeba ručně přepsat na primárním serveru. Až později byla přidána možnost dynamicky měnit záznamy v DNS [23]. Princip tohoto mechanismu spočívá v tom, že klient pošle žádost o update autoritativnímu serveru své zóny, tato žádost je předána primárnímu master serveru. Master server provede změny a pošle notifikaci o změnách všem svým podřízeným serverům. Podřízené servery by měli zahájit transfer zóny, aby získaly její aktuální verzi. Celý proces ilustruje Obrázek 5.



Obrázek 5: Sekvence zpráv pro dynamický DNS update.

Standard nenařizuje master serveru odeslat notifikaci okamžitě po provedení změn a ani slave server není povinen okamžitě po obdržení notifikace zahájit transfer zóny. Většinou master server čeká asi minutu mezi každým odesláním notifikace. Důvod tohoto chování je snaha limitovat počet souběžných přenosů zón a kontrolovat tak zatížení serveru.

Tento mechanismus nedokáže zajistit dostatečně rychlou propagaci změn. Pro

11 DNS má ve skutečností více funkcí ale v tomto kontextu se budeme zabývat pouze touto funkčností.

účely mobility však potřebujeme aby změny v DNS byli propagovány pokud možno okamžitě. Pappas ve své práci [24] uvádí několik možností pro vylepšení tohoto mechanismu směrem ke zrychlení:

1. Master server odesílá notifikace okamžitě po provedení změn. Tato změna není složitá na implementaci, ale zůstává problém se zatížení serveru. V případě jednoho master serveru, který obsluhuje mnoho slave serveru, může zatížení vzrůst podstatným způsobem. Toto řešení volá po nějaké sofistikovanější hierarchii serverů.
2. Jmenný server, který změní svojí zónu, zahájí přesun zóny na všechny své podřízené servery bez toho, aby je předtím informoval notifikační zprávou. Nejlépe s použitím inkrementálního přenosu zóny (IXFR) s jednou nebo několika málo položkami.

Proces převodu doménového jména na IP adresu také v hojné míře využívá cache. Cache ale není explicitně informována o změnách a hrozí tak nebezpečí, že staré informace budou v cache drženy ještě dlouhou dobu. Pomocí TTL (time to live) lze ovlivnit délku držení informací o daném záznamu v cache. Pappas [24] uvádí tři alternativy řešení cachování DNS informací pro mobilní uzly:

1. Zakázat cachování celé zóny. V případě, že DNS server neumožňuje nastavit TTL pro konkrétní záznam, ale pouze pro celou zónu. Způsobíme tím ale zbytečné zatížení autoritativních jmenných serverů.
2. Nastavit TTL pro konkretní záznam podle toho, jestli předpokládáme mobilitu. Dotazy na mobilní uzly nebudou cachovány. Tato metoda je poněkud složitější na správu.
3. Dynamická kriteria pro cachování. TTL stanoveno na základě frekvence updatů tak, aby se minimalizovala šance, že je záznam změněný dřív než vyprší TTL. Zároveň je potřeba řešit i možnost, kdy tento případ nastane, například registrováním cache a její explicitní notifikací. Je možná i spolupráce mobilního uzlu, který pomůže stanovit správné TTL .

Je zřejmé, že proces dynamického update musíme zabezpečit proti neoprávněnému přístupu. Bezpečnostní aspekty DNS řeší několik samostatných RFC a jejich popis je nad rámec tohoto dokumentu. Na tomto místě je nutné podotknout, že se zajištěním bezpečnosti je spojena další režie, která se negativně promítá do rychlosti s jakou je změna propagována a do zátěže, které je server vystaven.

Mnoho poskytovatelů dynamických DNS služeb využívá k update DNS záznamů raději jednoduchý protokol založený na http. Bezpečnost je v tomto případě založena pouze na jménu a heslu, které jsou odesílány v otevřeném tvaru nebo, v lepší případě, přes https. Nejznámějšími zástupci těchto služeb jsou www.dyndns.com a www.no-ip.com. Oba poskytovatelé dokázali zajistit okamžitou propagaci změn do všech autoritativních jmenných serverů. Hodnota TTL u dynamických záznamů byla 60 sekund.

Systém DNS má další problémy. Na vrcholu hierarchie doménových serveru se nachází několik takzvaných kořenových nameserverů. A právě ty jsou dnes

nejzranitelnější místem Internetu. Systém DNS má i další úzká místa. Počet autoritatvních jmených serverů pro doménu je omezen a jejich kompromitací dojde ke kompromitaci celé domény. Podle údajů v [25] je 78,44% domén obsluhováno pouze dvěma nameservery (minimum doporučené ve standardu), a dokonce 0,82% pouze jedním nameserverem. Častý je případ, kdy oba nameservery jsou za jedním směrovačem. Jeho kompromitací pak dojde k vyřazení obou nameserverů.

3.3.2 Migrace spojení

Snoeren a Balakrisham ve své práci [26] navrhují rozšíření TCP protokolu, které by umožnilo migraci TCP spojení v případě, že jeden z komunikujících uzlů změní IP adresu. Mobilní uzel může restartovat dříve navázané spojení z nové IP adresy. Využije k tomu speciální Migrate SYN paket, který obsahuje token identifikující předchozí spojení. Následuje běžný TCP handshake, s tím rozdílem, že se nevytváří nové spojení, ale dochází pouze k synchronizaci spojení s novou koncovou IP a pokračuje se v původní komunikaci. Stav spojení, včetně prostoru sekvenčních čísel, je během migrace zachován, případné nedoručené pakety jsou přeposlány.

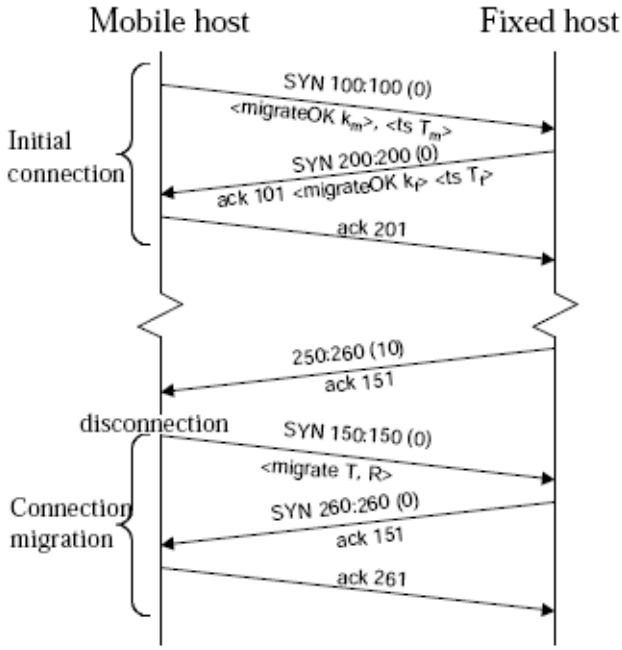
Token identifikující spojení je dohodnut už při prvním navázání spojení, tedy při prvním TCP handshaku. Už při navázání spojení tedy dochází k rozhodnutí, jestli se bude TCP migrace používat.

Příklad migrace TCP spojení ilustruje Obrázek 6. V první části mobilní uzel naváže spojení se statickým uzlem. Úvodní SYN paket je rozšířen o volbu migrate-permitted, parametry k_m a T_m slouží k výpočtu tokenu. Pokud statický uzel podporuje TCP migraci, uvede v odpovědi stejnou volbu. Spojení pokračuje standardním způsobem. Za nějaký čas změní mobilní uzel svojí IP adresu a zahájí proces migrace. Odešle statickému uzlu SYN paket obsahující volbu migrate request s parametrem obsahující token, který byl předtím spočítán. Proces dále pokračuje stejně jako standardní TCP handshake.

a) Zabezpečení migrace

V klasickém TCP má útočník, který odhadne sekvenční čísla, možnost částečně převzít spojení. Pokud povolíme migraci, má útočník, který odhadne sekvenční čísla a migrační token, možnost převzít spojení kompletně. Navíc některé metody, které mají chránit spojení proti útočníkům nacházejícím se na cestě paketů, jsou v tomto případě neúčinné. Proto musí být věnována velká pozornost zabezpečení migračního tokenu.

Jedna možnost je použít zabezpečení, které už nabízí síťová vrstva (IPsec). Při správném použití poskytuje dostatečné zabezpečení komunikace. Bezpečnostní asociace, které využívá IPsec, jsou založeny na IP adresách, proto je nutné před obnovením spojení ustanovit novou bezpečnostní asociaci.



Obrázek 6.: Migrace TCP spojení.

Protože s použitím IPsecu vzniká další režie, tak autoři TCP migrace nabízejí také vlastní metodu, která je založena na algoritmu Diffie-Hellmanově výměně klíčů s použitím eliptických křivek a algoritmu SHA1. Uzly, kteří používají IPsec, můžou tuto výměnu zakázat a ušetřit výpočetní zdroje.

Metoda založená na eliptických křivkách byla zvolena proto, že má nejlepší poměr počtu bitů vůči síle šifry. V hlavičce TCP paketu můžeme totiž přenést omezené množství dat.

3.3.3 Zhodnocení

End-to-End přístup nepožaduje žádný další prvek uvnitř sítě, který by se o mobilitu staral. To má význam s ohledem na bezpečnost. Není nutné ustanovovat nějaký důvěryhodný vztah s neznámým prvkem v síti. Jediný prvek v síti, který musí být mobilnímu uzlu nápomocen, je DNS server. Vlastní mobilní přístup si zajišťuje mobilní uzel ve spolupráci se svým protějškem.

Největší nevýhodou je nutnost měnit TCP stack na koncových stanicích. Tedy nejen na koncových uzlech ale i na statických, které o mobilitě nechtějí nic vědět. Na straně mobilního uzlu je potřeba změnit dokonce i aplikace, které chtějí mobilitu využívat.

DNS infrastruktura také není připravena na rychlé propagování změn do celého Internetu. Tato oblast by si zasloužila podrobnější výzkum.

Metoda mobility na transportní vrstvě je sice přínosná a otvírá nové možnosti, ale díky posledním dvou nevýhodám má malé šance prosadit se v praxi.

3.4 Využití distribuovaných hašovacích tabulek

3.4.1 Motivace

Mobilní IP vychází z předpokladu, že každý mobilní uzel je někde „doma“ a občas vyrazí na „výlet“ do jiných IP sítí. Nikdy však neztratí kontakt se svojí domovskou sítí a je vždy k zastižení pod svojí domovskou adresou. Nabízí se paralela s reálným světem. Na IP adresu se můžeme dívat jako na poštovní adresu, kam nám chodí dopisy. Pokud jsme na dovolené, požádáme domovníka, aby nám všechny dopisy přeposílal na adresu hotelu, kde svoji dovolenou trávíme.

Toto řešení je neúčinné v případě, že bydlíme v karavanu a každý den jsme někde jinde. Nemáme k dispozici žádnou domovskou adresu a ani žádného domovníka, který by hlídal, kde zrovna jsme. V tomto případě by bylo výhodnější použít jakýsi interaktivní adresář, do kterého budeme pravidelně zapisovat svoji aktuální adresu a každý si nás tam bude moc najít. Nelze tedy jako náš jednoznačný identifikátor použít poštovní adresu (IP adresu), ale je potřeba vymyslet něco jiného.

Řešení, popsané v kapitole 3.3.1, využívá k tomuto účelu jméno, uložené v systému DNS. Systém DNS však nebyl navržen na dynamické změny a jména uzlů, která se v něm používají, byla navržena tak, aby byla snadno čitelná pro člověka. Z tohoto důvodu se nezdá příliš vhodné použít tato jména jako jednoznačný identifikátor uzlu.

Zkusme se podívat na celý problém více obecně. Předpokládejme, že každý uzel má svoji jednoznačnou číselnou identifikaci. Pro tyto účely se dá použít MAC adresa síťové karty, případně IMEI mobilního telefonu. Tato identifikace je nerozlučně spjata s konkrétní identitou zařízení v rámci sítě a nevykazuje žádnou hierarchickou strukturu. Naopak na IP adresu se díváme jako na informaci o poloze. Každá IP adresa je nerozlučně spjata s konkrétní pozicí v rámci topologie sítě. Systém DNS pak můžeme chápat jako rozhraní, které zpřístupňuje některé informace ve formě čitelné pro člověka. DNS jména jsou chápána jako logická jména (např. jména virtuálních serverů), která nemusí být spjata s konkrétním zařízením. Data v DNS jsou spíše statického rázu.

Tím redukujeme problém mobility na problém převodu mezi identifikací zařízení a identifikací polohy. Tento převod musí být dostatečně dynamický, aby dokázal reagovat na rychlé změny polohy mobilního zařízení a zároveň bezpečný, aby znemožnil odcizení identity.

Zavedeme tyto zkratky:

- NodeID – jednoznačná, neměnná, číselná identifikace uzlu (např. MAC adresa)..
- LocID – identifikace konkrétní polohy v síti. (např. IP adresa).
- Name – jméno spojené s uzlem, forma čitelná pro člověka.

Za těchto předpokladů můžeme k mobilitě přistupovat dvěma způsoby:

1. End-to-End přístup: Síť poskytuje službu převodu NodeID na aktuální LocID. Pokud chce uzel komunikovat s jiným uzlem, zjistí si jeho aktuální LocID a na jeho základě naváže se vzdáleným uzlem spojení. Pokud dojde ke změně LocID vzdáleného uzlu, je navázané spojení převedeno na novou lokaci stejným způsobem, jaký je popsán v kapitole 3.3.2.
2. Route-Based přístup: NodeID je použito jako jediná informace o cíli v hlavičce odesílaného paketu. O vlastní převod mezi NodeID a LocID se starají směrovače v sítí. Ty zajistí, aby byl paket doručen na aktuální polohu cílového uzlu. V tomto případě by se pro převod mezi Name a NodeID použil systém DNS.

O dynamický převod mezi NodeID a LocID se pokusíme pomocí techniky zvané Distributed Hash Table (DHT).

3.4.2 Distribuované hašovací tabulky

Distribuované hašovací tabulky je třída decentralizovaných distribuovaných systémů, které rozdělují vlastnictví množiny klíčů mezi účastnické uzly a umožňují efektivně směrovat zprávy k vlastníkovi daného klíče. Každý uzel je analogie oblasti v klasické hašovací tabulce. DHT jsou typicky navrhovány tak, aby dokázaly pracovat s velkým počtem uzlů a umožňovaly kdykoliv připojit nebo odpojit uzel.

DHT vznikaly pro prostředí peer-to-peer sítí. Měly zajišťovat sdílení většího množství dat mezi uživateli bez centrálního serveru. Většina DHT používá nějakou variantu konzistentního hašování pro mapování klíčů na uzly. Konzistentní hašování nabízí funkci $\delta(k_1,k_2)$, kterou lze chápat jako vzdálenost mezi klíčem k_1 a klíčem k_2. Každému uzlu, který se účastní DHT, je přidělen klíč, který uzel identifikuje. Klíč k je přidělen tomu uzlu, jehož identifikace je nejblíže (ve smyslu funkce δ) klíči k. Tato vlastnost zajistí, že v případě připojování nebo odpojování uzlů z DHT je potřeba přeřadit jen malou skupinu klíčů, které jsou v okolí daného uzlu.

V každé DHT topologii by mělo platit, že daný uzel je buď vlastník klíče k, nebo má spojení na uzel s identifikací, která je v menší vzdálenosti od klíče k. Směrování se pak děje podle hladového algoritmu, kdy se v každém kroku přiblížíme k hledanému klíči. Tento způsob se nazývá key based routing.

Základní funkce, které poskytuje DHT jsou:

- Funkce *route(key,msg)* – doručí zprávu uzlu, který je nejblíže klíči *key*.
- Funkce *join(key)* – připojí do DHT nový uzel s identifikací *key*.
- Funkce *leave(key)* – odpojí od DHT uzel s identifikací *key*.

Kromě toho se DHT musí vyrovnat se selháním libovolného uzlu.

3.4.3 Existující technologie DHT

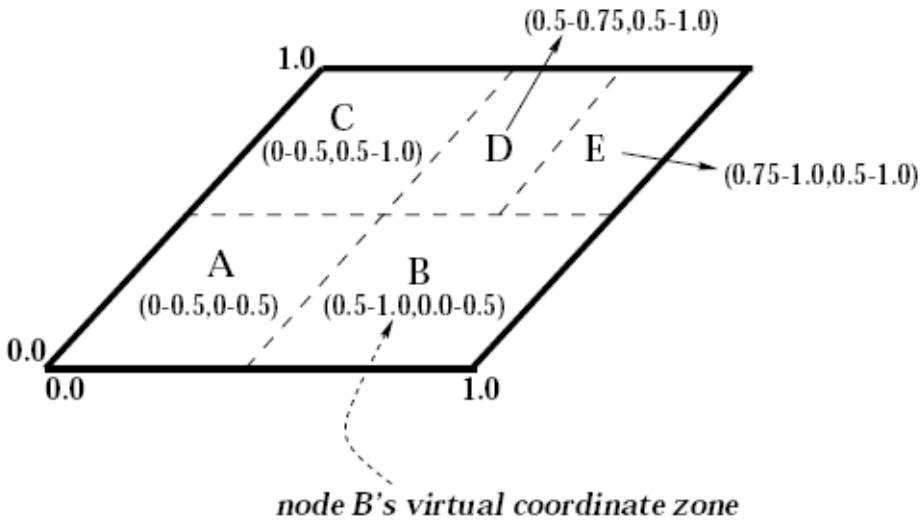
Uvedeme zde čtyři nejrozšířenější typy distribuovaných hašovacích tabulek: CAN, Chord, Pastry a Kademlia.

Budeme sledovat dvě základní vlastnosti algoritmu. Počet stavových informací, které si musí držet každý uzel a průměrný počet přeskoků mezi dvěma uzly. Nejedná se o přeskoky na síťové vrstvě, ale na vrstvě aplikační. Dva sousední DHT uzly mohou být na síťové vrstvě od sebe značně vzdáleny.

Pro většinu DHT technologií se obě tyto hodnoty pohybují okolo $O(log_2 n)$.

a) Content-Addressable Network (CAN) [27]

CAN je postavený na virtuálním d-rozměrném kartézském souřadnicovém systému aplikovaném na d-torus. Tento systém je plně virtuální a nemá žádnou spojitost s fyzickou topologií. Souřadnice bodu v tomto systému představují hašovací klíče. Celý prostor je rozdělen na jednotlivé zóny a každé zóně je přidělen jeden uzel, který je jejím vlastníkem. Příklad CAN v 2-rozměrném prostoru s pěti uzly ukazuje Obrázek 7.



Obrázek 7.: CAN v 2-rozměrném prostoru s pěti uzly.

Úkolem každého uzlu je držet si přímé spojení na uzly, které vlastní sousední zóny. Nejjednodušší směrování v této síti je pak přes zóny, které leží na spojnici zdrojového a cílového bodu. Zpráva je doručena uzlu v jehož oblasti leží cílový bod.

Pokud se do sítě připojuje nový uzel, provede následující kroky:

1. Nalezne nějaký uzel, který je již zapojen do CAN. Tento uzel mu zprostředkovává přístup do sítě.
2. Náhodně zvolí bod v prostoru klíčů, který použije jako svojí identifikaci. Použije směrovací mechanismus v CAN, aby našel uzel, v jehož zóně leží zvolený bod.
3. Dohodne se s nalezeným uzlem na rozdělení jeho zóny a stane se

vlastníkem jedné její poloviny. Upozorní všechny sousední uzly, že nastala změna.

Když se uzel odpojuje od sítě, musí zajistit, že jeho zónu převezme nějaký jiný uzel. Pokud se odpojuje plánovaně, není problém to zajistit. V případě výpadku uzlu musí tento výpadek detekovat sousední uzly a dohodnout se, který z nich převezme uvolněnou zónu.

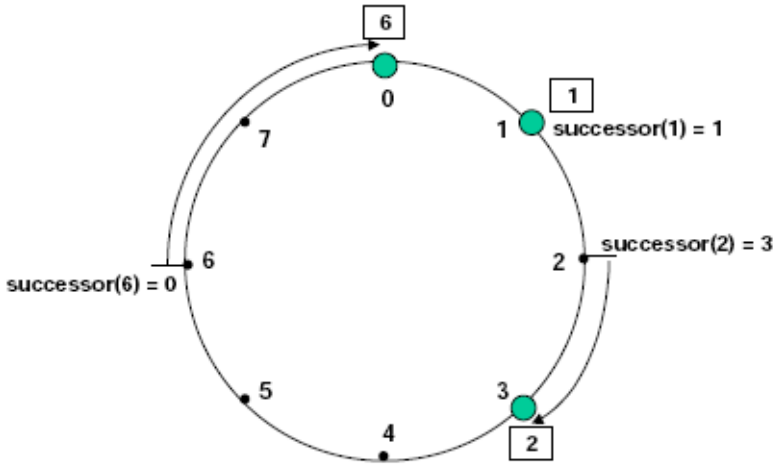
V některých případech může být výhodné aplikovat jiný algoritmus pro připojování uzlů. Případně další metody, které mají zvýšit efektivitu celého systému. Například směrování může brát v potaz i vzdálenost dvou uzlů na IP vrstvě. Některé metody jsou také popsány v článku [27].

Počet stavových informací, které si musí každý uzel držet (počet sousedů), je u CAN $O(d)$. Kde d označuje dimenzi prostoru souřadnic. Průměrný počet přeskoků mezi dvěma uzly je $O(d\, n^{1/d})$. Což není typická hodnota pro DHT síť. Pokud ale volíme $d = (\log_2 n)/2$, dostáváme se k hodnotám, které jsou běžné pro ostatní DHT technologie, tedy $O(\log_2 n)$.

Závislost d na počtu záznamů v síti je samozřejmě nepříjemná. Pro velké sítě s velkým počtem záznamů nepřijatelná.

b) Chord [28]

Klíče v tomto algoritmu jsou m-bitové číselné identifikátory a jsou uspořádány do kružnice. Klíč k je přiřazen k prvnímu uzlu, jehož identifikátor je rovný nebo větší k. Tento uzel je označován jako následník k a značíme ho *successor(k)*. Je to tedy první uzel ve směru hodinových ručiček od identifikátoru k. Situaci pro m=3, uzly 0,1,3 a klíče 1,2,6 ilustruje Obrázek 8.

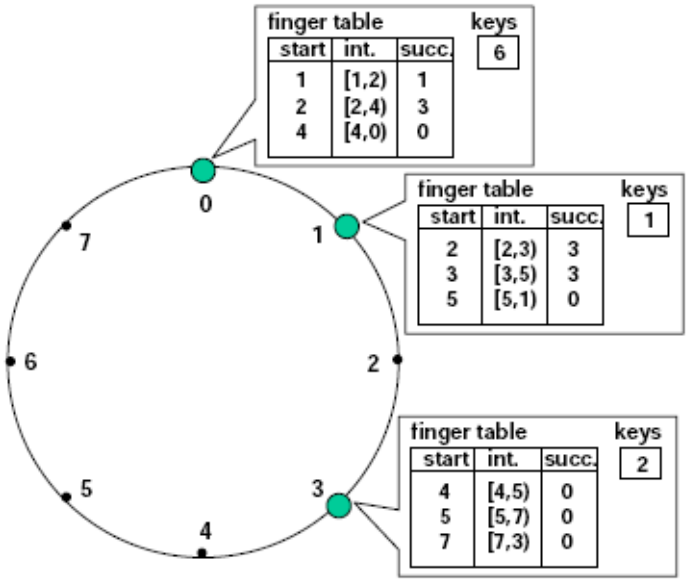


Obrázek 8: Ukázka topologie CHORD pro m =3 s uzly 0,1,3 a s klíči: 6 – přiřazeno uzlu 0, 1 – přiřazeno uzlu 1, 2 – přiřazeno uzlu 3.

Směrovací tabulka (finger table) každého uzlu má velikost m. Záznam i v

tabulce uzlu *u* obsahuje identifikaci uzlu *s*, který následuje uzel *u* na kružnici nejméně o 2^{i-1} , tedy $s = successor(u + 2^{i-1})$. Záznam pokrývá část kružnice $[u+2^{i-1},u+2^{i})$ a obsahuje identifikaci uzlu *s* i informace jak ho kontaktovat (IP adresa, port, ..). Směrovací tabulky ilustruje Obrázek 9.

Pomocí těchto tabulek si uzly mezi sebou předávají zprávu tak dlouho, dokud nedojde k příjemci. Je vidět, že maximální počet přeskoků mezi uzly je $O(\log_2(n))$ a každému uzlu si stačí držet informace o $log_2(n)$ uzlech.



Obrázek 9: Směrovací tabulky v CHORD.

Poněkud složitější je mechanismus připojení nového uzlu. Aby bylo možné zajistit tento proces je nutné, aby si každý uzel držel ukazatel také na svého bezprostředního předchůdce. Každý uzel také musí poskytovat funkce *najdi_předchůdce(k)* a *najdi_následníka(k).* Následující ukázka kódu ukazuje realizaci těchto funkcí:

```
#define successor finger[1].node


// požadavek na uzel n, aby našel následníka id.
n.find_successor(id) {
      n' = find_predecessor(id);
      return n'.successor;
}

// požadavek na uzel n, aby našel předchůdce id.
n.find_predecessor(id) {
      n' = n;
      while ( id  (n', n'.successor] ) {
            n' = n'.closest_preceding_finger(id);
      }
```

```
      return n';
}

// požadavek na uzel n, aby našel prst nejblíže předcházející id.
n.closest_precending_finger(id) {
      for i = m downto 1 {
            if (finger[i].node   (n,id)) return finger[i].node;
      }
      return n;
}
```

Pro samotné připojení nového uzlu se musí provést tři operace:

1. Inicializace směrovacích tabulek a ukazatelů na nově připojeném uzlu.
2. Aktualizace směrovacích tabulek a ukazatelů na stávajících uzlech tak, aby reflektovaly přidání nového uzlu.
3. Informovat vyšší vrstvy tak, aby mohly provést případný transfer přiřazených klíčů.

Následuje ukázka kódu, který implementuji připojení nového uzlu:

```
// požadavek na uzel n, aby se připojil do sítě přes uzel n'.
n.join(n') {
      if(n') {
            init_finger_table(n');
            update_others();
      } else { // n je jediný uzel v sítí.
            for i = 1 to m {
                  finger[i].node = n;
            }
            predecessor = n;
      }
}

// inicializace směrovací tabulky přes uzel n'.
n.init_finger_table(n') {
      finger[1].node = n'.find_successor(finger[1].start);
      predecessor = successor.predecessor;
      successor.predecessor = n;
      for i = 1 to m-1 {
            if (finger[i+1].start   [n,finger[i].node)) {
                  finger[i+1].node = finger[i].node;
            } else {
                  finger[i+1].node =
                        n'.find_successor(finger[i+1].start);
            }
      }

// oprava tabulek všech uzlů, jejichž tabulky mohou
```

```
// ukazovat na n.
n.update_others() {
      for i = 1 to m {
            p = find_predecessor(n – 2^(i-1));
            p.update_finger_table(n,i);
      }
}

// zkusit opravit i-tý prst hodnotou s.
n.update_finger_table(s,i) {
      if (s   [n,finger[i].node)) {
            finger[i].node = s;
            predecessor.update_finger_table(s,i);
      }
}
```

V prvním kroku požádáme uzel, který již participuje na síti, aby nám vyhledal všechny údaje, které potřebujeme k inicializaci své směrovací tabulky. V druhém kroku postupujeme proti směru hodinových ručiček a upozorňujeme uzly, jejichž směrovací tabulku mohl nově přidaný uzel ovlivnit.

Třetí krok musí být řešen na vrstvě, která spravuje přiřazené klíče. Od vrstvy chord se pouze očekává, že upozorní aplikační vrstvu v případě, že se změnil předchůdce daného uzlu.

Algoritmus popsaný výše požaduje na připojení uzlu čas $O(log_2^2 n)$. Po několika dalších optimalizacích se můžeme dostat až na $O(log_2 n)$. Nevýhodou tohoto algoritmu je skutečnost, že využívá směrovací tabulky velkého množství uzlů a předpokládá, že tyto tabulky jsou v konzistentním stavu. To v případě, že se připojuje (nebo odpojuje) více uzlů najednou, nelze zajistit. Během připojování uzlu také může dojít k situaci, kdy dotaz na klíč není korektně zodpovězený.

Jiný algoritmus na připojení uzlu předpokládá periodické spouštění stabilizačních procedur.

```
// Požadavek na uzel n, aby se připojil do sítě přes uzel n'.
n.join(n') {
      predecessor = nil;
      successor = n'.find_successor(n);
}

// Uzel periodicky kontroluje svého následníka.
n.stabilize() {
      x = successor.predecessor;
      if ( x   (n,successor)) {
            successor = x;
            successor.notify(n);
      }
}
```

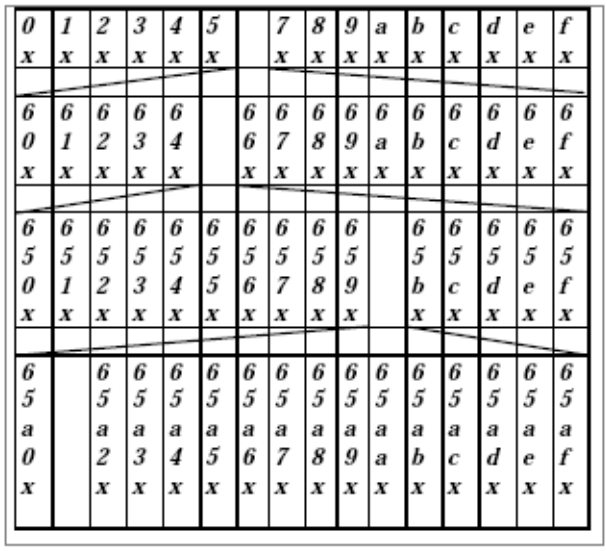
```
// n' si myslí, že je můj předchůdce.
n.notify(n') {
      if (predecessor is nil or n'  (predecessor,n)) {
            predecessor = n;
      }
}

// Uzel periodicky obnovuje svojí směrovací tabulku.
n.fix_fingers() {
      i = náhodný index ze směrovací tabulky větší než 1.
      finger[i].node = find_successor(finger[i].start);
}
```

Pro korektnost vyhledávání je důležitý především ukazatel na následníka. Pokud jsou ve zbytku tabulky zastaralé údaje, záznam se přesto vyhledá, může to ale trvat delší dobu. Při připojení uzlu se inicializuje pouze ukazatel na následníka, ostatní údaje jsou získávány později díky periodickému spouštění stabilizačních procedur.

Aby mohl systém lépe reagovat v případě havarie některého uzlu, drží se každý uzel seznam několika svých dalších následníků. V případě selhání bezprostředního následníka uzel kontaktuje dalšího v seznamu. Stabilizační procedury se upraví tak, aby udržovaly aktuální i tento seznam.

c) Pastry [29]

Podobně jako CHORD organizují pastry uzly do kruhové topologie. Každý uzel má přiřazený svůj klíč, *NodeID*. Pro účely směrování jsou klíče rozděleny na sekvenci číslic se základem 2^b (*b* je konfigurační parametr, typicky 4). V každém kroku směrování uzel směruje zprávu tomu uzlu, jehož *NodeID* sdílí s klíčem prefix, který je nejméně o jednu číslici delší než prefix, který sdílí klíč s aktuálním uzlem. Díky tomu je maximální počet přeskoků $log_{2^b}(n)$. Směrovací tabulka (routing table - R) obsahuje $log_{2^b} N$ řádků, každý po $(2^b - 1)$ záznamech. Řádek *n* (jsou počítány od nuly) obsahuje záznamy, které sdílejí s identifikátorem uzlu alespoň *n* číslic. Ve sloupcích pak jsou další možná pokračování záznamu. Kromě toho, který má stejné pokračování jako identifikátor aktuálního uzlu. Takový záznam je rozveden na dalším řádku. Situaci ilustruje Obrázek 10. Záznamy, které odpovídají prefixu identifikátoru uzlu v každém řádku, nejsou vyplněny. Tabulka má čtyři řádky, na prvním řádku jsou prefixy délky jedna (kromě prefixu 6, který se stejný s identifikátorem uzlu). Pokud se klíč shoduje s nějakým záznamem na prvním řádku, je směrován na uzel, který je s tímto záznamem spojen. Jinak se postupuje na další řádek. Směrovací tabulka má velikost $log_{2^b} N * (2^b - 1)$.



Obrázek 10: Směrovací tabulka pro PASTRY uzel s nodeID=65a1x. Kde b=4, x značí neznámou koncovku. Přiřazené IP adresy nejsou zobrazeny.

Každý uzel si také drží množinu uzlů (leaf set – L), které jsou numericky nejbližší jeho *NodeID*. Velikost této množiny je další konfigurační parametr značený l (doporučuje se volit 2^b nebo 2^{b+1}). Polovina z této množiny obsahuje klíče větší než je identifikátor daného uzlu, druhá polovina zase klíče menší než identifikátor uzlu. Vlastní směrování pak probíhá podle následujícího algoritmu:

```
// D – Klíč zprávy, která má být směrována.
// A – Identifikátor aktuálního uzlu.
// R_l^i – Záznam ve směrovací tabulce na řádku l, ve sloupci i.
// L_i – i-tý nejbližší uzel v leaf set.
// D_l - l-tá číslice v klíčí D.
// share(A,B) – délka sdíleného prefixu mezi klíči A a B.

if ( L_-l/2   D   L_l/2 ) {   // použít leaf set
      přepošli na L_i takové, že | D – L_i | je minimální.
} else {    // použij routing table.
      len = share(D,A);
      dig = D_len;
      if ( R_len^dig   nil) {
            přepošli na R_len^dig;
      } else {    // pokud ve směrovací tabulce chybí záznam.
            přepošli na T takové, že:
                  T   ( L   R );
                  share(T,D)   len;
                  | T – D | < | A – D |
      }
}
```

Připojení nového uzlu se děje podle jednoduchého scénáře. Nově připojovaný uzel (označíme ho X) využije nějaký již připojený uzel (A) a pošle speciální zprávu *join* s klíčem X. Tato zpráva je směrována na uzel (Z), který je numericky nejbližší klíči X. Každý uzel, který je na cestě této zprávy, odešle uzlu X své stavové tabulky. Z těchto informací je schopen uzel X sestavit si svoji kompletní směrovací tabulku i leaf set.

V druhém kroku odešle uzel X své stavové tabulky všem uzlů, které má ve své směrovací tabulce a leaf setu. Příjemci těchto informací pak mají možnost zanést příslušné opravy do svých tabulek.

Celá operace připojení nového uzlu by neměla stát více než $O(log_2{}^b)$ přenesených zpráv (konstanta je okolo $3 * 2^b$).

Pastry také řeší případy, kdy se uzel neočekávaně odpojí nebo havaruje. Uzel, který se pokoušel kontaktovat havarovaný uzel musí zajistit opravu ve svých tabulkách.

V případě, že je havarovaný uzel v jeho leaf setu, musí kontaktovat uzel, který se leaf setu nachází blízko havarovaného a požádat ho o jeho leaf set. Z vlastního leaf setu pak může vypustit havarovaný uzel a nahradit ho jiným uzlem ze získaného leaf setu. Toto funguje pokud simultánně vypadne méně než l/2 *l* uzlů v bezprostřední blízkosti.

V případě, že havarovaný uzel je ve směrovací tabulce (řekněme na pozici $R_l{}^d$), kontaktuje jiný uzel ze stejného řádku ($R_l{}^i$, i d) a zeptá se ho na jeho záznam pro $R_l{}^d$. Pokud žádný uzel z l-tého řádku nemá kontakt na žijící uzel, může uzel kontaktovat uzly na nižším řádku. Tímto postupem s vysokou pravděpodobností najde příslušný uzel, pokud existuje.

Pro každý záznam ve směrovací tabulce existuje více možných uzlů, na které může ukazovat. Rozhodnutí, který uzel se do směrovacích tabulek umístí, může být ovlivněno další metrikou. Například počet IP přeskoků. Pro využití této metriky si každý uzel drží také neighborhood set, která obsahuje |M| uzlů, které jsou nejblíže danému uzlu ve smyslu metriky IP přeskoků. Udržování aktuálnosti této tabulky se děje podobně jako u leaf setu.

d) Kademlia

S jiným přístupem přichází technologie nazvaná Kademlia [30]. Používá metriku postavenou na funkci XOR, $\delta(x,y) = x \oplus y$. Pro lokaci uzlů v blízkosti zadaného ID se využívají paralelní asynchronní dotazy, které mohou výrazně snížit latenci dotazu a zajistit nalezení cílového uzlu i ve velmi dynamicky měnící se síti.

Identifikace uzlu v síti je 160 bitů dlouhá. Každý uzel si drží 160 seznamů, v terminologii kademlie nazývané k-buckets. *S*eznam *i* obsahuje uzly (a kontaktní informace na ně), jejichž vzdálenost od daného uzlu je mezi 2^i a 2^{i+1}. Každý seznam má *k* položek, kde *k* je konfigurační parametr. Díky tomu, že

XOR je symetrická funkce, může uzel získat směrovací informace z každé zprávy přijaté od jiného uzlu. Po přijetí zprávy postupuje podle následujícího schématu:

- Pokud odesílatel zprávy už v příslušném seznamu existuje, je posunut na konec seznamu.
- Pokud odesílatel zprávy není v příslušném seznamu a seznam má méně než *k* položek, je odesilatel přidán na konec seznamu.
- Pokud odesílatel zprávy není v příslušném seznamu a seznam je již plný, provede se ping na uzel, který byl viděn před nejdelší dobou:
  - Pokud uzel odpoví na ping je přesunut na konec seznamu a informace o odesilateli původní zprávy se zahodí.
  - Pokud uzel neodpoví na ping je ze seznamu vymazán a odesilatel původní zprávy je přidán na konec seznamu.

Každý seznam efektivně implementuje least-recently seen mechanismus na údržbu směrovacích tabulek. S tím rozdílem, že živé uzly nejsou nikdy vymazávány.

Každý uzel implementuje čtyři základní funkce:

- *ping(node)* – ověří funkčnost daného uzlu..
- *store(key,value)* – uloží dvojici <klíč,hodnota> na daném uzlu.
- *find_node(id)* – vrátí *k* uzlů, které má daný uzel ve svých směrovacích tabulkách a jsou nejblíže parametru *id.*
- *find_value(id)* – obdobně jako *find_node,* pouze s tím rozdílem, že pokud má uzel k zadanému *id* uloženou hodnotu, tak je tato hodnota výstupem funkce.

K nalezení *k* uzlů, které jsou nejblíže zadanému ID používá kademlia rekurzivní algoritmus. Iniciátor dotazu vybere ze svých tabulek α uzlů, které jsou nejblíže hledanému ID a odešle jim dotaz *find_node(ID).* Z odpovědí, které uzel přijal, vybere opět uzly, které jsou nejblíže hledanému ID a znova odešle *find_node(ID)*. Takto rekurzivně pokračuje dokud nemá všech *k* uzlů, které jsou skutečně nejblíže hledanému ID.

Pokud je parametr α = 1 vyhledávací algoritmus připomíná Chord. Kademlia však dokáže směrovat zprávu rychleji, protože vyhledávání může probíhat paralelně.

Procedura pro uložení hodnoty v kademlii pak probíhá ve dvou krocích. Uzel nejprve lokalizuje *k* uzlů nejblíže ukládanému klíči. Následně všem takto získaným uzlům pošle zprávu *store.* Dvojice <klíč,hodnota> je tedy vždy uložena na nejméně *k* uzlech, proto se parametru *k* také říká replikační parametr.

Získání hodnoty probíhá podobně jako lokalizace uzlů, pouze se použije funkce *find_value* a rekurze skončí v okamžiku, kdy nějaké volání funkce vrátí hledanou hodnotu.

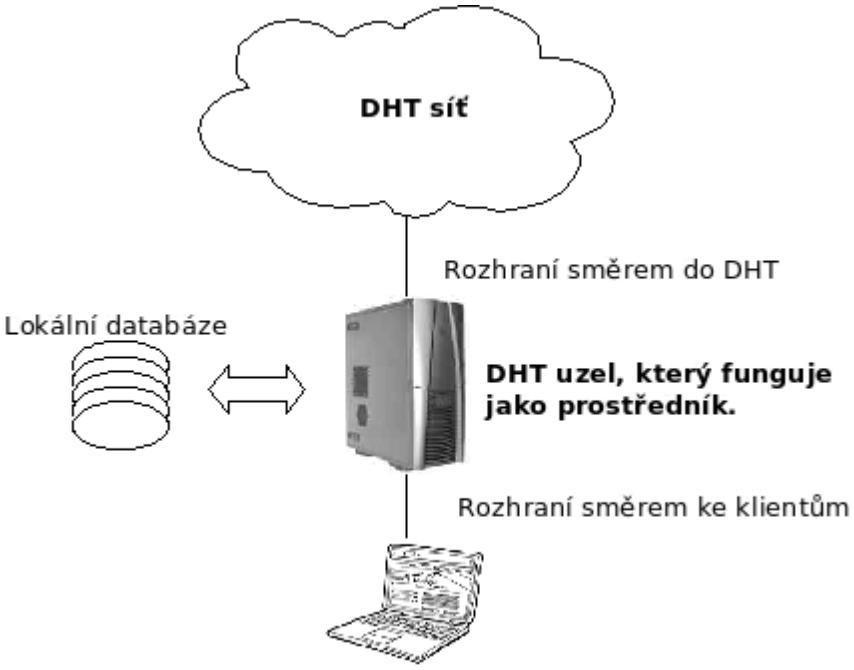
Nově připojovaný uzel musí znát alespoň jednoho účastníka sítě. Nejprve si vygeneruje své unikátní ID a do svých seznamů si jako první záznam vloží ID známého uzlu. Poté spustí vyhledávací algoritmus na své vlastní ID. Díky tomu si inicializuje všechny své seznamy a ostatní uzly si ho uloží do svých seznamů.

Pro odpojení uzlu se nepředpokládá žádný zvláštní postup. Síť je navržena tak, aby si s odpojováním uzlů snadno poradila.

3.4.4 Využití DHT pro lokalizaci uzlu (End-to-End přístup)

Pro End-to-End přístup vyžadujeme, aby v síti existovala služba, která nám na základě nějaké identifikace uzlu zjistí jeho aktuální IP adresu (LocID). Pro tyto účely využijeme dynamické hašovací tabulky.

Naší DHT síť budou tvořit relativně statické servery. Operace přidání a odebrání uzlu z DHT sítě je poměrně drahá, proto by k ní mělo docházet minimálně. Určitě není vhodné, aby se přímo mobilní uzly podílely na tvorbě DHT sítě. Každý mobilní uzel, který chce využívat služeb DHT, musí mít svého prostředníka. Prostředník je server, který se účastní DHT sítě a který umožňuje ostatním uzlům odesílat zprávy do DHT sítě. Situaci ilustruje Obrázek 11.

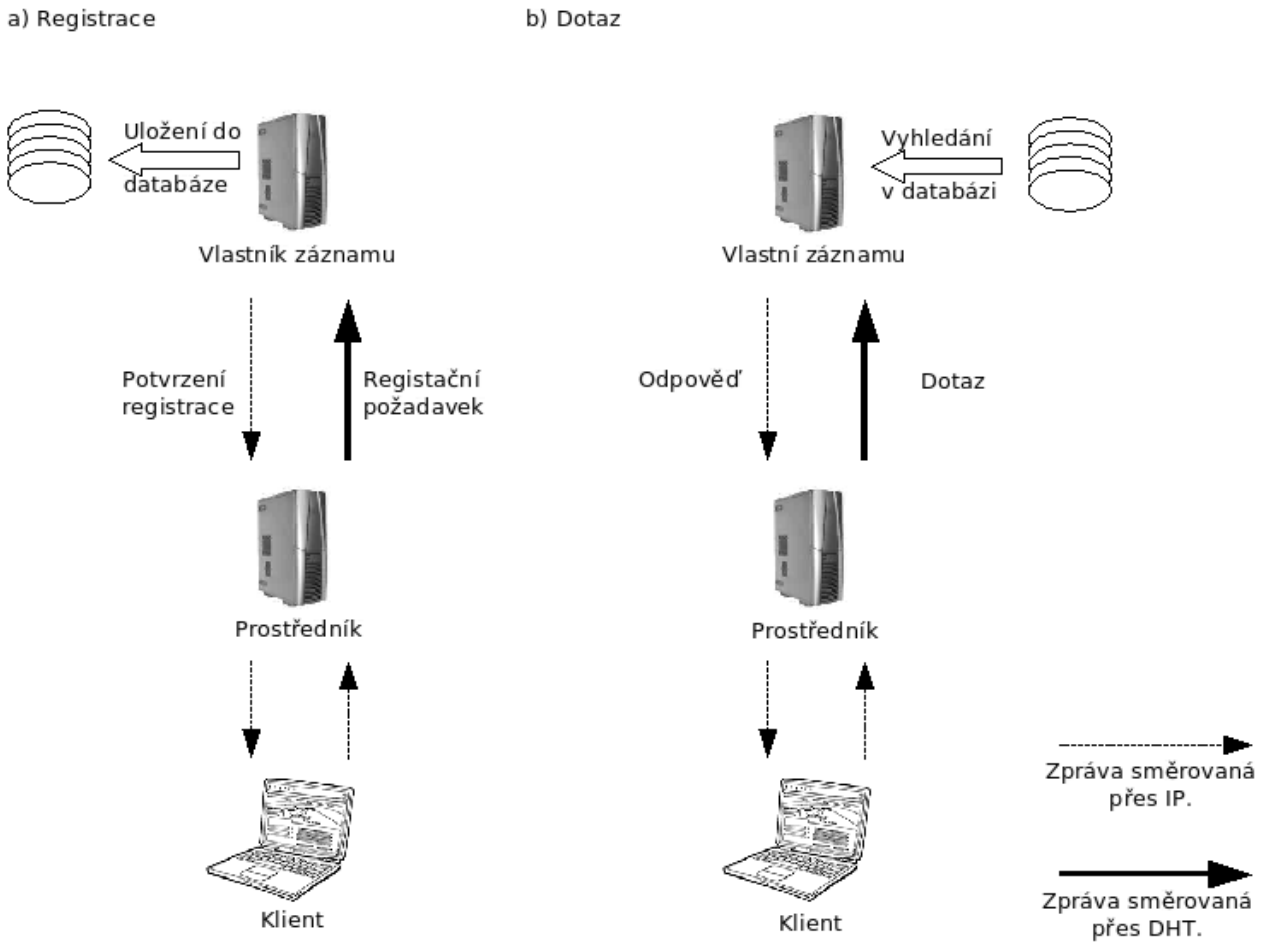


Obrázek 11: Struktura DHT sítě.

Každý DHT uzel má také svojí lokální databázi, která reprezentuje jeho část globální hašovací tabulky. Každý DHT uzel poskytuje svým klientům dvě základní služby:

1. Funkce *register(key,value)* – zanese dvojici <key,value> do hašovací tabulky.
2. Funkce lookup(key) – vyhledá v hašovací tabulce hodnotu <value>

odpovídající zadanému klíči a vrátí ji jako výsledek funkce.



Obrázek 12: Registrace a vyhledání záznamu v prostředí DHT.

Obrázek 12 ukazuje implementaci základních funkcí DHT sítě. Vlastníkem záznamu rozumíme takový DHT uzel, jehož identifikace je nejblíž klíči daného záznamu. Vlastník záznamu udržuje ve své lokální databázi aktuální hodnotu záznamu.

a) Spolupráce s DNS

Námi navržená DHT sít dokáže mapovat libovolný klíč na jeho hodnotu. Tedy podobnou funkci jakou nabízí dnešní DNS. Cílem návrhu End-to-End přístupu pomocí DHT je zajistit dynamičtější propagaci změn než je tomu dnes v DNS. Naproti DNS nabízí však pouze plochou strukturu klíčů.

Naší snahou není vytvořit systém, který má ambice zcela nahradit stávající DNS (takové snahy se objevili například v článku [25]). Ale vytvořit systém, který DNS doplňuje a vytváří prostředí pro mobilní uzly.

Jsou možné dva způsoby kooperace DHT s DNS:

DNS záznam obsahuje klíč do DHT

DNS je poměrně flexibilní systém, který se snadno rozšíří o další typ záznamu. Již dnes existuje v DNS poměrně velké množství různých typů záznamů. Nejpoužívanější je A-záznam, který mapuje doménové jméno na IP adresu. Další typy záznamů jsou třeba AAAA-záznam pro IPv6, CNAME-záznam, TXT-záznam, MX-záznam, nebo celá skupina záznamů pro DNSSEC. Není tedy nijak proti filozofii DNS, když ho rozšíříme o další typ záznamu. Nový typ záznamu nazveme ID-záznam a jeho úkolem bude zajišťovat mapování mezi doménovým jménem a neměnnou identifikací zařízení.

Identifikace zařízení bude postavena na asymetrické kryptografii tak, aby se zamezilo zneužití identity. Každý mobilní uzel si vygeneruje dvojici veřejný a privátní klíč. Na veřejný klíč použije hašovací funkci a výsledek funkce použije jako svůj identifikátor a klíč do DHT. Tento identifikátor předá správci své domény a požádá ho, aby pod jeho doménovým jménem zaregistroval ID-záznam s tímto identifikátorem.

Díky vlastnictví soukromého klíče (ten musí mobilní uzel pochopitelně udržovat v tajnosti) má uzel kdykoliv možnost prokázat, že daná identifikace k němu opravdu patří. A díky začlenění ID-záznamu do DNS serveru prokáže, že má nárok na dané doménové jméno.

Není důvod, aby mobilní uzel měnil svojí identifikaci. Snad pouze v případě vyzrazení soukromého klíče. Záznam v DNS tedy může být téměř statický a uchovávaný v cache po poměrně dlouhou dobu. Dynamické vlastnosti má pouze záznam v DHT.

Následující kód ukazuje postup klienta při získávání IP adresy:

```
// Překlad doménového jména na IP adresu.
lookup(domain_name) {
      id_rec = dns_query(domain_name,'ID');
      if (id_rec  nil) {
            ip = dht_query(id_rec);
            return ip;
      } else {
            ip = dns_query(domain_name,'A');
            return ip;
      }
}
```

Požadavek na registraci opatří klient elektronickým podpisem pomocí svého soukromého klíče a přiloží i svůj veřejný klíč. Vlastník záznamu v DHT síti pak má možnost ověřit oprávněnost požadavku.

DHT existuje jako alternativa k DNS

V tomto případě poskytuje DHT službu převodu doménového jména na IP adresu stejně jako A-záznamy v DNS. Jako klíč záznamu se použije hašovaná hodnota doménového jména.

Klient má možnost si vybrat, jestli dotaz položí nejprve do DHT nebo DNS. Obě služby mu vrátí IP adresu, na které může kontaktovat vzdálený uzel. Pokud chce využívat mobilitu, měl by nejprve kontaktovat DHT.

Trochu složitější situace je v případě registrace záznamu v DHT. Vlastník záznamu v DHT síti musí být schopen ověřit oprávněnost registračního požadavku. To může udělat jedině v součinnosti s DNS. V DNS musí být záznam typu DNSKEY, který obsahuje veřejný klíč. Soukromý klíč k tomuto záznamu vlastní mobilní uzel a pomocí něho podepíše žádost o registraci. Vlastník záznamu po přijetí žádosti o registraci tedy nejprve položí dotaz do DNS a na základě získaného veřejného klíče ověří pravost elektronického podpisu.

Srovnání obou metod

První metoda je šetrnější k DHT síti. Do DHT sítě se dostanou pouze dotazy na konkrétní klíč. Tedy dotazy na uzly, které s velkou pravděpodobností nějaký mobilní záznam mají. Naproti tomu více zatěžuje DNS, protože každý dotaz do DHT předchází dotaz na ID-záznam v DNS. Pro nemobilní záznam to tedy znamená jeden DNS dotaz navíc.

Druhá metoda je naopak šetrnější k DNS. Mobilní záznamy jsou zodpovězeny přímo pomocí DHT sítě bez jakéhokoliv kontaktování DNS. Pouze při registraci je pokládán záznam do DNS, aby bylo možno ověřit autenticitu registrace. Naproti tomu DHT síť je zatěžována množstvím zbytečných dotazů na statické záznamy.

Z tohoto pohledu bych upřednostňoval první variantu. Na DHT síť jsou kladeny velké nároky pro zajištění dynamičnosti záznamů, proto by neměla být zatěžována zbytečnými dotazy na statické záznamy. Dotazy na statické záznamy dokáže DNS efektivně cachovat. Navíc takto navržená DHT síť by neměla být chápána jako alternativa k DNS, ale jako jeho doplněk. DNS slouží jako převážně statická hierarchická databáze pro většinu uzlů v Internetu a obsahuje různé typy záznamů o jednotlivých uzlech. Naproti tomu DHT je dynamická plochá databáze pro potřeby lokace mobilních uzlů.

b) Využití cache

V DNS, jak ho dnes známe, hraje využití cache důležitou roli. Významně zmenšuje počet zpráv, které jsou přenášeny po síti za účelem vyřízení dotazu, a zmenšuje čas, který je na zodpovězení dotazu potřeba. Na druhou stranu umožňuje, aby tazatel získal záznam, který je již neplatný.

Při návrhu cache pro DHT je nutné brát v potaz dynamický charakter sítě. Podrobněji budeme diskutovat dva základní přístupy ke cachování.

Hodnotová cache

Jedná se o nejběžnější variantu cache, kdy jednotlivé položky v cache obsahují přímo dvojice <key,value>. Tato cache může být umístěna přímo na klientech,

na prostřednících nebo na libovolnému uzlu DHT sítě. Každá položka také musí mít nastavenu svoji dobu platnosti (TTL), po uplynutí této doby musí být z cache vymazána. Přesto se však nelze vyhnout situaci, kdy klient dostane na svůj dotaz již neplatnou odpověď.

Klient by tedy měl mít možnost ověřit platnost získaného záznamu. To může udělat například tak, že se pokusí kontaktovat uzel na získané IP adrese. Vlastní komunikaci však musí předcházet prokázaní identity. Lokální uzel má díky DNS k dispozici veřejný klíč vzdáleného uzlu, proto by prokázání identity neměl být problém. Pokud se vzdálenou stranu nepodaří kontaktovat nebo pokud se nedokáže správně identifikovat, je získaný záznam považovaný za neplatný. Je nutné odeslat dotaz do DHT znovu, tentokrát s příznakem, který zajistí vynechání cache. Tato metoda nedokáže rozlišit situaci, kdy se uzel přesune, od situace, kdy uzel havaruje.

Druhá otázka zní, jak volit hodnotu TTL. Čím více je uzel mobilní, tím menší hodnotu TTL by měl mít. Optimální hodnotu TTL může určit vlastník záznamu podle frekvence s jakou se uzel přesunuje. Pokud si bude schopen držet údaje o historii registrací, může z nich spočíst optimální hodnotu TTL. Pro nejvíce mobilní uzly by hodnota TTL byla tak malá, že by se tato cache vůbec nemusela uplatnit.

Směrovací cache

Tato cache je postavena na myšlence, že pokud se hodnota záznamu často mění, není výhodné uchovávat si v cache přímo jeho hodnotu. Položky v cache obsahují kontakt na vlastníka záznamu. Cache tedy slouží jako pomocná směrovací tabulka. Každý uzel na cestě zprávy může tuto cache využít a poslat zprávu přímo jejímu adresátovi.

Dotaz je vždy doručen vlastníkovy záznamu, proto je vrácená odpověď vždy platná. Pokud se nějak změní počet uzlů v DHT síti a dojde k přesunutí některých klíčů, může se stát, že záznam v této cache již nebude odpovídat skutečnosti. To ale příliš nevadí. Pokud se na adrese z cache nenachází žádný uzel a zprávu se nepodaří odeslat, bude odeslána standardním mechanismem. Pokud se na adrese z cache nachází jiný uzel, je tento uzel schopen zprávu převzít a dál ji směrovat podle svých tabulek. Pouze upozorní předchozí uzel, že příslušná položka v cache je neplatná.

Přidání nové položky do cache se děje v případě, že uzel obdrží odpověď na dotaz, který předtím odeslal. Záznamy v cache jsou platné tak dlouho, dokud není zpochybněna jejich platnost. Následující kód ukazuje jak DHT uzel pracuje se směrovací cache.

```
// Přijetí zprávy DHT uzlem.
receive(msg,from) {
      if (msg.key ∈ my_zone) {
            // Přijatá zpráva je odpověď na můj dotaz,
            // přidej záznam do cache.
            if (msg.type = answer) {
                  route_cache.add(msg.key,from);
```

```
                    };
                    if (msg.type = 'incorect cache for $key') {
                          // Vymazání neplatné položky v cache.
                          route_cache.del(key);
                    } else {
                          // Vlastní zpracování zprávy.
                          process(msg);
                    }
              } else {
                    // Přijatá zpráva byla odeslána na
                    // základě špatného záznamu v cache.
                    // Upozorni odesílající uzel.
                    if (msg.route_type = direct) {
                          send(from,
                          new message('incorect cache for '+msg.key));
                    }
                    if(next_hop = route_cache.find(msg.key)) {
                          // Klíč má zaznam v cache,
                          // pokusím se odeslat zprávu přímo.
                          msg.route_type = direct;
                          if (!send(next_hop,msg)) {
                                // Přímé odeslání se nezdařilo
                                // musím odeslat přes DHT.
                                msg.route_type = forward;
                                next_hop = dht_get_route(msg.key);
                                send(next_hop,msg);
                                // Vymažu neplatný záznam z cache.
                                route_cache.del(msg.key);
                          }
                    } else {
                          // Odeslání přes DHT.
                          msg.route_type = forwad;
                          next_hop = dht_get_route(msg.key);
                          send(next_hop,msg);
                    }
              }
        }
```

Systém zaručuje, že vrácená odpověď vždy odpovídá poslední zaregistrované hodnotě. Při úspěšném zásahu do cache hned na prostředníkovi vyžaduje čtyři přenesené zprávy.

Rozebereme případy, kdy může dojít k vyzvednutí neplatného záznamu ze směrovací cache:

1. Vlastník záznamu se odpojil od sítě, záznam je přiřazen jinému uzlu a na původní IP adrese nelze nikoho kontaktovat: Odesílatel dotazu nejprve zkusí kontaktovat uzel podle záznamu ve své cache. Pokud do určité doby nedostane odpověď, zruší položku v cache a zprávu odešle přes standardní mechanismus. Doba doručení zprávy se prodlouží pouze o čas, který je nutný pro ověření dostupnosti uzlu.

2. Do sítě se připojil nový uzel, který se stal novým vlastníkem záznamu: Původní vlastník záznamu příjme dotaz na záznam, který již nevlastní. Pošle upozornění původnímu uzlu a sám se pokusí směrovat zprávu. Nový vlastník záznamu ale nutně musí být někde blízko (ve smyslu DHT) a původní vlastník by tedy měl mít na něj ukazatel ve svých směrovací tabulkách. Doba doručení zprávy vzroste pouze o jeden přeskok.

V obou případech lze získat ještě některá vylepšení. V první variantě může odesílatel dotazu před jeho směrováním v DHT přidat k dotazu ještě upozornění na neplatnou položku v cache. Je možné, že ostatní uzly v DHT, přes které bude zpráva směrována, mají stejnou neplatnou položku v cache a je zbytečné, aby se i oni pokoušely kontaktovat nefunkční uzel.

V druhé variantě lze do upozornění o neplatnosti položky v cache vložit i informaci o novém vlastníkovi záznamu. Tedy v případě, že má původní vlastník tento údaj přímo dostupný ve svých směrovacích tabulkách.

Je vidět, že i neplatná položka nezpůsobí žádné velké zdržení. Není tedy třeba limitovat platnost záznamu. Systém dokáže neplatné položky eliminovat sám. Pouze pro udržení cache co nejmenší je vhodné vyhazovat položky, které nejsou dlouhodobě potřeba.

Srovnání obou metod

Efektivnost obou metod závisí na způsobu jakým je DHT síť využívána. Lze předpokládat, že pro silně mobilní uzly bude efektivnější využívat metodu se směrovací cache. Pro statické uzly bude výhodnější zase hodnotová cache.

Směrovací cache je více náchylná na změny přímo v DHT síti. Její efektivita se bude snižovat s četností jakou se budou do DHT sítě připojovat/odpojovat uzly. Hodnotová cache zase špatně reaguje na změnu pozice mobilních uzlů. Stanovit optimální hodnotu TTL pro uzly, které se pohybují velice nepravidelně, může být obtížné.

Další nezanedbatelnou výhodu směrovací cache je fakt, že dokáže optimalizovat čas pro přenos všech typů zpráv, které využívají DHT směrování. Tedy i registračních požadavků a to při použití hodnotové cache není možné.

Vliv různých faktorů v jednotlivých přístupech je vhodné ověřit simulačně.

c) Reverzní záznamy

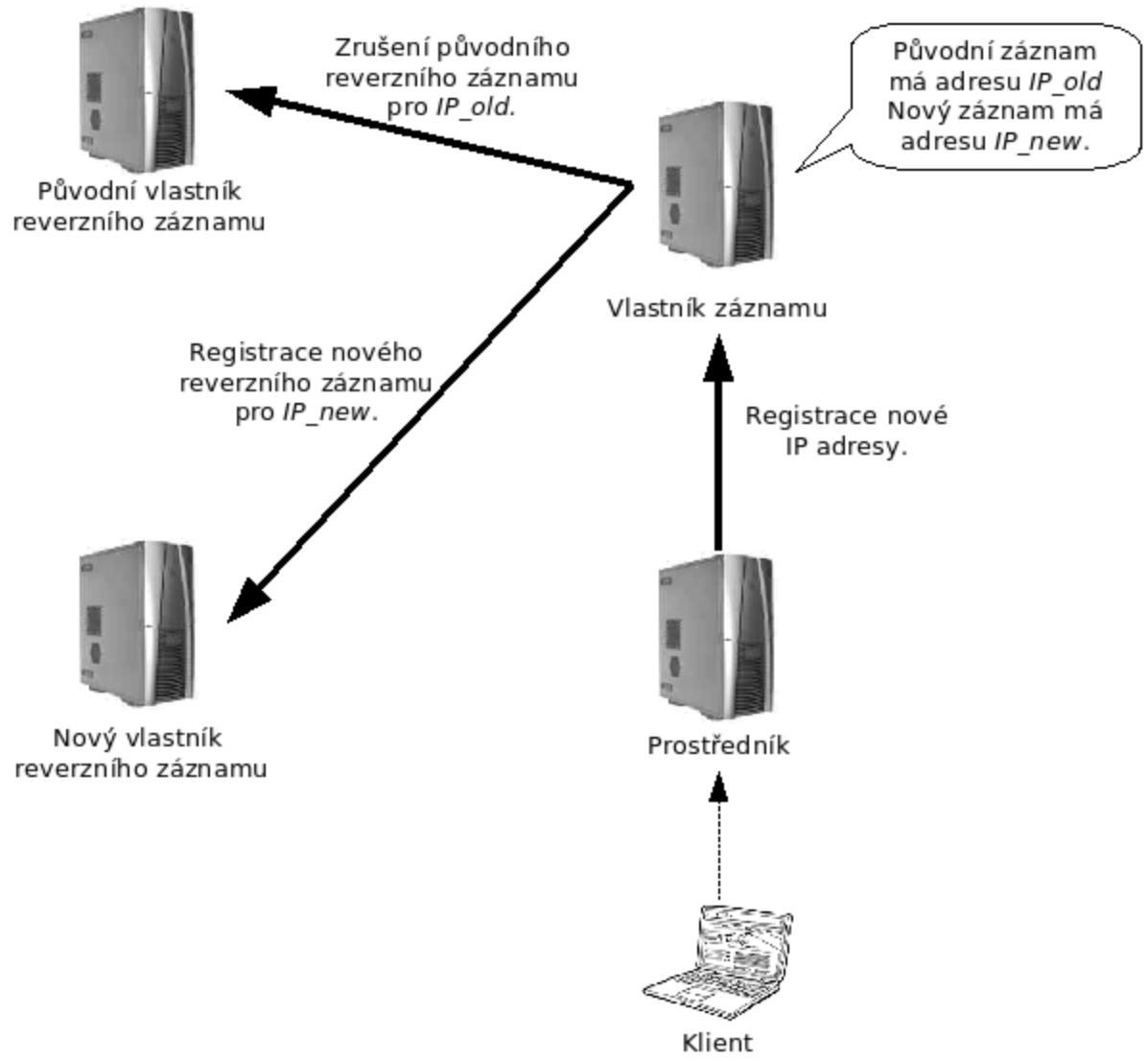
Může nastat situace kdy nepožadujeme převod jména (identifikace) na IP adresu, ale službu opačnou. Například k ověření identity uzlu, který se chce k našemu serveru připojit.

Rozšíření DHT sítě pro reverzní záznamy není nikterak obtížné. Spravováním reverzního záznamu pověříme vlastníka klasického záznamu. Vlastník reverzního záznamu se, na rozdíl od vlastníka klasického záznamu, mění s každou novou registrací. Vlastník klasického záznamu při každé změně registrované IP adresy provede následující kroky:

1. Z nové IP adresy záznamu spočte hašh, který bude sloužit jako klíč do DHT.
2. Podle získaného klíče směruje do DHT žádost o registraci reverzního záznamu. Příjemce žádosti (vlastník reverzního záznamu) zařadí tento záznam do své databáze a pošle zpět potvrzení.
3. Podobný proces provede pro původní IP adresu záznamu, s tím rozdílem, že posílá žádost o zrušení reverzního záznamu.

Dotaz na reverzní záznam pak probíhá stejným způsobem jako dotaz na klasický záznam. Reverzní záznam může obsahovat doménové jméno, číselný identifikátor nebo i veřejný klíč daného uzlu. Celou situaci ilustruje obrázek 13.

Při přesunu mobilního uzlu se reverzní záznamu aktualizuje po trochu delší době než záznam klasický. Od okamžiku odeslání registrace do okamžiku úpravy reverzního záznamu musí proběhnout dvojí DHT směrování (jedno k vlastníkovi záznamu, následně k vlastníkovi reverzního záznamu). Pomalejší update by však neměl vadit, využití reverzních záznamů není časté a lze ji chápat pouze jako doplňkovou službu.



Obrázek 13: Schéma registrace reverzního záznamu.

d) Notifikace

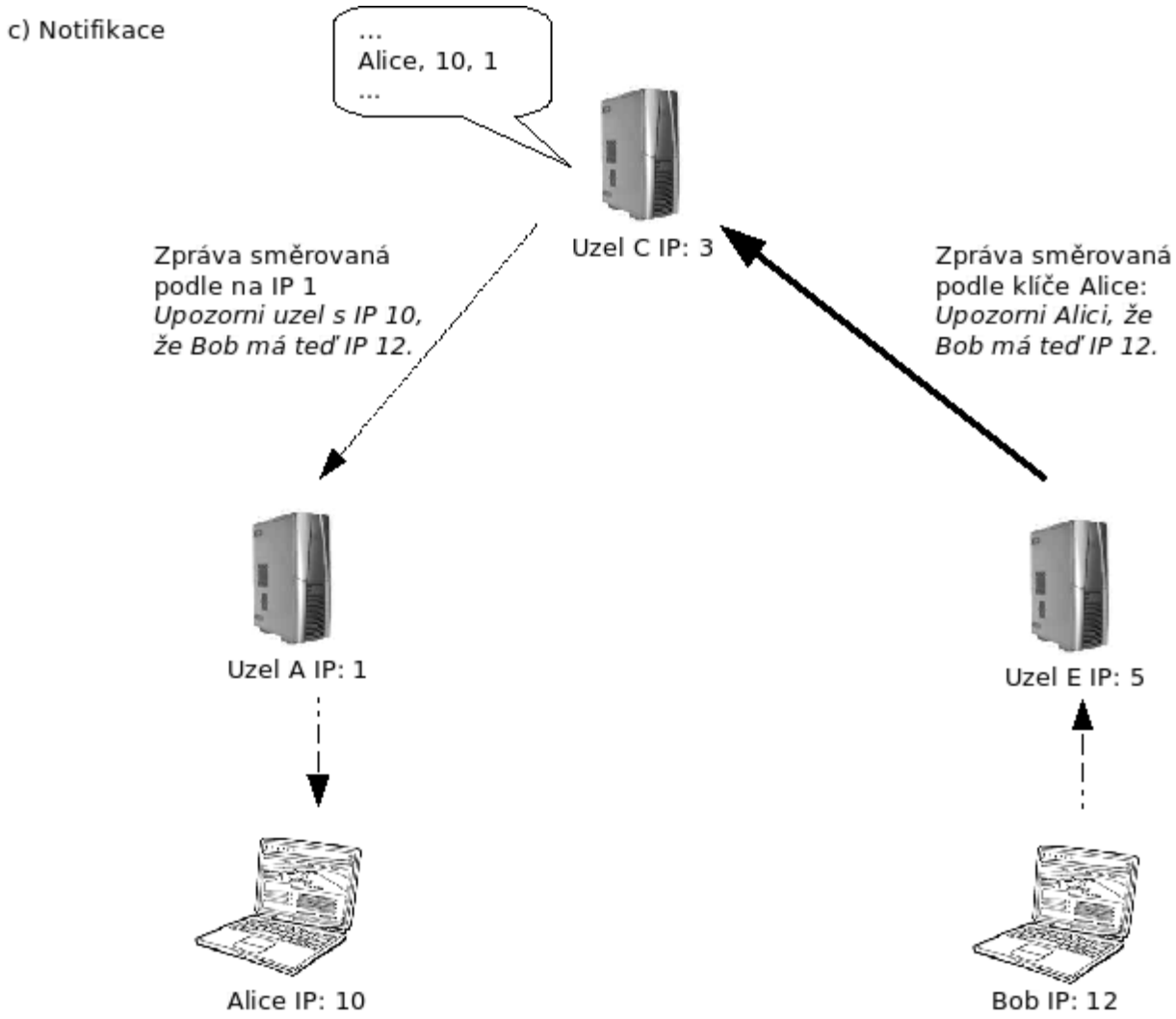
Rozeberme nyní případ, kdy jeden (nebo oba) komunikující uzly změní svojí polohu během komunikace. TCP migrace jak je popsána v kapitole 3.3.2 se dokáže vypořádat s případem, kdy změní polohu jeden z komunikujících uzlu. Jednoduše začne komunikovat se svým partnerem z jiné IP adresy. To, že tímto způsobem nemůže dojít k odcizení spojení, zajišťuje protokol TCP migrace.

Co ale v případě, kdy spolu komunikují dva mobilní uzly a stane se, že změní svojí polohu oba naráz? Nebo v případě, kdy je spojení mezi uzly nestavové (např. UDP) a princip použitý při TCP migraci nelze dobře použít?

DHT síť nabízí možné řešení. Komunikující uzly se mohou informovat o změně své polohy prostřednictví DHT uzlů. Jediné co je k tomu potřeba je, aby si

vlastník záznamu pamatoval IP adresu DHT uzlu, který daný záznam registroval.

Způsob, jakým se dá notifikace v prostředí DHT řešit ukazuje obrázek 14. Uzel *Bob* změnil svojí polohu (také svého prostředníka) a chce o tom informovat uzel s NodeID *Alice*. Zpráva je nejprve směrována na uzel, který drží informace o uzlu *Alice* a následně na prostředníka, který *Alice* využívá.



Obrázek 14: Notifikace v end-to-end řešení mobility přes DHT.

Pokud bychom chtěli, aby systém podporoval současnou změnu obou komunikujících uzlů, je potřeba aby si držitel záznamu nějakou dobu pamatoval notifikační žádost. Pokud by se do této doby záznam změnil, odeslal by notifikaci znovu, ale už na jiného prostředníka. Tato doba může být poměrně krátká, musí pouze pokrýt zpoždění, které vzniká přenosem registračních zpráv.

Hlavní využití ale vidím v případě, kdy uzly komunikují nestavovým protokolem. V takovém případě nemají mnohdy možnost zjistit, že jejich zprávy již nejsou doručovány a nebo jsou už doručovaný na jiný uzel. Také

nemají mezi sebou navázanou žádnou bezpečnostní asociaci a pokud by se o změně informovaly přímo, hrozilo by odcizení spojení.

DHT síť vlastně nabízí jednoduchou a zabezpečenou službu na posílání krátkých zpráv mezi uzly bez toho, aby byla nutná vzájemná znalost IP adres. Je možné využít tohoto mechanismu i k jiným účelům než pro uvedenou notifikaci. Například při ustanovování bezpečnostních asociací.

e) Zabezpečení

V souvislosti se zabezpečením se budeme věnovat několika okruhům:

Zneužití identity

Zneužití identity je základní problém mobility. První krok jak tomu zabránit, je vytvořit mechanismus, který jednoznačně identifikuje mobilní uzel. To bylo již z části probíráno v kapitole o spolupráci s DNS. Identifikace je postavena na asymetrické kryptografii a pro distribuci klíčů je využit právě systém DNS. Lze samozřejmě využít i jiný distribuční kanál, ale DNS je díky své rozšířenosti vhodný kandidát. Identifikační údaje jsou většinou statického charakteru a lze pro ně s úspěchem využít cache jaká se používá u DNS. DNS má však i své problémy, například existuje možnost podvržení záznamu. Důsledná implementace DNSSEC by však měla všem těmto problémům zabránit.

Díky asymetrické kryptografii má mobilní uzel možnost digitálně podepsat každý svůj záznam, který ukládá do sítě. Všechny ostatní uzly mají možnost ověřit pravost tohoto podpisu. Je na zvážení každého uzlu, zda bude kontrolu provádět. Díky DNS je celý systém snadno spravovatelný.

Pokud by chtěl útočník zneužít identitu mobilního uzlu, musel by zaútočit na samotný mobilní uzel a získat soukromý klíč nebo zaútočit na DNS a podvrhnout záznam o veřejném klíči. Útok na samotnou DHT nemá v tomto případě smysl.

Omezení služby (DoS)

Jak bylo rozebráno v předešlém odstavci, útočník nemůže podvrhnout záznam jiného uzlu. Může se však postarat o to, aby se záznam nedostal k jiným uzlům. Tento problém se týká spíše vrstvy samotné DHT. Vhodnými zprávami může útočník například zanést falešné hodnot do směrovacích tabulek DHT uzlů a způsobit tak rozpad celé sítě. Je tedy nutné, aby i mezi uzly v samotné DHT panoval určitý vztah důvěry. Tyto problémy jsou podrobně diskutovány v článku [31].

Výpadek DHT uzlu

Nelze se vyhnout situaci, kdy některý uzel v síti havaruje. V takovém případě dojde ke ztrátě všech dat na něm uložených. Proto je nutné udržovat záznamy v síti ve více kopiích. Některé DHT řeší redundanci již přímo ve svých návrzích (například Kademlia). Existují však metody, které lze použít univerzálně:

- Použitím k hašovacích funkcí - záznam je pak uložen v k kopiích v DHT. Hašovací funkce by měli být voleny tak, aby se minimalizovala možnost

umístění více kopií na jeden uzel.

- Použití *r* realit - *r* instancí DHT realizovaných na stejných uzlech. Záznam je ukládán do každé reality.
- Překrývající se oblasti - určitý úsek v prostoru klíčů obsluhuje *p* uzlů. Záznamy v tomto úseku jsou pak uloženy na všech těchto uzlech.

Jako nejméně výhodná metoda replikace se zdá být metoda s více realitami. Simulace provedené v [27] na DHT typu CAN ukazují, že přínos této metody je v nepoměru s nárůstem režie pro jednotlivé uzly (režii uzlů je *r*-násobná).

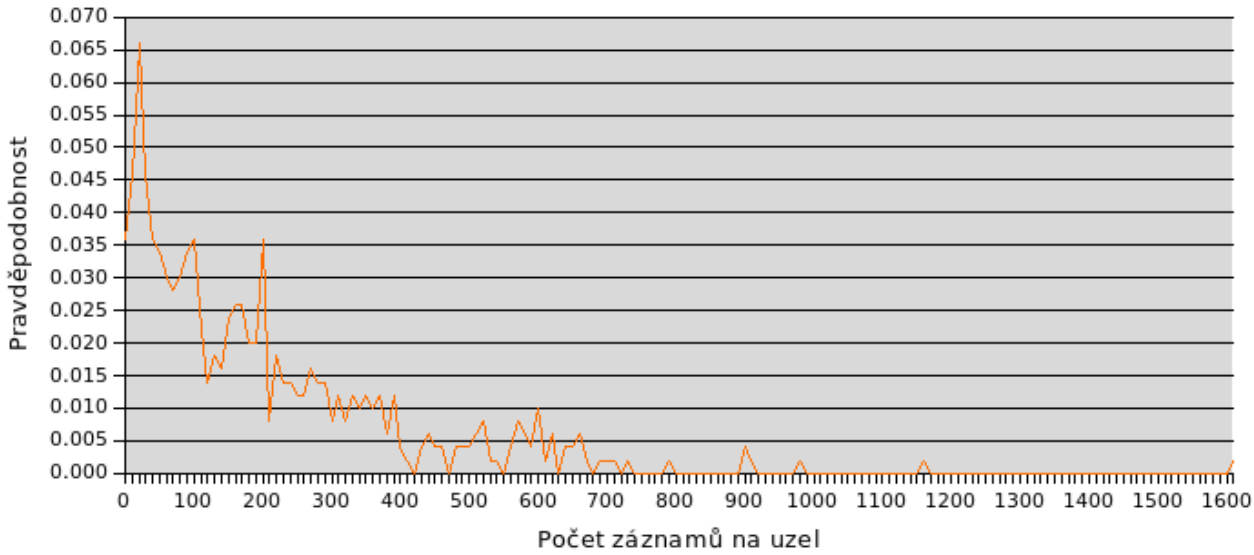
Stejné simulace ukazují, že z hlediska režie uzlů jsou metody s více hašovacími funkcemi a s překrývajícími oblastmi takřka rovnocenné. Metoda překryvných oblastí vykazuje nepatrně lepší chování, ale její implementace bývá složitější.

f) Rozložení zátěže

V navrženém systému očekáváme, že na každém uzlu bude uložen přibližně stejný počet záznamů a zátěž tak bude distribuována rovnoměrně na všechny uzly. Bohužel tomu tak není, klíče nejsou rovnoměrně distribuovány přes celý prostor. Když rozdělíme prostor na N stejně velkých oddílů, kde N je počet klíčů, očekáváme, že v každém oddílu bude jeden klíč. Ve skutečnosti pravděpodobnost, že daný oddíl bude prázdný je $(1-1/N)^N$. Pro velké N se tato pravděpodobnost blíží hodnotě $e^{-1} = 0.368$. V každé DHT síti pak existují uzly, které nemají žádné záznamy a také velká část uzlů, které mají záznamů i několikráte více než je průměr.

Rozložení pravděpodobnosti počtu záznamů na jednotlivých uzlech tak, jak bylo naměřeno v síti o 500 uzlech a při celkovém počtu záznamů 100 000, ukazuje obrázek 15.

Možné řešení tohoto problému je přes virtuální uzly. Každý reálný uzel bude na sebe mapovat několik virtuálních uzlů v DHT síti. Pokud budou virtuální uzly náhodně přiřazeny k reálným uzlům, bude rozložení záznamů přes reálné uzly více rovnoměrné. Zaplatíme za to zvýšenou zátěží reálného uzlu, který musí udržovat směrovací tabulky všech svých virtuálních uzlů.



Obrázek 15: Rozložení pravděpodobnosti zatížení jednotlivých uzlů v síti o 500 uzlech a 100 000 záznamech.

g) Simulace

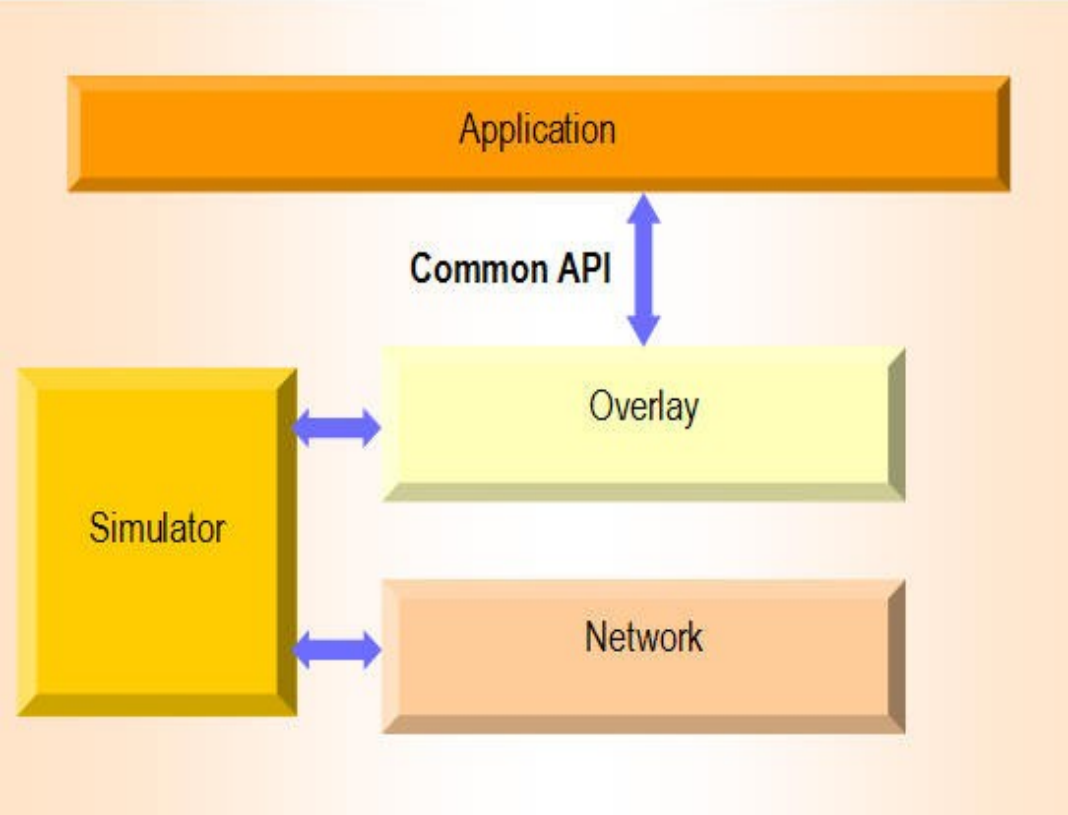
PlanetSim je objektově orientovaný simulační framework pro překryvné sítě a na nich postavené služby. Jako programovací jazyk je použita Java. Framework je založen na modulární třívrstvé architektuře s dobře dokumentovanými interface. Vývojáři používající PlanetSim ho mohou rozšiřovat na libovolné vrstvě. První vrstva simuluje síťové prostředí 'network layer'. Na ní je postavená překryvná vrstva 'overlay layer', která má za úkol simulovat různé překryvné sítě (Chord, Pastry, CAN, …). Poslední vrstvou je vrstva aplikační 'application layer' pro koncové aplikace.

Síťovou vrstvu lze nahradit wraperem, který přistupuje přímo k síťovému rozhraní. Tímto způsobem lze spustit simulaci na reálné síti. Standardní distribuce PlanetSim obsahuje základní implementaci síťové vrstvy, kde vzdálenost mezi každými dvěma uzly je jedna.

Pro účely naší simulace byly vytvořeny tří aplikace na aplikační vrstvě:

1. BasicDhtApp – simuluje DHT síť bez jakýchkoliv vylepšení.
2. RouteCacheDhtApp – simuluje DHT síť se směrovací cache.
3. ValueCacheDhtAp – simuluje DHT síť s hodnotovou cache.

Na nižší vrstvě byla využita implementace Chord, která je ve standardní distribuci PlanetSim. Síťová vrstva také využívá základní implementaci a vzdálenost mezi dvěma uzly na síťové vrstvě je jedna.



Obrázek 16: Schéma simulačního frameworku PlanetSim

BasicDhtApp

Jedná se o jednoduchou implementaci DHT sítě, která nevyužívá žádnou cache ani jiné metody na zrychlení. Je určena pro porovnání s ostatními metodami a určení efektivity jednotlivých přístupů.

RouteCacheDhtApp

Implementace, která využívá směrovací cache. Pomocí této cache jsou směrovány všechny zprávy na DHT vrstvě (registrace a dotazy). Záznamu v cache se snaží využít každý uzel na cestě směrovaného paketu. Pokud najde ve své cache odpovídající položku, je zpráva směrována přímo vlastníkovi záznamu.

Velikost směrovací cache na každém uzlu je omezena konfiguračním parametrem. Položky cache jsou stále platné. Pokud je cache plná a přijde požadavek na přidání nové položky, je z cache vymazána náhodná položka, aby se uvolnilo místo pro novou.

Není implementován mechanismus upozorňování na neplatnou položku ve směrovací cache. Vzhledem tomu, že simulovaná síť je statická, nemůže neplatná položka v cache vzniknout.

ValueCacheDhtApp

Implementace, která využívá hodnotovou cache. Pomocí této cache je

odpovídáno na dotazy klientů. Záznamů v cache využívají pouze prostřednici, které s její pomocí odpovídají na dotazy svých klientů.

Každá položka v cache má svojí hodnotu TTL, kterou stanoví vlastník záznamu podle frekvence s jakou se hodnota záznamu mění. Hodnota TTL je stanovena jako:

<čas od první registrace>/<počet změn registrace>

Pokud klient příjme odpověď a zjistí že na získané IP adrese není požadovaný uzel (v simulaci porovná s globální tabulkou, která odráží skutečné přiřazení IP adres), opakuje odeslání dotazu s parametrem, který zamezí využití hodnotovou cache.

Využití hodnotové cache na všech uzlech se ukázalo jako neefektivní. V případě špatného zásahu do cache někde na cestě dotazu může dojít až k dvojnásobnému prodloužení doby na vyřízení dotazu. Navíc na cestě dotazu může být více uzlů, které mají neplatnou položku ve své cache. V dynamické síti, jako je navržená DHT, je to skutečně problém. K eliminování tohoto problému by bylo nutné vyvinout poměrně složitý notifikační protokol.

Implementace hodnotové cache v simulaci používá cache pouze na prostřednících. V takovém případě se hodnotová cache ukazuje částečně efektivní.

Vstupní data

Na první pohled nejvhodnější vstupní data pro simulaci je záznam nějakého skutečného DNS provozu, který byl odchytávaný na nějaké větší síti po rozumně dlouhou dobu (například týden). Takový záznam ale není jednoduše dostupný a pro naše účely není vhodný. Statický systém DNS nepočítá s tím, že je možné záznamy měnit. Pro nás jsou důležité akce změny polohy uzlu, které by záznam DNS provozu neobsahoval.

Cílem bylo, aby vstupní data co nejvíce odpovídala reálnému provozu. Proto byla generována podle následujícího algoritmu:

1. Nejprve je vygenerována množina, která obsahuje použitá DNS jména spolu s údajem o jejich mobilitě. Mobilitu (M) chápeme jako poměr mezi počtem (pře)registrací a počtem dotazů. Pohybuje se v intervalu [1,0). Mobilita jedna reprezentuje nejvyšší hodnotu, kdy je poloha uzlu měněna po každém dotazu.
2. V další fázi jsou náhodně vybírána jména z vygenerované množiny a podle příslušné mobility je generována buď akce dotaz, nebo registrace.

Celou množinu vstupních dat lze parametrizovat pomocí těchto hodnot:

- N – Počet použitých DNS jmen (počet záznamů).
- M – Průměrná mobilita všech použitých DNS jmen (mobilita záznamu).
- A – Celkový počet akcí.

Pro účely testování byly použity následující soubory dat:

Soubor	Počet záznamů	Mobilita záznamů	Počet akcí	Počet registrací	Počet dotazů
R3_M1-3	1000	0.001	$4 * 10^6$	4083	3995917
R3_M1-2	1000	0.01	$4 * 10^6$	39238	3960762
R3_M5-2	1000	0.05	$4 * 10^6$	189654	3810346
R3_M1-1	1000	0.1	$4 * 10^6$	363205	3636795
R3_M5-1	1000	0.5	$4 * 10^6$	1331653	2668347
R3_M1-0	1000	1	$4 * 10^6$	1999779	2000221
R3_M5-1	100000	0.5	$5 * 10^5$	166558	333442

Samotná simulace probíhá tak, že z množiny vstupních dat jsou postupně čteny jednotlivé akce, které jsou přiřazovány náhodně vybraným uzlům. Vždy po čtyřech akcích následuje jeden simulační krok (Tato informace je obsažena přímo ve vstupních datech jako akce STEP). V každém simulačním kroku se provedou následující úlohy:

1. Klient vyzvedne od nižší vrstvy všechny zprávy, které mu byly doručeny v minulém kroku a vyhodnotí je.
2. Zpracuje nové požadavky a předá zprávy k odeslání nižším vrstvám.
3. Provede se jeden simulační krok na nižší vrstvě.

Výsledky testů

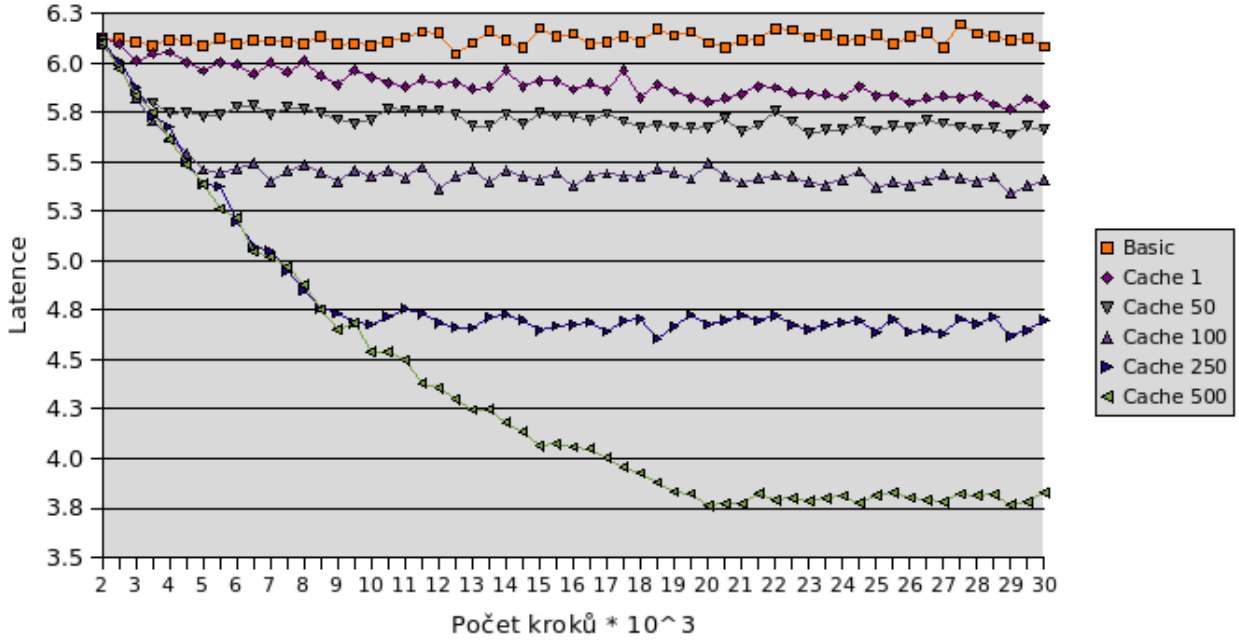
Ve všech testech je základní časovou jednotkou jeden krok. Ten v použitém simulačním frameworku znamená přenos jedné zprávy na druhé vrstvě (Overlay layer). Na aplikační vrstvě dojde v jednom kroku k načtení čtyř akcí ze vstupního souboru a jejich přiřazením náhodně vybraným uzlům.

Doba vyhledávaní záznamu je počítána v počtu simulačních kroků. De facto se jedná o počet uzlů, přes které je zpráva směrována. Cílem testů bylo ověřit funkčnost celého systému pro případné reálné nasazení a zároveň změření některých charakteristik.

Využití směrovací cache.

Pro první test jsem použil síť o velikosti 100 uzlů a mobilitu záznamu 0.001, tedy téměř statické záznamy. Cílem bylo ukázat, jak velikost dynamické cache ovlivňuje latenci vyhledávaní.

Test byl prováděn s aplikací RouteCacheDhtApp pro různé velikosti cache: 0 (s aplikací BasicDhtApp), 1, 50, 100, 250, 500, 750, 850 a 950 záznamů. Obrázek 17 ukazuje graf latence pro síť o 100 uzlech, mobilitou záznamů 0.001 a pro velikosti cache 0, 1, 50, 100, 250 a 500.



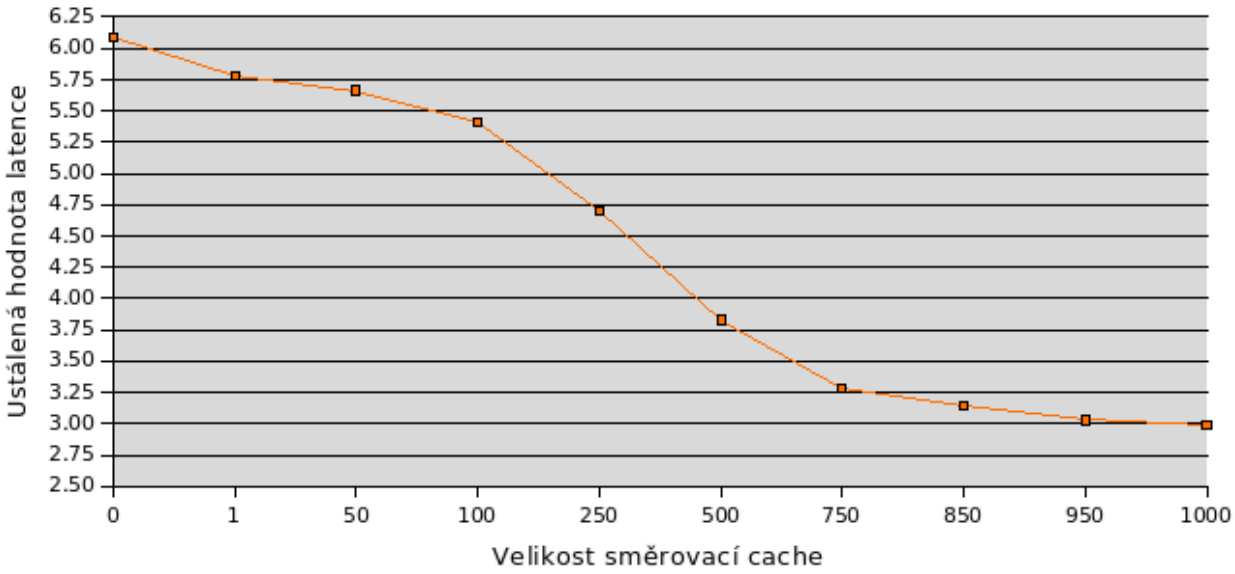
Obrázek 17: Závislost latence na velikosti směrovací cache. (Síť o 100 uzlech, mobilita záznamů 0.001)

Z obrázku 17 plyne, že již velikost cache 1 má poměrně velký vliv na výslednou latenci. Hodnota latence se ustálí přibližně při 95% naplnění cache. Tabulka 1 ukazuje ustálenou hodnotu latence při naplnění cache a počet kroků, po kterých je cache plná. Maximální velikost cache je 1000, to je i počet záznamů v našem testu. Pokud by došlo k zaplnění této cache, znamenalo by to, že každý uzel má kompletní směrovací informace pro všechny záznamy. Uzel, který hledá záznam, se může spojit přímo s vlastníkem záznamu. Taková operace zabere tři kroky (1. odeslaní dotazu 2. zpracování dotazu a odeslání odpovědi 3. zpracování odpovědi). Ve skutečnosti to bude o trochu méně než tři, pokud se uzel ptá na záznam jehož je sám vlastníkem, celá operace proběhne ve dvou krocích. Bohužel simulaci, která by se dostala až k zaplnění maximální velikosti cache, nebylo možné z důvodů vysokých nároků na výpočetní čas provést. Hodnota uvedená v tabulce odpovídá údaji při zaplnění přibližně 99% cache, po 300 000 krocích simulace.

Cache	Latence	Naplnění
0	6.084210	0
1	5.784000	2000
50	5.663664	4000
100	5.406500	5500
250	4.700949	11000
500	3.828657	22500
750	3.283567	42000
850	3.144717	59000
950	3.029515	170000
1000	2.990486	(300000)

Tabulka 1: Ustálená hodnota latence a bod naplnění pro různé velikosti cache.

Lepší představu o tom jak závisí latence na velikost cache je možné si udělat z obrázku 18.



Obrázek 18: Závislost latence na velikosti směrovací cache (Síť o 100 uzlech, mobilita záznamů 0.001).

Dalším důležitým kriteriem je mobilita záznamu. Vzhledem k navržení směrovací cache, nemá mobilita záznamu žádný vliv na efektivitu směrovací cache. To je pro návrh DHT sítě jako služby pro lokalizaci mobilních uzlů podstatné.

Doposud prováděné testy simulovaly síť o velikosti 100 uzlů. Velikost sítě má velký význam na výslednou latenci. Teorie o DHT hovoří o tom, že pro směrování na DHT vrstvě potřebujeme *log(n)* kroků (neplatí zcela pro CAN), kde n je počet uzlů v síti. Musíme také počítat s jedním přeskokem při doručení odpovědi. Tedy celkem *log(n)+1* přeskoků na DHT vrstvě, každý z nich zabírá jeden krok. V naší simulaci zabere jeden krok odeslání dotazu a jeden krok zpracování odpovědi. Platí tedy:

$$počet\ kroků = počet\ přeskoků + 3 = log(n) + 3$$

pokud je počet kroků menší než takto vypočtená hodnota, je to díky mechanismu dynamické cache. Zajímá nás, o kolik se výsledná latence vlivem cache snížila.

Efektivitu cache definujeme tímto vzorečkem:

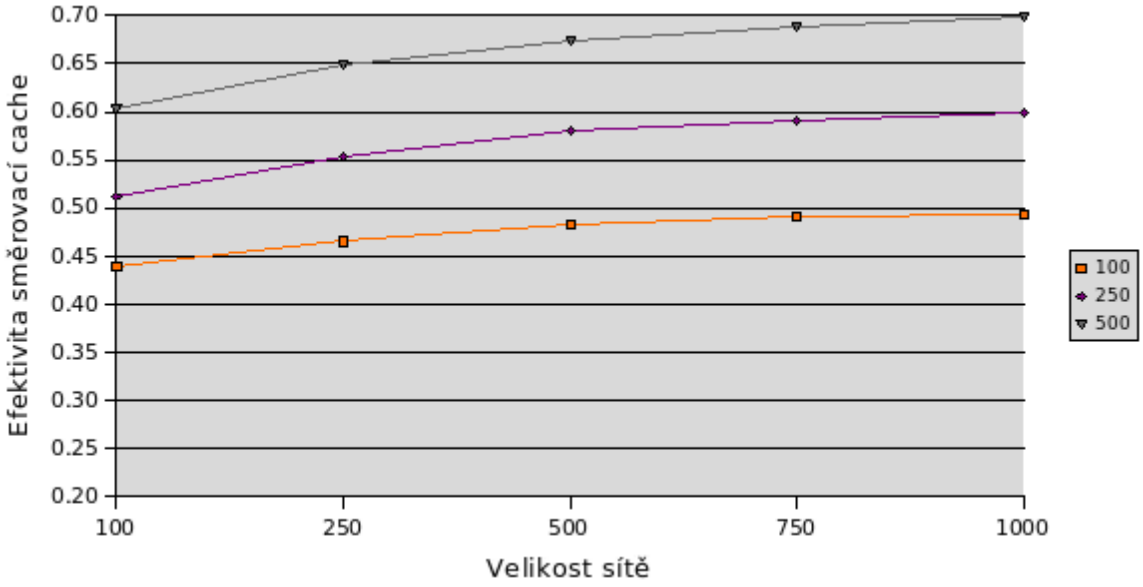
$$E = (100 - (100 * počet\ kroků) / (log(n) + 3)) [\%]$$

Určuje, o kolik procent pomohl mechanismus dynamické cache snížit latenci oproti očekávané hodnotě. Efektivity cache pro sítě o velikost 100, 250, 500, 750 a 1000 a cache velikosti 100, 250 a 500 jsou zachyceny v tabulce 2 spolu s naměřenou latencí.

Velikost sítě	Velikost Cache	100	250	500
100	**Latence**	5.406500	4.700949	3.828657
	Efektivita	43.94%	51.25%	60.30%
250	**Latence**	5.865634	4.895635	3.857357
	Efektivita	46.51%	55.36%	64.82%
500	**Latence**	6.183592	5.016016	3.915331
	Efektivita	48.32%	58.08%	67.28%
750	**Latence**	6.387275	5.143932	3.911412
	Efektivita	49.11%	59.01%	68.84%
1000	**Latence**	6.563691	5.187594	3.897397
	Efektivita	49.38%	59.99%	69.94%

Tabulka 2: Efektivita směrovací cache pro různé velikostí sítě a cache.

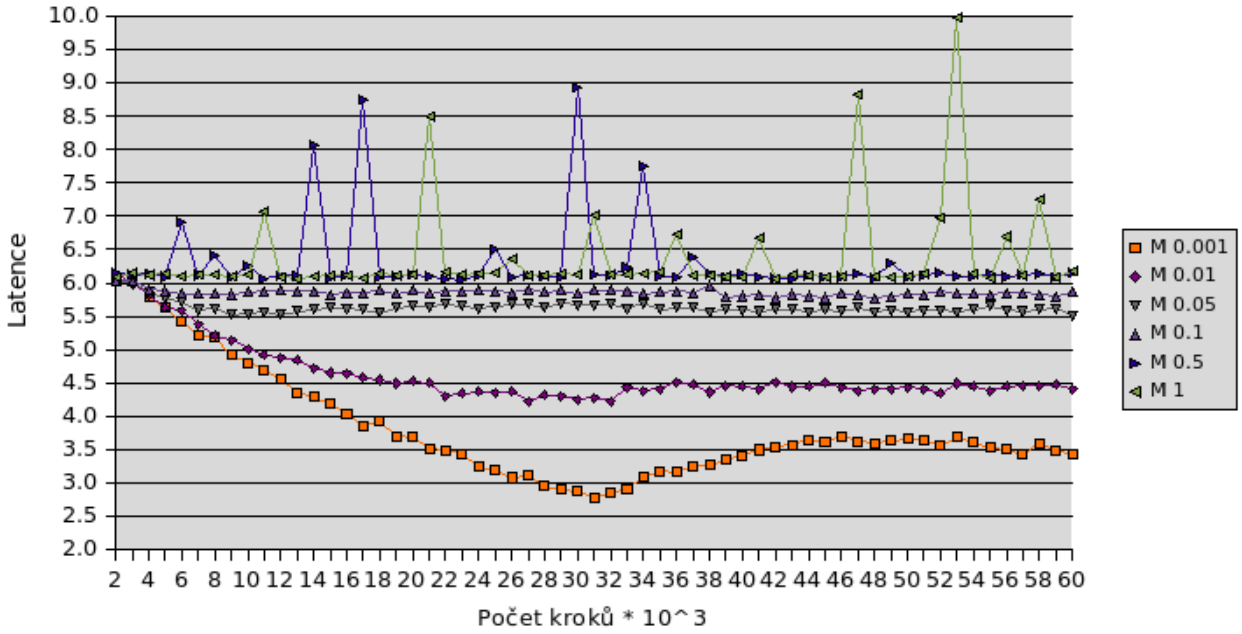
Vývoj efektivity zachycuje graf na obrázku 19.



Obrázek 19: Vývoje efektivity směrovací cache v závislosti na velikosti sítě.

Využití hodnotové cache.

U hodnotové cache má pro výslednou latenci největší význam mobilita záznamů. Graf na obrázku 20 ukazuje vývoj latence pro mobility 0.001, 0.01, 0.05, 0.1, 0.5 a 1. Simulace byla prováděna na síti o 100 uzlech.



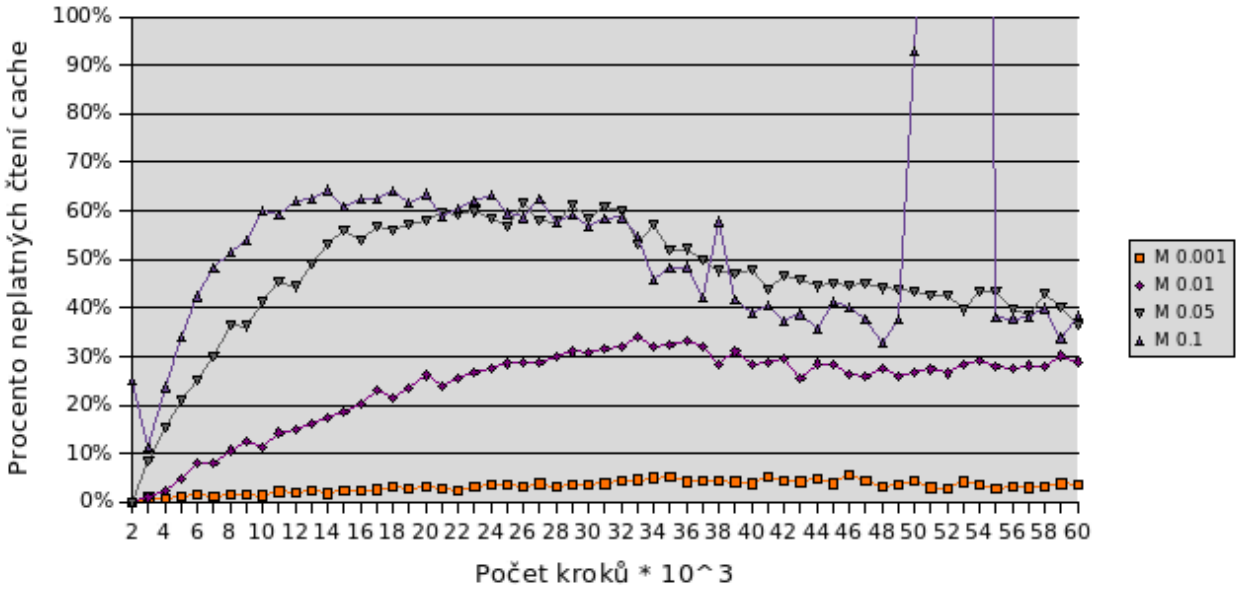
Obrázek 20: Závislost latence na mobilitě záznamů při využití hodnotové cache v síti o 100 uzlech.

Je vidět, že efektivita hodnotové cache poměrně strmě klesá s rostoucí mobilitou záznamů. Při mobilitě nad 0.05 (20 dotazů na jednu registraci) je její efekt již zanedbatelný. Při mobilitách nad 0.1 dochází k situaci, kdy krátkodobě latence prudce vzroste. To je způsobeno častým výskytem neplatné položky v cache a nutnosti opakování celého dotazu. Vzhledem k dynamičnosti záznamu se může stát, že dotaz se opakuje i několikrát.

Obrázek 21 ukazuje, kolik procent z celkového počtu čtení z hodnotové cache vrátilo hodnotu, která již neodpovídá skutečnosti a je nutné opakovat dotaz.

Graf je omezen na hodnoty do 100%, ale již při mobilitě 0.1 je tato hodnota překročena. To znamená, že dochází k situaci, kdy se dotaz opakuje vícekrát. Pro vyšší mobility je situace ještě horší. Mobilita 0.1 se dá považovat za hraniční mobilitu, kdy má použití cache v síti o 100 uzlech ještě smysl.

Nyní budeme zkoumat efektivitu cache tak, jak byla definována v kapitole o směrovací cache. Tabulka 3 ukazuje změřené efektivity pro sítě o 100, 250, 500, 750 a 1000 uzlech a pro mobility do 0.1.



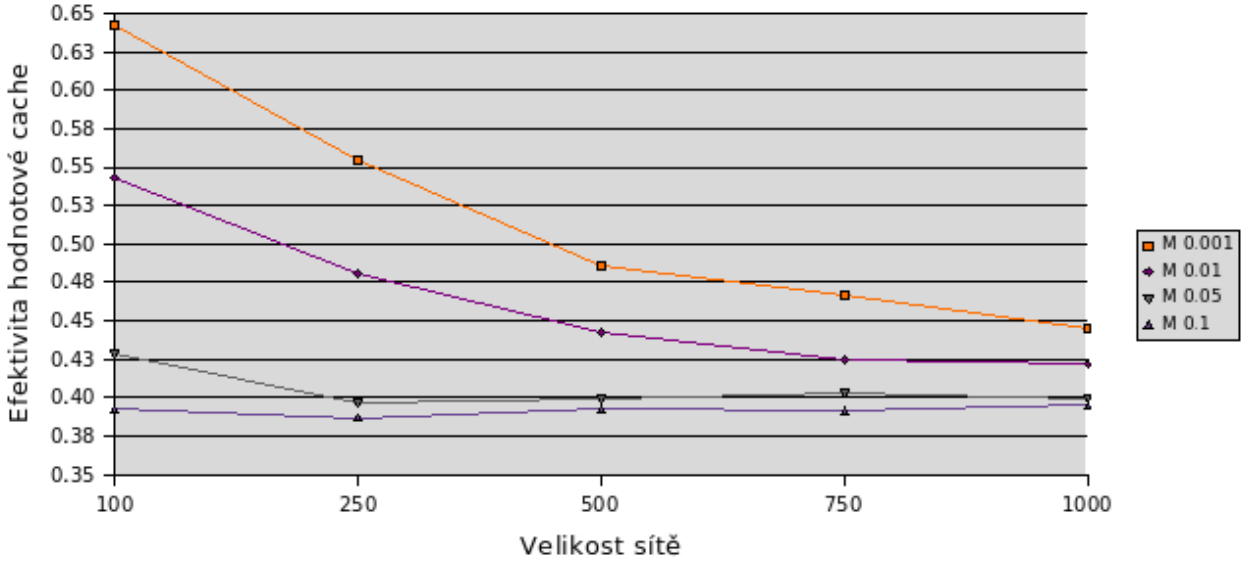
Obrázek 21: Procento zásahů do hodnotové cache, které vrátili neplatnou hodnotu. Simulace na síti o 100 uzlech.

Velikost sítě	Mobilita	0.001	0.01	0.05	0.1
100	**Latence**	3.444500	4.406068	5.509557	5.861210
	Efektivita	64.28%	54.31%	42.87%	39.22%
250	**Latence**	4.883500	5.692697	6.613553	6.721749
	Efektivita	55.47%	48.09%	39.69%	38.70%
500	**Latence**	6.154328	6.675082	7.191300	7.260726
	Efektivita	48.57%	44.22%	39.90%	39.32%
750	**Latence**	6.687172	7.219499	7.487958	7.637393
	Efektivita	46.72%	42.48%	40.34%	39.15%
1000	**Latence**	7.192346	7.491550	7.785452	7.840123
	Efektivita	44.53%	42.22%	39.95%	39.53%

Tabulka 3: Efektivita hodnotové cache pro různé velikosti sítě a mobility záznamů.

Vývoj efektivity zachycuje graf na obrázku 22.

Za povšimnutí stojí, že zatímco u směrovací cache s velikostí sítě efektivita cache stoupala, zde je tomu naopak.



Obrázek 22: Vývoj efektivity hodnotové cache v závislosti na velikosti sítě.

Porovnání směrovací a hodnotové cache

Z tabulek zachycujících efektivitu jednotlivých typů cache je vidět, že hodnotová cache je výhodnější jen při nejnižší uvažované hodnotě mobility. Důležitým faktorem je ale také počet položek v cache, které jsou nutné pro jednotlivé přístupy. Tabulka 4 porovnává efektivitu a maximální počet položek v cache pro hodnotovou a směrovací cache pro různé mobility. Maximální počet položek v cache je chápán jako maximální součet přes všechny uzly dosažený v průběhu simulace. Poslední sloupec pak přepočítává efektivitu na jednu položku v cache.

U hodnotové cache efektivita klesá se zvyšující se mobilitou. Také počet položek v cache klesá, protože klesá i TTL jednotlivých záznamů. Na směrovací cache nemá mobilita žádný vliv, rozhoduje pouze velikost směrovací cache.

Pokud nám jde o dosažení co největší možné efektivity, zvolíme pravděpodobně směrovací cache. Pouze pokud očekáváme mobilitu menší než 0.001 je lepší volbou cache hodnotová. V případě, že chceme minimalizovat prostor, který cache zabírá, můžeme použít hodnotovou cache i pro vyšší mobility, kde dosahuje lepšího poměru efektivita/velikost.

Mobilita 0.001, Síť 100 uzlů			
Typ	**Efektivita**	**Maximum položek v cache**	**Na položku cache**
ValueCache	64.28%	69870	0.00092%
RouteCache 100	43.94%	10000	0.00439%
RouteCache 250	51.25%	25000	0.00205%
RouteCache 500	60.30%	50000	0.00121%
Mobilita 0.01, Síť 100 uzlů			
Typ	**Efektivita**	**Maximum položek v cache**	**Na položku cache**
ValueCache	54.31%	61283	0.00089%
RouteCache 100	43.94%	10000	0.00439%
RouteCache 250	51.25%	25000	0.00205%
RouteCache 500	60.30%	50000	0.00121%
Mobilita 0.05, Síť 100 uzlů			
Typ	**Efektivita**	**Maximum položek v cache**	**Na položku cache**
ValueCache	42.87%	37375	0.00115%
RouteCache 100	43.94%	10000	0.00439%
RouteCache 250	51.25%	25000	0.00205%
RouteCache 500	60.30%	50000	0.00121%
Mobilita 0.1, Síť 100 uzlů			
Typ	**Efektivita**	**Maximum položek v cache**	**Na položku cache**
ValueCache	39.22%	22382	0.00175%
RouteCache 100	43.94%	10000	0.00439%
RouteCache 250	51.25%	25000	0.00205%
RouteCache 500	60.30%	50000	0.00121%

Tabulka 4: Porovnání efektivit jednotlivých cache.

h) Shrnutí

Navržený systém nemá ambice zcela nahradit stávající systém DNS. Jeho cílem je DNS doplnit o dynamickou databázi mobilních uzlů. V mnoha případech se opírá o záznamy, které DNS poskytuje. Hlavní účel celého systému je umožnit lokalizovat mobilní uzly v Internetu bez toho, aby se zasahovalo do adresního schématu sítě. Vylepšuje řešení, které poskytuje samotné DNS (popsané v kapitole 3.3) a činí ho nezávislým na výpadcích libovolných uzlů i na snaze systém regulovat.

Z časového hlediska je nejnáročnější směrování dotazu přes DHT. Navržená implementace se snaží využití tohoto směrování minimalizovat. V každé akci se provádí pouze jednou, ostatní komunikace se odehrává přes IP. Také přínos směrovací cache je především v tom, že umožní zkrátit směrování dotazu přes DHT.

Velký význam bude mít i zvolená DHT technologie, která musí co nejlépe odrážet strukturu navrženého systému.

Technologie kademlia je navržena tak, aby odolala i poměrně vysoké fluktuaci uzlů a tomu odpovídá počet replikování záznamů. Je tedy optimalizována spíše pro peer-to-peer sítě a pro náš systém nevhodná.

U sítě typu CAN je zase problém, že směrování trvá příliš dlouho. A pokud chceme dosáhnout obvyklé latence *log(n),* musíme parametr *d* volit podle velikosti sítě. V [27] byla provedena simulace CAN sítě s 2^{18} uzlů s topologií,

kde na páteřní síti byla latence 100ms, okrajové sítě byly připojeny s latencí 10ms a mezi uzly v okrajových sítích byla latence 1ms. Po aplikování všech optimalizačních metod bylo zapotřebí mezi dvěma uzly průměrně 5 přeskoků na DHT vrstvě, latence mezi dvěma uzly byla průměrně 82.4ms. To znamená, že čas na jedno DHT směrování je 412ms. Pokud přičteme čas na zpracování požadavku a směrování odpovědi přes IP, vejde se registrace, dotaz i notifikace pod 1s.

Jako nejlépe použitelné metody se jeví Chord nebo Pastry.

3.4.5 Využití DHT pro směrování (Route-Based přístup)

Hlavní myšlenka tohoto přístupu je začlenění převodu mezi NodeID a LocID přímo do směrovacích algoritmů sítě. Koncové uzly od transportní vrstvy výše budou používat jako identifikaci pouze NodeID. Každý uzel v takovéto síti musí být členem globální DHT a mít tedy přiřazené své jednoznačné NodeID. Zprávy mezi jednotlivými uzly jsou pak směrovány podle směrovacích tabulek použité DHT technologie.

Mechanismus, který je u DHT použit pro lokalizování dat, používáme i pro lokalizaci uzlu. Pokud je cílový uzel nedostupný, skončí paket na uzlu, který je nejblíže cílovému uzlu (ve smyslu funkce δ). Takový uzel musí poslat odesílatelovy zprávu o nedostupnosti příjemce.

Aby paket došel svému příjemci, používá dva nezávislé mechanismy směrování. Mezi dvěma DHT uzly se využívá směrování podle IP adresy (IP směrování) a mezi zdrojovým a cílovým uzlem směrování podle NodeID (DHT směrování). Oba dva typy směrování se vzájemně doplňují, IP směrování odráží topologii přenosové sítě a DHT směrování fyzické umístění jednotlivých uzlů v rámci této topologie.

Pokud se uzel přesouvá (mění své LocID), musí o tom informovat všechny uzly, které na něj ukazují ve svých DHT směrovacích tabulkách. Tuto informaci ale nemá ve všech DHT technologiích uzel automaticky k dispozici.

Zahrnutí všech uzlů do DHT v sobě skrývá několik nebezpečí:

- Vzdálenost mezi dvěma sousedními uzly na DHT vrstvě by mohla být na IP vrstvě neúměrně veliká. Pokud by nějaký přenos využíval tyto dva uzly, docházelo by k velkému (a zbytečnému) zpoždění. Sousedními DHT uzly myslíme takové uzly, kde jeden ukazuje na druhý ve svých DHT směrovacích tabulkách.
- Komunikace mezi dvěma koncovými uzly by mohla být směrována přes uzel, který je připojen linkou s malou kapacitou. Přestože oba koncové uzly by byly připojeny na rychlé lince, jejich vzájemná komunikace by byla omezena touto pomalou linkou. Navíc uživatel, který má zpoplatněný přenos dat, nebude rád, když přes jeho (koncovou) linku poteče cizí komunikace.
- Většina uzlů v Internetu je pro účely přenosu dat nespolehlivá. Znamená to, že mohou být náhodně vypnuty nebo přesunuty. Takovéto případy

negativně ovlivňují přenosy odehrávající se na těchto uzlech. Nezanedbatelné je i riziko odposlechu komunikace.

To klade velké nároky na použitou DHT technologii a hlavně na její optimalizační mechanismy. Optimalizace musí využívat metriku postavenou na vzdálenosti mezi IP uzly, rychlosti (ceny) linky i spolehlivosti uzlu.

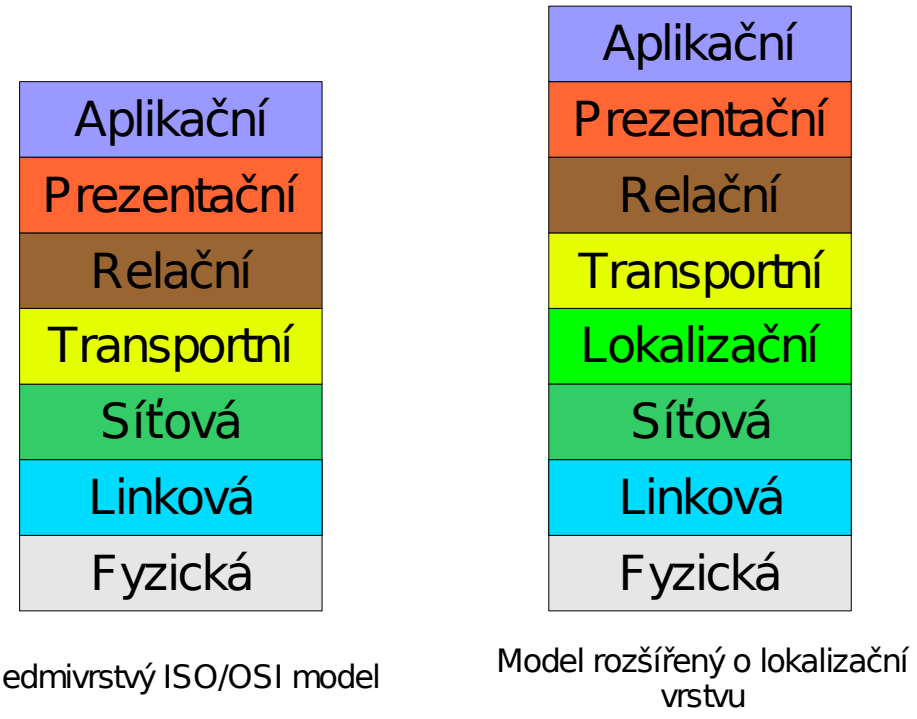
a) Návrh implementace

Návrh počítá s rozšířením tradičního protokolového modelu o další vrstvu. Mezi síťovou a transportní vrstvou by vznikla další vrstva, která by měla na starosti směrování podle NodeID. Nazveme ji lokalizační vrstva (na obrázku 23 je označena zeleně).

Lokalizační vrstva přidá do hlaviček odesílaného paketu informaci o NodeID odesílatele a příjemce. Podle NodeID příjemce vyhledá IP adresu DHT uzlu, na který má být paket v následujícím kroku směrován a předá tuto informaci síťové vrstvě.

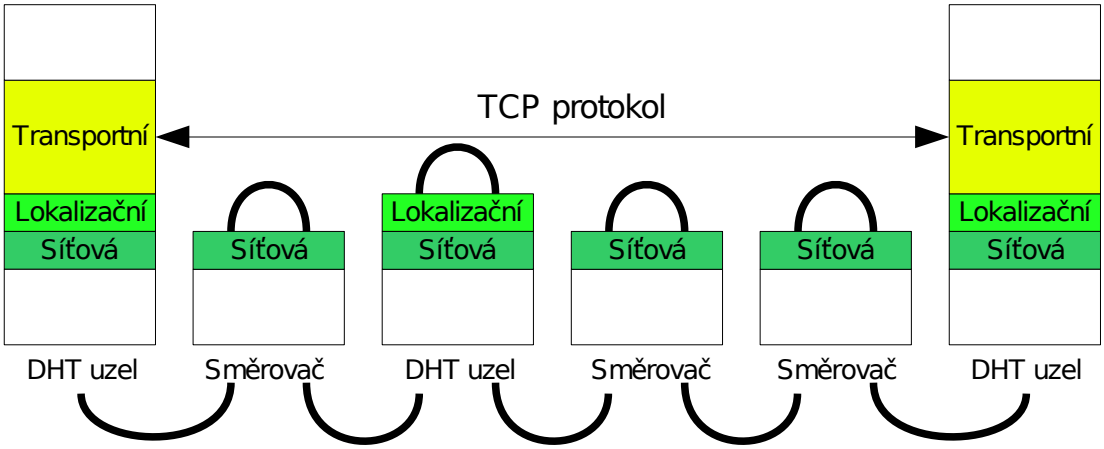
Příjemce paketu s lokalizační hlavičkou nejprve rozhodne, jestli je paket určen jemu. Pokud ano, předá ho vyšším vrstvám. Pokud ne, předá ho dalšímu uzlu na cestě k cílovému uzlu, případně oznámí odesílateli nedostupnost cílového hostitele.

Celý proces přenosu paketu je ilustrován na obrázku 24. Přidání nové vrstvy způsobí nárůst režie přenosu. Jednak dojde ke zvětšení hlaviček přenášených paketů (toto zvětšení je však zanedbatelné) a pak dojde také k prodloužení doby přenosu paketu a to až několikrát.



Obrázek 23: Síťový model v route-based přístupu

Právě prodloužení přenosové doby je největší slabinou tohoto přístupu. V síti s přibližně jedním milionem uzlů by za předpokladu neoptimalizované DHT sítě bylo potřeba asi dvacet přeskoků na lokalizační vrstvě. Pokud bychom nemohli nic předpokládat o rozmístění DHT uzlů v rámci IP sítě, vyjde nám zhruba dvacetinásobné prodloužení přenosové doby. A to je něco s čím se nemůžeme smířit.



Obrázek 24: Přenos paketu v síti s použitím route-based přístupu.

V každé DHT technologii existuje postup optimalizace směrování. Jeho základem je stanovení ceny jednotlivých linek k dalším DHT uzlů a na jejich základě rozhodovat o směrování. Úkol najít a dokázat změřit vhodnou metriku, na jejímž základě bychom mohli stanovit cenu linky, není úplně jednoduchý.

b) Optimalizace

Nabízí se několik metrik, podle kterých lze stanovit cenu linky. Linkou v tomto kontextu máme namysli IP spojení mezi dvěma sousedními DHT uzly. Linka tedy může procházet přes více IP uzlů.

Uvažujme následující metriky:

- Latence (zpoždění) – čas, za který data doputují k cílovému uzlu. Označme *L*.
- Přenosová rychlost – počet bitů, které je linka schopna přenést za jednotku času. Označíme *R* .
- Nespolehlivost – procento nedoručených paketů za jednotku času. Například díky výpadkům IP uzlů na lince nebo nekvalitnímu spojení mezi IP uzly. Označíme *S*.

Jakým způsobem může uzel tyto metriky změřit?

V případě měření latence lze použít ICMP pakety echo request a echo reply (tedy klasický ping). V případě přenosové rychlosti se lze spoléhat na průběžné měření přenesených dat nebo udělat jednorázový test. V případě

měření spolehlivosti můžeme také použít nárazový test nebo výsledky průběžně získávat během přenosů a spolehnout se, že nedoručené pakety jsou hlášeny pomocí ICMP zpráv.

Je třeba dát pozor, aby výše zmíněné testy příliš nezahltily síť. Pokud uzly na koncích linky nemění svojí IP adresu, není příliš pravděpodobné, že se budou měnit parametry linky. Vyloučit to ale nelze, například změna směrování nebo zvýšení/snížení kapacity spoje. Je tedy vhodné zjišťovat parametry linky pouze v případě změny adresy koncových uzlů. Narazíme ale na dva problémy:

1. V případě, že se uzel pohybuje příliš rychle, budou opakované testy příliš zatěžovat síť.
2. V případě, že se uzel nepohybuje vůbec, nebude docházet k měření a změřené parametry mohou zastarat.

Zavedeme dvě omezení, která by měla tento problém řešit.

1. Maximální počet testů za jednotku času. Pokud se uzel pohybuje příliš rychle, nebudou už testy prováděny. Označme T_{MAX}.
2. Minimální počet testů za jednotku času. Pokud se uzel nepohybuje vůbec, bude přeci jen jednou za čas test proveden. Označme T_{MIN}.

Z toho vyplývá, že u příliš mobilních uzlů se nelze příliš spolehnout na naměřené hodnoty. Přes takové uzly není vhodné přenášet data k dalším uzlům.[12] Přirozeným způsobem se tedy dostáváme k další metrice, kterou bychom měli uvažovat:

- Mobilita – kolik změn IP adres provedl uzel za jednotku času. Například za poslední hodinu. Označme M.

Jakým způsobem může uzel na základě metrik určit cenu linky?

Předpokládejme, že požadavky na přenos jsou dvojího druhu. Jednak přenos s co nejmenším zpožděním, ale s menšími nároky na přenosovou kapacitu (např. IP telefonie) a pak přenos s velkými nároky na přenosovou kapacitu, ale velikost zpoždění bude vedlejší (např. přenos velkých souborů). Pro oba tyto druhy přenosů je vhodné volit jinou metodu výpočtu ceny linky.

c) Praktické důsledky

Pokud bychom chtěli implementovat uvedené řešení, narazíme pravděpodobně na dva zásadní problémy. Nárůst zátěže směrovacích DHT uzlů a na nárůst zátěže sítě. Otázkou je, nakolik tyto dva handicapy ovlivní použitelnost tohoto přístupu.

Na rozdíl od předchozího přístupu je zde zajištěna plná anonymita fyzické polohy uzlu. Komunikující uzly nemají žádnou možnost zjistit LocID svého protějšku (tedy pokud nejsou DHT směrovače s přímou linkou mezi sebou).

Další výhodou je minimální nutnost zasahovat do protokolů transportní vrstvy. Jediné, co je potřeba změnit, jsou údaje identifikující koncové uzly v

12 Předpokládáme, že v síti je více statických uzlů a směrování přes ně je výhodnější.

hlavičkách protokolů. Tedy dnes používané LocID nahradit za (na přesunu nezávislé) NodeID.

Souběžné přesuny obou komunikujících stran nejsou ani v tomto přístupu problémem.

Vzhledem ke značným problémům, které tento přístup přináší, není vhodné ho nasazovat ve velkých sítích jako je Internet. Přínos by snad mohl mít pouze v menších komunitních sítích v rámci jedné velké sítě. Museli bychom se ale smířit s velkými latencemi.

4. Mobilita v bezdrátových sítích

V dnešní době existuje velké množství technologií pro bezdrátový přenos dat. Liší se dostupnou přenosovou rychlostí, rozsahem frekvencí, dosahem i stupněm podpory mobility.

Mobilitu v bezdrátových sítí můžeme chápat trochu jinak, než jak jsme o ní dosud mluvili. Změna fyzického umístění uzlu v rámci dosahu jednoho vysílače je vlastně samozřejmostí, i když rychlost této změny může ovlivnit přenosové parametry. Možnost pohybu v oblastech pokrytých různými vysílači (základnovými stanicemi - BS) už vyžaduje navrhnout příslušné protokoly. Proces přesunu mobilní stanice (MS) z jedné základnové stanice na druhou se nazývá handover. Rychlost tohoto procesu je pro mobilitu důležitá. Pokud se handover stihne do 50ms bez přerušení toku dat, mluvíme o takzvané seamless handoveru.

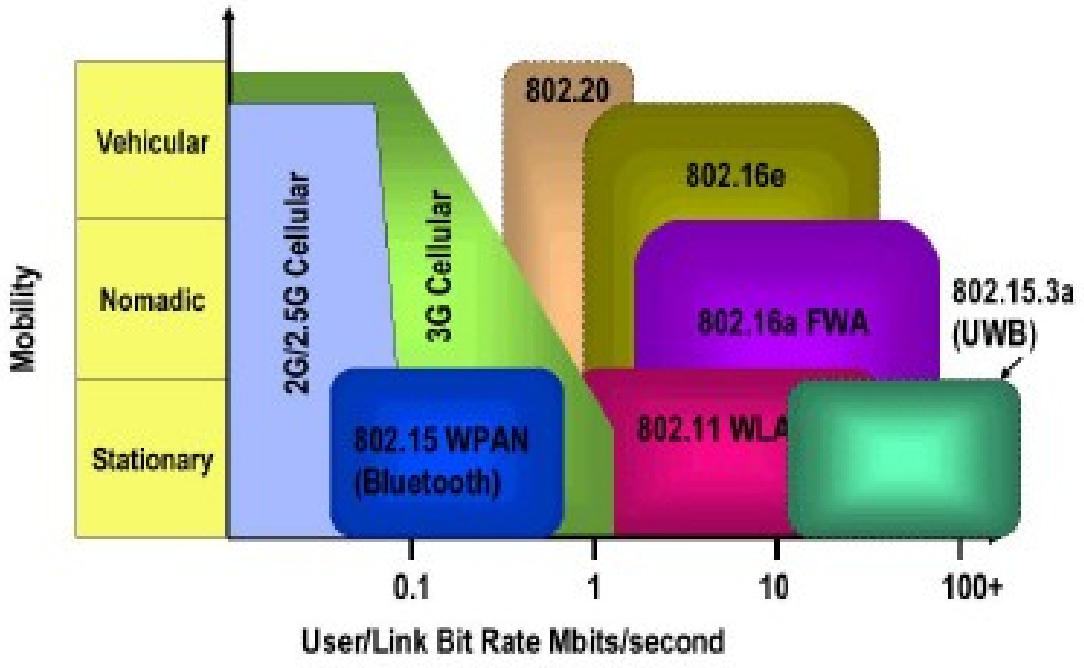
U mobilních stanic připojených přes bezdrátovou síť lze předpokládat, že se budou vůči základnové stanici pohybovat, a to i velkou rychlostí. Právě rychlost pohybu mobilních stanic, při které je síť ještě schopna zajišťovat mobilitu, je dalším důležitým kriteriem pro mobilní bezdrátové sítě. V tomto kontextu mluvíme o stacionárních uzlech, kočujících (nomadic) nebo pojízdných (vehicular).

Většinu času jsou mobilní zařízení napájena z přenosné baterie, proto je cílem výrobců těchto zařízení snížit jejich spotřebu na minimum. Součástí standardu každé mobilní bezdrátové sítě je i power management, který má za úkol snížit spotřebu elektrické energie potřebné pro obsluhu bezdrátového připojení.

V předchozích částech jsme mluvili o mobilitě na úrovni IP vrstvy (případně transportní vrstvy). V bezdrátových sítích se bude jednat o mobilitu spíše na úrovni linkové vrstvy. Ta se uplatní v případě, kdy se MS přesouvá mezi jednotlivými BS stejného poskytovatele. Pro zajištění skutečně globálního mobilního přístupu je nezbytné, aby mobilita na linkové i síťové vrstvě spolupracovala.

V souvislosti s mobilitou v bezdrátových sítích se objevuje další pojem. Roaming, tedy možnost využít infrastrukturu jiné sítě, než ve které je zařízení registrováno (domovské sítě). Roaming nastává v případě, když uživatel jedné sítě začne využívat služeb jiné sítě, aniž by měl s touto druhou sítí uzavřenou nějakou dohodu. Dohodu za uživatele uzavírají provozovatelé jednotlivých sítí a nemusí se jednat pouze o bezdrátové sítě. V rámci roamingu je nutné řešit otázky autorizace, zpoplatnění a bezpečnosti. Vzhledem k tomu, že neexistuje jedna celosvětová bezdrátová síť, je pro globální mobilitu roaming nutnou podmínkou. Někdy je termínem roaming označován také přechod zařízení z jednoho přístupového bodu na druhý i v rámci jedné bezdrátové sítě.

Obrázek 25 porovnává mobilitu a přenosové kapacity nejpoužívanějších technologii.



Obrázek 25: Přehled mobility a přenosové kapacity bezdrátových technologii. (zdroj: WiMAX Forum)

4.1 Handover

Handover je důležitý proces v mobilní bezdrátové síti, který umožňuje mobilním uzlům přesunout se z jedné základnové stanice na druhou. Používá se nejčastěji v případech, kdy mobilní uzel změnil své fyzické umístění tak, že nemůže nadále přijímat signál z původní základnové stanice. Základnová stanice, ke které byl uzel připojen před zahájení handoveru, se nazývá obslužná BS, Stanice, ke které se uzel chce připojit, budeme nazývat cílová BS.

Rozlišujeme dva základní druhy handoveru:

Hard handover

Tento typ handoveru se často také nazývá break-before-make. MS nejprve přeruší spojení s obslužnou BS a teprve pak naváže spojení s cílovou BS. Než začne MS komunikovat s cílovou BS, musí se k ní nejprve přihlásit. To také zabírá nějaký čas.

Používá se v případě, že není dostupný soft handover nebo pokud se přepojujeme mezi různými typy sítí. Je to nejjednodušší typ handoveru, který musí být vždy implementovaný.

Soft handover

Nazývá se také make-before-break. MS nejprve naváže spojení s cílovou BS, provede registraci a teprve v případě, že vše proběhne korektně, odpojí se od obslužné BS. Mobilní zařízení je po celou dobu handoveru připojené do komunikační sítě, proto je tento způsob více šetrnější k vyšším vrstvám. Na druhou stranu stoupne zatížení základnových stanic a celý proces je složitější

na implementaci.

Na handover se lze dívat i z pozice architektury sítě, pak ho rozlišujeme na dva typy:

Horizontal handover

Jedná se o přepojení mezi BS jedné bezdrátové technologie. Většinou je součástí standardu každé mobilní bezdrátové technologie. Může být jak soft tak hard.

Vertical handover

Zde se přepojuje mezi základnovými stanicemi různých technologií. Jeho implementace bývá zpravidla těžší, protože musí odrážet standardy obou technologii. Většinou se volí jednoduší přístup a použije se hard handover, ale je možné použít i soft handover.

Existují standardy, které se vertical handover snaží řešit obecně (např. IEEE 802.21), ale i návrhy, které počítají s propojení dvou konkrétních bezdrátových sítí.

4.2 Technologie bezdrátových sítí

Zaměříme se na některé konkrétní technologie:

4.2.1 Osobní bezdrátové sítě (WPAN).

Tyto sítě se vyznačují malým dosahem (řádově desítky metrů) a jsou určeny spíše pro domácí použití. Pro komunikaci mezi mobilními komunikačními zařízeními (mobilní telefony, PDA, ..) , periferiemi (zejména tiskárnami) či počítači.

První standard, který v této oblasti vznikl, byl průmyslový standard bluetooth. Později převzato jako standard IEEE 802.15.1. Skupina IEEE 802.15 se zabývá právě WPAN a z její dílny vyšly i další standardy 802.15.3 – WPAN High Rate, 802.15.4 – WPAN Low Rate.

Dosah bluetooth je 10 metrů. Pokud chceme zajistit pokrytí většího území, musíme vybudovat infrastrukturu. Jednu možnost popisuje Valaisanne v článku [32]. Je založena na budování složitějších bluetooth sítí (tzn. scaternetů). Jiný přístup se snaží zkombinovat bluetooth a WiFi. Tento přístup je rozveden v článku [33]. Základní myšlenka je vybavit některá zařízení oběma komunikačními rozhraními (Bluetooth i WiFi). Tyto zařízení (Bluetooth Wireless Gateway – BWG) pak zprostředkovávají přístup do sítě ostatním bluetooth zařízením. Celý systém umožňuje omezenou mobilitu pro bluetooth zařízení. Stále pouze na poměrně malém prostoru – například přednášková místnost. Bohužel bluetooth a WiFi používají stejné frekvenční pásmo a je potřeba řešit otázku případného rušení.

Vzhledem ke krátkému dosahu a primárnímu určení těchto sítí, není otázka

mobility příliš aktuální. Hovořit o maximální rychlost při jaké je mobilita podporována nemá smysl.

4.2.2 Lokální bezdrátové sítě (WLAN)

V současné době nejpoužívanější lokální bezdrátovou sítí je WiFi (podle standardu IEEE 802.11). Základní verze standardu (802.11a, 802.11b a 802.11g) roaming podporují, jeho implementace je však příliš pomalá a hodí se pouze pro data. Čas na předání uzlu mezi dvěma stanicemi se pohybuje v řádu stovek milisekund.

Pro účely VOIP (voice over IP) je vyvíjen standard 802.11r (Fast Roaming), který by měl zvládnout předaní do 50ms. Tento čas je nezbytný, aby kvalita hovoru vnímaná lidským uchem nebyla narušena. Roaming v základních verzích standardu trpí ještě jedním nedostatkem. Mobilní zařízení nemůže vědět, jestli na novém přístupovém bodu jsou k dispozici QoS (Quality of Service) prostředky, které využíval na původním přístupovém bodu a které potřebuje pro danou aplikaci. To může být u VOIP problém.

Protokol 802.11r umožňuje mobilnímu zařízení autorizovat se vůči cílovému přístupovému bodu a zarezervovat si QoS prostředky ještě v době, kdy je připojen ke obslužnému přístupovému bodu. Může dokonce využít obslužný přístupový bod jako prostředníka pro komunikaci s cílovým přístupovým bodem. Poté co se fyzicky připojí k cílovému přístupovému bodu pouze potvrdí předchozí registraci. Pro tento proces jsou důležité i dvě další normy 802.11i (Enhanced security) a 802.11e (QoS).

Pro komunikaci mezi přístupovými body se používá protokol IAPP (Inter-Access Point Protocol) normalizovaný v IEEE 802.11f. Obslužný přístupový bod jeho prostřednictvím informuje cílový přístupový bod o novém klientovy. Obslužný přístupový bod si musí MAC adresu klienta vymazat ze své asociační tabulky. Cílový přístupový bod informuje také mosty a přepínače připojené k lokálnímu segmentu, aby si aktualizovaly své tabulky MAC adres.

802.11r mluví o roamingu na druhé vrstvě síťové infrastruktury, tedy roaming mezi různými IP sítěmi nelze v tomto standardu použít.

4.2.3 WiMAX

Pracovní skupina IEEE 802.16 pracuje od roku 1999 na normě, která má být řešením pro pevný širokopásmový bezdrátový přístup. Až dosud se pro bezdrátové připojení domácností a firem používaly uzavřené firemní technologie. Vývoj nového otevřeného standardu má přispět k většímu rozvoji v této oblasti. V oblastech, kde není možné použít telefonní vedení nebo natáhnout optickou linku, je výhodné použít tuto technologii pro řešení poslední míle. Standardům z dílny této skupiny se začalo přezdívat WiMAX (Worldwide Interoperability for Microwave Access Forum).

První zajímavou normou je 802.16a, která používá pásma od 2 do 11 GHz, nepožaduje přímou viditelnost, podporuje QoS, je schopna komunikovat do vzdálenosti 3 – 5 km v husté zástavbě a až 40 km ve venkovských oblastech a

agregovanou rychlostí do 70 Mbit/s.

Pro nás je však důležitá norma 802.16e, která byla schválena v prosinci 2005 a hovoří o mobilním WiMAX. Specifikace vychází s technických principů 802.16a, ale zahrnuje podporu pro předávání stanice mezi buňkami a roaming. Cílem je podporovat jak pevný, tak mobilní přístup. Pracuje v nižších pásmech 2 až 6 GHz a nabízí i nižší rychlost okolo 3 až 5 Mbit/s. Podporuje mobilitu až do rychlosti 150 km/h v lokálním až regionálním dosahu s latencí menší než 50ms. Jako koncové zařízení se předpokládá laptop s omezenou možností napájení, takže 802.16e podporuje i řízení spotřeby a režim spánku.

Standardizační organizace IEEE se zaměřila pouze na fyzickou (PHY) a MAC (media access control) vrstvu. Původní specifikace byla rozšířena o vlastnosti nutné pro mobilní přístup. Nejdůležitější z nich je handover. 802.16e podporuje samozřejmě hard handoff a také dvě možnosti soft handoveru: Fast Base Station Switching (FBSS) a Macro Diversity Handover (MDHO). V obou případech si MS udržuje seznam všech dostupných BS (tzv. active set).

- FBSS – Mobilní uzel komunikuje pouze s jednou BS v active set. Data určená pro MS jsou ovšem doručována všem BS v active set. MS se tedy může kdykoliv přepnout na jinou BS a nepřijít tak o žádná data.
- MDHO – V tomto režimu může mobilní uzel přijímat a odesílat data z více BS ve stejný časový okamžik. BS vysílají k MS stejné data.

Protože IEEE se zabývalo pouze linkovou vrstvou technologie, WiMAX Forum přišlo se svým návrhem řešení síťové architektury. Obsahuje procedury a protokoly pro podporu mobility, zabezpečení, identifikace a účtování. Sítový model WiMAXu je na obrázku 26.

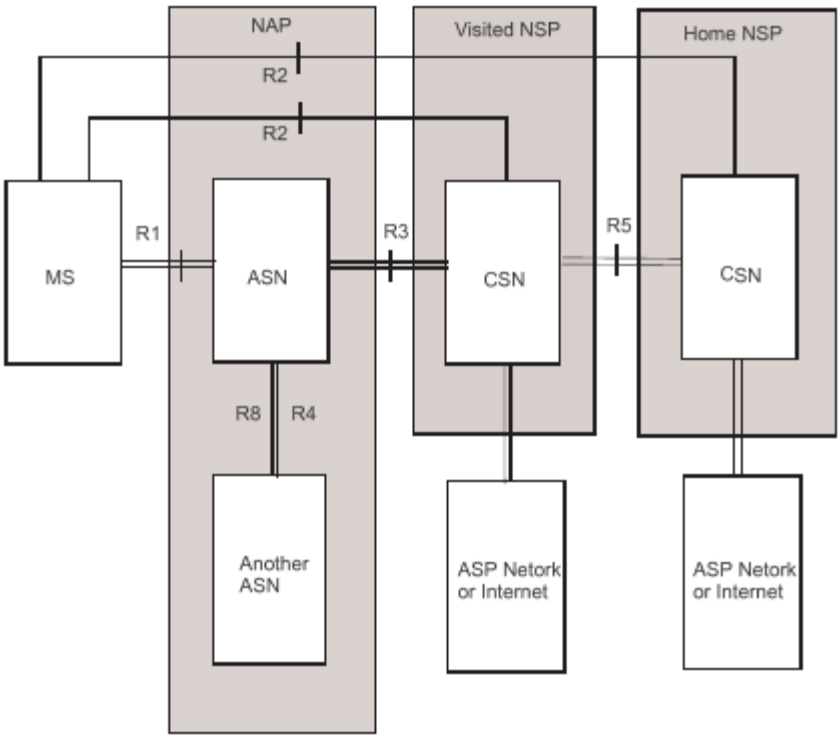
Obsahuje tyto části:

- MS – Mobile station.
- ASN – Access service network.
- CSN – Connectivity service network.
- ASP – Application service provider.

Access service network

Je tvořena jednou nebo několika ASN-gateway a základnovými stanicemi. Jejím úkolem je poskytnout rádiové pokrytí WiMAX sítí pro danou oblast. Obsluhuje MAC vrstvu, vyšší vrstvy přenechává na CSN.

Provozovatel této sítě je v terminologii WiMAX nazýván NAP (Network Access Provider). ASN-gateway slouží jako prostředník s CSN.



Obrázek 26: Síťový model WiMAX

Connectivity service network

CSN je vrstva, která poskytuje WiMAX stanicím připojení do IP sítě. Obsahuje brány, proxy servery, uživatelskou databázi, účtovací servery apod. Provozovatel této sítě se nazývá NSP (Network Service Provider). Koncový uživatel uzavírá smlouvu právě s tímto provozovatelem a pro připojení do jeho sítě využívá dostupnou ASN.

Rozdělení sítě na ASN a CSN přináší širší možnosti škálování. Jedna ASN může být připojena do více CSN a samozřejmě i naopak. Záleží pouze na dohodě jednotlivých provozovatelů.

Pokud se mobilní uzel přesouvá mezi BS v rámci jedné ASN, mluvíme o takzvaném Intra ASN Handoveru. Tento handover dokáže být velmi efektivní a rychlý. Pokud MS používá IP služby, nedojde ke změně IP adresy, protože přesun není vidět mimo danou ASN.

Situaci, kdy se MS přepíná mezi BS, které nejsou součástí jednoho ASN, nazýváme Inter ASN Handover. V tomto případě je nutná koordinace ASN-gataway z obou dotčených ASN.

Podrobnější seznámení s mobilním WiMAX lze nalézt například v [34].

4.2.4 MBWA

V rámci IEEE se pracuje ještě na jednom projektu, který je od začátku koncipován jako mobilní přístup. Pracovní skupina, která se jím zabývá, má

označení 802.20 a technologie je označována jako MBWA (Mobile Broadband Wireless Access). Má umožnit mobilním uživatelům, pohybujícím se až rychlosti 250 km/h, využívat širokopásmové služby rychlostí odpovídající kabelové nebo DSL přípojce. Maximální kapacita směrem k uživateli by měla být minimálně 1 Mbit/s a směrem od uživatele minimálně 300 kbit/s. V jedné buňce o průměru asi 15 km by měla být agregovaná rychlost 4 Mbit/s k uživateli a 800 kbit/s od uživatele.

Lze nalézt společné rysy s technologií 802.16e, obě jsou koncipovány jako rychlý mobilní přístup. Je tu však řada rozdílů například v použitém pásmu. 802.20 používá frekvence do 3,5 GHz. 802.16e je vybudováno na standardu 802.16a, kdežto skupina 802.20 bude své řešení budovat od začátku. Také 802.20 by měl podporovat mobilitu i v rychlovlacích (250 km/h) zatímco 802.16e se dá použit maximálně v automobilech (150 km/h). Standard 802.20 bude spíše přímou konkurencí mobilních sítí 3G.

4.2.5 Celulární mobilní sítě

Tyto sítě dosahují dnes již téměř globálního pokrytí a mobilita byla vždy jejich prvořadým cílem. Jsou však primárně určené pro přenos hlasu a tomu odpovídají i nízké přenosové rychlosti. Systém GSM nabízí nejvyšší přenosové rychlosti 384 kbit/s (EDGE). U sítí třetí generace (UMTS) je to sice až 2 Mbit/s, ale pouze pro stojícího uživatele. Pro pohybujícího se uživatele rychlost klesne na 384 kbit/s a pro rychle se pohybujícího dokonce na 144 kbit/s.

Přirozeným požadavkem uživatele je nepřerušení hovoru v době přechodu od jedné základnové stanice k druhé. GSM síť je schopna toto zaručit v rozsahu sítě jednoho poskytovatele, ne však v rámci roamingu mezi více poskytovateli.

Mobilní zařízení monitoruje kvalitu signálu z 16 nejbližších základnových stanic a každou sekundu posílá informace o šesti nejsilnější svému BSC (Base Station Controler). Rozhodnutí o přepnutí k jiné základnové stanice muže vzejít od mobilního zařízení nebo od samotné GSM sítě.

Mobilitu v GSM lze použít i v rychle jedoucích vlacích.

4.3 Heterogenní sítě

Každá bezdrátová síť má své výhody a nevýhody. WiFi nabízí relativně velkou kapacitu, ale nemá dostatečně široké pokrytí. Naproti tomu GSM dosahuje téměř globálního pokrytí, ale zase je problém s malou přenosovou rychlostí, případně s nevýhodným účtováním (pro uživatele). V době, kdy většina mobilních zařízení podporuje více bezdrátových sítí, by bylo výhodné, kdyby toto zařízení využívalo právě tu nejlepší dostupnou. K tomu je potřeba, aby zařízení dokázalo předávat spojení mezi různými bezdrátovými technologiemi bez ztráty spojení.

4.3.1 IEEE 802.21

Právě o to se snaží skupina IEEE 802.21. Její řešení by mělo umožňovat

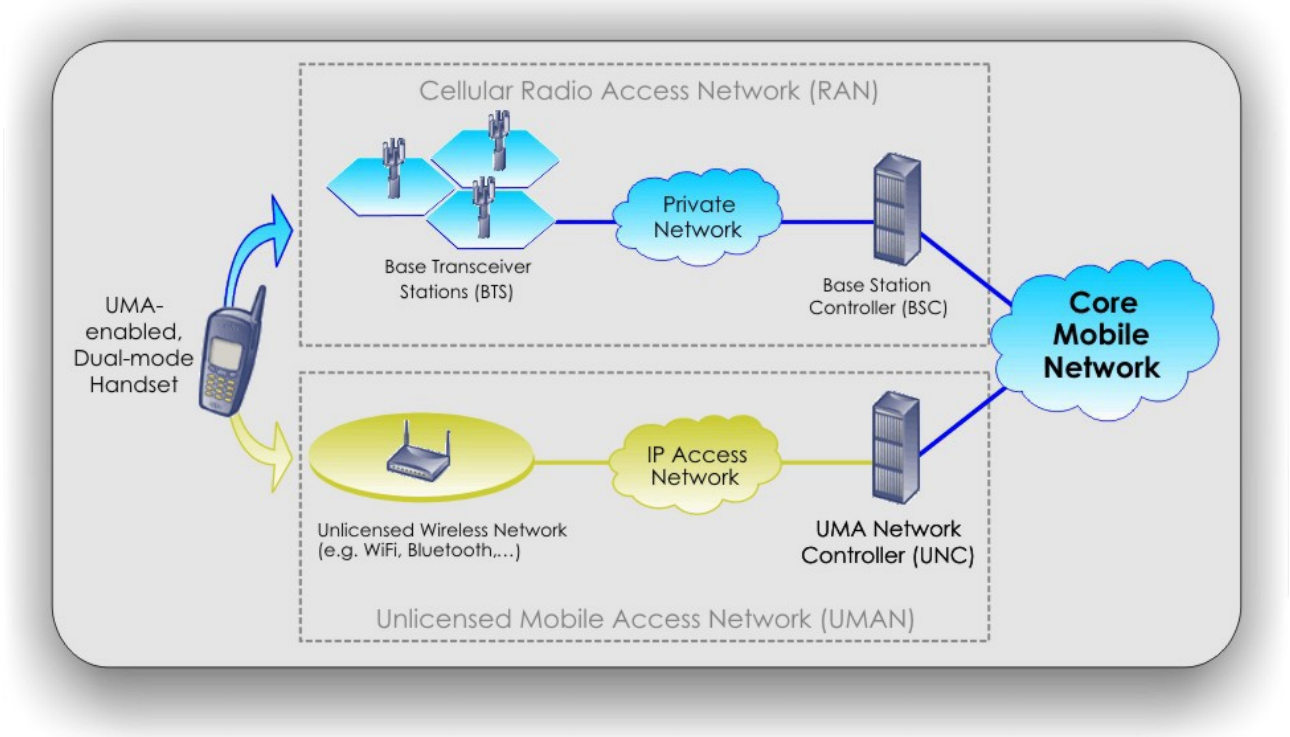
předání spojení bez ohledu na přípojnou technologii, a to nejen v rámci standardů IEEE 802, ale i z/do GSM nebo UMTS. Tato skupina zahájila činnost v březnu 2004 a v současné době je ve stavu prvních návrhů protokolu.

4.3.2 UMA

O trochu dál je aliance UMA (Unlicensed Mobile Access), která sdružuje výrobce hardwaru, softwaru a karet. Nabízí alternativní přístup do GSM/GPRS sítě přes spojení na bázi IP. Specifikace definuje nový síťový element v GSM síti (UNC – UMA network controler) a protokol, který poskytuje bezpečný přenos pro GSM/GPRS signalizaci přes IP síť. Integraci UMA technologie do GSM sítě ilustruje Obrázek 27.

Cílem technologie je zajistit hladký přechod hlasového i datového přenosu mezi GSM sítí a bezdrátovou sítí. Je postavena na rozšíření stávající GSM sítě a poskytuje stejnou identifikaci mobilního zařízení na GSM síti i bezdrátové síti. Cílem projektu je také zajistit stejnou úroveň zabezpečení jaka panuje v běžných GSM sítích.

Mobilní zařízení komunikuje s UNC pomocí zabezpečeného IP tunelu. Jeho prostřednictvím se přenášejí zprávy GSM protokolu. UNC vypadá z pohledu jádra GSM sítě jako BSC (Base Station Controler). Přechod od základnové stanice GSM do bezdrátové sítě se tedy jeví z pohledu jádra GSM sítě jako přechod mezi dvěma základnovými stanicemi.



Obrázek 27.: Rozšíření GSM sítě o technologii UMA (zdroj: http://www.umatechnology.org/).

Bezdrátová síť většinou nabízí větší přenosové rychlosti oproti GSM. Uživatel

připojený prostřednictvím bezdrátové sítě může teoreticky využívat kvalitnější hlasové i datové služby, protože bezdrátová síť poskytuje lepší konektivitu do jádra GSM sítě než systém základnových stanic.

Jedním ze členů sdružení UMA je společnost Motorola, která publikovala své řešení v článku [35].

4.4 Shrnutí

V bezdrátových sítí je význam mobility větší než v sítích „drátových“. Skutečně mobilní zařízení se potřebují připojovat bezdrátově a navíc požadují, aby se mohla libovolně pohybovat bez přerušení komunikace nebo zhoršení kvality jejího přenosu. Proto je v bezdrátových sítích třeba řešit problém mobility především na druhé vrstvě síťového modelu. Kritický je čas potřebný na asociaci k novému přístupovému bodu. Minimalizace tohoto času se dosahuje především neustálým skenováním dostupných přístupových bodů a vyhodnocování situace. Samotný přesun už je pak připraven dopředu a zabere pouze minimální čas. V podstatě každá dnes běžně používaná bezdrátová technologie se touto otázkou zabývá. Některé se dokonce snaží řešit problém i na třetí vrstvě (např. GPRS,WiMAX), jiné přenechávají třetí vrstvu mobilnímu IP.

Důležitá je také snaha umožnit rychlý přechod mezi sítěmi založenými na různých technologiích. To vyžaduje hlubší zásah do síťové infrastruktury a uzavření roamingových dohod mezi jednotlivými poskytovateli. Situaci komplikuje i neexistence obecně uznávaného standardu, většina firem pracuje na svých vlastních řešeních.

5. Mobilní ad-hoc sítě (Manet)

Mobilní ad-hoc sítě se také někdy nazývají bezdrátové sítě se smyčkami (překlad anglického Wireless mesh networks). Jsou to sítě mobilních uzlů spojených bezdrátovými linkami, kde se všechny uzly mohou náhodně a nepředvídatelně pohybovat nebo se vypínat a zapínat. Ad-hoc sítě tak tvoří libovolnou topologii, která je proměnná v čase. Jednotlivé bezdrátová spojení mohou být obousměrná nebo jednosměrná. Výsledná topologie je reprezentována orientovaným grafem.

Mobilní ad-hoc sítě nepožadují žádnou pevnou infrastrukturu a konfiguraci, každý uzel může soužit jako směrovač. Mohou pracovat samostatně nebo být připojeny do rozsáhlejší sítě (Internetu). V ad-hoc sítích si uzly na základě vzájemné domluvy sami budují směrovací topologii.

Minimální konfigurace a možnost rychlého vybudováni dělá z mobilních ad-hoc sítích kandidáta na nasazení v krizových situacích, kdy není možné vybudovat pevnou infrastrukturu. Navíc se ad-hoc sítě vyznačují velkou odolností, protože výpadek libovolného uzlu nebo i většího množství uzlů nemusí znamenat ujmu pro funkčnost sítě. Základní požadavek na ad-hoc sítě je rychlá a automatická rekonfigurovatelnost. Tato síť byla vytvořena na míru mobilním uživatelům a jejich potřebám. Může spontálně vzniknout všude tam, kde se nachází nějací mobilní uživatelé v dosahu svých bezdrátových linek.

V jednom manetu mohou vedle sebe existovat různé bezdrátové technologie. Mobilní uzel se tak může přímo spojit se všemi uzly, které jsou v dosahu a jsou schopny komunikovat alespoň jednou bezdrátovou technologií. Dva různé uzly se mohou spojit i pomocí více bezdrátových technologií. Obecně se tedy můžeme na topologii manetu dívat jako na orientovaný multigraf.

Směrování mezi všemi uzly v manetu se odehrává na síťové vrstvě (dále předpokládejme IP). Manet tak tvoří konzistentní síť, která v sobě může zahrnovat různé technologie na fyzické vrstvě. Aby toto mohlo fungovat, je potřeba, aby každý uzel měl přiřazenou jednu identifikaci (IP adresu) i v případě, že má více bezdrátových rozhraní a všechna tato rozhraní se účastní jednoho manetu. To je důležitý rozdíl oproti běžným „statickým“ sítím, kde je požadavek na unikátní adresu pro každé rozhraní.

Manet je navržen jako autonomní síť nezávislých uzlů. Lze ho použít jako koncovou sít rozsáhlejší sítě (Internetu). Nevhodné je ale použití jako transportní sítě, tedy na přenášení dat mezi dvěma sítěmi spojených manetem.

Uzly v manetu nemají z počátku žádnou informaci a topologii sítě a musí ji sami dynamicky objevovat a vypořádat se také s častými změnami. Pro tyto účely jsou vyvíjeny speciální algoritmy pro směrování se specifickými požadavky:

- Udržovat směrovací tabulku v rozumné velikosti.
- Vybrat vždy nejlepší cestu k danému cíli (to může znamenat nejrychlejší, s největší kapacitou, nejlevnější, atd.).

- Držet směrovací tabulku aktuální i v případě, že uzel změnil pozici nebo se odpojil/připojil.
- Minimalizovat počet přenášených řídících zpráv a čas pro vlastní algoritmus.

Standardizací směrovacích protokolů na třetí vrstvě v ad-hoc sítích se zabývá pracovní skupina IETF MANET http://www.ietf.org/html.charters/manet-charter.html.

Kromě toho vzniklo i poměrně velké množství návrhů, které pocházejí z akademické sféry nebo od komerčních subjektů. Některé z nich mají i implementaci.

Základní dělení směrovací protokolů v ad-hoc sítích je na proaktivní a reaktivní. Proaktivní protokol se snaží mít v každém okamžiku k dispozici směrovací tabulku, která obsahuje směrovací informace o celé síti. Na budování této tabulky neustále pracuje. Naproti tomu reaktivní směrovací protokol zjišťuje směrovací informace na základě aktuální potřeby.

Některé protokoly se snaží implementovat kombinaci obou přístupů, jiné spoléhají čistě na jeden přístup.

5.1 Optimized Link State Routing Protocol (OLSR)

OLSR je čistě proaktivní protokol. Jedná se o optimalizovanou verzi LSR (Link State Routing Protocol). OLSR oproti LSR minimalizuje počet kontrolních zpráv, které se šíří záplavově a šetří tak kapacitu sítě. Pouze vybrané uzly (nazývané MPR – Multi Point Realy) předávají kontrolní zprávy. Každý uzel si vybere z množiny svých sousedů množinu uzlů, které pro něj budou sloužit jako MPR. Pouze po těchto uzlech se bude šířit záplavová vlna kontrolních zpráv. Na základě těchto kontrolních zpráv si každý uzel spočte svoji směrovací tabulku, která bude obsahovat směrování pro všechny dostupné uzly v síti.

OSLR definuje tři typy zpráv: Hello, Topology Control (TC) a HNA.

Implementace protokolu řeší následující problémy:

- Snímání linky – každý uzel musí periodicky kontrolovat dostupnost vybraných sousedních uzlů.
- Detekce sousedů – k tomu se používá periodické vysílání zprávy Hello.
- Výběr MPR – snaha je vybrat dostatečný počet MPR s kvalitní obousměrnou konektivitou.
- Posílání kontrolních zpráv – uzel vysílá kontrolní zprávu (TC) obsahující seznam všech uzlů s možností přímé komunikace.
- Výpočet směrovacích informaci – na základě získaných informací uzel počítá směrovací tabulku. Na konci tohoto procesu má každý uzel informaci o směrování na všechny uzly v síti.

HNA zprávu (Host and Network Association) vysílá uzel, který funguje jako

brána do jiné sítě. Obsahuje informace o prefixu pro konfiguraci adresy a o síti, do které umožňuje přístup. Takový uzel může sloužit například jako brána do Internetu.

Hlavní nevýhodou OSLR je zbytečně velký a nepřetržitý traffic, který generují jednotlivé uzly. Některé takto získané informace uzel nikdy nevyužije, přesto musí investovat procesorový čas, paměť a přenosovou kapacitu sítě na jejich získání. Směrovací tabulka obsahuje všechny uzly v sítí a pro rozsáhlé sítě může být poměrně veliká

Na druhé straně získáme spolehlivý směrovací mechanismus, který je schopen okamžitě najít nejlepší cestu k danému cíli.

Protokol OSLR je standardizován v RFC 3626 [36] z roku 2003. V současné době je připravována druhá verze tohoto protokolu [37], která používá stejný základní mechanismus, ale poskytuje flexibilnější signalizační framework a některá zjednodušení ohledně přenosu zpráv.

5.2 Ad-hoc On-demand Distance Vector (AODV)

AODV narozdíl od OLSR funguje čistě reaktivně (on-demand). To znamená, že požadované směrování je vytvořeno až v okamžiku, kdy je potřeba. AODV umožňuje mobilnímu uzlu získat rychle požadované směrování a nevyžaduje na mobilním uzlu držet informace o směrováních, které mobilní uzel nevyužívá.

Protokol je postaven na třech základních zprávách, Route Request (RREQ) , Route Replies (RREP) a Route Error (RERR). Pokud uzel potřebuje zjistit směrování k určitému cíli, vyšle broadcastem zprávu RREQ. Směrování je určeno v okamžiku, kdy RREQ dorazí k cílovému uzlu nebo k uzlu, který má dostatečně čerstvý záznam o směrování k cílovému uzlu. Za čerstvý směrovací záznam je považován takový záznam, který má sekvenční číslo nejméně tak velké jako sekvenční číslo RREQ. Cílový uzel odpoví zprávou RREP zpět odesilateli požadavku. Každý uzel na cestě, který přeposílá RREQ, si ukládá do cache směrování zpět k odesilateli. Díky tomu může být RREP směrován unicastově až uzlu, který odeslal požadavek.

Každý uzel také monitoruje linku k dalšímu uzlu v každém aktivním směrování, které přes něj prochází. Jakmile spojení, které je součástí aktivního směrování, přestane fungovat, jsou o této skutečnosti informování ostatní uzly, které se daného směrování účastní. K tomuto účelu slouží zpráva RERR.

Protokol AODV je poměrně jednoduchý. Nezatěžuje zbytečně komunikační kanály a nespotřebovává další paměť ani procesorový čas na jednotlivých uzlech. Nevýhodou je, že potřebuje delší čas na počáteční inicializaci spojení.

Experimentální verze je standardizován v RFC 3561 [38] z roku 2003

5.3 Hazy Sighted Link State (HSLS)

Tento protokol [39] nepochází z dílny pracovní skupiny IETF MANET, ale z laboratoří BBN Technologies. Kombinuje reaktivní a proaktivní přístup. Snaží

se o efektivní využití přenosové kapacity a nabízí pouze suboptimální směrování.

5.5 Zhodnocení

Předchozí odstavce obsahovaly pouze úvod do složité oblasti mobilní ad-hoc sítí. Už z definice je ale vidět, že mobilita je pro tyto sítě prioritní otázka a dokáží si s ní celkem dobře poradit. Složitá topologie sítě a její časté změny vyžadují komplikované směrovací protokoly, které spotřebovávají značné síťové zdroje. Nasazení těchto protokolů v globálním měřítku je proto nereálné a lze je využít pro zajištění mobility pouze v menších koncových sítích.

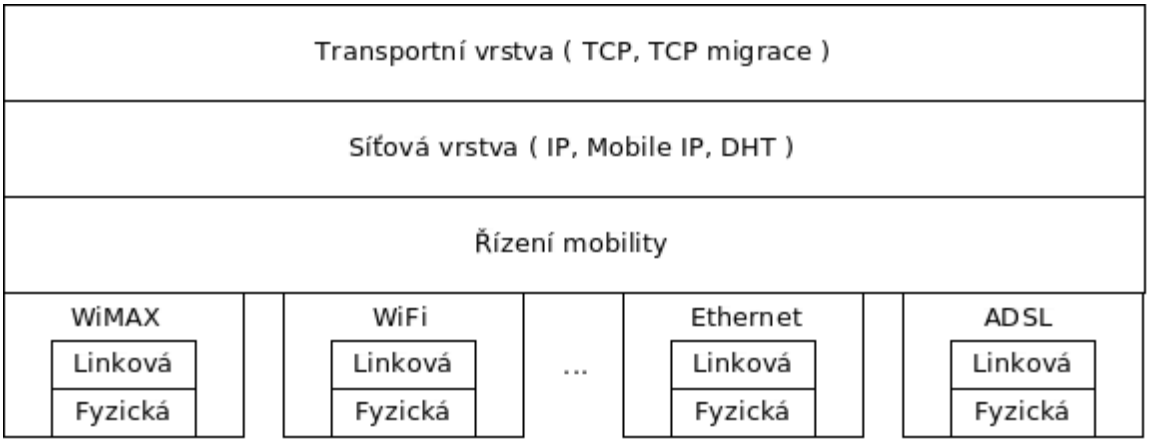
6. Závěr

Z předchozího textu vyplývá, že mobilitu lze řešit na linkové (bezdrátové sítě), síťové (Mobile IP), transportní (TCP migrace) a nakonec i aplikační (Peer-to-peer sítě) vrstvě síťového modelu. Většina těchto řešení si nekonkuruje, spíše se doplňuje. Jak by tedy měl vypadat síťový subsystém plně mobilního uzlu?

6.1 Idea plně mobilního uzlu

Předpokládejme mobilní uzel, který má několik síťových rozhraní s konektivitou do různých sítí. Nemusí se jednat pouze o sítě bezdrátové (WiFi, WiMAX, BlueTooth, UMTS, ...) ale třeba i o sítě drátové (Ethernet, ADSL, ...). Operační systém na mobilním uzlu se dnes na všechna tyto rozhraní dívá zvlášť a je na uživateli rozhodnout, jakou konektivitu chce zrovna využívat.

Plně mobilní uzel by však měl nabízet uživateli pouze jedno síťové rozhraní, které zastřešuje všechnu konektivitu. Operační systém by měl řešit, jaké fyzické rozhraní právě použít a zajišťovat hladký přechod mezi nimi. Obrázek 28 ukazuje představu uspořádání vrstev v síťovém systému plně mobilního uzlu (Vrstvy od transportní výše nejsou zobrazeny).



Obrázek 28: Vrstvy síťového systému u plně mobilního uzlu.

Přibyla vrstva řízení mobility, jejímž úkolem je monitorovat jednotlivé rozhraní, rozhodnout, které se má primárně používat, a zajišťovat hladký přechod mezi nimi. Nemusí se jednat pouze o přepnutí mezi dvěma technologiemi, ale i o přepnutí mezi dvěma sítěmi pracujícími na jedné technologii.

Pravidla pro použití různých sítí by měla být uživatelsky konfigurovatelná a měla by odrážet kriteria jako:

- Dostupnost – u bezdrátových sítí se jedná o kvalitu signálu, u drátových o fyzické připojení kabelu.
- Propustnost – rychlosti jednotlivých technologií se většinou diametrálně liší. Pro uživatele je samozřejmě výhodnější používat tu nejrychlejší.

- Cena – Pro většinu koncových uživatelů je to velice podstatné kriterium. Díky marketingové strategii mnoha firem bývá však cenová politika dosti komplikovaná. Někde se platí za data, jinde za připojený čas, někde se rozlišuje i denní doba. Někdy platíme svému domovskému operátorovi, který nám zpřístupnil síť roamingového partnera, jindy zas přímo provozovateli sítě (například v hotelu).

Lze použít i jiná kriteria, například úrovně zabezpečení, nastavené omezení (firewall), možnost provést rychlý handover z aktuální technologie nebo i osobní preference uživatele.

Automatické zjišťování těchto kriterií nemusí být vždy bezproblémové. Například po připojení k telefonní lince lze jen těžko zjistit dostupné ADSL připojení a jeho propustnost. Ale ADSL je v zvláštní případ a její použití není u mobilních uzlů příliš pravděpodobné.

Palčivější problém je s cenou. Uživatel může buď zadat tarify jednotlivých poskytovatelů ručně, nebo jeho domovský poskytovatel musí dát k dispozici seznam svých roamingových partnerů spolu s aktuálním ceníkem. Pokud se připojuji v hotelu, musí ceník (v elektronické formě) dodat provozovatel sítě. V případě bezplatného připojení (v některých restauracích) je nutné, aby tato síť sama propagovala svojí bezplatnost.

Můžeme říci, že řízení mobility na mobilním uzlu vyžaduje, aby se jednotlivé sítě dokázaly bezpečně identifikovat a umožnily získat veškeré parametry svého provozu. Na základě těchto parametrů se mobilní uzel rozhodne, které ze sítí bude využívat. Tyto požadavky nelze v dnešní době u všech technologií plně splnit a tato problematika zatím zůstává mimo zájem standardizačních organizací.

Nad vrstvou řízení mobility jsou již standardní vrstvy jak je známe ze síťového modelu ISO/OSI. Pouze s rozšířením o mobilní prvky diskutovanými v této práci.

Pokud vrstva řízení mobility při přepnutí na jinou technologii změní IP adresu, musí na tom spolupracovat se síťovou vrstvou. Ta zajistí potřebné kroky pro propagaci změny. Pokud uzel používat mobilní IP, zajistí přeregistraci u svého domovského agenta. Pokud se používá End-to-End, řešení zanese novou adresu do DHT a informuje transportní vrstvu o nutnosti provést TCP migraci.

Z tohoto pohledu se zdá, že mobilní IP a End-to-End přístup si přímo konkurují. Principiálně lze ale provozovat oba současně, každý pracuje na jiné vrstvě síťového modelu, každý má své výhody a nevýhody:

Při použití mobilního IP je nutné, aby každý mobilní uzel měl svojí domovskou IP adresu a také svého domovského agenta. Nově navázané spojení jde nejprve přes domovského agenta a teprve po optimalizaci cesty lze komunikovat přímo. Po přesunu do jiné sítě se musí registrace u domovského agenta i optimalizace cesty opakovat. Vyžaduje zásahy pouze do síťové vrstvy.

Pokud se bude používat End-to-End přístup, je potřeba existence DHT sítě, která bude uchovávat záznamy o mobilních uzlech. Každé spojení jde přímo optimální cestou. Po přesunu do jiné sítě se provádí nová registrace v DHT a

případná TCP migrace. U nestavových protokolů (UDP) lze využít notifikaci, kterou navržená DHT nabízí. Vyžaduje zásahy do protokolů transportní vrstvy.

Nelze jednoznačně říci, který způsob je výhodnější.

6.2 Shrnutí

Přestože je na mobilitu v poslední době kladen velký důraz, nejsou ještě všechny aspekty mobility plně dořešeny. Největší přínos pro mobilitu slibuje přechod na IPv6, který ovšem není tak rychlý, jak se původně předpokládalo. Obsahu dostupném přes IPv6 je málo, koncovým uživatelům chybí motivace vyžadovat IPv6 a díky tomu většina poskytovatelů IPv6 stále nenabízí.

Jiný problém je na linkové vrstvě. Především u bezdrátových sítí existuje více standardů, jejichž vzájemná kooperace není ještě plně dořešena (vertical handover). Zde se však na příslušných standardech pilně pracuje. Hnací silou je snaha o konvergenci datových a mobilních telefonních sítí (GSM,UMTS). Původně telefonní sítě nabízejí stále rychlejší přenos dat a datové sítě mají ambice přenášet telefonní hovory. Pro GSM sítě je mobilita přirozená vlastnost a dnes funguje prakticky na globální úrovni. Pokud chtějí bezdrátové datové sítě konkurovat GSM, musí nabízet mobilitu na minimálně stejné úrovni.

Mobilní ad-hoc sítě znamenají zvláštní kapitolu. Představují spíše alternativu ke klasickým bezdrátovým sítím než nějaké vlastní řešení mobility. Jejich primární cíl je umožnit komunikaci uzlů bez budování infrastruktury. Mohou vznikat spontánně všude tam, kde je více mobilních uzlů pohromadě. Pohyb mobilního uzlu v rámci ad-hoc sítě je přirozený a síť si s ním dokáže poradit. Přesto lze říci, že sítě s vlastní infrastrukturou dokáží mobilitu zajistit lépe. Pohyb uzlu je monitorován ze staticky umístěných základnových stanic a případný horizontální handover dokáže být řešen velmi rychle. Ad-hoc sítě je tedy nutno chápat jako sítě bez infrastruktury vhodné pro rychlé propojení několika uzlů (například v krizových situacích) a ne jako plnohodnotné mobilní řešení. Proto je jim také v této práci věnován menší prostor.

Mobilita by mohla mít i další výhody. Při implementaci plně mobilního uzlu, jak je popsán v kapitole 6.1, dokáže mobilní uzel využívat nejlevnější dostupnou síť. Například automaticky opustit síť svého poskytovatele a využívat volně přístupný access point ve svém dosahu. Uživatel také může mít smlouvu s více poskytovateli (provozovateli více bezdrátových sítí) a z dostupných sítí využívat tu nejvýhodnější. Takovéto řešení by zvýšilo konkurenční prostředí a dalo větší šance lokálním poskytovatelům, kteří se snaží prosadit před nadnárodními firmami.

Na rozvoji mobility se v posledních letech usilovně pracuje. Pro koncového uživatele tato práce zatím není příliš vidět. Jediná mobilita, která dnes opravdu masově funguje, je horizontální handover v některých bezdrátových sítích. Tím ale potenciál mobility není ani zdaleka vyčerpán.

Příloha A – seznam použité literatury

[1] C. E. Perkins: IP mobility support for IPv4 http://rfc.net/rfc3344.html

[2] C. E. Perkins: IP Encapsulation within IP http://rfc.net/rfc2003.html

[3] S. Deering: ICMP router discovery messages http://rfc.net/rfc1256.html

[4] D. L. Mills: Network time protocol (version 3) http://rfc.net/rfc1305.html

[5] S. Glass, T. Hiller, S. Jacobs, C. E. Perkins: Mobile IP autentication, authorization and accounting requirements. http://rfc.net/rfc2977.html

[6] M. Oschwald, J. Sievers: Mobile IP citeseer.ist.psu.edu/oschwald03mobile.html

[7] P. Ferguson, D. Senie: Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing. http://rfc.net/rfc2827.html

[8] G. Montenegro (Editor): Reverse tunneling for mobile IP http://rfc.net/rfc3024.html

[9] H. Levkowetz, S. Vaarala: Mobile IP traversal of network address translation devices. http://rfc.net/rfc3519.html

[10] C. Perkins, D. B. Johnson: Route optimization in mobile IP

[11] H. Chen, L. Trajkovic: Simulation of Route Optimization in Mobile IP http://www.ensc.sfu.ca/~ljilja/papers/WLN02_Chen.pdf

[12] E. Gustafsson, A. Jonsson, C. Perkins: Mobile IPv4 Regional Registration http://www.ietf.org/internet-drafts/draft-ietf-mip4-reg-tunnel-00.txt

[13] K. El Malki (Editor): Low Latency Handoffs in Mobile IPv4 http://www.ietf.org/internet-drafts/draft-ietf-mobileip-lowlatency-handoffs-v4-11.txt

[14] D. Johnson, C. Perkins, J. Arrko: Mobility support in IPv6 http://rfc.net/rfc3775.html

[15] D. Harkins, D. Carrel: The Internet Key Exchange (IKE) http://rfc.net/rfc2409.html

[16] M. Roe, T. Aura: Authentication of Mobile IPv6 Binding Updates and Acknowledgments http://research.microsoft.com/users/mroe/cam-v3.pdf

[17] T. Aura, J. Arkko: MIPv6 BU Attacks and Defenses http://research.microsoft.com/users/tuomaura/MobileIPv6/draft-aura-mipv6-bu-attacks-01.txt

[18] P. Nikandern, J. Arkko, T. Aura, G Montenegroo: Mobile IP version 6 (MIPv6)Route Optimization Security Designt

http://research.microsoft.com/users/tuomaura/Publications/nikander+-vtc2003f.pdf

[19] J. Arkko, V. Devarapalli: Using IPsec to Protect Mobile IPv6 Signaling Between Mobile Nodes and Home Agents http://rfc.net/rfc3776.html

[20] R. Koodli, Ed.: Fast Handovers for Mobile IPv6 http://rfc.net/rfc4068.html

[21] H. Soliman, C. Castelluccia: http://rfc.net/rfc4140.html

[22] P. Mockapetris: Domain names - concepts and facilities

[23] B. Wellington: Secure Domain Name System (DNS) Dynamic Update

[24] Andreas Pappas: Project Research Proposal: Real-Time DNS

[25] V. Ramasubramanian, E. G. Sirer: The Design and Implementation of Next Genertion Name Service for the Internet http://www.cs.cornell.edu/People/egs/beehive/codons.php

[26] Alex C. Snoeren and Hari Balakrishan: An End-to-End Approach to Host Mobility

[27] S. Ratnasamy, P. Francis, M. Handley, R. Karp a S. Shenker : A Scalable Content-Addressable Network http://www.acm.org/sigs/sigcomm/sigcomm2001/p13.html

[28] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications http://pdos.csail.mit.edu/papers/chord:sigcomm01/

[29] A. Rowstron, P. Druschel: Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems http://freepastry.org/PAST/pastry.pdf

[30] P. Maymoukov, D. Mazieres: Kademlia: A Peer-to-peer Information System Based on the XOR metric. http://kademlia.scs.cs.nyu.edu

[31] L. Divac-Krnic, R. Ackermann: Security-Related Issues in Peer-to-Peer Networks

[32] Hauto École Valaisanne: Roaming for Bluetooth http://www.holtmann.org/lecture/bluetooth/Bluetooth_Roaming.pdf

[33] CARLOS DE M. CORDEIRO, SACHIN ABHYANKAR, RISHI TOSHIWAL and DHARMA P. AGRAWAL: BlueStar: Enabling Efficient Integration between BluetoothWPANs and IEEE 802.11 WLANst

[34] Wimax Forum: Part I: A Technical Overview and Performance Evaluation http://www.wimaxforum.org/news/downloads/Mobile_WiMAX-_Part_1-Overview_and_Performance.pdf

[35] Motorola: Seamless Mobility Solutions

http://www.motorola.com/mot/doc/5/5550_MotDoc.pdf

[36] T. Clausen, P. Jacquet: Optimized Link State Routing Protocol (OLSR) http://www.ietf.org/rfc/rfc3626.txt

[37] T. Clausen, C. Dearlove: The Optimized Link-State Routing Protocol version 2 http://www.ietf.org/internet-drafts/draft-ietf-manet-olsrv2-01.txt

[38] C. Perkins, E. Belding-Royer, S. Das: Ad hoc On-Demand Distance Vector (AODV) Routing http://www.ietf.org/rfc/rfc3561.txt

[39] BBN Technologies: Hazy Sighted Link State Protocol http://www.cuwireless.net/downloads/HSLS.pdf

Příloha B – abecední rejstřík

Příloha C – seznam obrázků