VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

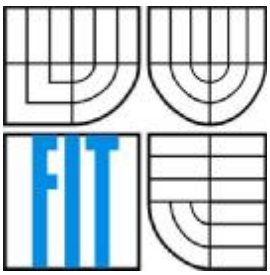
INFORMAČNÍ SYSTÉM PRO SPRÁVU FAIR USER POLICY

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE VLADIMÍR OBERREITER
AUTHOR

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

INFORMAČNÍ SYSTÉM PRO SPRÁVU FAIR USER POLICY

INFORMATION SYSTEM FOR FAIR USER POLICY MANAGEMENT

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE
AUTHOR
VLADIMÍR OBERREITER

VEDOUCÍ PRÁCE
SUPERVISOR
ING. JIŘÍ TOBOLA

BRNO 2008

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačových systémů Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Oberreiter Vladimír**
Obor: Informační technologie
Téma: **Informační systém pro správu Fair User Policy**
Kategorie: Web

Pokyny:

1. Seznamte se s technologiemi pro tvorbu webových informačních systémů (HTML, CSS, PHP, Javascript, MySQL apod.).
2. Stručně se seznamte s technologií NetFlow pro monitorování sítí.
3. Proveďte analýzu požadavků pro systém umožňující nastavování a kontrolu FUP, účtování za překročení FUP a tvorbu reportů na základě NetFlow dat.
4. Vytvořte detailní návrh tohoto systému a vhodně jej modelujte.
5. Navržený systém realizujte a otestujte, funkčnost systému demonstrujte na vhodně zvoleném vzorku dat.
6. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího rozšíření systému.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Splnění prvních tří bodů zadání.

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Tobola Jiří, Ing.**, UPSY FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno, Božetěchova 2

doc. Ing. Zdeněk Kotásek, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Vladimír Oberreiter**
Id studenta: 78826
Bytem: Stítež 119, 674 01 Třebíč 1
Narozen: 29. 09. 1983, Třebíč
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
bakalářská práce

Název VŠKP: Informační systém pro správu Fair User Policy
Vedoucí/školitel VŠKP: Tobola Jiří, Ing.
Ústav: Ústav počítačových systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:
tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☐ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

....

Nabyvatel Autor

Abstrakt

Náplní bakalářské práce je implementace Informačního systému pro správu Fair User Policy. Tento systém umožňuje mimo jiné evidovat internetová připojení jednotlivých zákazníků, nastavovat těmto zákazníkům různé typy ceníků s různými omezeními a při překročení daného limitu omezení jim zaslat informaci o překročení. Informace o množství dat odeslaných a přijatých zákazníkem jsou získávány pomocí technologie NetFlow.

Klíčová slova

FUP, PHP, MySQL, HTML, CSS, databáze, internetové připojení, informační systém, internet, ceník, přijatá data, odeslaná data

Abstract

The aim of this bachelor's thesis is the implementation of information system for administration of Fair User Policy. This system can monitor internet traffic of each customer, set adequate types of pricelists according to the customer behaviour, apply different regulations if any limit violation will occur and inform the customer about processes and crossed limits etc. The information about download and upload data is got by the technology NetFlow.

Keywords

FUP, PHP, MySQL, HTML, CSS, database, internet connection, information system, internet, pricelist, download, upload

Citace

Oberreiter Vladimír: Informační systém pro správu Fair User Policy. Brno, 2008, bakalářská práce, FIT VUT v Brně.

Informační systém pro správu Fair User Policy

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Jiřího Toboly a uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Vladimír Oberreiter
4.5.2008

Poděkování

Tímto bych chtěl poděkovat svému vedoucímu Ing. Jiřímu Tobolovi za poskytnutou pomoc a konzultace při tvorbě bakalářské práce.

Obsah

Úvod

Bakalářská práce Informační systém pro správu Fair User Policy se zabývá získáváním informací o komunikaci po sítí a následným zpracováním získaných dat.

Během chvilky vás jistě napadne bezpočet situací, kdy je možné tento typ dat využít. Jako jednu z mnoha těchto situací můžeme uvést případ internetového připojení od poskytovatele, ať už přes dial-up, wifi či ADSL. Vy jako uživatel internetového připojení většinou požadujete přesné informace o množství dat, ať už přijatých nebo odeslaných. Pokud máte navíc od poskytovatele pro stahovaná data nastaveny limity, budete se o to více zajímat, kolik dat už jste stáhli a jestli vám nehrozí od poskytovatele omezení rychlosti stahování dat, nebo zda nedostanete přirážku k ceně, či nějakou jinou formu postihu za překročení limitu. Takové omezení z důvodu překroční limitu nazýváme zkratkovým výrazem FUP (Fair User Policy). Ale FUP není jen strašák, který vás omezuje ve vašem stahování, ale naopak i vás chrání před lidmi, kteří příliš stahují. Chrání vás v případě, kdy si pořídíte levnější připojení, u kterého sdílíte datové pásmo s více uživateli, a takový uživatel, který neúměrně stahuje data, by vás přespříliš omezoval. Úkolem FUP je tedy omezit nebo případně jiným postihem potrestat uživatele, který během určité doby (den, týden, měsíc…) stáhnul více dat, než měl povoleno, a zpravidla mu na toto období snížit rychlost, aby vás dále nemohl omezovat.

Cílem této práce bylo vytvořit takový informační systém, který bude získávat ze sítě informace o odeslaných a přijatých datech, na jejichž základě bude možné vytvářet jednotlivé ceníky, u kterých lze různým způsobem nastavovat omezení pro zákazníky. Tento informační systém bude také informovat zákazníky o tom, že došlo u jejich připojení k omezení v důsledku překročení FUP limitu.

V první kapitole bakalářské práce jsou zařazeny informace o technologii NetFlow, pomocí které lze získávat data ze sítě, a o nástrojích, které s touto technologií pracují a umožňují ji využívat. Druhá kapitola obsahuje analýzu požadavků a podmínek potřebných pro činnost informačního systému. Ve třetí kapitole se zabývám návrhem informačního systému Fair User Policy. V kapitole číslo čtyři vás seznámím s použitými vývojovými programy pro vytváření webových informačních systémů PHP, MySQL, Perl, HTML a JavaScript. Pátá kapitola obsahuje popis implementace vytvořeného informačního sytému FUP. V závěru bakalářské práce se zaměřím na možnosti budoucího rozšíření systému a na návrhy jeho možného vylepšení - podle zlatého pravidla „vždy je co vylepšovat.“

1 NetFlow

NetFlow je otevřený protokol vyvinutý společností Cisco Systems. Z počátku byl využíván u Cisco směrovačů jako doplňková služba [1]. Prioritou tohoto protokolu je monitorování a sledování sítě, ke kterému využívá IP toků. Díky těmto informacím můžete sledovat tok dat v síti, odkud kam putují pakety, jejich množství, dobu komunikace, rychlost a mnoho dalších užitečných vlastností, na jejichž základě můžete danou síť spravovat, plánovat její rozšíření a předcházet nejrůznějším problémům. Další neocenitelnou výhodou je skutečnost, že data lze prakticky získávat v reálném čase. NetFlow je zároveň chápán i jako celý proces měření toků.

Využití NetFlow v praxi [2]:

- Účtování a fakturace, kontrola FUP. Měření IP provozu (informace o počtu bajtů, paketů, čase, typu služby, IP adresách a portech) a na základě těchto dat vytváření faktur a vyúčtování.
- Monitorování využití a vytížení Internetu.
- Sledování a analýza aplikací, využití například u sledování používaných aplikačních serverů v různých časových intervalech a na základě výsledků rozvrhnutí plánování záloh (kdy je server nejméně zatížen a záloha nebude zpomalovat uživatele), zjišťování, zda server nestíhá (tzn. rozložení aplikace v čase nebo přidání nového serveru). Dalším využitím může být průběžná analýza dimenzace pásma např. pro vyhrazená pásma u služeb VoIP.
- Sledování a analýza uživatelů, využití například ve firmách, kde firemní politika nedovoluje používat některé internetové aplikace nebo navštěvovat určitý typ stránek; případně kontrola, jestli uživatel nestahuje nepovolená data, a mnoho dalších.
- Detailní znalost kompletního provozu na síti.
- Zjištění nesprávných konfigurací sítě.
- Dohledávání incidentů a útoků v síti.
- Zvýšení bezpečnosti sítě, detekce vnitřních i vnějších útoků.
- Dlouhodobé skladování informací o přenesených datech.
- Kontroly peeringu a SLA (service-level agreement).
- Plánování kapacity sítě a datových linek, předcházení kolapsu sítě zahlcením.
- Dodržování zákona o elektronické komunikaci.

1.1 Cisco Systems

Cisco Systems je jednou z největších společností zabývajících se síťovými produkty. Mezi její produkty patří mimo jiné také routery, ethernetové i ATM switche, VoIP gatewaye a IP telefony. Tato firma byla založena skupinou vědců ze Stanford University v roce 1984. Její název „Cisco“ je zkratka ze San Francisco.

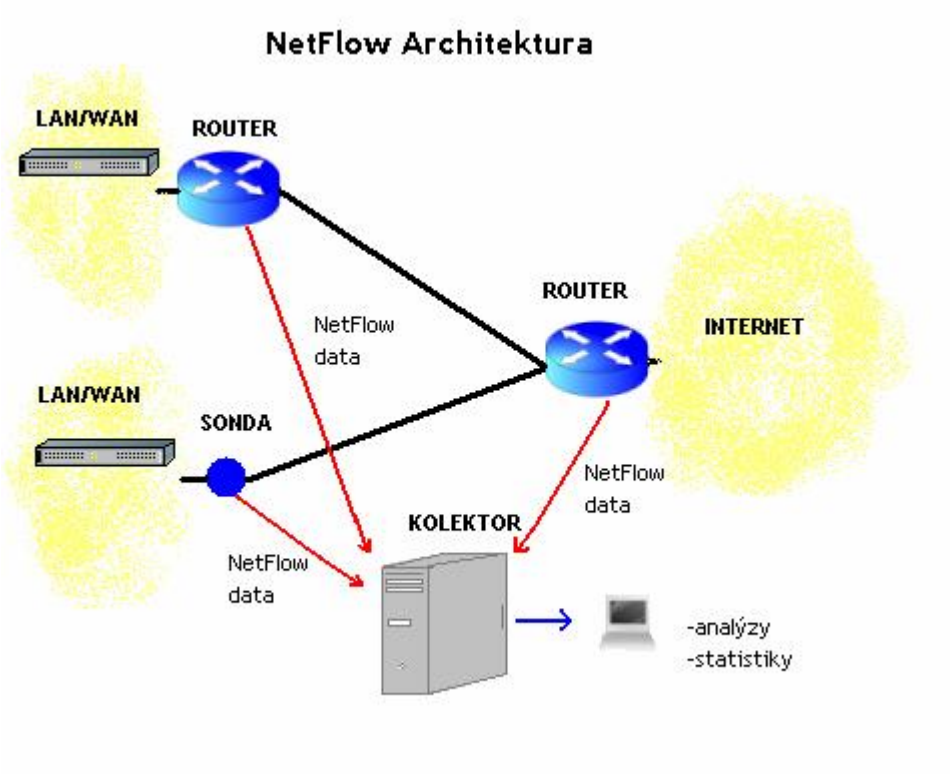
1.2 Definice toku v NetFlow

Tok je posloupnost paketů, které mají shodná klíčová pole (klíčové pole je index do tabulky, kde se dané statistiky ukládají) [2]:

- Zdrojová a cílová IP Adresa
- Zdrojový a cílový aplikační port (např. http=80)
- Protokol (např. ICMP=1, IGMP=2, TCP=6, UDP=17)
- ToS (Type of Service – nastavení paketů)
- Číslo rozhraní

1.3 NetFlow Architektura

NetFlow architektura se skládá z exportérů, které sbírají a posílají NetFlow záznamy o tocích [2]. Exportér může být sonda, která vytváří NetFlow záznamy a posílá je, nebo router, který tyto záznamy exportuje. Pomocí těchto exportérů, kterých může být v síti libovolný počet, získáváme ze sítě data. Data jsou přijímány kolektorem (jeden kolektor může přijímat záznamy o tocích z více exportérů), který je ukládá na disk nebo do databáze. Poté se data mohou zobrazovat pomocí webového rozhraní nebo dotazů nad databází (databáze je pomalejší, ale lépe se s ní pracuje). Ze zobrazených dat lze provádět různé analýzy a pomocí programu také zobrazovat grafy a statistiky, které umožňují sledovat a analyzovat provoz na síti i běžnému uživateli.



Obrázek 1.1 NetfFlow architektura

1.3.1 NetFlow Exportér

Exportér přijímá data ze sítě ve formě paketu, z něhož data extrahuje [2]. Exportér se pokusí nalézt příslušný záznam a pokud tento záznam existuje, aktualizuje jej podle zpracovávaných dat [2]. Jestliže paket neexistuje, pak jej exportér vytvoří. Poslední akcí poté, co je NetFlow záznam ukončen, je jeho odeslání z exportéru do kolektoru.

Možné způsoby, kterými exportér ukončuje Net Flow záznam:

- spojení TCP je ukončeno (FIN,RST)
- po vypršení časového intervalu naposledy zpracovaného paketu daného toku (využití např. UDP)
- kvůli uvolnění paměti z důvodu přetečení čítačů
- po vypršení pevného časového intervalu při dlouhých tocích (kdyby poslal NetFlow data po ukončení dlouhého toku, už by mohlo být pozdě zareagovat na tento stav, např. pokud by klient překročil limit)

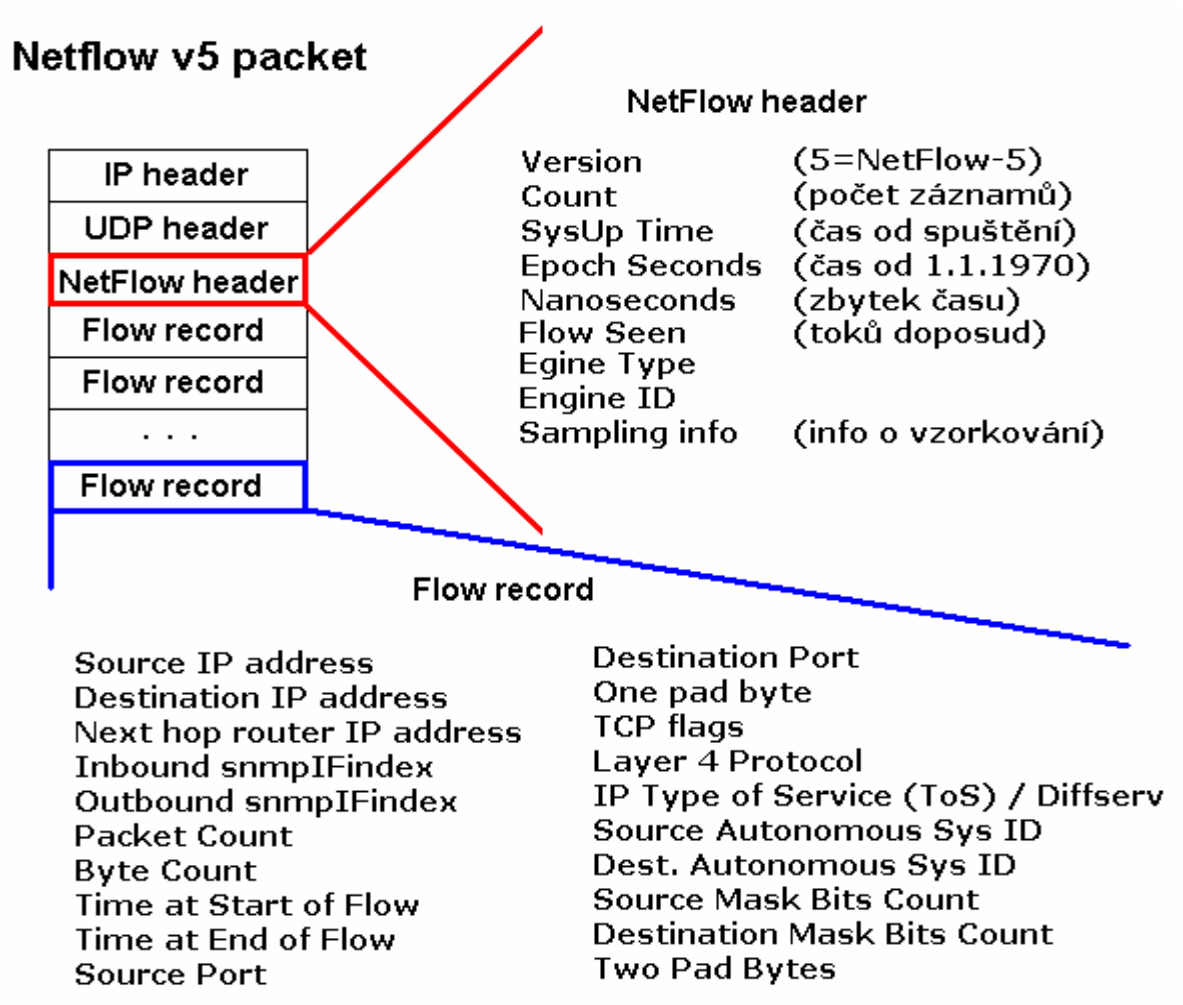
1.3.2 NetFlow Kolektor

Kolektor přijímá NetFlow pakety. Extrahuje z nich data a ukládá je do úložiště dat (disk, databáze) [2]. Má také možnost vzorkovat NetFlow pakety. Nad takto zpracovanými daty můžeme provádět dotazy, agregovat je podle různých kritérií (např. IP adresy, protokoly) a kontrolovat tímto způsobem komunikaci na síti, mimo jiné zjišťovat, zda někdo nepoužívá zakázané programy.

1.3.3 Verze NetFlow

Otevřeného protokolu NetFlow existuje několik verzí. Od první verze NetFlow v1 až po verzi v10, přičemž verze NetFlow v2-v4 nebyly výrobcem uvedeny [2].

- **verze NetFlow v1:** V dnešní době se už prakticky nepoužívá. Nahradila jí verze 5.
- **verze NetFlow v5:** Jedna z nejpoužívanějších verzí, je pro ní napsáno nejvíce uživatelských programů. Má fixní záznam. Uživatel si nemůže volit, které položky budou v záznamu (Flow record). Pro přenos používá pouze protokol UPD.

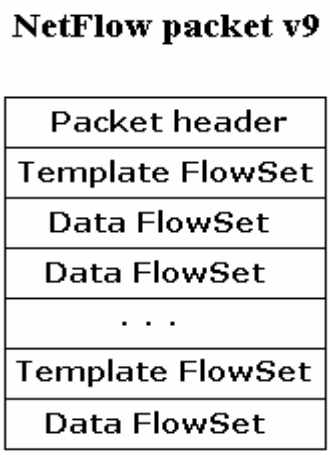


Obrázek 1.2 NetFlow v5 packet

- **verze NetFlow v7**: Byla speciálně vyvinuta pro switche, jsou v ní navíc obsaženy informace o přepínání**.**

- **verze NetFlow v8:** Umožňuje přidávat různá agregační schémata, podle nichž měří toky. Výhodou těchto agregačních schémat je ušetření místa v tabulkách, jelikož jsou ukládána pouze data, podle kterých agregace probíhá. Další nezanedbatelnou výhodou je menší vytížení routerů.

- **verze NetFlow v9:** Verze 9 využívá flexibilní strukturu záznamu. Díky ní si uživatel může zvolit, jaká data chce sledovat a vyhodnocovat. Struktura je dána šablonou (template), ve které je popsána struktura záznamu nebo nastavení monitorovacího procesu (option FlowSet). Šablony jsou zasílány kolektoru, který je schopen interpretovat podle nich jednotlivé záznamy. Šablonu stačí poslat pouze jednou, a to na začátku monitorovacího procesu. Každá šablona má vlastní ID, podle kterého se na ni kolektor odkazuje. V rámci jednoho monitorovacího procesu můžeme využívat dokonce i více druhů šablon.



Obrázek 1.3 NetFlow packet v9

- **verze IPFIX (NetFlow v10):** IPFIX (Internet Protocol for Flow Information Export) je nástupcem verze 9 a snaží se o RFC definici monitorovacího procesu i protokolu. S největší pravděpodobností je tento protokol budoucností pro NetFlow data.

1.4 Nástroje pracující s NetFlow

Existuje celá řada nástrojů, které pracují s NetFlow [2]. Většina z nich využívá NetFlow v5. U nástrojů, které využívají vyšší verze, se může stát, že routery vám neumožní využívat jejich plné schopnosti, protože to neumí. Budete tak komunikovat například pomocí verze NetFlow v9, ale výsledek bude úplně stejný, jako kdyby jste komunikovali pomocí verze NetFlow v5.

Požadavky na tyto nástroje jsou zejména rychlost, jednoduchost používání, udržení dat po určitý časový úsek, jednoduchá navigace při hledání uložených dat, flexibilní a rychlé vyhledávání, flexibilní agregace dat a mnoho dalších.

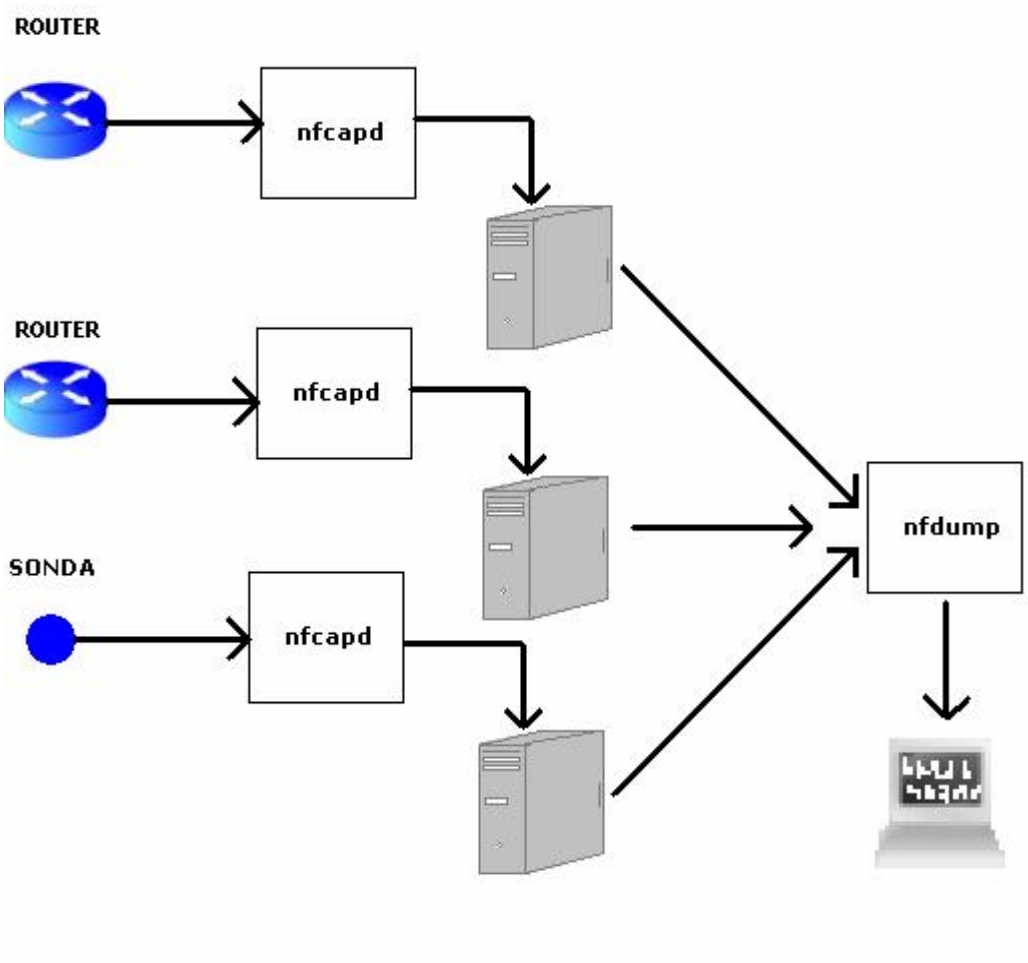
Níže jsou uvedeny příklady nástrojů pro práci s NetFlow daty [3]. Tyto nástroje vyhovují většině z výše zmíněných požadavků, patří do skupiny nástrojů NFDUMP, jsou distribuovány pod BSD licencí a pracují na úrovni příkazové řádky. Výjimku představuje nástroj NfSen, pracující přes webové rozhraní. Všechny tyto nástroje jsou podporovány NetFlow verzemi v5, v7 a v9.

- **Nfcapd** - tento nástroj čte data ze sítě a ukládá je do souborů na disk. Ukládání probíhá automaticky, uživatel si může délku intervalu zvolit (zpravidla probíhá ukládání v periodě každých pět minut). Pro každý NetFlow tok je potřeba vlastní nfcapd proces, který tok zpracovává.
- **nfdump** čte data ze souborů (kam je ukládá nfcapd) a zobrazuje je. Toto zobrazení je možné podle nejrůznějších agregací a nastavení, například podle cílové nebo zdrojové IP adresy, podle portu atd.
- **nfprofile** je nástroj, který čte data ze souboru (kam je ukládá nfcapd). Data filtruje podle speciálního filtru a výsledná data ukládá zpět do souborů pro pozdější použití.
- **nfreplay** čte data ze souboru (kam je ukládá nfcapd) a posílá je přes síť jinému uživateli, který je následně může zpracovat.
- **nfclean.pl** slouží k mazání souborů (které vytvořil nástroj nfcapd), kterým vypršel čas exspirace. Lze ho spouštět pravidelně v určitých intervalech a zabránit tím zahlcení disku.
- **nfSen (Netflow Sensor)** umožňuje využívat výhody příkazové řádky, ale také umožňuje grafický výstup pro zobrazení NetFlow dat. Data lze agregovat, vytvářet nejrůznější statistiky, nastavovat varování a nejrůznější podmínky. Jednou z podstatných výhod je možnost napsat vlastní plugin pro zobrazování NetFlow dat.

1.5 Princip zpracování dat

Všechna data jsou před zpracováním uložena na disky, čímž je oddělen proces ukládání dat a proces jejich zpracování [3]. Data jsou organizována v souborech, které nfcapd ukládá v pravidelných intervalech. Pojmenování těchto souborů se určuje podle data a času, kdy byly informace o toku

pořízeny, což usnadňuje orientaci v záznamech dat. Formát názvu souboru je následující: nejprve je uvedeno jméno programu nfcapd a za ním je uvedeno datum a čas ve formátu YYYYMMddhhmm. Z toho vyplývá, že výsledný název pořízeného souboru nástrojem nfcapd 18.04.2008 v 18 hodin 3 minuty bude vypadat následovně: nfcapd.200804181803. Podle zvoleného časového intervalu, ve kterém se mají data ukládat, je vždy vygenerován nový soubor s patřičným názvem.



Obrázek 1.4 Princip zpracování dat

Nástroje NFDUMP umožňují různé uspořádání souborů s uloženými daty:

- data uložená v jednom souboru v jednom adresáři
- data uložená v jednom adresáři ve více souborech
- data uložená v jednom souboru ve více adresářích
- více souborů uloženo ve více adresářích.

2 Analýza informačního systému

2.1 Požadavky na systém FUP

Systém by měl být přehledný, jednoduchý a mělo by se v něm dát rychle orientovat. Základním pilířem systému je sběr informací o zákazníkovi - kolik odeslal či přijal dat prostřednictvím sítě. Údaje o množství odeslaných či přijatých dat by měl systém uchovávat v databázi, a pokud zákazník přesáhne limit nastavený podle jím zvoleného ceníku, zašle mu systém email s touto informací.

Zákazník by měl mít přehled o svých připojeních, které mu poskytovatel poskytuje, a také by měl mít možnost zjistit, jakou má aktuální rychlost pro stahování a posílání dat a jakou má aktuální cenu, kterou za připojení platí. Dále si zákazník může měnit v systému své osobní informace (kontaktní adresu, kontakty, jméno). V neposlední řadě si zákazník může zvolit, zda chce mít od příštího vyúčtování jiný ceník.

Z hlediska administračního by měly existovat dvě skupiny uživatelů, kteří mohou spravovat tento informační systém.

Správce, který může spravovat jednotlivé zákazníky a přiřazovat jim připojení, nastavovat IP adresy a starat se o ceníky, které jsou zákazníkům poskytovány. Správce však nemá právo tyto ceníky nějak ovlivnit, pokud už jsou někde použity.

Administrátor je funkce s rozšířenými právy správce. Navíc, oproti správci, může přidávat jednotlivé uživatele a editovat je (administrátor avšak nemá právo editovat sám sebe). Dále má administrátor větší pravomoc pracovat s nabízenými ceníky.

2.2 Získávání dat o komunikaci

Získávat data ze sítě budeme pomoci NetFlow technologie pro sledování toků dat na síti (tato technologie byla popsána výše). Pro získání dat o komunikaci budeme potřebovat nějaký rychlý nástroj, který dokáže efektivně zpracovávat NetFlow data a umožní je agregovat tak, abychom mohli pracovat s co nejmenším počtem informací, které jsou nezbytně nutné pro práci systému. Takovým nástrojem je nfdump. Protože množství informací, které se budou zpracovávat, je velké, budeme muset zpracování dat provádět pravidelně, v předem zvolených intervalech, abychom předešli zahlcení serveru. Zpravidla se jedná o interval pěti minut, protože Netflow je většinou nastaven na ukládání dat o tocích také každých pět minut.

2.3 Zpracování dat o komunikaci

Zpracovaní dat o komunikaci bude nejnáročnější operací pro celý systém, a proto by mělo být naprogramováno co nejefektivněji, aby tyto operace nezabíraly příliš času a systémových prostředků. Uvědomme si, že přes router nebo sondu, kterou sledujeme síť, mohou projít během několika minut statisíce záznamů o komunikaci mezi uživateli, a že zpracování takového množství dat i velmi jednoduchým skriptem zabere značnou výpočetní kapacitu.

Po zamyšlení nad těmito fakty je jasné, že se budeme muset snažit přesunout co největší část těchto operací na stranu serveru, na kterém běží náš informační systém, abychom zbytečnou komunikací nezatěžovali síť. Efektivním řešením při zpracování dat o komunikaci je přesunutí procesu aktualizujícího data v databázi přímo do uložené procedury do MySQL databáze

2.4 Údržba databáze

Charakter vytvářeného informačního systému vyžaduje pravidelnou údržbu systému. Znamená to například pravidelné zpracování uživatelských dat, vytváření vyúčtování a nastavení požadovaných nových ceníků pro zákazníky. Tento požadavek vede k napsání skriptu v některém skriptovacím jazyce (PHP, Perl...), který musí být pravidelně spouštěn v přesně daných časových intervalech a který také bude požadované akce provádět.

2.5 Zamykání databáze, transakce

MySQL databázový systém je systém klient-server. Z toho vyplývá, že spousta uživatelů může přistupovat současně k databázi, číst, ale také měnit data, což může vést k problémům, vznikajícím při současném přístupu více uživatelů ke stejným datům. Jeden uživatel si data načte a než je stihne zpracovat, jiný uživatel mu je změní. U některých systémů může neošetření zamykání databáze způsobit velké problémy (jedná se například o finanční instituce). Pokud si dva uživatelé načtou zároveň data o účtu z databáze a první uživatel z účtu vybere peníze a ukončí práci dřív než druhý uživatel, ovlivní tím data uložená v databázi, aniž by ovlivnil ta data, která si načetl druhý uživatel. Pokud druhý uživatel například přidal nějaké peníze na účet a operaci ukončil po prvním uživateli, přepíše hodnotu v databázi a tím způsobí to, že na stavu účtu se v důsledku tohoto přepisu neprojeví výběr prvního uživatele.

Řešením výše popsaného problému může být zavedení zamykaní. Zamykání pracuje na principu umožnění práce pouze s odemčenou tabulkou. To znamená, že jen jeden uživatel (program) může aktuálně pracovat s daty v tabulce. Pokud chce uživatel s danou tabulkou pracovat, musí být tabulka odemčena. S uživatelovým vstupem se tabulka zamkne a pouze tento uživatel má

pak právo do ní zapisovat a modifikovat ji. Po provedení všech potřebných operací uživatel tabulku opět odemkne a ta je připravena k dalšímu použití.

Dalším řešením je používání transakcí (transakčního zpracování). Transakční zpracování pracuje na principu provedení několika příkazů jako celku. Pokud tedy začnete zpracovávat blok příkazů jako transakci a z nějakých důvodů bude tento blok příkazů ukončen předčasně (spadne server nebo nějaký z příkazů není proveden aj.), zruší se všechny dosud provedené operace v rámci probíhající transakce a databáze se vrátí do stavu před započetím provádění transakce. Transakční zpracování lze používat pouze u tabulek InnoDB.

Po zralé úvaze jsem došel k přesvědčení, že ve vytvářeném informačním systému FUP ani jednu z těchto technik nepoužiji, neboť do části systému, kde by mohlo docházet k problémům výše popsaným, bude mít přístup jen systémový skript, který bude vždy spuštěn jen jako jedna instance.

2.6 Zodpovědnosti za změnu dat v databázi

V neposlední řadě je nutné se zamyslet nad tím, jak bude v databázi řešeno zaznamenávání změn jednotlivými uživateli - záznamy o změnách. Tento systém by měl zcela určitě obsahovat historii změn, popřípadě nějakým způsobem určovat odpovědnost za uložená data v databázi. Musí existovat možnost při chybně vyplněných datech nebo při změnách dat identifikovat odpovědnou osobu, která daný úkon provedla a která je schopna vysvětlit důvod změny nebo ochotna změnu označit za uživatelskou chybu. Tento požadavek je o to důležitější, o kolik více uživatelů má možnost s danými daty pracovat.

Jedním z možných způsobů řešení je ten, že při každé změně dat v nějaké tabulce (ať už se jedná o vytvoření, smazání či editaci sloupce) se data uloží do další tabulky, v níž bude uloženo, který uživatel změnu provedl, jakou změnu provedl a případně které sloupce ovlivnil. Tento způsob je dle mého názoru z hlediska odpovědnosti a dohledání změn nejlepší, ale má i své nevýhody. Jednou z hlavních nevýhod tohoto způsobu řešení je fakt, že při každé změně v systému (v tabulce s daty) je vytvořen záznam do jiné tabulky, čímž velmi rychle roste velikost databáze. Tím se stane, že i celkem malý systém může během krátké doby používaní zabírat až několik GB paměti.

Další možným způsobem je v případě změny uvést ke změněným datům údaj kdo a kdy je změnil, díky čemuž bude při řešení odpovědnosti možné obrátit se vždy na uživatele, který s danými daty pracoval naposledy. Z tohoto řešení plyne, že uživatel, který s danými daty pracoval naposledy, převzal pro daný záznam veškerou zodpovědnost za data v něm uvedená. Výhodou oproti předchozímu řešení je nenarůstání velikosti databáze metadaty, nevýhodou je ovšem nemožnost dohledání změněných informací. Částečně je možné tuto nevýhodu řešit zálohami databáze. Zálohování nám umožňuje zjistit aktuální stav v čase zálohy a díky tomu částečně dohledat požadovaná data.

Při vybírání metody musíme samozřejmě zohlednit typ vytvářeného informačního systému a také důležitost dat, se kterými systém pracuje.

3 Návrh informačního systému

3.1 Práce s ceníky

Jednou z důležitých částí systému jsou také ceníky, ve kterých by měla být nastavována jednotlivá omezení pro FUP. Ceník umožňuje nastavit jak omezení pro posílání dat, tak omezení pro jejich příjem. Dále bude umožňovat také finanční postihy za překročení limitu, které budou na sobě nezávislé. To znamená, že uživatel (administrátor, správce) si bude moci nastavit zvlášť omezení pro posílání dat a zvlášť pro příjem dat. Také si bude moci uživatel zvolit, zda bude postih při překročení stanoveného limitu proveden formou snížení rychlosti, nebo ve formě finančního postihu, či kombinací obojího. Pro jeden ceník bude možno nastavit libovolný počet omezení. Uživatel si například vytvoří ceník, který bude mít rychlost stahování 2Mbit/s, po překročení 2GB přijatých dat se mu rychlost stahování sníží na polovinu (1Mbit/s) a pokud dále překročí limit 4GB přijatých dat, rychlost mu znova klesne o polovinu - na 0,5Mbit/s a dostane přirážku 50Kč.

Ceníky bude moci vytvářet správce nebo administrátor. Nově vytvořené ceníky musí být nejprve nastaveny jako aktivní, aby bylo možné je zobrazit v nabídce pro zákazníky. Pokud má nějaký uživatel nastaven daný ceník jako využívaný (ať již jako v dané chvíli aktivní, nebo nastaven s tím, že mu bude platit od příštího měsíce, či je už daný ceník uložen ve vyúčtování), nelze tento ceník smazat, ani editovat. Je logické, že uživateli nemůžeme měnit ceník, jak se nám zlíbí. V některých případech se ale může stát, že si všimneme špatného nastavení ceníku, tudíž je v dané situaci nutné tento ceník nějakým způsobem editovat. Proto bude v rámci systému vytvořen nástroj, který umožní zaměnit jeden typ ceníků za druhý a automaticky poté přepočítat cenu a rychlosti připojení pro daný měsíc.

Zákazník si bude moci změnit ceník jednou za měsíc a změna bude provedena vždy k začátku následujícího zúčtovacího období.

3.2 Správa zákazníků a připojení

Při přidání nového zákazníka je tomuto zákazníkovi automaticky vygenerováno unikátní přihlašovací jméno a jeho emailová adresa, na kterou mu budou zasílány informace o případném překroční limitu. Tuto adresu poskytuje poskytovatel k připojení. Ke každému zákazníkovi bude možno přiřadit libovolný počet kontaktů. Dále bude možné k zákazníkovi přiřadit libovolný počet internetových připojení a ke každému tomuto připojení libovolný počet IP adres (minimálně však musí existovat jedna IP adresa u každého připojení).

Vytvořeného zákazníka není možné již odstranit. Lze mu pouze nastavit stav „archive“, což znamená, že zákazník je neaktivní. Tímto způsobem bude zabráněno neoprávněnému smazání

zákazníka a tím také ztrátě historie o zákaznících. Další výhodou je skutečnost, že pokud se k nám zákazník po určité době vrátí, můžeme v systému vyhledat historii jeho připojení, vyúčtování a stačí nám tohoto zákazníka už jen znovu aktivovat.

3.3 Správa uživatelů

Uživatel je osoba, která bude spravovat informační systém. Při vytváření takovéto osoby bude možné vybrat skupinu práv, která jí budou přiřazena. Uživatele lze následně libovolně mazat a editovat.

3.4 Jednotky

Nastavení systému umožní zadávání množství objemu dat v celém systému pouze ve stejných jednotkách (GB, MB, ...), které půjdou změnit globálně pro celý systém. Stejné kritérium bude platit i pro jednotky rychlosti přenosu dat (Mbit/s, Kbit/s, ...). Pokud budou tato data zobrazována z databáze, jejich jednotky budou automaticky vygenerovány v závislosti na jejich velikosti. Například 1000 MB se zobrazí jako 1 GB.

3.5 Jazyk

V dnešní době, kdy je Česká republika členskou zemí Evropské unie, by měl být dobrý informační systém podle mého názoru minimálně dvojjazyčný. A protože velmi rozšířeným světovým jazykem je angličtina, bude možné ve vytvořeném systému kromě češtiny pracovat i v angličtině.

3.6 Vyúčtování

V pravidelných intervalech proběhne skript, který vytvoří vyúčtování za právě skončené zúčtovací období. Pokud bude mít zákazník nastavený požadavek na nový ceník, nastaví se mu pro následující zúčtovací období ceník nový. Tato časová perioda je zpravidla nastavena na jeden měsíc.

3.7 Informace o množství dat

Informace o množství přijatých a odeslaných dat budou získávány pomocí NetFlow, což bude zajišťovat skript, spouštěný každých pět minut. Tento skript zjistí, kolik dat uživatel přijal a kolik odeslal, a tyto informace zapíše do databáze. Poté zkontroluje, jestli uživatel nepřekročil limit svého připojení. Pokud tato situace nastane, bude uživateli zaslána systémová zpráva s informací o překročení limitu a o novém nastavení.

3.8 Řešení záznamu odpovědnosti za změnu dat v databázi

Po zvážení dvou výše uvedených možností řešení odpovědnosti (viz. kap. 2.7) jsem si vybral řešení druhé, ve kterém jsou ke každému záznamu dat přidávány záznamy o tom, kdo a kdy s těmito daty pracoval naposledy. Z hlediska povahy zpracovávaného informačního systému je tento způsob dostačující. Jak jsem se v praxi přesvědčil u firmy, ve které pracuji jako správce informačního systému, může tento způsob řešení odpovědnosti fungovat bez větších problémů i na velkých systémech.

Neplatí to však u jednoho typu záznamu, kterým je záznam o změně ceníku. Je to z toho důvodu, že ceník na příští měsíc může měnit jak uživatel (administrátor, správce), tak zákazník. Je proto nutné uchovávat historii těchto změn. Například pokud si zákazník změní ceník na příští měsíc sám a posléze mu jej přemění uživatel a zákazník se bude dožadovat informace, proč nemá jím zvolený ceník, bude potřeba dohledat, kdo a kdy mu jej změnil, a vysvětlit, proč k této změně došlo.

3.9 Role uživatelů – případy užití

V systému budou tři základní role. Ty se od sebe liší jednotlivými pravomocemi manipulace s daty a přístupy k datům. Základní požadavek je na tyto tři role – administrátor, správce (diagram případu užití pro správce a administrátora je uveden ve schématu 3.2) a zákazník (diagram případu užití pro zákazníka znázorňuje schéma 3.3).

Dalším účastníkem je čas, díky kterému jsou v pravidelných intervalech prováděny předem naplánované akce (diagram případů užití času následuje ve schématu 3.1).

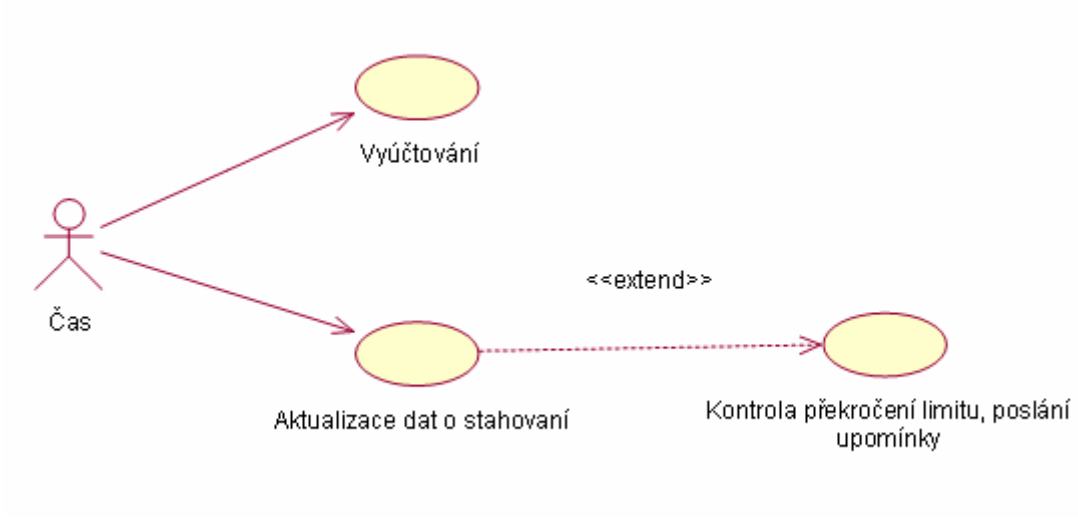


Schéma 3.1 Diagram případů užití času

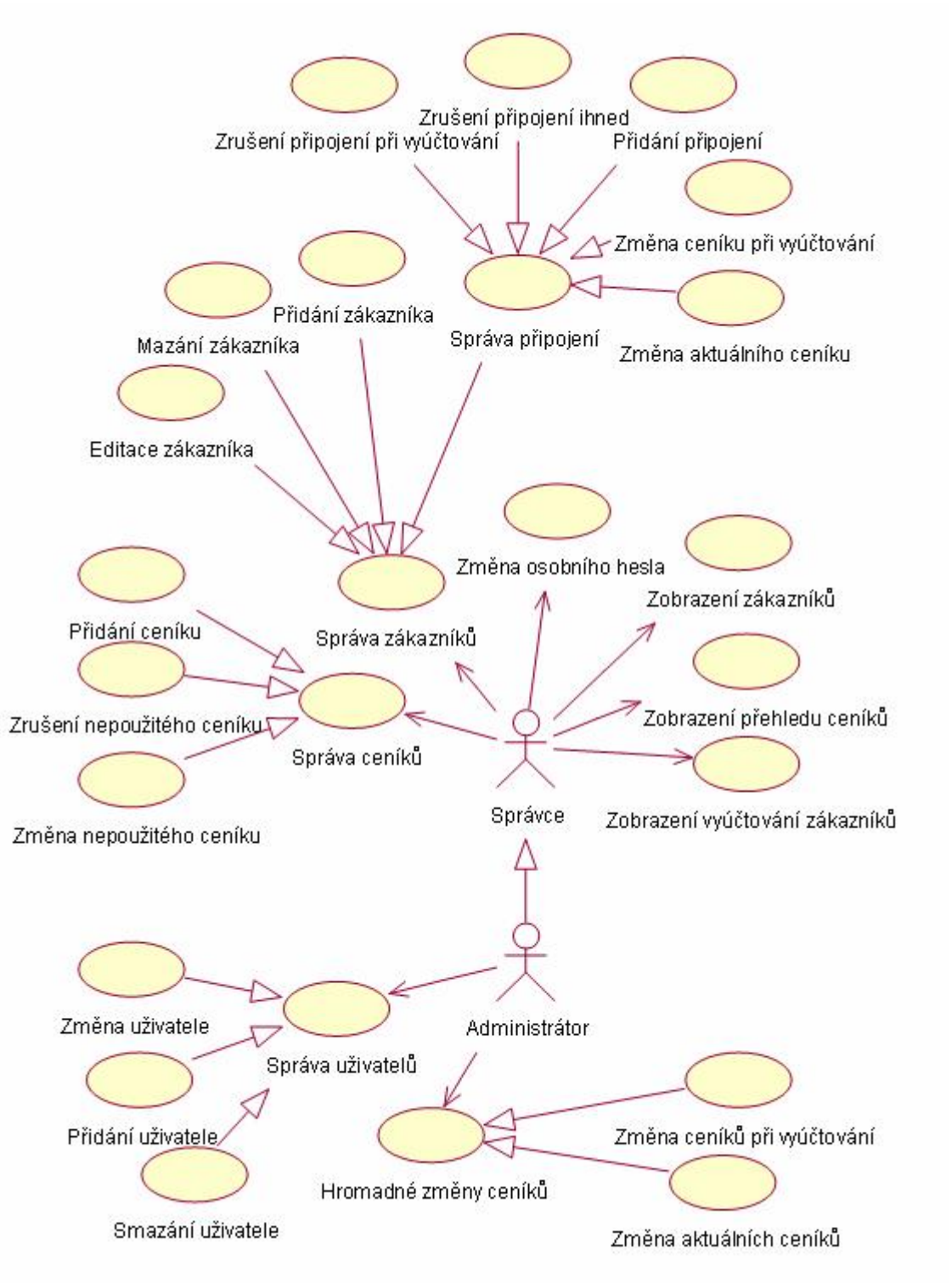


Schéma 3.2 Diagram případů užití administrátora a správce

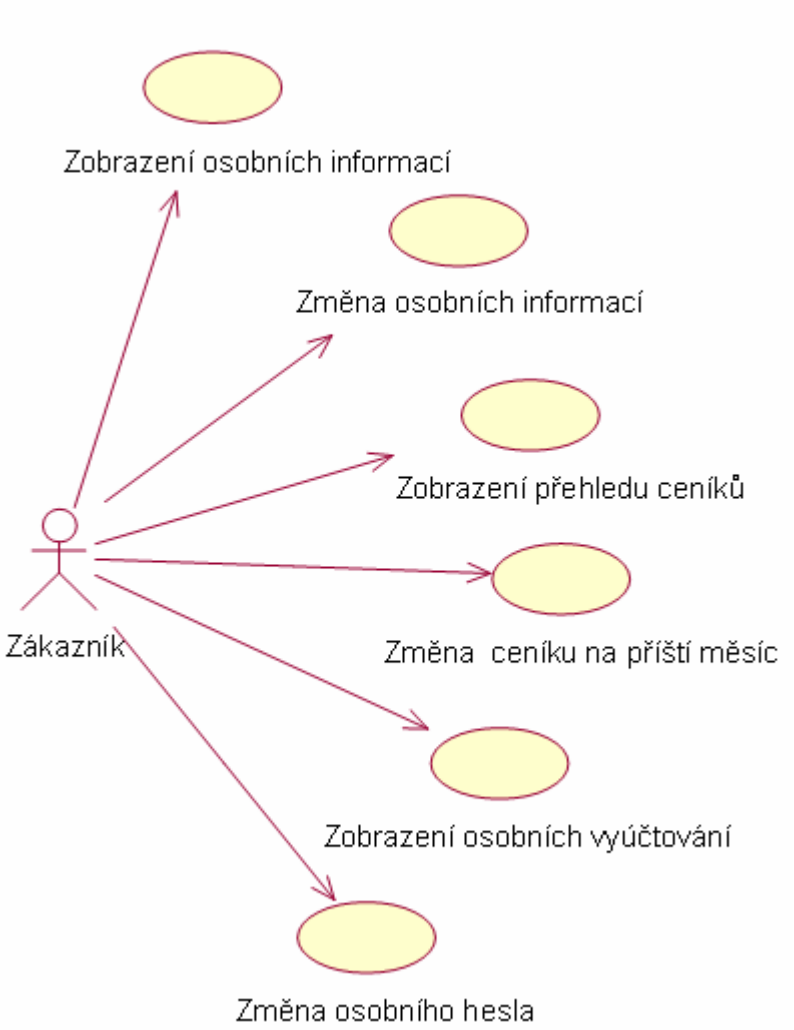


Schéma 3.3 Diagram případů užití zákazníka

3.10 Návrh databáze

Databáze je navržena s ohledem na možnosti rozvíjení systému. Tabulky jsou typu InnoDB, který umožňuje používat cizí klíče a integritní omezení. ER diagram návrhu databáze je znázorněn ve schématu 3.4

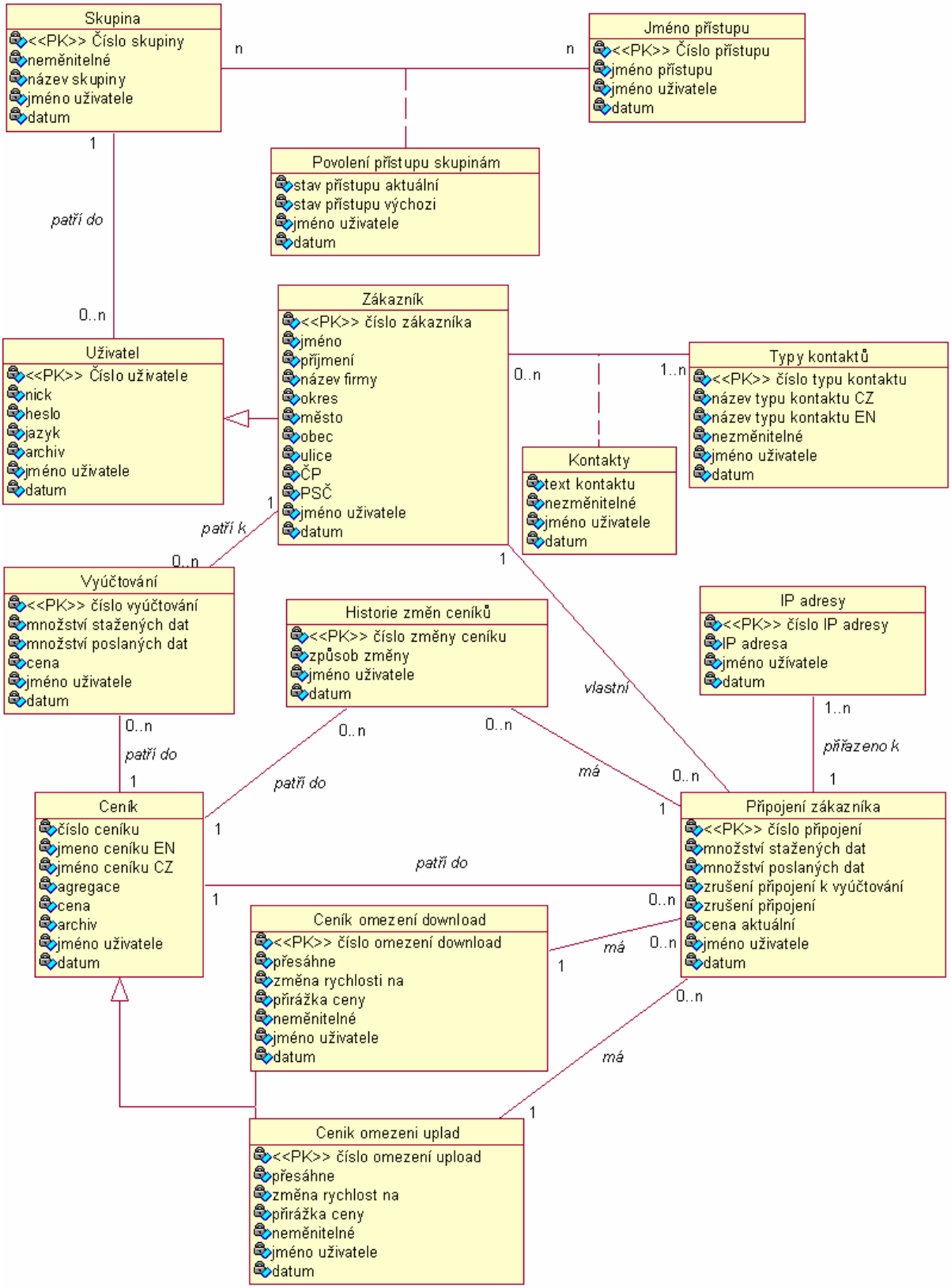


Schéma 3.4 ER diagram

Databáze obsahuje tabulky pro uložení práv jednotlivých skupin uživatelů. Tato práva lze do databáze přidávat neomezeně a lze je také přiřazovat k jednotlivým skupinám. Také tabulka se skupinami není omezena. Z toho vyplývá, že správa práv je plně flexibilní a ke změně práv není potřeba jediné změny kódu informačního systému, pouze stačí změnit data v databázi. Rovněž je zřejmé, že přidávání nových skupin není problémem a při rozšiřování systému lze přidat nové skupiny a nová práva. Jelikož však pro navrhovaný informační systém bohatě postačují pouze tři skupiny a práva těchto skupin jsou přesně daná, nebude potřeba pro správu práv a skupin vytvářet webové rozhraní. Pokud by se systém v budoucnu rozšiřoval, nic nebrání vytvoření tohoto rozhraní, jelikož je na něj vše připraveno.

Tabulky pro uložení kontaktů a typů kontaktů (email, ICQ, telefon, mobil...) jsou také navrženy tak, aby dané položky bylo možno rozšiřovat v případě, že by tento požadavek vyvstal. Tudíž na každého zákazníka může být libovolný počet kontaktů.

4 Implementační jazyky

V této kapitole jsou uvedeny základní informace o jednotlivých implementačních jazycích, použitých při tvorbě programu Fair User Policy.

4.1 PHP

PHP (Hypertext Preprocessor) je skriptovací jazyk, který umožňuje procedurální i objektově orientované programování (není ovšem podmínkou psát objektově) [4]. Je to beztypový jazyk, což znamená, že jednotlivé proměnné mohou nabývat libovolných hodnot a měnit svůj typ. PHP je šířeno jako Open Source.

PHP je nástroj pro programování na straně serveru (server-side). Lze ho použít k tvorbě interaktivních stránek. S určitým zjednodušením můžeme říct, že skript je spuštěn na serveru podle zadaných kritérií (parametrů) a tím vygeneruje HTML kód, který je zaslán uživateli. Jakmile uživatel stránku načte, nelze ji už modifikovat pomocí PHP, ale je to možné pouze pomocí skriptů na straně uživatele, například pomocí JavaScriptu. Nevýhodou JavaScriptu je slabá bezpečnost, a proto jej mají někteří uživatelé zakázány.

4.1.1 Historie PHP

Počátky se datují od roku 1995, kdy Rasmus Lerdorf napsal skript Perl/CGI, který dokázal zjistit, kolik návštěvníků četlo jeho online résumé. Tento skript uměl protokolovat informace o návštěvnících a tyto informace (počet návštěvníků) zobrazit na webové stránce. Tento nástroj byl prvním svého druhu. Značný zájem, který vyvolal, přiměl Rasmuse Lerdorfa, aby jej dal k dispozici ostatním uživatelům. Určil mu název Personal Home Page (PHP).

Velký zájem o PHP vedl Rasmuse Lerdorfa k dalšímu vývoji nástrojů PHP, využitelných například při převodu dat z formuláře HTML do symbolických proměnných tak, aby bylo možné zasílat data i na jiné systémy. Další a další přidávání nových funkcí do nástrojů PHP vykrystalizovalo k vydání další verze PHP, a to PHP 2.0 (Personal Home Page – Form Interpreter). PHP se u programátorů těšilo čím dál větší oblibě, což vedlo k jeho mnohým rozšířením.

V roce 1997 byl změněn význam názvu nástrojů PHP. Z Personal Home Page se změnil na Hypertext Preprocessor.

V roce 1998 vyšla další verze, ve které bylo implementováno mnoho nových funkcí a došlo k masivnímu rozšíření PHP. Nyní se implementací zabývají hlavně vývojáři Zeev Suraski a Andi Gutmans, kteří přepsali parser PHP.

Rok 2002 přináší čtvrtou verzi PHP, ve které se objevují objektové prvky. Po ní je vydána doposud poslední verze PHP, a to verze PHP 5, jejímž největším přínosem je zdokonalení objektově orientovaného programování.

4.2 MySQL

MySQL (My Structured Query Language) je relační databáze typu DBMS (database managment system), která vychází z deklarativního programovacího jazyka SQL (Structured Query Language) [5]. Ten je šířen jako Open Source. Jedná se o architekturu client/server.

Do databáze je možné ukládat různá data počínaje textem přes obrázky atd. Databáze je složena z tabulek a data jsou ukládána po řádcích, přičemž sloupec určuje typ uložených dat. MySQL databáze se velmi často používá s PHP, pomocí kterého je možné s daty v databázi pracovat (ukládat, mazat, editovat). Existuje řada dalších jazyků, pomocí kterých můžete s MySQL pracovat, například Java, Perl, Python, Visula Basic a mnoho dalších.

Pro správu databáze je nejčastěji používán PhpMyAdmin, což je program napsaný pomocí PHP, který je Open Source. Tento nástroj umožňuje kompletní práci s databází.

4.2.1 Historie MySQL

MySQL vyvinuli zaměstnanci ze švédské firmy TCxDataKonsult. Roku 1996 bylo poprvé MySQL uvolněno pro veřejnost. Roku 2001 vzniká firma, která se převážně zabývá jen MySQL, takže se MySQL stalo brzy velmi populárním a rychle se šířilo do celého světa.

Od počátku byl kladen velký důraz na výkon. Díky tomu vznikl optimalizovaný produkt, kterému ale ze začátku chyběly některé schopnosti, jako například uložené procedury a triggery. V dnešní verzi MySQL 5 už jsou veškeré schopnosti implementovány.

4.3 HTML

HTML je zkratka Hypertext markup Language (hypertextový značkovací jazyk) [6]. HTML je značkovací jazyk, který používá definované značky (tagy) k vytváření a formátování dokumentů pro webové stránky. Jazyk je aplikací dříve vyvinutého rozsáhlého univerzálního značkovacího jazyka SGML (Standard Generalized Markup Language). HTML je jakoby návod, jak zobrazit přijatá data na obrazovce pomocí programu obecně zvanému prohlížeč.

Poslední aktuální verzí HTML jazyka je verze 4.01. HTML jazyk se v dnešní době už nevyvíjí. Normy pro HTML jazyk schvaluje mezinárodní konsorcium W3C. Náhradou za HTML se stává nový značkovací jazyk XHTML a XML

4.3.1 Historie HTML

V roce 1990 byl navržen jazyk HTML, na jehož vývoji se podíleli Tim Berners-Lee a Robert Caillau. K HTML vznikl protokol pro přenos po síti HTTP (HyperText Transfer Protocol – přenosový protokol hypertextu). Následovně také Tim Berners-Lee vytvořil první prohlížeč WorldWideWeb.

4.4 CSS

CSS (Cascading Style Sheets) je oddělením layoutu (formátování vzhledu) od struktury dokumentu HTML [7]. CSS vzniklo z potřeby sjednotit vzhled stránek dokumentů, protože během vývoje byla objevena celá řada nových tagů. Většina z nich byla podporována pouze některými prohlížeči a každý prohlížeč měl navíc své specifické tagy, z toho důvodu se tedy stránka zobrazovala v každém prohlížeči trochu jinak. Se záměrem vyřešit tento problém začalo konsorcium World Wide Web Consortium (W3C) vyvíjet standart CSS, který by měl daný problém řešit.

CSS je tedy definice stylů zobrazení stránky. Tyto styly mohou být vkládány přímo do stránky nebo mohou být nalinkovány, díky čemuž je možné jejich využití pro více stránek či dokonce pro celý server. Velká síla CSS tkví ve flexibilitě změny vzhledu celé stránky, jež je umožněna pouhým zaměněním definice CSS. Díky tomuto zaměnění se změní vzhled stránky, aniž by programátor musel zasáhnout do struktury dokumentu.

4.4.1 Historie CSS

V roce 1994 v Chicagu ukázal Hakon Wium Lie návrhl „Cascading HTML Style Sheets," ze kterého vycházejí kaskádové styly do dnes.

O několik dní později byl vydán prohlížeč Netscape, který stylování dokumentu řešil po svém. Nevyužíval totiž stylového jazyka, ale zabudoval základní formátovací prostředky přímo do jazyka HTML. Toto řešení nebylo příliš šťastné a jeho následky neseme až dodnes, jelikož tím byla značně zkomplikována kompatibilita a použitelnost webových dokumentů.

Po vydání návrhu v roce 1994 se ještě dlouhý čas vedly spory o konkrétní podobu, které byly ukončeny až v roce 1996, kdy bylo CSS 1 v prosinci doporučeno W3C. V dnešní době je CCS 1 už téměř stoprocentně podporováno všemi prohlížeči. W3C dále kaskádové styly rozvíjelo a v květnu 1998 vydalo další specifikaci, a to CSS 2, která kromě grafické podoby dokumentů umožňuje definovat také styly pro dokumenty, jako jsou hlasové dokumenty atd. Po CSS 2 se začala vyvíjet další verze CSS 3.

4.5 Perl

Perl je interpretovaný programovací jazyk [8], který je velmi populární pro tvorbu CGI skriptů. Je možné jej používat ke zpracovávání různých dokumentů (například textových), umožňuje také práci s databází a mnoho dalších.

4.5.1 Historie Perlu

Perl byl vytvořený Larry Wallem v roce 1987. Původně byl používán jako náhrada jazyka AWK a interpretru sh. Velké rozšíření zaznamenala verze 4 z roku 1991. Další důležitou verzí byla verze 5, která přinesla výkonné datové struktury a možnost objektového programování. V dnešní době se vyvíjí nová verze, a to verze 6, která by měla běžet na zcela novém jádře.

4.6 JavaScript

JavaScript je objektový beztypový skriptovací jazyk, používaný při tvorbě HTML stránek [9]. Lze v něm jednoduše vytvářet a manipulovat s objekty, řídit vzhled a obsah zobrazovaného dokumentu, plně ovládat formuláře, částečně řídit prohlížeč, provádět libovolné matematické výpočty, manipulovat s vloženými obrázky a ukládat a číst data na straně klienta v podobě Cookies. Skripty jsou zapisovány přímo do HTML kódu stránky a prohlížeč je interpretuje až na klientském počítači, díky čemuž nejsou potřeba pro tvorbu skriptů žádné speciální nástroje nebo překladače. Výhodou je také všeobecná dostupnost zdrojových kódů. Problémem při používání JavaScriptu je ovšem nekompatibilita jednotlivých prohlížečů, které podporují různé verze jazyků a často je také různými způsoby interpretují. Může se také stát, že některý z prohlížečů nepodporuje JavaScript vůbec. Přesto je tento skriptovací jazyk velmi oblíbený nejen na internetových stránkách, ale také v aplikacích díky své jednoduchosti a svému rozšíření.

4.6.1 Historie JavaScriptu

U zrodu tohoto jazyka stojí Brendan Eich, zaměstnanec firmy Netscape, který dostal za úkol vytvořit skriptovací jazyk schopný integrace do internetového prohlížeče Netscape Navigator [9]. Zhruba v té samé době přišla firma Sun s jazykem Java a Eich měl tedy dvě možnosti. Buď Javu přijmout, nebo vytvořit jazyk vlastní. Nakonec se přiklonil k možnosti druhé, a tak vznikl jazyk s názvem Livescript.

Jeho první implementace proběhla v září 1995 v prohlížeči Netscape Navigator 2.0 a ještě v témž roce (4. 12. 1995) byl tento jazyk přejmenován na JavaScript.

Od svého vzniku prošel JavaScript bouřlivým vývojem. Přes verze 1.2 (implementovaná v prohlížeči Netscape Navigator 3) a 1.3 (implementovaná v Netscape 4.5) až po aktuální verzi 1.5, implementovanou prohlížeči Netscape 7.0 a Mozilla 1.0.

5 Implementace

Implementace informačního systému a databáze byla provedena podle návrhu v kapitole 3 a pomocí výše popsaných implementačních jazyků (PHP, MySQL, HTML, CSS, Perl, JavaScript). Z tohoto důvodu je v této kapitole uveden pouze popis implementace skriptů, která je poněkud zajímavější. Tento popis zde uvádím jako jediný také proto, aby zde nebyly znovu uváděny již jednou napsané informace z kapitoly číslo 3 Návrh informačního systému.

5.1 Implementace skriptu pro vyúčtování

Skript vyúčtování je napsán jako skript PHP/CGI, jenž je možno pravidelně spouštět utilitou cron, která je standardně obsažena v každém Linuxovém systému, a jež umožňuje naplánování spouštění skriptů v libovolných časových periodách. Zpravidla se v cronu nastaví, aby se spouštěl každý měsíc.

Tento skript vynuluje data v tabulce připojení a nastaví nová výchozí data pro daný měsíc. Před tím, než jsou všechna data vynulována, přepíše skript do tabulky vyúčtování informace o množství stažených a poslaných dat a zapíše také cenu z tabulky připojení. Tato cena je již cenou skutečnou, kterou má zákazník platit, a obsahuje už všechny přirážky.

Pokud uživatel nebo zákazník v průběhu měsíce nastavili změnu ceníku na příští měsíc, provede se tato změna nyní a zároveň se při této změně zapíše do tabulky, obsahující informace o změnách ceníku, informace o tom, že tato změna proběhla.

Jestliže uživatel v průběhu měsíce nastavil požadavek na zrušení připojení při vyúčtování, tento požadavek bude proveden také ihned po vytvoření vyúčtování.

5.2 Implementace skriptu pro zjištění množství přijatých a odeslaných dat

Skript pro zjišťování množství odeslaných a přijatých dat je napsán v jazyce Perl. Tento jazyk jsem si zvolil záměrně, protože jsem v něm ještě nikdy nepracoval a chtěl jsem při zpracovávání bakalářské práce získat nové poznatky a zkušenosti. Ta část skriptu, která pracuje s databází, je z něj přesunuta přímo do vložené procedury do Mysql databáze, aby se tak zvýšila rychlost prováděného skriptu a snížila režie na komunikaci mezi skriptem a databází. Ze skriptu je jen volána vložená procedura s příslušnými parametry, která vrátí požadované informace o tom, zda byl limit ceníku překročen.

Skript může být spouštěn, podobně jako skript pro vyúčtování, pomocí cronu. A protože data jsou pomocí NetFlow ukládána zpravidla každých pět minut, bylo by na místě tento skript také nastavit tak, aby se spouštěl každých pět minut. Pokud bychom nastavili spouštění skriptu po delším

časovém intervalu, byl by systém z důvodu velkého množství zpracovávaných dat nerovnoměrně vytížený.

Funkce skriptu spočívá ve zjištění množství stažených a odeslaných dat pro jednotlivé IP adresy a uložení těchto dat do databáze. V případě, že dojde k překročení limitu, je odeslán zákazníkovi, který tento limit překročil, automatický email s informacemi o překročení limitu a následných omezeních. Tento email může být zaslán pouze na adresu, která je označena jako nesmazatelná (není umožněno jí smazat). Zákazník má sice možnost nastavit si více svých emailových adres, ale pokud by mu měl být email zasílán na každou z nich, mohlo by to zpomalovat práci skriptu, což je při takovém množství zpracovávaných dat nežádoucí. Zákazník si poté může samozřejmě nastavit přesměrování z emailové adresy, která mu byla poskytnuta při jeho zadání do systému, na jakýkoliv jím zvolený e-mail.

V první řadě si skript získá z pomocných konfiguračních souborů informace o tom, která data má zpracovávat. Pokud jsou daná data k dispozici, zpracuje je a po zpracování nastaví konfigurační soubory pro příští použití. V případě chyby při zpracování dat nebo problému s načítáním konfiguračních souborů, pošle skript informaci o chybě na email, který je nastaven na začátku tohoto skriptu jako email správce.

Závěr

Úkolem bakalářské práce bylo seznámit se s nástroji pro tvorbu webových informačních systémů a s technologií NetFlow pro monitorování sítí. Dále pak provést analýzu požadavků pro systém umožňující nastavování a kontrolu FUP, účtování za překročení FUP, tvorbu reportů na základě NetFlow dat a na těchto poznatcích vytvořit Informační systém pro správu Fair User Policy. Dané požadavky zadání jsem splnil, seznámil jsem se z dostupnými technologiemi pro tvorbu informačních systémů (viz. kapitola 4), a také s technologií NetFlow (viz. kapitola 1). Na základě získaných obecných informací a po provedení analýzy požadavků na systém FUP jsem navrhnul informační systém (viz. kapitola 3) a posléze provedl jeho implementaci.

Grafické prostředí jsem vytvořil co nejintuitivnější a co nejpřehlednější, aby se uživatel během krátkého seznámení s daným prostředím rychle sžil. Navržené grafické prostředí jsem nechal testovat skupinou lidí s různou úrovní znalostí práce na PC. Jejich reakce byly různorodé, což bylo především způsobeno předchozími zkušenostmi se stejně nebo podobně řešeným uživatelským rozhraním. Po krátkém zasvěcení však byla většina těchto osob schopna s tímto systémem pracovat bez problému. Do budoucna je to jistě možná oblast pro další vylepšování.

Možných vylepšení a rozšíření může být pro daný systém celá řada, jak už to tak bývá při vzniku téměř každého nového systému. Strukturu databáze jsem navrhl tak, aby případná rozšíření nečinila z hlediska databáze velké potíže.

Jednou z možných oblastí rozšíření je správa uživatelských práv. Pro toto rozšíření je databáze přímo navrhnutá, ale z podstaty systému zatím nebylo potřeba implementovat, protože dosavadní role vyhovují daným požadavkům. Rozšíření o správu rolí by se dalo využít například v případě, kdy by se do systému naimplementovala správa účetnictví, kde by byl potřeba daleko flexibilnější přístup práv. Dalším zajímavým rozšířením by byla možnost zobrazovat ceny ceníku podle aktuálního kurzu zvolené měny. V dnešní době, kdy je například ve velké části Evropy běžně používáno Euro, by tato funkce byla bezesporu výhodou. V neposlední řadě by také bylo možné rozšířit systém pro používání IPv6 a mnoho dalších.

Práce na této bakalářské práci mě zaujala. Přínosem pro mě bylo hlavně bližší seznámení s technologií NetFlow a práce nad daty touto technologií získanými. Dalším velkým přínosem pro mě bylo zdokonalení se při práci s implementačními jazyky, a to především s PHP a MySQL. S MySQL se sice setkávám v zaměstnání, ale pracuji s ním spíše jen na úrovni získávání dat z databáze, a tak implementace databáze od začátku mi pomohla k lepšímu pochopení práce s dotazy. Jsem přesvědčen, že získané znalosti mi budou platné i v dalším profesním životě.

Literatura

[1] *Wickipedia : otevřená encyklopedie* [online]. 25 April 2008 [cit. 2008-04-26]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Netflow>.

[2] ŽÁDNÍK , Martin. *NetFlow* [online]. 18.12.2007 [cit. 2008-04-20]. Prezentace k předmětu Síťové aplikace a správa sítí. Dostupný z WWW: <https://wis.fit.vutbr.cz/FIT/st/course-files-st.php/course/ISA-IT/lectures/isa_netflow.pdf>.

[3] *Sourceforge.net : nfdump* [online]. Oct 15 2007 [cit. 2008-04-27]. Dostupný z WWW: <http://nfdump.sourceforge.net/>.

[4] GILMORE, W. Jason. *Velká kniha PHP a MySQL 5 : kompendium znalostí pro začátečníky i profesionály*. 1. vyd. Brno : ZONER Press, 2007. 864 s. ISBN 80-86815-53-6.

[5] KOFLER, Michael, et al. *Mistrovství v MySQL 5 : kompletní průvodce webového vývojáře*. 1.vyd. Brno : Computer Press, 2007. 805 s. ISBN 978-80-251-1502-2.

[6] *Wickipedie : otevřená encyklopedie* [online]. 8.4.2008 [cit. 2008-04-29]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/HyperText_Markup_Language>.

[7] PROKOP, Marek. CSS pro webdesignery. 2.vyd. Brno : CP Books, 2005. 288 s. ISBN 80-251-0487-7.

[8] *Wickipedie : otevřená encyklopedie* [online]. 2.3.2008 [cit. 2008-05-03]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Perl>.

[9] HOLZNER, Steven. JavaScript profesionálně : kompletní referenční příručka. 1. vyd. Praha : Mobil Media, 2003. 1071 s. ISBN 80-86593-40-1.

Seznam obrázků a schémat

Seznam příloh

Příloha 1. CD s následujícím obsahem:

[a] zdrojové kódy systému – složka FUP

[b] návod instalace – soubor INSTALACE.pdf

[c] manuál – soubor MANUAL.pdf

[d] technická zpráva – soubor BP.pdf