Česká společnost uživatelů otevřených systémů EurOpen.CZ
Czech Open System Users' Group
www.europen.cz

45. konference

Sborník příspěvků

Hotel Šumava, Kašperské Hory
5.–8. 10. 2014

Programový výbor:

Zdeněk Šustr
Jiří Sitera
Jan Kynčl

Sborník příspěvků z 45. konference EurOpen.CZ, 5.–8. 10. 2014

Plzeň 2014. První vydání.
Editor: Vladimír Rudolf
Sazba a grafická úprava: Ing. Miloš Brejcha – Vydavatelský servis, Plzeň
e-mail: servis@vydavatelskyservis.cz

Tisk: Typos, tiskařské závody, s. r. o.
Podnikatelská 1160/14, Plzeň

Příspěvky neprošly redakční ani jazykovou úpravou.

ISBN 978-80-86583-28-0

Obsah

Platby mobilním telefonem

Martin Chlumský

E-mail: martin.chlumsky@hp.com

V průběhu posledních let přistoupila většina českých bank k migraci na bezkontaktní platební karty. Zatímco někteří své první zkušenosti s bezkontaktními kartami teprve získávají, jiní již začali používat alternativní formy – bezkontaktní nálepky, přívěšky na klíče nebo telefony. Tento příspěvek je věnován právě mobilním telefonům. Nejde o detailní technický popis protokolů a standardů, nýbrž o stručné seznámení s dvojicí zcela odlišných způsobů placení telefonem – s mobilními platbami na běžných bezkontaktních terminálech prostřednictvím technologie NFC a s Internetovými platbami mobilním telefonem zapojeným do programu MasterCard Mobile Remote Payment.

1 NFC card emulation

V první části se budeme věnovat placení telefonem emulujícím kartu pomocí technologie NFC. V tomto případě telefon nahrazuje skutečnou kartu a lze s ním tedy platit na běžných terminálech u obchodníků. Popis průběhu bezkontaktní transakce není předmětem tohoto článku a lze jej nalézt v jiném příspěvku této konference.

1.1 Potřebné vybavení

1.1.1 Hardware

Aby mohl být telefon použit k bezkontaktním platbám, musí obsahovat dvě základní komponenty – NFC obvod s anténou a Secure Element.

NFC (Near Field Communication) představuje množinu standardů zahrnujících bezkontaktní technologie určené k snadné výměně informací na krátkou vzdálenost v řádech centimetrů. NFC se opírá především o RFID standardy ISO/IEC 14443 a 18092, zahrnuje ale také standard ISO/IEC 15693 s delším dosahem a nižší přenosovou rychlostí. Technická specifikace standardu definuje tři operační módy:

- „Peer-to-Peer“ sloužící k vzájemné výměně dat nebo sdílení souborů (využití např. k jednodušší konfiguraci zařízení se složitějšími protokoly jako je Wi-Fi)

- „Reader/writer“ umožňující číst nebo zapisovat NFC tagy (např. v reklamních plakátech)
- „Card emulation“, kdy NFC zařízení vystupuje jako bezkontaktní karta ISO/IEC 14443 nebo FeliCa (platby, hromadná doprava apod.)

My se dále budeme věnovat režimu emulace karet komunikujících protokolem ISO 14443, který umožňuje mobilním telefonům s rozhraním NFC, aby mohly provádět transakce na bezkontaktních platebních terminálech. Všechny telefony ovšem anténou a NFC čipem vybaveny nejsou. Naštěstí existují i řešení, která mohou pomoci překlenout tento nedostatek do doby, kdy bude na trhu dostatek zařízení s touto technologií. Takovou alternativou může být speciální microSD karta nebo ochranný obal s anténou a potřebným čipem.

U platebních čipových karet je kladen důraz na bezpečnost, proto by měla být data a samotná aplikace představující platební kartu i v případě mobilního telefonu uložena a provozována v bezpečné oblasti. Tou není paměť telefonu a CPU, nýbrž Secure Element (SE), který vyhovuje všem potřebným bezpečnostním kritériím a certifikacím. V režimu emulace karty komunikuje NFC kontrolér přímo se Secure Elementem bez účasti CPU. Takové prostřední garantuje požadovanou úroveň bezpečnosti. SE může být rovněž implementován jako dodatečný čip v telefonu, speciální microSD, nová generace SIM (UICC) nebo jako čip v ochranném obalu na telefon. Největší standardizace ovšem panuje kolem UICC.

1.1.2 Software

V telefonu musí být rovněž nainstalována následující sada aplikací – platební aplikace reprezentující kartu, mobilní aplikace plnící funkci grafického rozhraní a aplikace PPSE s katalogem bezkontaktních platebních aplikací.

Platební aplikace uložená v SE je ekvivalentem aplikace na platební kartě. V implementaci jsou ale přeci jen drobné rozdíly. Aplikace podporuje novou metodu ověření držitele prováděnou přímo na mobilním telefonu (Passcode), nikoliv na platebním terminálu. Aplikace také nikdy nemůže nutit držitele, aby strčil telefon do kontaktního rozhraní, kde obvykle probíhají aktualizace interních objektů karet. Tyto operace musí být nově dostupné prostřednictvím sítě mobilního operátora (MNO) nebo Wi-Fi. Platební aplikace za tímto účelem obsahuje nové datové elementy sloužící k předávání informace o stavu nebo potřebě komunikace s bankou či držitelem.

Kromě výše zmíněné platební aplikace obsahuje telefon i mobilní aplikaci poskytující grafické rozhraní. Ta umožňuje držiteli výběr z většího množství karet, prohlížet historii transakcí, měnit platební preference nebo ověřit držitele během platby na telefonu. Tato aplikace není uložena v SE, ale spolu s ostatními programy v paměti telefonu. Implementace může představovat specializovanou aplikaci nabízející rozhraní k jedné platební kartě nebo mobilní peněženku, do které lze vložit více karet, přijímat slevové kupóny, reklamy apod. Aplikace je schopna na základě stavu platební aplikace nebo přijetí notifikace z banky (Talk-to-me

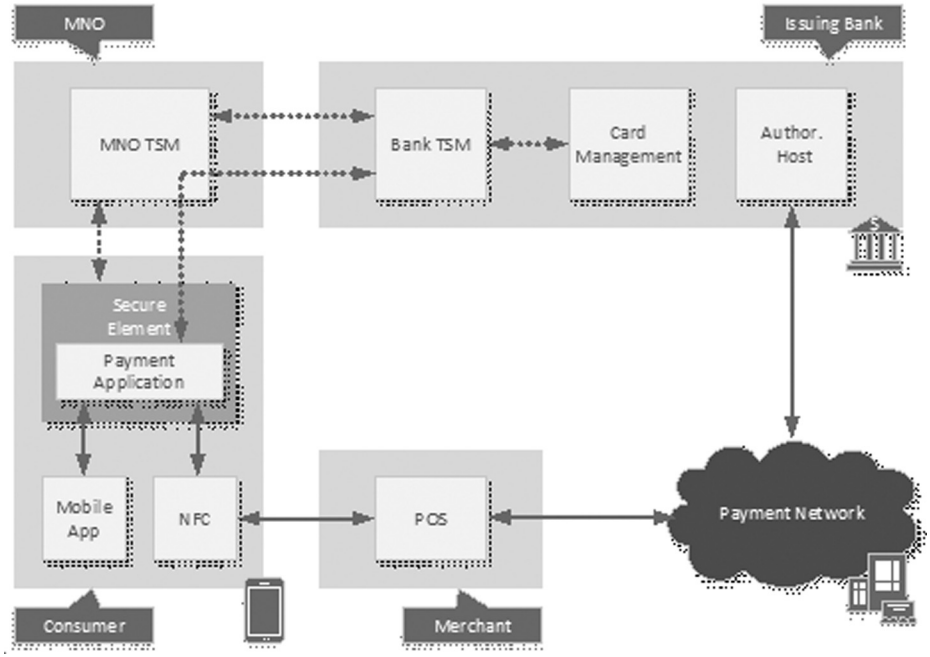
push message) zajistit spojení s bankou a provést aktualizaci interních objektů. K dalším službám může patřit dynamické generování kódu CVV2, který je běžně uveden na zadní straně karty a je používán při platbách na Internetu, dále také dobíjení kreditu, generování jednorázového hesla pro přístup do eBankovnictví apod. Mobilní aplikace také ovlivňuje konfiguraci poslední důležité aplikace – PPSE.

PPSE (Proximity Payment System Environment) představuje katalog dostupných bezkontaktních aplikací, který je načítán platebním terminálem ještě před zahájením transakce. Tradičně obsahuje PPSE statický seznam, v případě telefonu musí být ale konfigurace aplikace PPSE aktualizována na základě množiny aktivovaných platebních aplikací a potřeb uživatele. Katalog může např. obsahovat více platebních aplikací s různými preferencemi. Pokud se majitel rozhodne, že nechce, aby byly platební aplikace pro okolní svět stále viditelné, musí být seznam PPSE smazán. Pokud si naopak přeje, aby mohl zaplatit kdykoliv pouhým přiložením bez nutnosti nejprve nastartovat mobilní aplikaci, musí v PPSE zůstat alespoň jediný záznam s informací o preferované kartě.

1.1.3 Instalace aplikace

Průběh instalace programového vybavení a dat je závislý na použitém HW a potřebách banky. Uveďme dva rozdílné příklady. Banka se rozhodla pro řešení založené na ochranných obalech obsahujících NFC čipy, antény a SE. Důvodem mohla být podpora klientů vlastnících telefony iPhone a také nezávislost na SE v UICC kartách mobilních operátorů. V takovém případě může personalizace platební aplikace a dat karty probíhat v běžném personalizačním centru, víceméně jako tomu je u normálních karet. Zákazník převezme HW a nainstaluje si běžným způsobem mobilní aplikaci. Od této chvíle je schopen provádět platby.

Proces může být ale mnohem komfortnější. Takový model je obvykle spojen s telefony vybavenými NFC čipem a SE ve formě UICC. Držitel zavolá do banky a sjedná si požadovanou službu. Banka mezi tím spustí proces, který prověří vhodnost telefonu a UICC k danému účelu. Pokud je vše v pořádku, může být vzdáleně (OTA – Over The Air) nainstalováno veškeré SW vybavení a potřebná data. Celý ekosystém se v takovém případě opírá o TSM (Trusted Service Manager), který zajišťuje vzdáleně bezpečnou správu SE – Provisioning and Management. TSM si lze představit jako mobilní personalizační centrum, které je schopné online řešit požadavky banky (ale i jiných poskytovatelů služeb) při zachování end-to-end bezpečnosti. Celý model je ještě komplikovanější tím, že banka není vlastníkem SE a část prostoru na UICC si pouze pronajímá od MNO. TSM je obvykle provozován MNO nebo společnostmi, které se zabývají výrobou a personalizací čipových karet. Závislost bank na operátorech, vysoká cena implementace TSM a zdržování projektů regulačními úřady v případech, kdy se operátoři snaží rozložit náklady spojením sil, výrazně přibrzdily rozšíření NFC plateb.



Obr. 1

1.2 Placení na POS terminálu

Vlastní placení na terminálu je podobné bezkontaktním kartám. U podlimitních transakcí (u nás do 500 Kč) může být transakce provedena bez verifikace držitele, nad tento limit je terminálem požadováno ověření. Slovo „může“ nebylo použito v předchozí větě náhodně, neboť v závislosti na nastavení vydavatele a preferencí držitele může být pro jistotu požadováno telefonem ověření při každé platbě.

V předchozích odstavcích byla zmíněna nová verifikační metoda označovaná jako Passcode u asociace VISA nebo mPIN u MasterCard. Pokud se vydavatel rozhodl využívat tuto metodu (obvykle ano), a ta je podporována i terminálem, proběhne verifikace držitele v případě potřeby přímo na mobilním telefonu. U bezkontaktních transakcí dochází k verifikaci držitele vždy až po dokončení komunikace terminálu s kartou. U této metody je po dokončení vzájemné komunikace terminálu a telefonu zobrazena držiteli výzva k ověření a po vlastním ověření je telefon přiložen podruhé. Teprve v tuto chvíli může dojít k transakci obsahující informace o výsledku ověření držitele. Dvojímu přikládání se lze vyhnout tak, že držitel, který očekává vyšší útratu, provede např. při čekání ve frontě ověření předem, ještě před prvním přiložením telefonu k terminálu.

Na rozdíl od karet, které jsou v dosahu elektromagnetického pole kdykoliv ochotny provést transakci, umožňují telefony řídit své chování, pokud není mobilní aplikace aktivní. V zásadě jsou zajímavé následující příklady nastavení: Transakce je možná pouze tehdy, pokud má držitel na displeji spuštěnou mobilní aplikaci a potvrdil, že chce telefonem zaplatit. Jiné nastavení naopak dovolí automaticky probudit nebo spustit mobilní aplikaci přiložením telefonu k terminálu. Poslední zajímavý příklad je závislý na možnostech HW a SW a umožňuje provedení některých plateb i v případě, kdy je vybitá baterie telefonu. Energie elektromagnetického pole terminálu je dostatečná k napájení NFC čipu a SE. V tomto případě musí jít ale o transakci, která nesmí vyžadovat interakci na telefonu.

1.3 Placení na Internetu

S běžnou kartou můžeme ve většině případů platit také na Internetu. Karta v mobilním telefonu není fyzická a neobsahuje tedy zadní stranu s CVC2/CVV2 kódem. Platební aplikace uložená v SE ale může obsahovat i objekty s citlivými údaji karty. Číslo karty, doba platnosti a CVC2/CVV2 pak mohou být zobrazeny po úspěšném ověření držitele na displeji telefonu. Pro zvýšení bezpečnosti připravila asociace Visa možnost dynamického generování hodnoty nahrazující tento kód.

1.4 Risk Management

Čipové platební karty obsahují řadu datových elementů potřebných pro vyhodnocování rizika offline transakcí. Na základě risk analýzy pak karta transakci provede offline, bez potřeby spojení s bankou, nebo naopak online, kdy terminál čeká na odpověď vydavatelské banky. Objekty, které představují různé akumulátory offline utracených částek nebo čítače počtu offline provedených transakcí, je nutné občas resetovat, jinak by veškeré platby po nějaké době probíhaly pouze online. K resetování dochází u normálních karet v kontaktním rozhraní, kdy je karta delší dobu ve styku s terminálem a kdy dostane zpět podepsaná data z banky s informací, jak má s interními objekty karty naložit. U telefonů ke vkládání do kontaktního rozhraní nedochází, a tak komunikaci s bankou zajišťuje mobilní aplikace na základě příznaků obdržených z platební aplikace. V případě potřeby provede mobilní aplikace obdobu platební transakce s nulovou částkou zaslané kontaktním rozhraním do platební aplikace. Kontaktním rozhraním je myšleno rozhraní mezi SE a telefonem. Vygenerovaný kryptogram (ARQC) je pak zaslán prostřednictvím MNO nebo Wi-Fi skrze VISA Mobile Gateway, resp. Reset Server (MasterCard), do vydavatelské banky. Přijaté pokyny z autorizačního centra banky jsou pak mobilní aplikací předány zpět platební aplikaci, která provede ověření dat a potřebné operace. Resetování čítačů, a tedy lepší připravenost na další offline použití (např. pro transportní služby), může být iniciováno

rovněž uživatelem. V obou případech je ale obvykle držitel karty ověřen, aby nemohlo dojít k nekontrolovanému „dobíjení“ karty umožňující provádění podlimitních plateb při ztrátě telefonu.

Aplikace může rovněž využívat čítače a akumulátory, které slouží výhradně k ochraně před zneužitím ztraceného nebo ukradeného telefonu. S každou transakcí jsou tyto objekty aktualizovány a při dosažení stanoveného limitu je požadováno ověření držitele. K resetování těchto čítačů postačí úspěšná verifikace majitele telefonu při libovolné operaci i bez kontaktu aplikace s bankou.

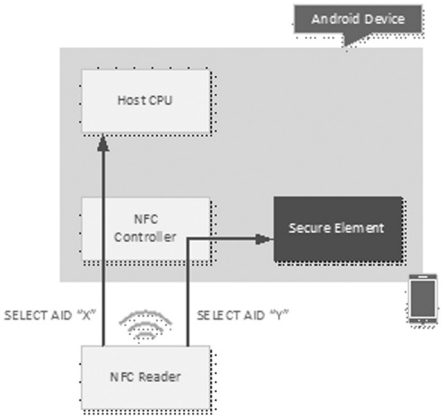
Aby nedocházelo zbytečně k opakovanému dotazování držitele na Passcode/mPIN, je u ověření možné definovat, pro jakou množinu následujících operací je toto ověření platné (VISA – platba, resetování čítačů, generování jednorázového hesla) nebo také, zda je ověření platné pouze pro jednu transakci nebo po stanovenou dobu (MasterCard). MasterCard umožňuje kromě ověření mPIN rovněž využívat pouhé potvrzení na displeji a obdobně jako u verifikace držitele lze definovat situace, kdy má být potvrzení vyžadováno.

1.5 Budoucnost s HCE?

Jedním ze základních předpokladů mobilních plateb přes NFC byla přítomnost SE. Tento prvek je jediným místem, kam jsou při emulaci karty z NFC čipu směrována data. Nová verze OS Android 4.4 KitKat (a také Blackberry OS 10) přichází s alternativou pojmenovanou HCE (Host Card Emulation), která dovoluje na základě interní směrovací tabulky systému určit, zda pro danou aplikaci směrovat data do SE nebo do normálního CPU („the host“). To znamená, že platební aplikace, která doposud musela být umístěna v SE, může být nově uložena ve standardní paměti telefonu a citlivá data např. v „cloudu“ nebo TEE (Trusted Execution Environment). To je obrovský průlom, který může usnadnit rozšíření technologie, neboť odpadá výměna SIM karet, komplikovaná správa SE, závislost na MNO apod. Na druhou stranu tím může být snížena bezpečnost systému. Z pilotních projektů v některých zemích je ale patrné, že platební asociace nechtějí zaspat. Hlavní výrobci SE (SIM aliance) naopak nabádají k obezřetnosti.

Hlavní odlišnosti NFC řešení založených na SE nebo HCE:

- Zprovoznění služby – řešení SE NFC vyžaduje náhradu SIM a vlastní distribuce SW do mobilního telefonu pak obvykle napojení na TSM a závislost na MNO. Instalace SW pro HCE NFC řešení naopak probíhá jako u kterékoliv jiné aplikace na dané platformě.
- Použitelnost – SE NFC je koncipováno tak, aby fungovalo všude jako běžné bezkontaktní karty. HCE NFC může narazit na omezenou platnost některých transakčních dat (např. jednorázových klíčů), neboť vydavatel může být nedostupný nebo je latence online spojení pro potřeby provedení transakce příliš vysoká. Přepracována musí být rovněž implementace offline verifikace Passcode/mPIN, neboť uložení ve standardní paměti telefonu není



Obr. 2

bezpečné. Trhy, kde se běžně používá online PIN, se mohou vrátit k této metodě. HCE rovněž vyžaduje zapnutý mobilní telefon, odpadá tedy možnost platby s vybitou baterií.

- Bezpečnost – SE NFC nabízí vysokou úroveň ochrany aplikace a potřebných klíčů. HCE NFC je obvykle spojováno s cloudovými technologiemi. Citlivé informace jsou pak uloženy v zabezpečených serverech datového centra, aplikace v telefonu obsahuje pouze informace, které umožní telefonu získat data s omezenou platností potřebná k provedení transakce. Pokud má být takové řešení bezpečné, musí být založeno na vícevrstvé bezpečnosti zahrnující jednorázová data, white-box kryptografii, real-time analýzu transakcí a kontrolu otisku mobilního zařízení. Číslo karty (PAN – Primary Account Number) je rovněž citlivou informací, proto by mělo být řešení integrováno s tokenizačními službami, kde je PAN nahrazen dočasnou hodnotou (token) s omezeným využitím. Zmíněná opatření nesnižují rizika odcizení pověření nebo citlivých dat, ale omezují jejich platnost, a tedy i možnost zneužití, na základě pravidel vydavatele, např. na jednu transakci, po omezenou dobu, do určité sumy, pro dané zařízení, obchodníka nebo platební kanál.
- Zralost technologie a standardů – koncept SE NFC je definován již řadu let. I přes problémy s rozšiřováním je dobře popsán a má již za sebou období dětských nemocí. HCE je zcela čerstvé a pilotní implementace teprve ukazují, které oblasti je třeba dále řešit.

Pokud ani HCE nepomůže k popularizaci NFC, může se stát, že emulace karet bude jedinou využívanou aplikací NFC, neboť v ostatních předpokládaných oblastech využití této technologie se začíná prosazovat také BlueTooth Low Energy (BLE).

2 Telefony s aplikací MasterCard Mobile

MasterCard Mobile představuje zcela odlišný pohled na použití mobilního telefonu při placení. Nejde o snahu učinit z telefonu platební kartu, ale umožnit z telefonu bezpečný přístup ke svým kartám při placení na Internetu nebo přímo z aplikací v telefonu. Hlavní myšlenka tedy spočívá ve vytvoření virtuální peněženky u poskytovatele služby (Service Provider – SP) a v mobilní aplikaci, která bezpečně přistupuje k této peněžence. Samotný obchodník nemá přístup k citlivým údajům karty. V České republice je poskytovatelem této služby společnost Wincor Nixdorf, její řešení nese název WN Wallet.

2.1 Potřebné vybavení a instalace

Na rozdíl od předešlého konceptu postačí libovolný moderní telefon nebo tablet s OS Android či iOS. Aplikace MasterCard Mobile (MCM) je volně dostupná v obchodech s aplikacemi dané platformy a nevyžaduje žádný SE. Při instalaci aplikace je totiž bezpečné úložiště karet vytvořeno v datových centrech SP. Aplikace má jednoznačný identifikátor náležící konkrétní instanci na konkrétním zařízení, tento identifikátor je spojen s virtuální peněženkou u poskytovatele služby. Aktivace aplikace probíhá vložením kódu zasílaného formou SMS na zvolené telefonní číslo. Uživatel si před dokončením aktivace zvolí mPIN, který slouží k přístupu do aplikace a potvrzení platby vybranou kartou. Aktivační proces končí výměnou komunikačních klíčů mezi mobilním telefonem a serverem SP. Opakovaná instalace aplikace na mobilním zařízení vede k vytvoření nového identifikátoru aplikace a nové peněženky u SP.

Kartu vkládá do peněženky přímo držitel bez jakékoliv potřeby žádosti v bance. V systému SP je uloženo číslo karty, její platnost a hodnota CVC2/CVV2. V mobilní aplikaci je pak dostupné pouze maskované číslo karty, zvolená barva a zvolený název karty pro její snadnou identifikaci v telefonu. Při vkládání karty do systému (karty MasterCard, Maestro, Visa i Visa Electron) může proběhnout autentizace držitele prostřednictvím 3-D Secure, v takovém případě je ihned k dispozici výsledek ověření a karta se stává okamžitě aktivní. Pokud karta není zapojena do programu 3-D Secure, je provedena aktivace až na základě vložení kódu, který držitel získá z bankovního výpisu nebo elektronického bankovnictví.

2.2 Placení telefonem na Internetu

V současnosti je placení telefonem s aplikací MasterCard Mobile dostupné zejména při nákupech na Internetu. Po přesměrování na platební bránu je místo vložení údajů karty možné kliknout na logo MasterCard Mobile. V prohlížeči je následně zobrazen QR kód představující identifikátor platby ve WN Wallet. Ten

je přepsán do mobilní aplikace nebo pohodlněji naskenován z aplikace fotoaparátem telefonu. Transakce je u SP spárována s peněženkou držitele, který následně zvolí kartu, která má být použita. Pokud si držitel nenastavil platby nižších částek bez mPINu, proběhne jeho ověření. Po verifikaci informací odeslaných telefonem ve WN Wallet je poskytovatelem služby bezpečně předána informace o zvolené kartě platební bráně, kde je dokončena eCommerce transakce.

2.2.1 Integrace do eCommerce systémů

Aby bylo patrné zapojení nové služby do Internetových plateb, pokusme se stručně popsat, jak může vypadat běžná Internetová transakce:

1. Držitel u obchodníka vloží zboží do virtuálního nákupního košíku a přistoupí k platbě.
2. Po stisknutí tlačítka zaplatit je zákazník přesměrován na platební bránu, webový prohlížeč vytvoří s platební bránou bezpečný kanál (SSL/TLS).
3. Platební brána zaeviduje transakci do systému, provede základní kontroly a vrátí částečně vyplněný HTML formulář zákazníkovi, aby mohl doplnit údaje o platební kartě.
4. Držitel odešle formulář s vyplněným číslem a platností karty, včetně kódu CVC2/CVV2 ze zadní strany karty.
5. Brána pošle obchodníkovi bezpečným kanálem k revizi informace o transakci. Tento krok slouží jako potvrzení, že nedošlo k modifikaci dat transakce.
6. Obchodník potvrdí platební bráně správnost dat.
7. Pokud brána podporuje program 3-D Secure, je proveden pokus o autentizaci držitele:
   a) Brána zjistí dotazem do adresářové služby platební asociace, zda je karta zapojena do programu 3-D Secure.
   b) Pokud ano, je držitel přesměrován na ACS (Access Control Server) vydavatelské banky.
   c) ACS obvykle vygeneruje jednorázový kód, který je zaslán držiteli na předem domluvený mobilní telefon.
   d) Držitel přepíše získaný kód do formuláře, ACS ověří kód a s podepsaným výsledkem přesměruje držitele zpět na platební bránu.
   e) Brána provede verifikaci výsledku ověření a na základě výsledku autentizace se rozhodne o pokračování v transakci.

8. Brána pošle transakci k autorizaci do vydavatelské banky. Pokud je platební brána provozována bankou, která zároveň vydala danou kartu, je požadavek odbaven v rámci banky. V opačném případě jsou ke komunikaci s vydavatelem využity sítě příslušné asociace.

9. Na základě odpovědi z autorizačního centra vydavatele platební brána transakci schválí nebo zamítne.

10. Obchodník i držitel jsou informování o provedení a výsledku transakce.

11. Webový prohlížeč je přesměrován zpět na stránky obchodníka.

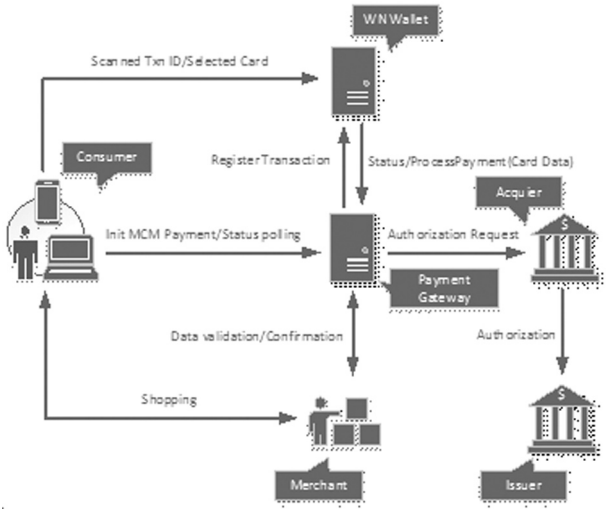
Po propojení platební brány s poskytovatelem služby MCM se některé kroky mohou změnit. Pokud místo vložení informací o kartě do platebního formuláře klikne zákazník na ikonku MasterCard Mobile, pak:

- Platební brána provede prostřednictvím WS (Web Service) registraci transakce u poskytovatele služby MCM. Na straně poskytovatele je této transakci přiděleno unikátní ID. Toto ID je vráceno zpět platební bráně a následně zobrazeno v prohlížeči jako číslo a QR kód. Prohlížeč začne od této chvíle periodicky zjišťovat stav transakce v platební bráně, neboť se další kroky odehrávají výhradně na mobilním telefonu.

- Držitel nastartuje mobilní aplikaci MCM, zadá mPIN a přepíše číslo transakce nebo naskenuje zobrazený QR kód. Telefon odešle toto číslo poskytovateli služby MCM, kde dojde ke spárování peněženky se zaregistrovanou transakcí. Poskytovatel pošle do platební brány informaci, že transakce byla naskenována. V závislosti na implementaci může platební brána od této chvíle blokovat jiné metody platby nebo naopak stále umožnit navrácení ke klasické platbě kartou.

- Držitel zvolí na mobilním telefonu kartu, kterou chce zaplatit. Současně může být znovu dotázán na mPIN. Mobilní telefon pošle poskytovateli služby identifikátor zvolené karty. Systém provede verifikaci požadavku z mobilního zařízení a poté zašle do platební brány požadavek na vykonání transakce s detailními informacemi o zvolené kartě (číslo karty, platnost, CVC2/CVV2). Platební brána změní stav transakce a zajistí autorizaci u vydavatele karty. Webový prohlížeč periodickými dotazy zjistí, že transakce je již vybavována a zobrazí pokyn k vyčkání na výsledek. Po obdržení autorizační odpovědi je SP informován o výsledku transakce.

Z pohledu držitele tedy odpadá nutnost znát údaje své karty, postačí aplikace na telefonu a znalost mPIN. Brána již neprovádí ověření držitele (3-D Secure), neboť bylo provedeno poskytovatelem služby MCM (mPIN).



Obr. 3

2.3 Přímé platby z telefonů

Výše popsaný postup odpovídal situaci, kdy nakupujeme např. na PC a následně platíme telefonem. Mnoho obchodníků má ale aplikace přímo pro mobilní telefony nebo tablety. Proto služba počítá také s přímým placením z těchto aplikací bez nutnosti klikat na stránkách platební brány v mobilním webovém prohlížeči. Z aplikace obchodníka je provedena registrace platby u poskytovatele a poté automaticky nastartována aplikace MCM s předaným identifikátorem transakce. Po potvrzení transakce a verifikaci držitele je odeslán požadavek k poskytovateli služby MCM na provedení autorizace transakce ve vydavatelské bance. Po provedení platby je řízení předáno zpět do aplikace obchodníka, která pošle poskytovateli služby MCM dotaz na zjištění stavu platby.

Vlastní vytvoření požadavků registrace platby a zjištění výsledku platby mobilní aplikací obchodníka může být realizováno třemi způsoby:

- Scénář „backend“ to „backend“ představuje případ, kdy spolu komunikují servery zúčastněných stran, tj. server obchodníka a server poskytovatele služby MCM. Telefon komunikuje se serverem obchodníka, který následně generuje a podepisuje požadavky posílané poskytovateli služby.
- Server obchodníka pro generování podpisu. V tomto případě telefon vždy žádá vyhrazený server na straně obchodníka o podepsání připravených požadavků. Následnou komunikaci s poskytovatelem služby zajišťuje telefon. I v tomto případě je klíč bezpečně uložen na serveru obchodníka.

- Poslední scénář přímé komunikace z mobilní aplikace vyžaduje, aby byl v mobilním telefonu bezpečně uložen privátní klíč obchodníka pro podepisování požadavků. Tento způsob je z hlediska bezpečnosti nejnáročnější.

2.4 Zabezpečení

Komunikace mobilní aplikace MCM se servery poskytovatele služby MCM je chráněna protokolem SSL/TLS zajišťujícím privátnost a integritu přenášených dat. Ověření každého požadavku mobilní aplikace na straně serveru zahrnuje ověření podpisu požadavku a otisku mPIN. Do podpisu požadavku (Message Authentication Code – MAC) vstupuje vždy identifikátor aplikace, otisk mPIN, symetrický klíč a vlastní data požadavku. Symetrický klíč je s každým požadavkem měněn. Mobilní zařízení ani servery poskytovatele si neukládají hodnotu mPIN a pracují výhradně s otiskem tohoto kódu.

Výměna dat mezi poskytovatelem MCM a platební bránou probíhá prostřednictvím Web Services. Klíčové služby jsou zabezpečeny standardem WS-Security, citlivé datové elementy XML zpráv (číslo karty, platnost, hodnota CVC2/CVV2, telefonní číslo) jsou šifrovány a zprávy jsou digitálně podepsány. Jako security token slouží X.509 certifikáty. Kanál je navíc zabezpečen protokolem SSL/TLS.

Při třech neúspěšných pokusech o verifikaci mPIN je MCM aplikace zablokována na 24 hodin. Pokud se po uplynutí této doby opět nepovede třikrát zadání mPIN, dojde k nenávratné blokaci. Uživatel musí v takovém případě aplikaci odinstalovat a následně provést novou instalaci s aktivací včetně registrace všech karet.

2.5 Další využití

Poskytovatelem služby MasterCard Mobile je v České republice společnost Wincor Nixdorf. Za omezení lze zatím považovat fakt, že systém pokrývá pouze obchodníky u nás a na Slovensku. Pokud tedy držitel hodlá platit touto službou v jiné zemi, musí si nainstalovat aplikaci MCM dané země a zaregistrovat karty do peněženky v systémech tamního poskytovatele. Do budoucna by bylo dobré, aby jednotlivé implementace byly integrovány skrze služby asociace MasterCard.

Jak již bylo zmíněno, kromě placení v Internetových obchodech je služba připravena i na přímé platby z aplikací mobilního telefonu. Koncept lze využít i k dobíjení kreditu, pro dárcovské platby, posílání peněz z jedné karty na druhou na základě programu MasterCard MoneySend apod. Platební systém by se mohl rozšířit i na jiná místa, nelze vyloučit např. aplikaci na platebním terminálu. Skenování QR kódu je možné i z displeje platebního terminálu nebo nově se objevujících Smart POS mobilních aplikací. Aplikace může také nově začít využívat technologii NFC, kde by načtení přiděleného čísla MCM transakce proběhlo bezkontaktně pouhým přiložením telefonu k terminálu.

Bezkontaktní platební karty – radost nebo starost?

Jan Okrouhlý

E-mail: jan.okrouhly@hp.com

Bojíte se nových metod placení nebo zaplatíte bez mrknutí oka?

Kdysi technologicky stačil při platbě kreditní záznam vytepaný v plíšku, viz obr. 1. Dnes při bezhotovostní platbě řešíme, zda budeme v obchodě platit mobitem, nálepkou, telefonem, přes tablet nebo zda zůstat u plastové karty.

Obr. 1: Ukázka kovového cestovního šeku/úvěrové známky (r. 1870)

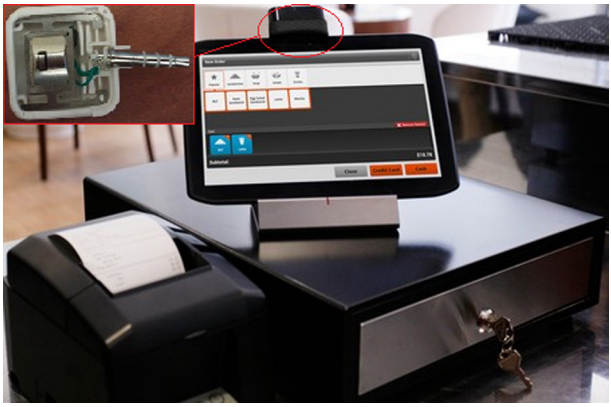
Udělejme si nejprve představu o tom, jak probíhá komunikace mezi kartou a terminálem obchodníka při platbě jednotlivými technologiemi.

Kontaktní rozhraní

Pro srovnání s bezkontaktním režimem se v této kapitole budeme věnovat primárně čipovému rozhraní, označováno též EMV dle zakládajících asociací Europay, MasterCard a VISA. Patří sem ale i z bezpečnostního hlediska značně problematické rozhraní magnetického proužku, který se pro plastové karty bohužel dosud využívá a to již přes čtyřicet let. V realitě zjistíme, že není tak

jednoduché bez něj fungovat, resp. karty vydávat, protože je ještě mnoho míst, kde jsou stále využívány:

1. různé starší parkovací automaty (např. placené parkoviště pod Národním divadlem), či různé zahraniční mýtné brány;
2. používají je stále i obchodníci, převážně v USA a dokonce je v tom podporuje i třeba produkt Square, viz obr. 2, popř. http://www.squareup.com;
3. částečně bankomaty pro „kontrolu“ vložené karty a dokonce pasivně některé banky k ověření přístupu k bankomatu mimo úřední hodiny dané pobočky;
4. u obsluhovaných terminálů je stále tato technologie akceptována jako záložní pro v případ nečitelnosti čipu.

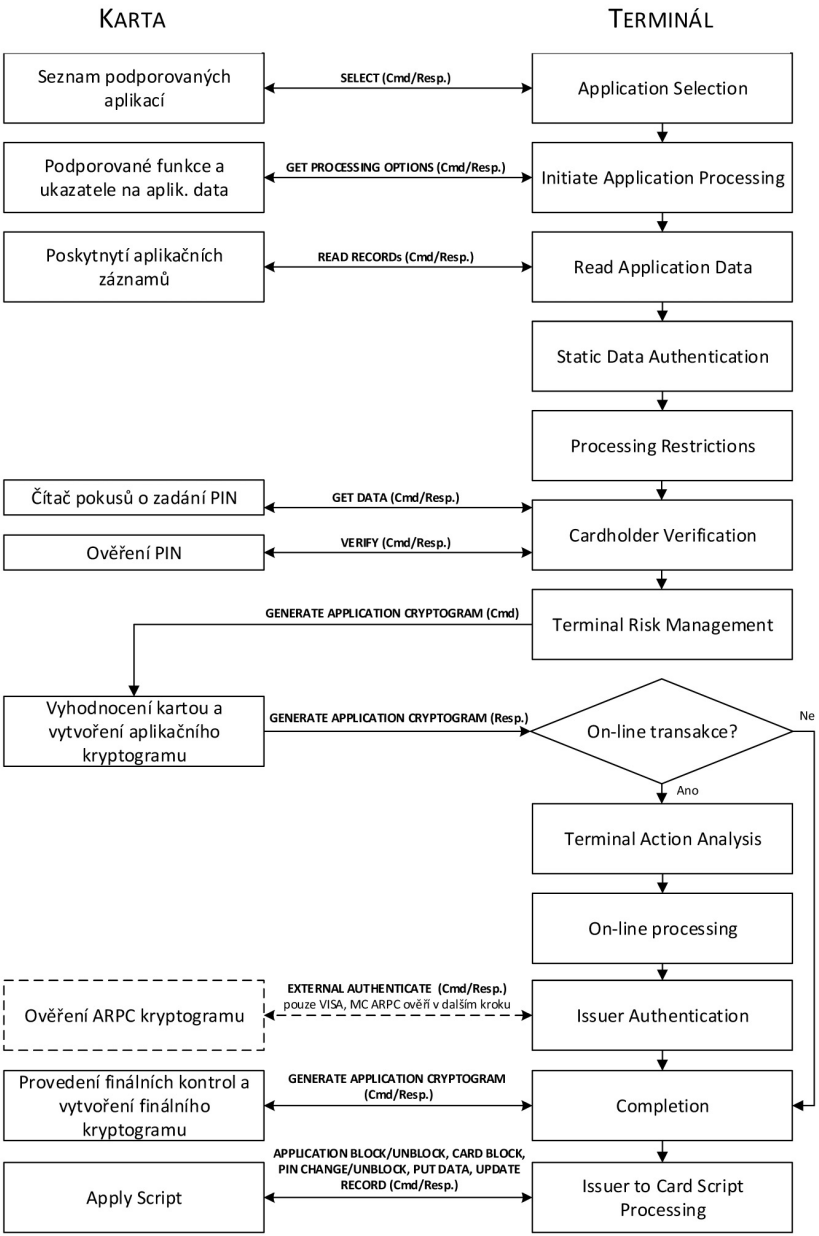


Obr. 2: Ukázka „iPad POS“ a konstrukce Square

Důsledkem je, že klienti bank dnes mají běžně i několik platebních karet a to jak bezkontaktní kartu, např. v telefonu, pro běžné placení, tak ještě minimálně jednu EMV kartu pro potřeby cestování či vykrytí jiných míst, kde s telefonem či bezkontaktní nálepkou dosud nezaplatí.

Ale zpět k čipovému rozhraní, jehož vývoj započal jen o jedno desetiletí později a jež finálně přineslo v podobě tzv. Smart karty jádro nezbytné k úspěchu:

- velkou kapacitu paměti;
- obtížnost výroby kopie;
- interaktivní rozhodování;
- nové metody ověření držitele karty;
- možnosti aktualizace obsahu;
- provozovatelnost více aplikací atp.



Obr. 3: Možné flow EMV transakce

Pro více detailů zde odkážeme např. na „Tutoriál – Co obnáší současný standard platebních karet“, který je k dispozici ve sborníku 30. konference Europen konané na ranči Malevil. Pro účel srovnání s průběhem bezkontaktních transakcí je zde postačující zjednodušené obecné flow kontaktní EMV transakce.

Z průběhu je patrný pozvolný stupňovitý průběh transakce. Na počátku může držitel manuálně vybrat i z více platebních aplikací na kartě, poté jsou načteny nezbytné podpisem chráněné informace a doplňkové informace o kartě. Následuje vyhodnocení terminálem a určení samotné potřeby a způsobu ověření držitele karty, poté dochází k risk managementu na kartě a teprve pak případné zpracování offline, pokus o zpracování online či opětovný pokus o schválení offline nebo online odpovědi. V realitě zde připadá v úvahu široká paleta variant nastavení konkrétní karty, konkrétního terminálu a placené částky, které vedou na možnost schválení offline, nezbytnost schválení online či na zamítnutí transakce. Při schvalování, zda transakce bude akceptována offline má velký význam stav interních čítačů karty a samozřejmě typ a hodnota prováděné transakce. Karty typicky obsahují jak čítač na počet realizovaných transakcí offline (v případě, že není měna kartě známa) tak i čítač na sumu zaplacenou offline v měně karty (v některých případech se uplatňuje dokonce orientační přepočet částky dle přibližného kurzu uloženého na kartě). Pokud některý z nich přesáhne kartou stanovený dolní limit, bude se požadovat autorizace online, která v kladném případě skončí resetem obou čítačů. Pokud není možné online zpracování, uplatní se test nepřesáhnutí horního limitu, kdy karta ještě jaksi nouzově schválí transakci offline. Nastavení těchto limitů záleží na vydavateli karty – pro lepší představu můžeme uvažovat například nastavení dolního limitu počtu transakcí hodnota 3, horní 5 a limity nasčítaných částek např. na 1 000 Kč a horní limit na 2 000 Kč. Rozhodující slovo má samozřejmě terminál, který může náhodně vybrat transakci ke zpracování online. V realitě se konfiguračně nastavují pravděpodobnostní křivky tohoto velocity checkingu pro náhodné vyžádání zpracování online i pro malé částky transakce nezávisle na stavu karty.

Teoreticky (asociace k tomuto kroku nebyly motivovány a ani terminály, resp. banky na to nejsou připraveny) lze příhodnou konfigurací tabulky dostupných metod ověření držitele akceptaci platby do jistého finančního limitu i bez potřeby zadávat PIN. Z hlediska celkového času transakce by se značně přiblížily bezkontaktním platbám nízkých částek, kdy se běžně nemusí držitel ověřovat zadáním PIN. Časová režie je pouze v zasunutí karty do čtečky, popř. nezbytnosti předání karty obsluze terminálu. Z hlediska uživatele je ale také pozorovatelnou nevýhodou EMV zastaralé časování komunikace – relativně pomalé výměny větších objemů dat. Standardní přenosová rychlost dat z karty 9.6 kb/s (max. 115 kb/s – záleží na výrobci čipu). Při bezkontaktní komunikaci je to běžně až 423 kb/s (max. 848 kb/s). Svou roli zde výjimečně hrají i problémy čtení způsobené opotřebením či znečištěním kontaktů nebo dokonce statická elektřina.

Bezkontaktní rozhraní

MasterCard

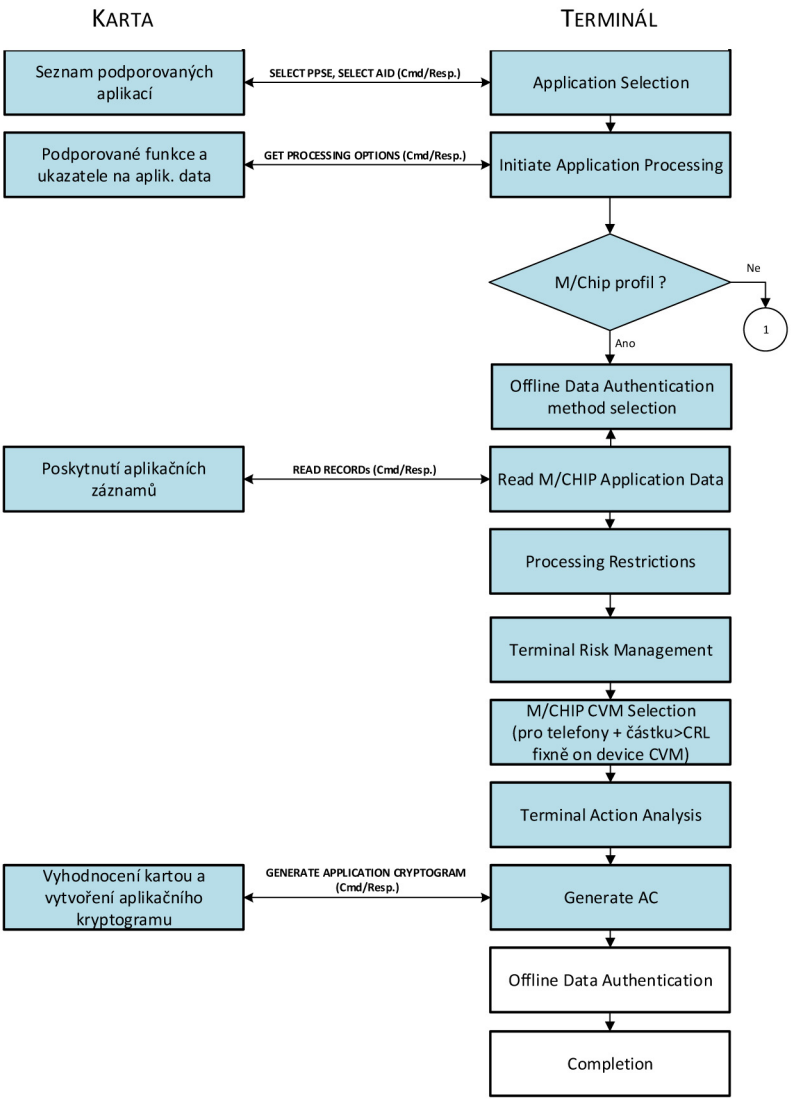
MasterCard od roku 2004 staví bezkontaktní platby na stávající síťové infrastruktuře pro výměnu transakcí, co se týká komunikačních zpráv, pravidel a politik. To vydavatelům karet zjednodušilo uvedení bezkontaktních plateb do života v USA během dvou let. Zavedl proto hned dvě bezkontaktní technologie – jednu MasterCard PayPass M/Chip pro trhy, kde se používá čipová technologie a druhou MasterCard PayPass Mag Stripe pro trhy, kde čipová technologie zavedena nebyla. Součástí designu je řada bezpečnostních opatření pro snížení rizikovosti bezkontaktních transakcí jakými je odposlech či krádež dat z karty a následné falešné magstripe karty či zneužití dat při e-komerce transakcích.

Pro srovnání s EMV nejprve uvedeme typický průběh bezkontaktního flow MasterCard PayPass M/Chip transakce (modře jsou podbarveny části, kdy je přiložena karta ke čtečce) – viz obr. 4.

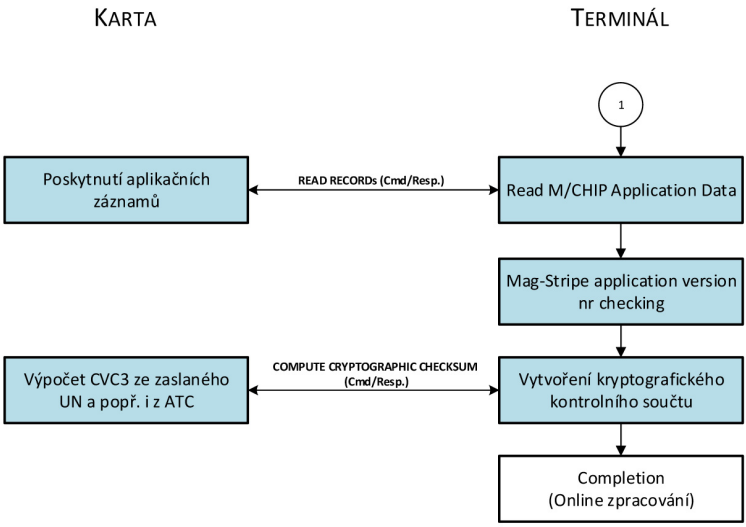
Z obr. 4 je patrno, že průběh PayPass M/Chip transakce je velmi podobný EMV, ale obsahuje maximálně jedno generování kryptogramu – není zde možné čekat na realizaci online komunikace a tudíž ani možnost aplikace resetování offline čítačů či jiné změny v datech, např. Issuer skripty (příkazy zaslané na kartu digitálně podepsané od jejího vydavatele). Drobnou slabinou tohoto flow je i přesto nemalé množství zpráv mezi kartou a terminálem, které logicky prodlužuje dobu nezbytnou pro jejich výměnu a tedy celkovou nezbytnou dobu přiložení karty ke snímači. Ta je sice shora omezena a nesmí přesáhnout **600 ms**, to ale může způsobovat problémy jak vývojářům terminálových aplikací, tak držitelům karty při nedostatečně dlouhém přikládání karty a není proto zcela jednoduché nasadit jej např. pro průchod turnikety. Tyto nedostatky se snaží Mastercard řešit v aktuální verzi PayPass zavedením podpory obnovy nedokončené transakce – Torn Transaction recovery. Její implementace je ale v terminálech nepovinná, takže se spíše spoléhá na aktivní NFC komunikaci mobilních zařízení, či prostě na již získaný „grif“ uživatelů karet...

Z pohledu bezpečnosti přenáší PayPass M/Chip do čtečky vzduchem mimo jiné v otevřené formě pouze tyto dva citlivé údaje: číslo karty (PAN) a její platnost (Expiraci). Proti potenciálnímu zneužití těchto údajů je MasterCard vyzbrojen použitím tzn. PPAN, což je zavedení odlišného čísla karty pro bezkontaktní komunikaci a jeho pozdější mapování na klasický PAN až v síti asociace. Použití služby maskování PAN však dosud také není povinné...

Téměř až do roku 2014 se v USA považovaly informace o kreditních kartách za dostatečně zabezpečené a mělo se za to, že čipová technologie nebude potřeba. Bezkontaktně tam tak i našinec mohl zaplatit kartou pouze v mag stripe režimu, viz pokračování flow MasterCard PayPass MagStripe transakce na obr. 5.



Obr. 4: Příklad flow PayPass M/Chip transakce



Obr. 5: Příklad pokračování flow PayPass transakce v režimu MagStripe

Komunikace s kartou je v tomto případě již mnohem kratší – celá komunikace musí proběhnout v časovém limitu **300 ms**. Zato následná autorizace transakce musí z bezpečnostních důvodů probíhat online, což přináší z hlediska klienta rychlost odbavení srovnatelnou s protažením magnetické karty čtečkou typicky následované ověřením držitele pomocí PIN či podpisem.[1]

Teprve od letošního roku v USA MasterCard zavedl podporu EMV transakcí – na základě společné výzvy asociací Visa, MasterCard, AmEx a Discover jsou do října 2015 všichni tamní vydavatelé karet povinováni přejít na čipovou technologii (bezkontakt nevyjímaje). Kdo z obchodníků či bank nepřejde na EMV, bude zodpovědný za podvodné transakce v jejich prostředí, které budou způsobené zákazníkům vlastnícím EMV kartu.[2]

Stále tedy není možné zbavit se reálně rozhraní magnetického proužku ani není možné vydávat karty bez MagStripe profilů, ale plány platebního systému na odstranění magnetického pásku jsou o krok blíže k jejich naplnění v blízké budoucnosti.

[1] Tímto způsobem bude pravděpodobně Vaše karta obsahující oba profily (M/Chip i Magstripe) komunikovat v USA a naopak držitel PayPass Magstripe only karty z USA v Evropě dle nařízení asociací bezkontaktně nebude moci platit a bude tak nucen použít magnetický proužek. Fakt, že v USA byla implementace kontaktní čipové technologie „přeskočena“ a rovnou byla zavedena technologie PayPass Magstripe s minimem nezbytných změn v mezi bankovních komunikačních systémech, tak do jisté míry ovlivnil bezpečnost karet mezinárodně.

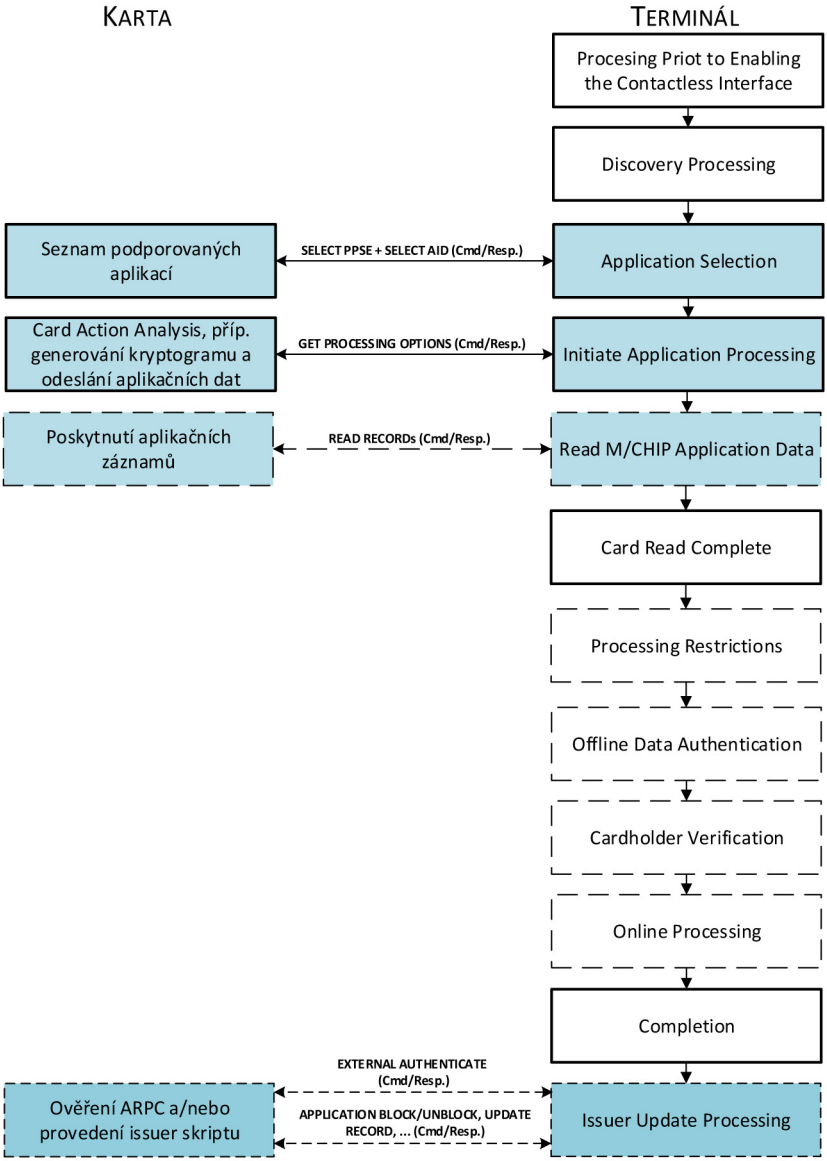
[2] Oproti většině světa bude k našemu překvapení převážná část EMV karet nově vydávaných v USA s podporou ověření držitele podpisem, tzn. bez ověření držitele prostřednictvím PIN, tak jak jsme na něj zvyklí. Karty s PIN tam ale mají, resp. musí být akceptovány bez problému.

VISA

Obdobně jako MasterCard zavedla i VISA ze stejných důvodů v roce 2007 dvě bezkontaktní technologie souhrnně označované jako PayWave nebo též VISA Contactless – pro trhy s čipovou technologií je určena qVSDC (quick Visa Smart Debit/Credit) a pro ostatní MSD (Magnetic Stripe Data). S cílem zkrátit čas interakce mezi kartou a terminálem radikálně změnila průběh bezkontaktní platby vůči tradičnímu EMV flow a vhodným uspořádáním dat dosáhla snížení počtu EMV related příkazů potřebných k realizaci transakce. Při qVSDC se maximálně snaží zkrátit čas transakce tak, že provádí terminal risk management před použitím karty a následný risk management již během jejího počátku. Pro online transakce postačuje po Selectu provedení Get Processing options, který již vrací potřebný kryptogram. Pro offline transakce následuje navíc čtení záznamů na kartě příkazy Read Record. Čtečka ověří dynamický podpis, až když už je komunikace s kartou dokončena. Specifikem VISA je zde použití metody fDDA (fast Dynamic Data Authentication), která s menším počtem údajů o transakci, což přináší mírné urychlení, ale na úkor bezpečnosti. Detaily k DDA, viz „Trendy v oblasti čipových platebních karet“ sborník 36. konference Europen.CZ v Maxičkách.

Pro maximální urychlení plateb zaváděla VISA v počátcích akceptace bezkontaktních karet tzv. Hard limit režim akceptace karet, kdy bylo možné platit bezkontaktně jen do ekvivalentu 500 Kč a bez ověření držitele a tudíž převážně offline. Při překročení čítačů na kartě (pro bezkontaktní platby má vydavatelská banka na kartách k dispozici sadu čítačů včetně možnosti čítání hodnot používaných též pro kontaktní limity) nebo vyšší částce k zaplacení bylo nezbytné zaplatit jinak než kontaktně. Později i v souvislosti s nástupem placení pomocí telefonů či jinými formami karty, kterými nelze kontaktním způsobem platit, došlo k nezbytnému přechodu na tzv. Soft limit režim (proklamovaný MasterCardem již od samého počátku), který umožňuje platit bezkontaktně prakticky libovolnou částku, ale nad stanovený limit je transakce zpracována výhradně s ověřením držitele a probíhá samozřejmě online.

Resetování offline čítačů je tak na klasických kartách víceméně ponecháno na náhodném kontaktním použití karty v místech, kde není bezkontaktní technologie podporována (typicky bankomaty), popř. karta může po přiložení nově informovat čtečku o tom, že transakci je nutné provést kontaktně anebo naopak informovat o tom, že není schopna provádět kontaktní transakce. Resetování čítačů je v tomto případě prakticky nemožné - na mobilních zařízeních prostřednictvím datových zpráv se zatím z finančních důvodů/roamingu neimplementovalo. Teoreticky lze nově provést aktualizaci karty vyžádaným druhým přiložením (viz nástin Issuer update ve spodní části výše uvedeného obrázku), v praxi se ale tento mechanismus zatím neimplementoval a v praxi počítá se s dostatečnou rychlostí a spolehlivostí online odbavení.



Obr. 6: Příklad flow PayWave transakce v režimu qVSDC

Aby bylo možné platit i v USA, obsahují karty pro mezinárodní použití i podporu MSD režimu. Flow je prakticky stejné jako u qVSDC výše.

VISA MSD obdobně jako PayPass Mag Stripe poskytuje přes bezkontaktní rozhraní prostřednictvím EMV příkazů obsahový ekvivalent dat z magnetické pásky. Pod VISA MSD se ve skutečnosti skrývají ještě dvě varianty realizace transakce, na níž se terminál s kartou domluví. Starším způsobem je zabezpečení realizováno pomocí dCVV (Dynamic Card Verification Value) a novějším je generování kryptogramu verze 17. Jelikož se jedná nutně o online transakci, ověření karty provádí až autorizační centrum a terminál v tomto režimu neprovádí kompletní načtení záznamů na kartě a tudíž ani offline data autentizaci. Ověření držitele záleží opět na parametrech terminálu. Mapování PAN tak jako např. MasterCard asociace nenabízí.

V rámci globální interoperability se vyžaduje, aby karty podporovaly oba režimy (qVSDC i MSD) a čtečky alespoň jeden z těchto režimů. Pokud čtečka podporuje oba, pak při komunikaci s kartou preferuje qVSDC (ale umožní i MSD pro případ domestic karet – netýká se Evropy).

Sumarizace

Rámcově lze různé formy bezkontaktních karet rozdělit takto:

Provedení karty	**Možná ověření držitele**	**Pozn.**
Plastová duální	Podpis No CVM Online PIN Offline PIN	Většinou i hybridní s magnetickým proužkem.
Nálepka nebo čip formátu SIM (např. v náramku hodinek)	No CVM Online PIN	Využívá se mag stripe režim (v Evropě povolen pouze PayPass MagStripe), kde technicky ani není možné čítače resetovat a využívá se jednoduššího (teoreticky levnějšího) čipu. Existuje ale implementace s plnohodnotným čipovým řešením, nicméně akceptace je možná pouze online – nelze vložit do čtečky.
Na SIM v mobilním zařízení	No CVM On Device CVM Online PIN	Dle konfigurace může také pracovat při vypnutém telefonu jako pasivní nálepka s Online PIN. Akceptace je možná pouze online – nelze vložit do čtečky.

Požadavky na konfigurace terminálů mohou být geograficky odlišné, tabulka uvádí příklad různých podporovaných metod ověření držitele závislých na základě typu použití terminálu:

Typ terminálu	**Typ ověření držitele**	**Povinný**
Obsluhované	Podpis No CVM Online PIN Offline PIN	Ano Ano Volitelně Volitelně
Neobsluhované Type C/CAT1 (Automated Delivery Machines/Fuel Dispensers)	Online PIN Offline PIN	Ano Volitelně
Neobsluhované Type B/CAT2 (Parkovné/Mýtné)	No CVM	Ano
Neobsluhované Type A/CAT3 (offline only mikroplatby/Kiosky)	No CMV Offline PIN	Ano Volitelně

Pozn. V realitě některé automaty v závislosti na placené částce mohou v rámci autorizace dané transakce změnit příhodně svůj typ.

Přehled zranitelností

- Tipovací útok – náhodná predikce dat, která by karta odeslala terminálu.
- Pasivní útok – odposlech dat např. přídavnou anténou u terminálu nebo např. uschovanou čtečko u útočníka při placení (se spec. vybavením lze odposlechnout data až. do 1 m) – zneužití se označuje Replay Attack.[3]
- Aktivní útok – použití falešného (opět lze dosáhnout realizace transakce na větší vzdálenost) či skutečného terminálu k vyvolání falešné transakce např. v přeplněném prostředku hromadné dopravy. Zachycená data jsou pak zneužita pro falešnou magstripe, čipovou nebo e-commerce transakci – označuje se jako Preplay Attack. Reálným příkladem je NFC Proxy aplikace (viz dále) v Proxy režimu, kdy simuluje zvolený terminál a uloží zaznamenanou komunikaci za účelem pozdějšího přehrání zachycených tagů.

[3]Příklad útoku od Kristin Paget prezentovaný na Shmoocon ve Washingtonu D. C., kdy stažená bezkontaktní data nahraje na magnetickou pásku (a také opačný útok, kdy se data z magstripe prezentují falešnou kartou jako bezkontaktní transakce), je čistě teoretický. Asociace od samého počátku vyžadují při autorizacích transakcí podporu datového elementu DE22, který mj. signalizuje, jakým rozhraním byla data z karty načtena. Korektně implementovaný terminál a kontroly v bance tuto transakci neumožní.

- Real-time útok – s využitím komplice jsou v reálném čase přenášena data a příkazy mezi skutečnou kartou a podvrženou kartou, se kterou komplic platí. S nástupem NFC a je bohužel tento útok s podvržením karty přes telefon reálný. Jeho snadnou realizaci předvedl Eddie Lee na Defcon hacker konference v Las Vegas prostřednictvím NFC Proxy aplikace v relay režimu (vyžaduje speciální verzi OS obsahující emulaci čtečky karet u komplice).[4]

- Invazivní útok – získání dat z karty bez vědomí držitele. Zneužitelný obsah polí s informacemi o držiteli karty se již v transakci nesmí vyskytovat, ale existují karty, ze kterých lze bezkontaktním rozhraním tyto informace vyčíst z čipové/EMV části. Bohužel záleží na vydávající bance, zda správně nastaví ochranu přístupu k daným objektům. Ačkoli se neporučuje log posledních n transakcí na kartě, karty některých bank tyto záznamy provádějí a mají je bezkontaktně přístupné. Na druhou stranu formát logu definuje vydavatel karty a neobsahuje informaci, zda se transakce zdařila.[5]

- Denial of Service: Pokusy o realizaci transakce na skutečném či falešném terminálu mohou vést ke znemožnění platby postiženou kartou.

Ochrana držitelů karty

- Pojištění transakce do ekvivalentu 500 Kč bez ověření držitele (u nás zadání PIN). V případě odcizení a **prokazatelném zneužití**, dojde-li k němu bez zadání PIN, získá držitel částku zpět. Obecně zde platí zákon č. 284/2009 Sb. Ten odpovědnost klienta omezuje §116 do výše 150 Eur, pokud ke škodě došlo ještě před nahlášením ztráty. Asociace povinují banky krýt rizika neautorizovaných transakcí. Proto banky často ve vlastním zájmu implementují v autorizačním systému nemožnost provedení řady následných neautorizovaných transakcí, jejichž celková suma přesáhne 150 Eur...

- Větší nákupy nad 500 Kč (popř. ekvivalent) jsou zajištěny PIN (či Podpisem). Teoreticky může být ve výjimečných případech (např. vstupné na masové kulturní akce) limit provozovatelem (Acquirer) terminálu navýšen, ale v tom případě dle pravidel asociací riziko zneužití kryje Acquirer, jež tento limit upravil.

[4] Předvedení NFCProxy bylo poslední kapkou, jež vedla k posílení bezpečnosti bezkontaktní technologie v USA, a od roku 2012 tam asociace začaly připravovat přechod na kontaktní čipové karty.

[5] Existuje malware, který na pozadí telefonu přes NFC hledá okolní bezkontaktní karty za účelem získání jejich čísel a odeslání do sítě hackerů.

- Náhodná platba je velmi nepravděpodobná. Kartu je nezbytné přiložit na minimální vzdálenost.[6]

- Relay Attack – 100% ochrana je pouze vodivý obal (a obecně neponechání karty bez dozoru), ale pozor na jeho provedení či nesprávné/nedostatečné zasunutí částí karty, kterými prochází anténa. Správné použití vodivého pouzdra kolem **celé** karty útoky dokonale znemožňuje, nelze pak samozřejmě zaplatit pouhým přiložením peněženky...

Obrana závislá na vydavateli karet

- Přístup k datovým objektům – při personalizaci karet by měl vydavatel zablokovat bezkontaktní přístup k většině objektů na čipu, resp. povolit čtení jen těch, které jsou absolutně nezbytné. Jméno a příjmení držitele nesmí být k dispozici a nedoporučuje se ani ponechání přístupu do logu transakcí.

- Limity na kartě – urychlení akceptace a odlehčení autorizační sítě motivuje banky k personalizaci karet tak, aby je bylo možné používat i pro offline autorizaci (netýká se tedy nálepek). Karta v tomto případě může kumulovaně limitovat počet či sumární hodnotu transakcí, může také zvyšovat některé čítače jen dle situace (lze např. využít datum z terminálu, měnu anebo kontrolovat vybrané stavové bity transakce). Vhodné nastavení sad použitých čítačů, v ideálním případě sdílených s EMV částí, a jejich příhodné resetování či modifikace při kontaktních operacích je záležitostí vydavatelské banky.

- Autorizační limity – některé banky umožňují klientům nastavit denní limity karty. V autorizačním centru se tak např. po 5 použitích karty bezkontaktně dovolí pouze kontaktní zpracování, resp. další bezkontaktní autorizace banka zamítne. Podmínkou je však vynucení online autorizace a vytrácí se zde možnost maximálně rychlého odbavení při placení offline.

- Anti-Replay mechanismy – při každé (i kontaktní) transakci karta zvýší interní čítač transakcí (ATC) a jeho hodnota je při autorizaci kontrolována. Lze tak detekovat zneužívání karty a odhalit Replay nebo Preplay Attack. Vydavatelská banka by při detekci automaticky měla zablokovat příslušnou kartu.

- Blokace karty – po zablokování v autorizačním centru přichází nelehký úkol zablokovat reálnou kartu. To je zatím možné pouze přes rozhraní EMV

[6] Přítomnost dalších např. MIFARE karet v peněžence značně narušuje možnost realizace transakce.

naplánovaným zasláním příslušného Issuer skriptu při autorizaci platby. Pokud je karta použita bezkontaktně, lze využít pouze speciální autorizační zprávu obsluze terminálu. Tyto mechanismy bohužel značně závistí na dané autorizační síti a vlastní implementaci terminálů.

Závěr

Bezkontaktní karta je při platbách samotných bezpečnější než magstripe, ale vyjma komunikační rychlosti, resp. uživatelského komfortu, zatím nebyly překonány výhody kontaktního čipu. Ideálním řešením do budoucna se zatím jeví karta v mobilu, kde lze NFC deaktivovat. Otázka je jak dobře, resp. jak dlouho toto bude bezpečné, viz snahy o SW přístup aplikací k řadiči secure elementu nebo NFC a jeho používání i pro jiné účely než placení. Bez klasických karet se ještě nějakou dobu neobejdeme, a protože zde neexistuje žádná nezávislá instituce, jež by oficiálně testovala kvalitu zabezpečení karet, lze jen doporučit šťastnou volbu vydavatele karty, vhodné pouzdro a obezřetné používání.

Zajímavé odkazy

http://creditcardforum.com/blog/chip-and-pin-credit-cards-usa/

http://blogs.wsj.com/corporate-intelligence/2014/02/06/
october-2015-the-end-of-the-swipe-and-sign-credit-card/

http://sourceforge.net/p/nfcproxy/wiki/Home/

Zákon č. 284/2009 Sb., O platebním styku.

Zabezpečení pomocí 3D-Secure při platbě na Internetu

Martin Zich

E-mail: martin.zich@hp.com

Současný stav

V dnešní době jsou stále mnohé Internetové transakce prováděné českými platebními kartami zabezpečeny jen do určité míry. Držiteli karty stačí k úspěšnému dokončení platby pouze tři údaje a to číslo karty (PAN), datum expirace a CVV2/CVC2 kód, který je k dispozici na zadní straně každého z „plastů“. To s sebou přináší rizika ve 2 rovinách:

- Držitel karty nemá žádnou jistotu, že v případě ztráty nebo zcizení karty nedojde k jejímu zneužití na Internetu. Všechny potřebné údaje jsou totiž dostupné přímo na kartě samotné a to komukoliv, kdo ji zrovna drží v ruce nebo se mu podařilo nějakým způsobem údaje z ní uchovat.

- V případě mezibankovních vypořádání reklamovaných transakcí (tzv. „chargeback“), prohrává spor o jejich hrazení v mnoha případech právě vydavatel, jehož platební karty nejsou zabezpečeny např. pomocí 3D-Secure nebo pomocí jiné adekvátní služby. Náklady reklamovaných transakcí jdou pak na vrub tohoto finančního ústavu. Sledované trendy navíc ukazují, že počet podvodných transakcí rok od roku nebývale roste.

Jedním z řešení, která se v tomto směru nabízí, je zavedením mezinárodně uznávané služby 3D-Secure podporované hlavními hráči (asociacemi) na trhu bezhotovostních plateb. Těmi jsou samozřejmě VISA a MasterCard. Zavedení 3D-Secure na straně vydavatele karet spočívá v implementaci autentizační vrstvy před v současnosti provozované řešení. Držitelova identita je tak spolehlivě ověřena jedním z vybraných způsobů a při následném zadání karetních údajů tak existuje mnohem větší jistota, že platební brána komunikuje přímo s osobou, která je vlastníkem použité platební karty.

3D Secure Issuing vs. Acquiring

Některé banky tvrdí a propagují, že aktivací 3D Secure na jejich kartě zajistí nutnost ověření každé z transakcí prováděné na Internetu, ale pokud dojde k vlastní platbě na některých platebních bránách, k žádnému ověření přesto nedojde. Jak je to možné? Je totiž nutné rozlišovat a zároveň požadovat zavedení 3D-Secure pro tzv. Issuing i Acquiring. Pro plnou funkčnost musí být k dispozici obě části této služby. To však není ve většině případů běžnému zákazníkovi srozumitelně komunikováno už z toho důvodu, že Acquiring část nemá banka logicky pod svojí kontrolou.

3D-Secure Issuing je právě ono ověření identity klienta provádějícího bezhotovostní eCommerce transakci. Parametry a konkrétní údaje vázající se ke klientovi (držiteli karty) jsou definovány vždy ke každé konkrétní kartě přímo v bance a jejich ověření probíhá při dočasném přesměrování na elektronický formulář („Issuer window“) konkrétní banky („Issuera“) při provádění platby.

Oproti tomu 3D-Secure Acquiring definuje procedury, které zajistí, že provádění transakcí při nákupech na Internetu u konkrétního obchodníka probíhají standardizovaným a zabezpečeným způsobem. Obchodník při těchto transakcích nepředává bance konkrétní informace sám, poté co je klient vyplnil do jeho formuláře, ale využívá k jejich „sběru“ přímo konkrétní platební bránu banky nebo spíše některou z univerzálních platebních bran, které tyto údaje do banky posílají šifrovaným kanálem. Tyto brány tedy obhospodařují celou transakci (posbírání údajů o kartě – PAN, expirace, CVV2/CVC2) a směrem k obchodníkovi zašlou pouze informací o výsledku jejich ověření. Do banky jsou zasílány všechny informace ve standardizovaném formátu (typicky pole v XML zprávě určeného formátu. Zprávy mají definovaná právě i pole, kde je příslušná asociace dotazována, zda daná karta má či nemá aktivovaný 3D Secure Issuing. V případě, že ano, pokračuje se ověřením identity, jak je popsáno v předchozím odstavci. V opačném případě transakce pokračuje bez tohoto ověření). Obchodníky podporující službu 3D Secure lze identifikovat např. tak, že na svých stránkách uvádějí loga služby 3D Secure a její adaptace VISA a MasterCard, konkrétně „Verified by VISA“ nebo „MasterCard SecureCode“.

Pokud jsou tedy při provádění platby obě tyto služby k dispozici, proběhne ono ověření identity. V jiném případě nikoliv.

Implementace 3D Secure v ČR

Některé bankovní ústavy v České republice se rozhodly v posledních letech toto řešení nasadit i pro české klienty. Způsobů zavedení je však několik a celý projekt integrace podobné služby dovnitř rozhlehlé banky neznamená zdaleka jen zapnutí jednoho boxu, který ověřuje klienty při provádění plateb. Ve většině

případů jde o rozšíření hlavních databází o potřebné údaje. Velké banky mají mnoho kanálů, které mohou posloužit k zadání požadavku na výrobu nové karty nebo aktivaci nové služby. Všechny tyto kanály jako jsou pobočky, internetová bankovnictví, bankomaty, webové konfigurátory nebo hlasové linky je pak nutné upravit. Ve velké bance se těmito jednotlivými agendami zabývá vždy specializované oddělení, které má zpravidla vlastní plánování úprav a změn. V těch pokročilejších ústavech lze pak s lehkostí využít připravených ITILovský change a release managementů. Ale sáhněme si do svědomí, kdo z nás dokázal ve své firmě nebo na svém pracovišti tyto procesy a standardy již plně adaptovat. Není výjimkou, že se z podobné a navenek drobné úpravy stává dlouhý a komplexní projekt, který požaduje přísné řízení.

Zavedení 3D-Secure znamená postupný nárůst počtu klientů využívajících tuto službu. Způsoby, jak dosáhnout pokrytí všech klientů té které banky je několik. Pro velký finanční ústav disponující desítkami i stovkami tisíc klientů není tento „náběh“ nic triviálního. Pro tento účel se kromě jiných opatření využívají definované stavy, které jsou mnohdy ve „front-end“ systémech viditelné i klientům (např. „Not Enrolled“ – neregistrovaný klient, „Semi-Enrolled“ – registrovaný ale neaktivovaný, „Enrolled“ – plně aktivovaný klient.). Jde pak o to, definovat scénáře a způsoby, jak klienty mezi těmito stavy vhodně přesouvat a zajistit jim tak plně funkční službu s potřebnými parametry. Velmi vhodným způsobem se jeví tzv. aktivace během nákupu („Activation During Shopping“), kdy je klient přidán do systému během provádění první ověřované platby. Dalšími kanály jsou pak pobočkové systémy nebo třeba i telefonní linky, internetová bankovnictví nebo dokonce bankomaty. Je však možné nutit klienty okamžitě službu využívat? Mnohdy k jejímu použití je třeba dalších nástrojů (např. mobilního telefonu) a nebylo by příliš vhodné blokovat v případě jejich momentální nedostupnosti provádění platby, zvlášť pokud se jedná o něco, s čím se klient setkává poprvé. Platby lze tedy ve většině implementací u konkrétních bank po omezenou dobu odkládat. Po plné aktivaci již však nikoliv.

Způsoby ověřování identity

Klient provádějící platbu může být v zásadě ověřován několika způsoby. První z nich je ten nejjednodušší a to pomocí statického hesla. S tím je spojeno množství problémů správy zvolených hesel, a proto se dnes hojně volí daleko hladší způsob ověření pomocí SMS kódů přes síť GSM. Mezi další metody patří ty méně známé, ale velmi zajímavé například i pro budoucí použití ve spojení s internetovým bankovnictvím apod. Řeč je o systémech CAP/DPA, které využívají speciální kapesní čtečky karet schopné generovat jednorázové autentizační kódy. Tyto čtečky mohou být také přímo integrovány do samotné platební karty („Display Cards“/„Code Secure“).

Obr. 1: DisplayCard firmy NagraID

Obr. 2: Čtečka CAP/DPA

Při použití CAP/DPA jsou generována „jednorázová hesla“ (tzv. tokeny). Uživatel musí mít k vygenerování jednorázového hesla k dispozici platební kartu a její PIN. Karty většinou generují tokeny o délce 9 znaků, když na svém čipu mají kromě platebních aplikací (např. pro VISA, MasterCard, JCB apod.) personalizovánu i separátní aplikaci právě pro CAP/DPA.

Vhodnost ověřovacích metod

Statické heslo se dnes jeví jako velmi zastaralý a slabý způsob ověření. To nic nemění na tom, že v mnoha případech je zcela funkční a dostatečným. Při použití s 3D Secure však praxe ukazuje, že jde o velmi neefektivní způsob, který vyžaduje poměrně značné úsilí pro vybudování systému správy.

V případě SMS kódů je otázkou, zda se jedná o ideální řešení. Tento způsob je velmi závislý na dostupnosti mobilního telefonu a také schopnosti jeho ochrany před případnou manipulací. Pokud k platbě přistupujete právě z telefonního prohlížeče a přijmete SMS kód, existují způsoby, které dokáží tento souběh manipulovat (samozřejmě v případě absence dalších zabezpečení chytrých telefonů apod.).

CAP/DPA se jeví jako velmi bezpečné řešení. Je však vhodné zavést u klientů tento „kapesní“ způsob prokázání vlastnictví platební karty? Pokud Vás někdo přepadne, vezme Vám platební kartu a bude chtít vědět její PIN, bude mít i možnost přímo si ověřit jeho správnost, v případě vašeho pokus o lež, bez nutnosti složitého shánění čtečky podobného typu (budete ji mít nejspíš u sebe

nebo ji v tomto nastavení bude vlastnit téměř každý). Ověřit správnost PIN lze dnes především u bankomatů, kde jsou ve všech případech instalovány kamerové systémy a s jejich pomocí lze alespoň s nějakou pravděpodobností zahlédnout případnou nestandardní situaci a angažované osoby.

V případě chybného nastavení služby není také příliš vhodné ztratit platební kartu s čtečkou, na které jsou vymačkána pouze 4 tlačítka. Pravděpodobnost uhodnutí kódu PIN se pak přeci jen značně zvyšuje. Samozřejmě toto lze řešit i správnějším nastavením, kde je při ověření platby zadáván do čtečky nejdříve jakýsi „feed“ opsaný z obrazovky monitoru a až poté je generován výsledný autentizační kód.

Při výběru vhodných metod jsou tedy vždy zvažovány pro a proti a hodnocena rizika. Většina bank v České republice se dnes uchyluje k autentizaci pomocí SMS zasílaných přes GSM.

Podpora u bankovních ústavů a platebních bran na českém území

K platbám v internetových obchodech (těch, u kterých je 3D Secure podporováno) jsou ve většině případů využívány platební brány PayU, GoPay (dříve PayMUZO). Vlastní platební bránu provozuje také Česká Spořitelna. Všechny tyto brány dnes službu 3D Secure podporují.

Pokud se budeme bavit o podpoře na platebních kartách, tak české banky službu 3D-Secure nabízí již velmi hojně a to v podobném nastavení. Situace je v současnosti tato:

Česká Spořitelna – 3D Secure Issuing je zaváděn od 17. 6. 2014. Ověřování plateb probíhá prostřednictvím SMS kódů. V současném nastavení je třeba plnou aktivaci provést povinně do 31. 12. 2014, pokud chcete mít limit pro Internetové transakce jiný než 0. Po zmíněném termínu nebude možné platby bez provedené aktivace dokončovat, což je případ i většiny ostatních bank.

Komerční banka – připravuje zavedení podpory pro své karty na listopad 2014. Zatím nebyly zveřejněny další podrobnosti.

Raiffeisenbank – zavedla 3D Secure na svých kartách na konci dubna 2014. Platby jsou ověřovány pomocí SMS kódu.

UniCredit Bank – zavedla podporu 3D Secure na svých kartách 3. 9. 2013. Platby jsou ověřovány pomocí SMS kódu.

ČSOB – podporuje u svých karet 3D Secure a ověřuje platby pomocí SMS.

Era (dříve Poštovní spořitelna) – zavedla podporu od května 2014. Ověření plateb je pomocí SMS kódu.

GE Money Bank – zavedení 3D Secure od 23. 7. 2013 pro platební karty. Ověření probíhá pomocí SMS kódu.

Citibank – byla první bankou, která v české platebním prostoru 3D Secure pro své karty zavedla. Službu zavedla již v roce 2011 a to 1. ledna. Ověření platby probíhá pomocí SMS kódu.

Fio banka – nepodporuje 3D Secure ověřování pro karty svých klientů.

ING Banka – zatím 3D Secure na svých kartách nepodporuje.

AirBank – 2. 5. 2014 informovala, že chystá zavedení služby 3D Secure Issuing v letošním roce. Otázkou je, zda k tomu letos ještě skutečně dojde.

mBank – zatím 3D Secure na svých kartách nepodporuje.

Equa bank – zatím 3D Secure na svých kartách nepodporuje.

LBBW Bank CZ – zatím 3D Secure na svých kartách nepodporuje.

Zuno Bank – zatím 3D Secure na svých kartách nepodporuje.

Sberbank – zatím 3D Secure na svých kartách nepodporuje.

Pocit falešné bezpečnosti?

Na některých Internetových fórech, hlavně pak těch, které provozují banky, jenž 3D Secure Issuing pro své karty ještě nezavedly, se objevují v mnoha variantách podobné příspěvky: „Dobrý den, chci se zeptat, kdy xxx banka plánuje zavést 3D Secure pro platby pres Internet. Pomalu se tato služba stává standardem a já osobně její absenci považuji za velkou bezpečnostní hrozbu pro držitele platebních karet. Dekuji za info.“

Bude však zavedení služby 3D Secure na platebních kartách té které banky tuto hrozbu skutečně eliminovat? Názory českých bank na zavedení nebo nezavedení 3D Secure v této souvislosti se dnes mnohdy liší a v některých případech podléhají určitému vývoji. Banky nám dnes rády tvrdí, že aktivací 3D Secure je naše karta zabezpečena proti nechtěnému úniku finančních prostředků prostřednictvím elektronických plateb. To však není celá pravda. Jak již bylo zmíněno, pro plnou funkčnost služby, tedy funkčnost dodatečného ověření identity, potřebujeme mít k dispozici obchodníka nebo lépe platební bránu podporující 3D Secure. Současně je třeba mít k dispozici platební kartu, která taktéž podporuje 3D Secure ověřování a má jej nastaveno a aktivováno. V jiných případech k ověření nedojde. V českém prostředí je tvrzení bankovních ústavů „téměř“ pravdivé. Zhruba 70 % obchodníků již 3D Secure podporuje a tedy zavedením služby u platebních karet lze tvrdit, že bude k ověření identity docházet. Není však pravda, že je karta od té chvíle zcela zabezpečena proti úniku prostředků při elektronických platbách. Při ztrátě důležitých údajů z „plastu“ nebo virtuální karty (PAN, datum expirace, CVV2/CVC2) je možné stále platit i při plné aktivaci 3D Secure na platební kartě a to bez jakéhokoliv dodatečného ověření. Tento scénář je uskutečnitelný u obchodníků nebo na eCommerce platebních bránách, které 3D Secure nepodporují. V prostředí světového Internetu je takových míst stále dost a dost.

Existuje tedy řešení?

Ano existuje a dokonce i v rámci standardu 3D Secure. Služba totiž ve svém popisu zavádí parametr „3D Secure Only“, který po své aktivaci zajistí, že není možné provést jinou než 3D Secure ověřenou transakci. Problémy jsou však především tyto:

- Banky musí tento parametr implementovat.
- Musí umožnit jeho použití. Současná situace je taková, že parametr zřejmě v mnoha případech existuje, ale klient o něm informován není. Banky si ho nechávají takříkajíc pro sebe (např. až v případě krizové situace nebo pro budoucí vývoj) a neumožňují jeho nastavení ani v rámci portálů jako je např. Internetové bankovnictví, které je jinak jedním z hlavních uzlů, kde lze parametry používané služby 3D Secure nastavovat.
- Při aktivovaném parametru „3D Secure Only“ není možné zaplatit na stále ještě značném počtu internetových obchodů a to především v zahraničí, ale v některých případech i u nás.

S dalším řešením přicházejí zejména ty banky, které 3D Secure Issuing ještě nezavedly a je otázkou, zda někdy zavedou. Tímto řešením je umožnění rychlé změny limitů pro Internetové transakce. Před vlastní platbou si např. pomocí Internetového bankovnictví zvýšit požadovaný limit, platbu provést a poté limit opět snížit. Tento postup uvedla např. Fio banka na své facebookové stránce 10. 2. letošního roku, konkrétně: „Fio karty nepodporují 3D Secure, tedy není možné platbu kartou potvrdit jednorázovým kódem zaslaným na mobil. Fio banka však nabízí pro zajištění bezpečnosti možnost okamžité změny limitů karty pro platbu na Internetu – je tedy možné mít tento limit nastaven na minimální úrovni, před platbou jej zvýšit a poté opět snížit, samozřejmě zdarma.“

Závěrem

Závěrem je tedy možné konstatovat, že služba 3D Secure s sebou přináší mnoho pozitivního a to zejména ve své Acquiring části, kdy zajistí, že uživatel poskytuje svoje karetní údaje přímo bance (platební bráně) a tyto vždy putují přes zabezpečený kanál. 3D Secure Issuing je samozřejmě také pozitivním posunem kupředu, ale je třeba ho dobře chápat. Nelze tvrdit, že je karta po aktivaci 3D Secure Issuing již zcela zabezpečena. Asi není zcela validní vyčítat některým bankám, že pro zajištění bezpečnosti na svých kartách volí jiné postupy. Otázkou je, zda dokáží nebo zda budou chtít odolávat možnému celospolečenskému tlaku k zavedení 3D Secure, které je mnohdy poněkud nešťastně prezentováno jako řešení k zajištění absolutního bezpečí při provádění Internetových transakcí.

Jak udělat banku

Václav Votípka

E-mail: Vaclav.Votipka@arbes.com

Běžný klient banky většinou vidí z bankovních systémů pouze internetové bankovnictví. Nitro moderní banky ale skrývá mnohem více zajímavých informačních systémů. Přednáška představuje jednotlivé systémy, jejich význam pro bankovní business a nároky, které jsou na tyto systémy kladeny.

Základ – bankovní systém

Používané IT systémy se samozřejmě liší podle velikosti a zaměření příslušné banky. Některé systémy ale musí mít každá, i ta nejmenší, banka. Moduly pokrývající základní funkcionalitu banky jsou někdy soustředěny do jednoho kompaktního bankovního systému, jindy jsou realizovány samostatnými systémy, mnohdy i od různých dodavatelů.

Pojďme si je ve stručnosti představit.

Transakční systém

Vybrané funkční požadavky:

- evidence účtů a jejich zůstatků
- zpracování transakcí
- blokace prostředků na účtu
- podpora více měn
- stornování transakcí
- závěrkové operace v on-line režimu

Zvýšené nároky:

- Propustnost systému. Téměř každá operace v bance končí vytvořením minimálně jedné transakce. Transakční systém tak musí být schopen zpracovávat značné množství transakcí, ideálně s preferencí zpracování transakcí, na jejichž výsledek uživatel čeká on-line (internetové bankovnictví, call centrum, pobočky)

- Detailní auditing. Jde „jen o peníze“, takže je nutné, mít možnost kdykoliv zpětně dohledat, kdo kdy a jakým způsobem inicioval převod peněz z jednoho účtu na druhý.

Businessové produkty

Jednotlivé produkty banky se velmi liší, ale z pohledu IT je můžeme shrnout do jedné kategorie. Mám zde na mysli produkty, jako jsou běžné účty, spořící účty, termínované vklady, úvěry, atd.

Zvýšené nároky:

- Rychlost implementace. Když si banka vymyslí nový produkt, tak ho potřebuje uvést na trh zpravidla „ještě včera“. Dobrý bankovní systém umožňuje definici nových produktů uživatelsky pomocí parametrizace systému. Pokud se zde vyhneme nutnosti programování, může být nový produkt otestován a spuštěn v řádu jednotek dní.
- Variabilita. Vlastnosti produktů jsou tím, čím se banky od sebe navzájem odlišují. Dobrý dodavatel bankovního systému umožní bance v oblasti produktů „cokoliv“.

Transakční styk s okolním světem

Banka není uzavřený systém – musí umět posílat a přijímat peníze z ostatních bank. Za tímto účelem je každá banka napojena na národní clearingový systém a (přímo nebo přes prostředníka) do mezinárodní sítě SWIFT.

Zvýšené nároky:

- transakce přicházejí zpravidla v dávkách. Je tedy nutné dávku zpracovat v rozumném čase ale zároveň bez negativních vlivů na výkonnost systému pro front-endové aplikace (internetové bankovnictví, pobočky, call centra).
- zejména protokol sítě SWIFT je poměrně komplikovaný – obsahuje několik desítek typů zpráv, každá zpráva pak má mnoho možností, v jakém formátu zapisovat jednotlivé položky.

Platební karty

Karetní modul zajišťuje napojení na agregátora karetních služeb. Mezi nejdůležitější funkce patří zodpovídání dotazů na disponibilní zůstatek karty, blokace prostředků na účtu a zpracování karetních transakcí.

Zvýšené nároky:

- Rychlost. Dotazy na zůstatek a požadavky na blokaci prostředků probíhají v synchronním režimu. Zde je kladen velký důraz na rychlost zpracování (klient nebude u bankomatu čekat věčně).

- Nezávislost na transakčním systému. Banka zpravidla nechce při odstávce bankovního systému (např. z důvodu nasazování nového produktu, instalace opravného patche apod.) omezovat karetní služby. Karetní modul tak musí základní operace (zjištění zůstatku na účtu a blokace prostředků) zvládat i v případě, kdy transakční systém není dostupný.

- Zpracování velkých dávek. Zatímco požadavky na blokaci prostředků přicházejí průběžně (on-line), samotné zpracování plateb se provádí dávkově s odstupem až několika dní. Požadavky na provedení plateb tak chodí ve velkých dávkách, se kterými si karetní modul (ale hlavně transakční jádro) musí umět rychle poradit.

Další podpůrné systémy

Dále uvedené systémy nemusí být nutně v každé bance přítomné. Moderní banka se bez nich ale neobejde, protože jí umožňují významně snižovat náklady na provoz a efektivněji prodávat své služby klientům.

Internetové bankovnictví

Je jen málo bank, které se dnes bez internetového bankovnictví obejdou. Moderní banka nabízí v internetovém bankovnictví kompletní paletu svých produktů a služeb, aby si klient mohl příslušnou službu objednat v pohodlí svého domova a hlavně bez nutnosti přímé komunikace s pracovníkem banky.

Zvýšené nároky:

- Bezpečnost. Banka musí počítat s tím, že internetové bankovnictví nebudou používat pouze experti na internetovou bezpečnost a tak musí pro běžné netechnické uživatele zajišťovat dostatečnou míru bezpečí pro jejich finanční prostředky. Kromě šifrování komunikace sem patří zejména ověření identity uživatele i jiným komunikačním kanálem (SMS, hardwarové tokeny apod.) a aktivní obrana proti známým druhům internetových útoků (XSS, XSRF, SQL injection, Clickjacking a mnoho dalších).

- Vzhled a použitelnost. Internetové bankovnictví je výkladní skříní banky a vzhled a použitelnost jeho rozhraní mohou hrát pro klienta klíčovou roli při výběru banky. S nástupem nových typů zařízení (chytré telefony, tablety, chytré televize, Ď) banky upouštějí od nestandardních technologií (ActiveX, Java plugin, Flash apod.) a stále více využívají čisté HTML a JavaScript.

CRM – Customer relationship management

CRM slouží jako centrální evidence klienta a jako master klientských dat pro všechny ostatní systémy v bance. Soustřeďuje se zde veškerá komunikace s klientem a dokumentace (např. smlouvy uzavřené s bankou). Dále se prostřednictvím CRM plánují nové marketingové kontakty s klientem (často ve spolupráci s bankovním DWH).

Základní CRM funkce může poskytovat i samotný bankovní systém. Moderní banky se ale většinou neobejdou bez mnohem sofistikovanějších řešení.

Zvýšené nároky:

- Rychlost. Protože je CRM master systémem pro klientská data, záleží často na rychlosti jeho odezvy i rychlost odezvy dalších systémů banky.
- Integrovatelnost. CRM musí nabízet univerzální konektory pro připojení ostatních systémů – dnes nejčastěji ve formě webových služeb.

DWH – Data warehouse

Slouží pro transformaci provozních dat banky z primárních systémů do podoby vhodné pro další analytické zpracování dat.

Zvýšené nároky:

- Velké datové úložiště. Objem dat v DWH zpravidla předčí velikost dat ve všech ostatních systémech banky.
- Neviditelnost pro ostatní systémy. DWH potřebuje přenášet z primárních systémů banky velké objemy dat. Tyto přenosy ale nesmí negativně ovlivnit rychlost odezvy primárních systémů.

DSS – Decision support system

Pokud chce banka fungovat efektivně, potřebuje v maximální možné míře nahradit manuální lidskou práci počítačovými systémy. Systémy pro podporu rozhodování umožní bance např. automaticky schvalovat úvěry nebo hledat transakce podezřelé na praní špinavých peněz.

Zvýšené nároky:

- Konfigurovatelnost. Změny v rozhodovacích procesech musí být možné dělat uživatelsky, za chodu a tedy bez nutnosti programování.

Technologické okénko

V čem je to všechno napsané? A na čem to běží?

Jednoznačně převládá trend nahrazování těžkých klientských aplikací za jejich lehké intranetové alternativy. Mezi programovacími jazyky tak nyní převládá Java, PHP a JavaScript.

Na databázové úrovni to pak je zpravidla Oracle, případně DB2 či MS SQL. Zvláště menší banky, které nemají problém s výkonem a snaží se ušetřit, kde to jde, občas sáhnou i po PostgreSQL.

I operační systémy se liší podle velikostí bank. Velké banky používají zpravidla komerční unixové či linuxové (RedHat, Debian) systémy, malé banky pak často sáhnou i po systému Windows Server.

Závěr

Banky se v současné době snaží minimalizovat své provozní náklady a v maximální míře automatizovat všechny interní procesy – od získávání nových klientů a vytvoření smluvního vztahu až po výběr a nabídku vhodných produktů a jejich schválení a aktivaci. Při tomto úsilí hrají IT systémy stále větší a důležitější roli.

Interní vývoj bankovního systému

Adrian Kantor

E-mail: adr@fio.cz

V době kdy, nadšené články o outsourcingu nahradily nadšené články o cloudu, není úplně běžné říci, že lze dlouhodobě vyvíjet a provozovat bankovní systém vlastními silami. Je na čase si přiznat, že celá IT udělala obrovský skok. Lze se tedy daleko více věnovat samotnému problému, než výběru konkrétní platformy pro jeho řešení.

Stejně tak vedle sebe existuje množství velmi protichůdných postupů vývoje. Každý z nich má své výhody a nevýhody. Krátké shrnutí konkrétních zkušeností z interního vývoje může být podnětem k zamyšlení nad používanými postupy a určením jejich silných a slabých stránek.

Specifika bankovního systému

Rozsah bankovního systému je opravdu velký. Zdaleka ne tolik, co se týče objemu vlastních dat, ale především složitostí aplikační logiky a množstvím jednotlivých subsystémů. Samotné jádro bankovního IS je nesmírně triviální – eviduje stav na kontě a pohyby mezi konty. Není problém navrhnout a realizovat projekt, který úspěšně obslouží konta menší banky. Nicméně existují tři zdroje, které nepřetržitě a závratným tempem dodávají složitost do bankovního IS.

Prvním jsou poskytované produkty banky. Tedy logika, jakým způsobem v čase a prostoru poskládat pohyby mezi konty, aby vznikla další abstraktní vrstva poskytující služby jako úročení, úvěr, měnová konverze, obchod s akciemi, exekuce na účtu...

Druhým je legislativa – veřejnoprávní i vnitřní předpisy, které specifikují, co IS nesmí dovolit, co musí po uživatelích vyžadovat, co musí uchovávat, co a jak musí vykazovat.

Třetím zdrojem jsou další subjekty, se kterými se banka musí elektronicky propojit – clearingy, karetní asociace, burzy, depozitáři, dozorové a státní orgány. V těchto případech je dáno pevné rozhraní, které lze pouze akceptovat a promítnout ho do IS. Často to přináší i další příděl legislativy a výkaznictví.

Je tedy zřejmé, že IS banky s rostoucím portfoliem služeb je v permanentním vývoji. Navíc legislativa se mění i pro technologicky stagnující banku. Bankovní IS je typickým příkladem říše, která čím je větší, tím větší má hranice, které musí

hlídat. Množství a načasování změn je nepředvídatelné a nelze je tedy v předstihu plně pokrýt univerzální logikou, která by se později pouze nakonfigurovala. Rychlejší a přesnější je realizovat konkrétní řešení, až přijde jeho čas.

Důvodem, proč lze udržet tempo i přes výše uvedené obtíže, je elektronická podstata současného bankovnictví. Jediné reálné objekty, se kterými je třeba pracovat, jsou klienti, bankovky a platební karty. Číslo v databázi není obrazem zůstatku na účtu, ale přímo zůstatek na účtu. V jádru bankovního IS není rozdíl mezi realitou a jejím modelem. Odlišení reality a modelu nastává až na hranici s uživatelem, klientem, dalším systémem.

Interní vývoj

S ohledem na předchozí část je zřejmé, že při interním vývoji je kladen větší důraz na pochopení problému a znalost kontextu, než na niterné znalosti nových technologií. Permanentní tok požadovaných změn automaticky vede k snaze řešit je nezávisle na sobě technikami, které se osvědčily. Důsledkem jsou krátké iterační cykly a podobnost s agilním vývojem. Zároveň to ale znamená značnou konzervativnost v použitých nástrojích a postupech. Pokud je potřeba udělat podobnou změnu, jaká se do systému zanášela před měsícem, bývá nejlepší implementovat ji podobně jako tu původní. Až v době, kdy se nějaký subsystém stává nevyhovujícím a stále více vývojářů naráží na jeho nedostatky, identifikuje se problém a vybere řešení. Jeho součástí je obvykle i menší migrace dat. Není to každodenní událost, ale ani nic zcela výjimečného. Rozhodnutí, že dozrál čas pro radikální změnu, je ryze pragmatické.

Interní vývoj je důsledkem i příčinou velkého sepětí mezi uživateli a vývojáři. V dobrém případě umožňuje neformální vazby a procesy, ale nebrání formalizovaným postupům a metodikám. Nepoměr mezi rozsahem IS a velikostí IT lze řešit pouze důsledným ořezáním zadání až na kost. Výslednou výhodou je IS podporující pouze skutečně potřebnou funkcionalitu. Při zachování jednoduchosti, lze vždy IS později rozšířit, pokud to bude skutečně potřeba.

Díky permanentnímu vývoji a množství kódu se postupně vždy vyprofiluje několik nástrojů, které jsou všeobecně používané. Aktuálně je to zcela neinvenčně především SQL a Java. Použití nových jazyků není nijak formálně regulováno. Pouze nejprve musí prokázat, že případné klady získané jejich zavedením, převáží objem existujícího kódu. Nebývá to často, ale přirozeně cyklicky to nastává. Do té doby je třeba z používaných nástrojů vytěžit maximum.

Samozřejmě občas nastane situace, že aktuální dovednosti týmu nestačí na vyřešení požadavku. V takovém případě je potřeba zajistit externí zdroje. Při vývoji jim poskytnout maximální součinnost – konzultace, připomínkování a testování. A po předání převzít kód do své správy a rozšířit své znalosti a dovednosti.

Vývojáři

U vývojářů se očekává solidní znalost používané technologie a ochota učit se technologie nové. A ještě více je potřebná ochota učit se chápat aplikační logiku a umět zasadit zadání do kontextu. Naštěstí se ukazuje, že běžný vývojář mívá pouze psychické zábrany a po překonání bariéry se ve finančním světě poměrně rychle rozkouká. I tak samozřejmě trvá měsíce, než získá samostatnost. I zkušený vývojář, pokud narazí na novou oblast, potřebuje během celého vývoje intenzivní konzultace se zástupci zadavatele. Aby měly, smysl musí rozumět základním pojmům a kontextu.

Po nasazení většího balíku kódu, typicky spojeného s novým produktem, následuje poměrně dlouhé období, kdy uživatelé i klienti požadují různé drobné změny a vylepšení. Ty, které jsou přijaty k řešení, je pak nejsnazší předat stejnému vývojáři. To je jistě nejrychlejší cesta k cíli. Praxe však ukazuje, že v dlouhodobém horizontu je lepší zadávat tyto požadavky někomu úplně jinému, kdo na původním balíku vůbec nepracoval. Čistý čas vyřešení je sice o něco delší, ale další vývojář se seznámí s novým subsystémem. Nový vývojář automaticky provede peer review a občas i nějaký refactoring. Původní vývojář ztratí status jediného znalce této oblasti a zmizí problém s jeho zástupem při dovolených. Přirozené sdílení kódu také intuitivně pomáhá nováčkům učit se úspěšně používané vzory. A podvědomě nutí psát kód čitelný a snadno pochopitelný.

Týmy pro řešení konkrétních zadání lze sestavovat ad hoc. Kromě zkušenosti a znalosti potřebné technologie je hlavním klíčem pro zařazení do týmu aktuální vytížení konkrétního vývojáře. To umožňuje snadné přeskupení týmu během vývoje a přiřazení vývojáře k týmu, jen na opravdu nutnou dobu. Úspěšnou technikou je rozdělení zadání na samostatné části a „naskicováním kódu“ základní architektury vedoucím týmu. Vývojář tak dostane kromě zadání i kostru, která je mu daleko srozumitelnější a lépe v ní chápe souvislosti. Až ve chvíli, kdy mají základy hotovou první verze, lze vývojářům předat k řešení další kusy, které na hotovou část navazují. Vývojář se tak nemusí zabývat průběžnými změnami v jiných částech. A protože další kusy navazují na hotovou část, není zde obvykle nutné nic skicovat a je poměrně zřejmé, jak to má k sobě pasovat.

Přestože, zde není vývoj zcela paralelní, je rychlý. Odpadá promarněný čas, kdy vývojář čeká na specifikaci rozhraní, nebo přepisuje kód v důsledku změn v jiné části. Vývojář několikrát přepsaný kód velmi nerad opouští, ačkoliv zkušenost jasně hovoří, že je lepší a rychlejší ho zahodit a napsat znovu. I pokud vývojář nakonec několikrát přepisovaný kód nakonec smaže, zůstane mu v hlavě jeho otisk. To naštěstí při tomto přístupu odpadá.

Nebezpečným nešvarem je implicitní rozšíření zadání do příliš obecné podoby. Snahou je pokrýt budoucí potřeby. Nejlépe nějakou sofistikovanou konfigurací. Když pak v budoucnu skutečně dojde na případ, který by mělo obecné řešení pokrýt konfigurací, ukáže se buď, že je to nemožné, nebo je stejně nutná nějaká

úprava obecného kódu. Je to naprosto logické, v okamžiku zobecnění nikdo neměl požadovaný reálný příklad mimo rozsah původního zadání. Je tedy daleko lepší realizovat zadání v jeho původním rozsahu za použití standardních technik a postupů. Když se později ukáže, že je potřeba zahrnout do řešení i další případy, je snazší upravit kód s konkrétní znalostí a případně migrovat data, než se pokoušet konfigurovat něco obecného, co na to nové jaksi nepasuje. Brzké zobecnění zadání neznamená ušetřenou práci v budoucnu, ale zanesení další zbytečné složitosti do systému nyní. Tu je nutno dokumentovat, otestovat, provozovat a při další změně znovu nastudovat a nakonec ji jako nevyhovující zahodit. Při sdílení kódu a krátkých iteračních cyklech je snadné pokus o příliš zobecňující rozšíření detekovat. Je ale velmi obtížné se ho zcela vyvarovat. Záchytným bodem je aktuálně řešené zadání, ke kterému je třeba se neustále vracet během konzultací.

Výhodou interního vývoje je nepochybně minimum pevných termínů. U většiny požadavků není tlak na konkrétní termín. Hotové věci, lze nasadit po úspěšném otestování. Tedy v době kdy všichni zúčastnění mají o změnách dobrý přehled a v době, kdy jim to vyhovuje. I když se to na první pohled nezdá, velkým problémem jsou pevně stanovené vzdálené termíny. Zakonzervované změny blokují další požadavky a před nasazením je třeba úpravy opětovně prozkoumat, nastudovat a leckdy i přetestovat.

Pochopitelně, kromě této „idylky“ bez termínů, přistávají někdy na stole i zadání s šibeničním termínem. Při interním vývoji si lze dovolit rychle odstavit ostatní projekty, u kterých pevný termín není. Současně je potřeba intenzivními konzultacemi s uživateli minimalizovat rozsah zadání v duchu pravidla 80/20. Tak, aby v termínu bylo hotové minimum, pokrývající nezbytný rozsah. A v den termínu práce rozhodně nekončí.

Uživatelé

Zásadním prvkem interního vývoje jsou bez pochyby jeho uživatelé a jejich intenzivní komunikace s vývojáři. To na ně klade požadavek alespoň základní orientace v IS. Výhodou je, že poměrně snadno detekují místa, která by chtěli upravit, nebo rozšířit. Velkou nevýhodou je, že základní orientace jim dává falešný pocit znalosti celého IS. Proto někdy vytvářejí bizarní požadavky na „rychlé a snadné“ poslepování čehosi z viditelných prvků IS.

Ve své podstatě je to stejný problém, jako vývojářova snaha o předčasné zobecnění, pouze promítnutý z druhé strany. V lidské rovině se pak jedná o konflikt mezi neformálními vazbami a vymezením kompetencí. Je potřeba důsledně přenechat na uživateli definování **CO** se má řešit a na vývojáři **JAK** se to má řešit. Snahy pomoci druhé straně – tedy vývojář definující zadání, nebo uživatel popisující implementaci, jsou zcela kontraproduktivní. Takto formulované je to

samozřejmě zcela jasné. V každodenní praxi je to někdy odhaleno až zpětně. Nezbývá než přiznat, že se jedná o nepřetržité střetávání a v neformálním prostředí nelze očekávat direktivní stanovení pevných hranic.

Přirozeně očekávanou výhodou interního vývoje je rychlá zpětná vazba. Možnost vytvořit prototyp a získat reakci uživatele v řádu hodin dovoluje velký průtok změnových požadavků při vývoji. Požadavek je obvykle dořešen tak rychle, že nenastane konflikt s další požadovanou úpravou. Uživateli to kromě nové funkcionality přináší i množství práce navíc. Jednak jejich specifikaci, následné konzultace, účast na testování, ale především rozhodnutí, které požadavky vybrat a prosadit k realizaci. Interní vývoj tedy rozhodně uživatelům práci neubírá.

Dvojsečnou vlastností je fyzická blízkost uživatelů a vývojářů. Pozitivně podporuje neformální vazby, které ale snadno můžou přerůst ve výše uvedené kompetenční problémy. Uživatelé sklouzávají k potřebě každý okamžitý nápad ihned a nejlépe osobně probrat. Akceptování takové praxe má naprosto devastující dopad na rychlost a chybovost vývoje. Striktní zabarikádování dveří zase vede uživatele k sepisování nesmyslně podrobných zadání, kde je důraz kladen na řadu technicky zcela nepodstatných detailů. A které je v důsledku své obsáhlosti dodáno obvykle pozdě. Jako vhodný kompromis se jeví na jedné straně jasně deklarovaná ochota vývojářů absolvovat neformální osobní schůzky po předchozí elektronické dohodě. A na straně druhé striktní ochrana teritoria vývojáře jeho vedoucím.

Důležitá poznámka – příspěvek je psán ryze z osobního pohledu vývojáře. Uživatel by měl možná výhrady, nebo zcela odlišný názor.

Závěr

Interní vývoj rozhodně není nejlepší, ani univerzální řešení. Nelze se pro něj rozhodnout jen tak. Pokud máte uživatele, které zajímá, jak váš IS funguje, pokud máte vývojáře, kteří se nebojí vysvětlení, co jejich kód znamená a pokud rostete, můžete zcela přirozeně získat IS, který si vyvíjíte a provozujete sami. Ale rozhodně to není jediná možná cesta.

Ať už při vývoji jdete po libovolné cestě, z času na čas, když se naskytne příležitost, udělejte krok stranou a zkuste něco jinak. A pak zkontrolujte, zda odvěká pravda, neochvějně zanesená v mysli, je stále na svém místě.

Souborové systémy v cloudu

Filip Hubík, Adam Tomek

E-mail: hubik@ics.muni.cz

Abstrakt

Distribuované či sdílené souborové systémy jsou v dnešní době nedílnou součástí moderních datacenter. Jedná se o systémy, které jsou svou strukturou odolné, rychlé, škálovatelné, propustné a často vysoce dostupné. Mají řadu výhod. Při výpadku určité části clusteru nemusí dojít přímo k pádu celého úložiště („No Single-Point-of-Failure"), systém může také zavádět redundanci dat, automatické vyvažování zátěže všech dostupných serverů, objektové úložiště, umožňovat paralelní práci s daty apod.

Pro optimální provoz cloudových služeb je třeba důsledně volit veškeré součásti infrastruktury, souborový systém nevyjímaje. V prostředí open source můžeme dnes vybírat z několika desítek kandidátů, které vyhodnocujeme dle provozovaných služeb a potřeb datacentra. Z širokého spektra vybereme současné lídry v oboru souborové systémy Ceph a GlusterFS. Za oběma stojí jak silná komunita, tak velké IT společnosti, které se snaží prosadit v konkurenci právě svůj projekt.

Cílem je představit vlastnosti a srovnat souborové systémy nejen po stránce výkonnostní, ale také po stránce vhodnosti použití v cloudovém datacentru. Je třeba také zhodnotit kompatibilitu se současnými virtualizačními technologiemi či vhodnost použití s určitými typy cloudových služeb či middlewaru (OpenNebula, OpenStack).

1 Úvod

Jádrem cloudu je proces virtualizace výpočetních prostředků, tj. na nejnižší úrovni (IaaS) zejména paměti, procesorů a disku. Naopak úkolem cloudových middlewarů je příprava důležitých součástí tohoto procesu, ať se jedná o správu infrastruktury, udržování metadat či důležitou část – distribuci obrazů virtuálních strojů na hostující stroje, vykonávající vlastní virtualizaci. Proces virtualizace procesoru a paměti není ze strany souborových systémů či cloudových middlewarů problematický – virtualizační platformě se předá požadavek na jejich množství a následně je mu vyhověno nebo ne. Problémem je však virtualizace obrazů disků virtuálních strojů, konkrétně jejich správa a distribuce napříč

cloudovou infrastrukturou. Správu obrazů ponechme na použitém middlewaru, tento článek bude orientován na jejich distribuci ve spolupráci s vhodným soub. systémem.

V tomto článku se budeme více zabývat dvěma v současné době nejskloňovanějšími jmény v prostředí cloudových open source souborových systémů – Ceph a GlusterFS. Bylo by také možno zahrnout menší projekty, např. MooseFS, XtreemFS, projekty ne primárně či sekundárně zaměřené na cloudové využití (HDFS, Lustre), nebo proprietární systémy (např. GPFS), tím by byl však překonán vyhrazený rámec tohoto článku. Jako referenční cloudové middlewary uvažujeme platformy OpenStack nebo OpenNebula, jež oba zvolené systémy v různorodé míře podporují. V oblasti virtualizačních softwarů budou sloužit jako případný kontext open source platformy KVM a Xen.

2 Distribuce obrazů

Aby dokázal cloudový middleware efektivně spravovat obrazy, musí využívat nejefektivnějších možných přístupů jejich distribuce. Nejčastější jsou dva možné přístupy – kopírování obrazů nebo sdílený souborový systém. Oba mají svoje výhody a nevýhody. Dalším možným přístupem je také deployment z úložišť třetích stran (objektového úložiště, webových marketplace, ...), tento přístup ale není z pohledu cloudových souborových systémů atraktivní.

Kopírování obrazů (pomocí middlewaru) je zpravidla konfiguračně jednodušší, problémem je však špatná škálovatelnost, pomalá rychlost a neflexibilita. Obraz disků je umístěn zpravidla v centrálním úložišti, ze kterého jej middleware musí nejprve zkopírovat do souborového systému cílového uzlu. Obraz je poté předán virtualizačnímu softwaru a po dokončení práce buď zahozen, nebo zkopírován zpátky do zdrojového úložiště. Neflexibilita a špatná škálovatelnost takového procesu je patrná. Roste-li počet souběžných procesů kopírování z úložiště, může dojít až v přehlcení celého pásma přenosového kanálu od úložiště směrem k cílovým uzlům. Uživatel zpravidla pracuje v cloudu vždy s minimalistickým obrazem, který v průběhu času roste, proces kopírování obrazu se tedy prodlužuje. Toto je nevhodné zvláště v případě používání „non-sparse“ RAW formátu obrazů, jejichž velikost je neflexibilně předem definovaná při tvorbě obrazu. Bez ohledu na množství uložených dat se provede kopie celého obrazu, což je zejména při iniciální práci s obrazem nežádoucí a uživatelsky nepřívětivé. Situace je lepší u formátů, jejichž velikost kopíruje množství uložených dat (QCOW, dynamic VDI), a je tak omezena již na úrovni formátu obrazu. Takový formát tak ze své podstaty omezuje množství dat kopírovaných po síti. Podobně tak činí i „sparse“ formát RAW, ale program realizující kopírování jej musí podporovat (např. rsync ano, scp ne). Kupní silou uvedeného přístupu je tedy jednoduchost, ale za cenu řady mandatorních nevýhod.

Další přístup spočívá v použití clusterového souborového systému. Existují dva typy: distribuovaný a sdílený. Sdílený systém umožňuje souběžný přístup k datům z více uzlů na úrovni blokového zařízení, naopak distribuovaný využívá ke sdílení síťový protokol. Souborový systém se rozloží po celé infrasrtuktuře alokované pro cloudové využití. Klientská část systému bude odpovídat hostujícím uzlům, úložná část bude připojena na úložiště či určitý typ úložné infrastruktury. Tady odpadá potřeba kopírování obrazů. Problematika flexibility a škálovatelnosti je tak delegována na vrstvu souborového systému, který má většinou vlastnosti vhodné pro tento typ použití. Zejména se jedná o možnost replikace dat, která zlepšuje charakteristiku vytížení přenosového pásma sítě bez škod na škálovatelnosti, která bývá často lineární ([3] a [2]). Síťová infrastruktura mezi úložištěm a virtualizačními uzly tak již není úzkým hrdlem, přes které probíhá veškerý přenos dat, ale nyní je plněji a tudíž účelněji využívána. Škálovatelnost do počtu klientských uzlů opět není v kombinaci s replikací problémem. U tohoto typu úložiště odpadá veškerá starost spojená s formátem či velikostí obrazů disků. Virtualizační vrstva přistupuje přímo k souborovému systému bez účasti middlewaru, což je značná optimalizace oproti předchozímu.

Druhý způsob se tedy jeví jako celkově vhodnější pouze za cenu složitější konfigurace. Je však třeba dodat, že ze strany souborového systému mohou být kladeny požadavky na middleware či virtualizační vrstvu a naopak. V praxi to může znamenat podporu nepřítomnosti obrazu v lokálním souborovém systému, ale možnost předání pouze odkazu na obraz, který middleware použije až v čase deploymentu např. tvorbou blokového zařízení (Ceph – rados block device). Dále je také třeba uvažovat, že ne každý virtualizační software musí být s vybraným soub. systémem kompatibilní, a to např. díky různorodé implementaci virtualizovaných ovladačů přípojných zařízení, chybějící nativní podpoře systému či různě implementované obsluze různých formátů obrazů.

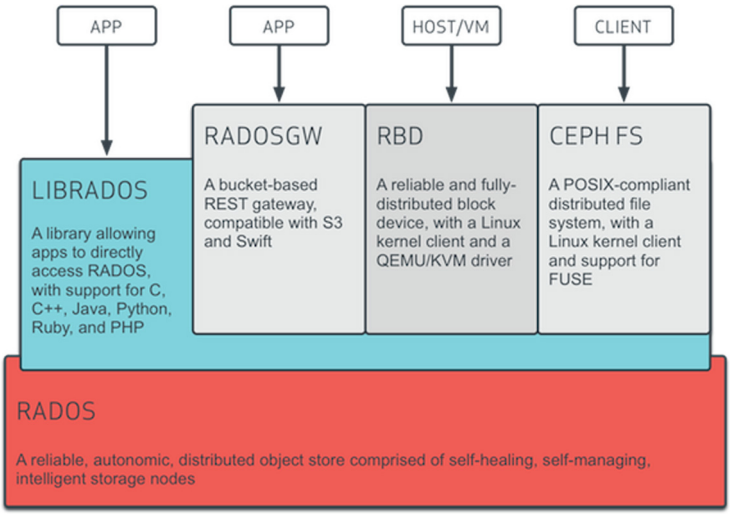
3 Ceph

Ceph je softwarový balík obalující sadu několika komponent sdružených do většího celku, tzv. „Ceph Storage Cluster“. Správcem projektu byla až do začátku roku 2014 firma Inktank Storage, později odkoupena firmou Red Hat. Cílem projektu je tvorba stabilního, škálovatelného, chybám odolného a jednoduchého řešení pro veškeré úložné potřeby v dnešním cloudu. Je vyvíjen od roku 2004 a již několik let je v produkčním stavu. Skládá se z následujících komponent:

- Ceph Object Store – objektové úložiště (RADOS)
- Ceph Object Gateway – rozhraní objektového úložiště (S3, OpenStack Swift)

- Ceph Block Device – sdílené úložiště exportující bloková zařízení
- Ceph Filesystem (CephFS) – sdílený souborový systém

Není podmínkou instalovat a konfigurovat všechny součásti balíku, je možné si vybrat komponentu dle požadavků na použití. Jádrem je však Ceph Object Store, implementován vrstvou „Reliable Autonomic Distributed Object Store“ (RADOS). Nad touto vrstvou je možné provozovat všechny nebo některou z dalších komponent. Architektura je znázorněna na obrázku 1. Výše uvedené komponenty se sestavují ze tří základních primitiv a klienta:



Obr. 1: Architektura Ceph

- Ceph OSD – správa dat
- Ceph Monitor – správa stavu clusteru, monitoring, distr. algoritmy
- Ceph Metadata Server (MDS) – správa metadat pro souborový systém CephFS
- Ceph client – klientský software

Pro minimální běh v základní konfiguraci je potřeba alespoň jednoho monitoru a dvou OSD. OSD uzel se stará o uchování, replikaci aktuálních dat, monitoring topologie a interakci s logickým úložištěm. Spolu s Ceph klienty využívají pseudonáhodného algoritmu CRUSH k optimálnímu rozložení zápisu či

čtení dat napříč clusterem. Tímto způsobem se zajišťuje optimální škálovatelnost. Ceph Monitor udržuje informace o struktuře a stavu clusteru, stará se také o zajištění vysoké dostupnosti. Udržuje tzv. mapu clusteru, řeší výpadky uzlů a interaguje s dotazujícími se klienty. MDS uzel je potřebný pouze pro POSIX soub. systém CephFS, udržuje jeho metadata formou stromové struktury a spravuje I/O operace. Je možné využití více MDS uzlů, stromová struktura se poté vhodně rozdistribuje tak, že se určité MDS uzly spravují pouze určitý podstrom. Tento souborový systém je v současné době stále v experimentálním stavu (fáze komunitního testování) a do produkce se nedoporučuje. Cílem autorů bylo eliminovat jakékoliv úzké hrdlo v systému, celý systém je tedy stavěn v duchu „No Single-Point-of-Failure“ (SPOF) filosofie. Ceph disponuje plnou podporou v OpenStacku, v OpenNebule je podporován v kombinaci s libvirt/KVM. Podporovány jsou soub. systémy xfs, ext4 a btrfs.

Nejdůležitější součástí z pohledu cloudového úložiště je Ceph Block Device. Umožňuje na klientském uzlu připojení obrazu z Ceph Object Store pomocí tvorby tzv. „thin provisioned“ blokového zařízení. Obraz připojený tímto způsobem zabírá v úložišti pouze svou aktuální kapacitu, ne však maximální velikost. Úložiště se tedy chová jako sdílený souborový systém exportující bloková zařízení místo vlastních souborů. V kombinaci s podporou na vrstvě virtualizace a cloudového middlewaru je možno provádět deployment v cloudu prakticky bez účasti lokálně připojeného souborového systému. Tímto přístupem dochází opět k optimalizaci celého procesu deploymentu, protože takto nevzniká potřeba připojovat obraz z úložiště jako blokové zařízení před vlastní virtualizací.

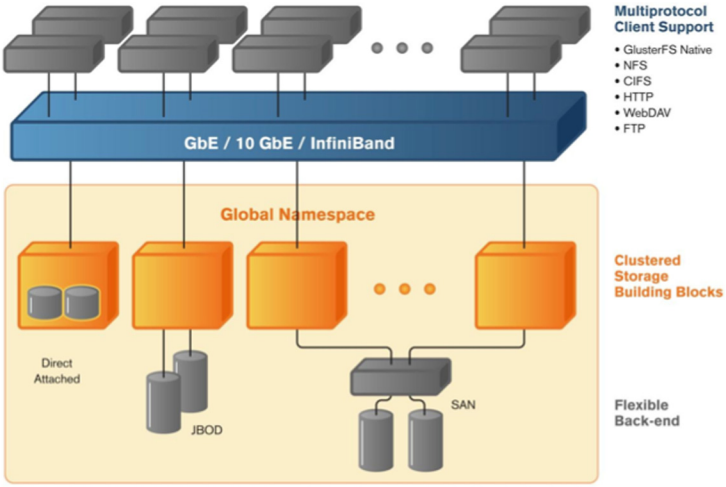
Zklamáním je absence podpory nativního infinibandu; Ceph podporuje maximálně technologii IPoIB. V současné době existuje implementace, zatím je však ve fázi ranného vývoje.

4 GlusterFS

GlusterFS je distribuovaný POSIX souborový systém vyvíjený původně firmou Gluster, od roku 2011 společností Red Hat. Jako výrazné vlastnosti autoři prezentují mirroring, striping a automatickou replikaci na přípojných bodů, dále vysokou dostupnost, odolnost proti výpadku či load balancing. Oproti systému Ceph je výhodou podpora nativního infinibandu (RDMA). Podobně jako u Cephu, i GlusterFS nepoužívá centralizovaný model metadat, čímž eliminuje SPOF při výpadku metadata serveru(ů) (je použit elastický hashovací algoritmus). Podporuje také dynamickou manipulaci s uzly. Výhodou je také jednodušší instalace oproti Cephu. GlusterFS disponuje také zajímavou vlastností – automatickou replikací na úrovní lokální (synchronní) či geografické (asynchronní). Na nižší úrovni podporuje souborové systémy xfs, ext3, ext4, reiserfs, jfs, výrazně se však doporučuje použití xfs. Přístup k GlusterFS je možný pomocí standard-

ního FUSE rozhraní. Dalšími podporovanými variantami jsou NFS (nativní podpora), SMB/CIFS či přístup pomocí unifikovaného objektového úložiště. Skládá se z následujících komponent:

- glusterd – serverový démon, správa oddílů
- glusterfsd – správa přípojných bodů, řízeno pomocí glusterd
- glusterfs – správa FUSE a NFS exportů
- gluster – CLI rozhraní



Obr. 2: Architektura GlusterFS

Data jsou organizována pomocí primitiv „bricks“, které reprezentují připojené zdrojové soub. systémy, do logických oddílů „volumes“. Vrstvu mezi nimi tvoří tzv. překladače („translators“), které tvoří logiku mezi daty a globálním jmenným prostorem. Je možné zvolit si typ organizace dat podobně jako u RAID, na úrovni „bricks“ je možné využít mirroringu či stripingu a různé možnosti replikace. Výhodou je také podpora HDFS či REST rozhraní, které umožňuje použít systém jako objektové úložiště. GlusterFS je plně podporován v OpenStack, je možné jej připojovat jako Cinder volume. OpenNebula jej také podporuje, avšak pouze v kombinaci s libvirt/KVM.

5 Výkonnostní testování

Jak již bylo řečeno, proces výkonnostního testování je značně závislý na provozním prostředí, způsobu testování a nemalém množství okolních proměnných. S jednoznačnými výsledky testování je také v rozporu heterogenní povaha obou systémů (je možné přidat zdrojová úložiště na úrovni fyzické, logické či souborové, různé RAID, apod.). Je třeba vybrat vhodné faktory, které mohou být ukazatelem výkonu souborového systému.

Nejprve je třeba se zamyslet nad tím, jak se systémy používají v praxi. V cloudovém prostředí virtualizujeme úložné kapacity formou obrazů uložených ve sdíleném úložišti. Tyto obrazy alokují kapacitu přiřazenou určitému virtuálnímu stroji a simulují jeho vnitřní úložiště. Veškeré I/O operace cestují od virtuálního stroje přes vrstvu soub. systému až do fyzických sfér úložiště. V globálním měřítku mohou však běžet z úložiště obrazů stovky až tisíce virt. strojů. Je potřeba zjistit, zda se s rostoucím počtem virtuálních stojů bude chovat systém dle specifikace výrobců, tj. jak bude vypadat propustnost, rychlost a tudíž i škálovatelnost systému. Pro měření propustnosti použijeme linuxový nástroj *„iozone“* v distribuovaném režimu.

Při testování je třeba uvažovat nejhorší možný případ, tj. běžící stroje využívající maximum svého I/O pásma; nástroj iozone dokáže takové prostředí simulovat. Kvůli snaze o zvýšení obecnosti výsledků testování však nebudeme uvažovat (pochopitelně) různé režie virtualizační platformy, ale simulovat běh strojů v cloudovém prostředí I/O operacemi interagujícími přímo se souborovým systémem ve více procesech na jeden uzel. Od takové informace a vzájemného srovnání je poté možné odrazit se k výběru jednoho z nich nebo k dalšímu testování. Propustnost při souběžném čtení/zápisu je tedy určujícím faktorem škálovatelnosti do počtu uzlů. Současný běh I/O operací ve více procesech na jeden uzel naopak simuluje hostování virtuálních strojů bez zkreslující vrstvy virtualizace.

Dalšími kandidáty na výkonnostní testování jsou také scénáře výpadků uzlů, práce vysoké dostupnosti, zjišťování vlivu různých parametrů systémů na rychlost a propustnost či změna parametrů testovacích nástrojů. Výsledky testů však budou uvedeny až po finálním shrnutí v doprovodné přednášce.

6 Závěr

Volba vhodného souborového systému pro cloudovou infrastrukturu není jednoduchý problém. Pokud zúžíme definiční obor na dva uvedené kandidáty, závisí volba na podobě vstupních požadavků či vhodných referencích. Oba systémy eliminují SPOF užitím decentralizovaných algoritmů pro umístění dat, využívají nejmodernějších přístupů k výpadkům provozu infrastruktury, výborně škálují

jak do počtu uzlů tak propustnosti síťové infrastruktury. Oba jsou určeny pro virtualizovaný provoz na heterogenní infrastruktuře a kompatibilní s uvedenými lídry v oboru cloudových middlewarů a virtualizačních platforem. Liší se v algoritmech pro decentralizovanou distribuci dat, architekturou či způsoby práce s daty (obrazy). Ceph je flexibilnější z manažerského pohledu kvůli možnosti výběru typu úložiště dle použití (blokový/souborový režim), naopak GlusterFS umožňuje variabilnější logickou manipulaci s daty na úrovni souborového systému. Cílem článku není výběr jednoho z nich, ale porovnání rozdílů a sestavení základního obrazu pro cloudového uživatele či administrátora vstupujícího do problematiky. Doplňkem může být také měření výkonu (propustnosti) obou systémů, které bude zveřejněno v doprovodné přednášce a může se stát podkladem pro volbu jednoho z nich. Měření jsou však infrastrukturně specifická.

Oba uvedené systémy jsou v současné době soupeři v oblasti cloudových úložišť. Jádrem problematiky je tedy rozdílný přístup k implementaci ze strany dvou majoritních skupin. V současné době není možné určit, která varianta je pro různá použití výhodnější, a to kvůli variabilitě heterogenních prostředí v cloudu a množství (ne)uvažovaných proměnných. Závěr je tedy ponechán na čtenáři. Je faktem, že i když jsou tyto systémy často v produkci stavěny na levnějším heterogenním hardwaru, poskytují oba relevantní variantu k současným proprietárním řešením. Doporučením autora je získání odpovědi na otázku vhodného soub. systému benchmarkingem obou souborových systémů na vlastní infrastruktuře v předpřipravených use-cases.

Literatura

[1] Offical Ceph documentation [online], http://www.ceph.com

[2] Offical GlusterFS documentation [online], http://www.gluster.org

[3] *CEPH: Reliable, Scalable, and high-performance distributed storage* [online], University of California, Sage A. Weil, December 2007.

[4] *Red Hat Storage Introduction to GlusterFS* [online], http://www.slideshare.net/Gluster

[5] *GlusterFS for SysAdmins* [online], Justin Clift, http://www.gluster.org/community/documentation/images/9/9e/Gluster_for_Sysadmins_Dustin_Black.pdf [online]

[6] *GlusterFS – Architecture & Roadmap* [online], Vijay Bellur, http://www.gluster.org/community/documentation/images/5/59/GlusterFS_Architecture_%26_Roadmap-Vijay_Bellur-LinuxCon_EU_2013.odp

OK

ZFS a BTRFS v praxi

Jan Krčmář

E-mail: honza801@gmail.com

Abstrakt

Už Vás unavuje řešit nafukování úložišť přes archaické nástroje a následné rozšiřování souborového systému, při kterém se musíte vždy pokřižovat? Pojďte to zkusit s modenými COW filesystémy. Jejich nasazením můžete získat stabilitu, škálovatelnost, redundanci a v neposlední řadě jednoduchou správu. Představíme si základní principy, pokročilé vlastnosti a některé výhody těchto souborových systémů, které mají oproti svým starším kolegům. Nakonec se zkusíme podívat, jestli jsou tyto modení přístupy opravdu v praxi použitelné a jak daleko je to s jejich implementací a vývojem.

Jan Krčmář vystudoval na Západočeské Univerzitě v Plzni obory Informatika a Kybernetika. Jeden z jeho hlavních koníčků je počítačová virtualizace. S tím je úzce spojen i jeho zájem o souborové systémy. Ve svém současném zaměstnaní, ČD Informační systémy, nasadil a již několik let úspěšně používá ZFS na operačním systému FreeBSD.

1 Úvod

Proč bychom se vůbec měli zajímat o nové souborové systémy? Žurnálovací systémy mají jeden velký problém. Starají se pouze o integritu a konzistenci metadat. Pokud bychom uvažovali nad tím, že budeme do žurnálu zapisovat i změny dat, narazíme na výkonostní problém. Oproti tomu Copy On Write (COW) představuje model pro ACID transakce, které jsou známé například ze světa databází, bez použití žurnálu. COW ve světě souborových systémů představuje jednu malou, ale velmi důležitou věc. Nikdy nepřepisuj data ani metadata, místo toho zapiš změny na volné místo na disku. Místo se uvolní až v případě, že jsou změny kopletní.

Tento model rovněž otevírá nové možnosti souborových systémů. Použití snapshotů se v tomto případě přímo nabízí. Pokud vyrobíme snapshot, postačí při každé změně zajistit, aby se stará data neuvolňovala, ale zůstala alokovaná pod jiným názvem.

Další novinkou těchto systémů je možnost rozložit data na více zařízení. K tomu využívá systém RAID modelů. Není tedy potřeba používat nástroje pro redundantní pole jako LVM nebo dmraid.

Poslední vlastností je podpora komprese, jejímž použitím se zvyšuje propustnost. Režie na kompresi a dekompresi se projeví na vytížení CPU.

Úkolem přednášky není kompletní popis COW file systémů. Představíme si základní a zajímavé vlastnosti a ukážeme některé rozdíly mezi ZFS a BTRFS.

2 Vlastnosti a správa ZFS

Tento systém byl navržen Sun Microsystems. Poprvé byl představen v roce 2005 v distribuci OpenSolaris. Je licencován CDDL, která není kompatibilní s GPL, pod kterou je vydáván Linux. Proto jsou jeho hlavní doménou operační systémy rodiny BSD a Solaris. ZFS byla původně zkratka pro Zettabyte File system, ale v současnosti již neznamená nic. Oproti většině file systémů používá proměnnou velikost bloku. Nepoužívá inody, ale různé struktury pro konkrétní operace (block-pointer, space-map, ...).

V názvosloví ZFS označujeme skupinu fyzických zařízení pro ukládaní dat jako pool. Do poolu můžeme zařízení přidávat a odebírat online. V poolu vytváříme další file systemy. Ty mají mountpoint, který defaultně odpovídá cestě v poolu. V ZFS můžeme také vytvořit volume, který je přístupný jako standardní zařízení.

ZFS vytvoříme tím, že vytvoříme pool.

```
[root@freebsd ~]# zpool create zpool /dev/ada0p3
[root@freebsd ~]# zpool status
  pool: zpool
 state: ONLINE
  scan: none requested
config:

        NAME        STATE     READ WRITE CKSUM
        zpool       ONLINE       0     0     0
          ada0p3    ONLINE       0     0     0

errors: No known data errors
[root@freebsd ~]# df
Filesystem  1K-blocks   Used   Avail Capacity  Mounted on
/dev/ada0p2   2031132 832404 1036240    45%    /
devfs               1      1       0   100%    /dev
zpool         8192947     31 8192916     0%    /zpool
```

Vytvořený pool je defaultně připojen v rootu.

File system vytvoříme příkazem zfs.

```
[root@freebsd ~]# zfs create zpool/test
```

Pro složitější příklad ukážeme defaultní rozdělení disku použitím automatického rozdělení – Automatic Root-on-ZFS, při instalaci FreeBSD. Velikost swapu jsme vybral při instalaci 1 GB. Po instalaci vypadají diskové oddíly takto.

```
[root@freebsd ~]# gpart show
=>      34  20971453  ada0  GPT  (10G)
        34      1024     1  freebsd-boot  (512K)
      1058   2097152     2  freebsd-swap  (1.0G)
   2098210  18873277     3  freebsd-zfs  (9.0G)

[root@freebsd ~]# zpool status
  pool: zroot
 state: ONLINE
  scan: none requested
config:

     NAME                                            STATE  READ WRITE CKSUM
     zroot                                           ONLINE    0     0     0
       gptid/f245d5b1-3350-11e4-88e0-a7ca1f524829 ONLINE    0     0     0

errors: No known data errors
```

Pool zroot se skládá z jednoho oddílu a jeho stav je v pořádku.

Další výpis zobrazuje informace o velikosti a volném místu v poolu.

```
[root@freebsd ~]# zpool list
NAME    SIZE  ALLOC   FREE    CAP  DEDUP  HEALTH  ALTROOT
zroot  8.94G   939M  8.02G    10%  1.00x  ONLINE  -
```

Podíváme se na jednotlivé file systemy.

```
[root@freebsd ~]# zfs list
NAME                 USED  AVAIL  REFER  MOUNTPOINT
zroot                939M  7.88G   144K  none
zroot/ROOT           887M  7.88G   144K  none
zroot/ROOT/default   887M  7.88G   887M  /
zroot/tmp            176K  7.88G   176K  /tmp
zroot/usr            576K  7.88G   144K  /usr
zroot/usr/home       144K  7.88G   144K  /usr/home
zroot/usr/ports      144K  7.88G   144K  /usr/ports
zroot/usr/src        144K  7.88G   144K  /usr/src
zroot/var           48.9M  7.88G  48.2M  /var
zroot/var/crash      148K  7.88G   148K  /var/crash
zroot/var/log        208K  7.88G   208K  /var/log
```

```
zroot/var/mail       144K  7.88G   144K  /var/mail
zroot/var/tmp        152K  7.88G   152K  /var/tmp
```

Rozdělění odpovídá myšlence ZFS. Založí se jeden velký oddíl a ten můžeme následně dále dělit.

Připojení file systemu na jiné místo provedeme nastavením parametru mountpoint

```
[root@freebsd ~]# zfs set mountpoint=/var/log zroot/var/log
```

Rozdělení poolu reflektuje standardní potřeby FreeBSD uživatele. Na jednotlivé datasety lze aplikovat kvóty a rezervace. Můžeme takto zaručit, že příliš obsáhlé logy nezasáhnou do provozu databáze. Kvóty a rezervace lze nastavit online.

```
[root@freebsd ~]# zfs set quota=2g zroot/var/log
```

Tato vlastnost je výhodou při použití jailů, kdy můžeme omezit obsazený prostor pro jednotlivé jaily.

```
[root ~]$ zfs list -o name,used,avail,refer,mountpoint,quota,reserv
NAME                   USED  AVAIL  REFER  MOUNTPOINT        QUOTA  RESERV
zpool                 2.44T   238G    31K  /mnt/zpool         none    none
zpool/jail            68.7G  41.3G    38K  /jail              110G    none
zpool/jail/backup     4.03G  41.3G  3.32G  /jail/backup       none    none
zpool/jail/honeypot    462M  41.3G   421M  /jail/honeypot     none    none
zpool/jail/ldapradius 11.7G  41.3G  10.9G  /jail/ldapradius   none    none
zpool/jail/snort      27.0G  41.3G  6.98G  /jail/snort         80G    none
zpool/jail/syslog     16.6G  41.3G  11.2G  /jail/syslog       none    none
zpool/jail/webs       8.83G  41.3G  2.26G  /jail/webs         none    none
zpool/ports           1.08G  2.92G  1.08G  /usr/ports           4G    none
zpool/tmp             20.3M  1004M  20.3M  /tmp                 1G      1G
zpool/usr             2.45G  3.55G  2.45G  /usr                 6G      6G
zpool/var             1.21G  2.79G  1.21G  /var                 4G      4G
```

Snapshoty se vytváří příkazem zfs.

```
[root ~]# zfs snapshot zpool/jail/snort@Wed
```

Pro vytváření periodických záloh využijme automatického skriptu. Vlastnoručně napsaného, nebo jako v tomto případě nástrojem zfSnap.

```
[root ~]# zfs list -t all
...
zpool/jail/snort                                27.0G 41.3G 6.98G /jail/snort
zpool/jail/snort@weekly-2014-07-12_05.42.00--2m    2.38G    -  6.86G  -
zpool/jail/snort@weekly-2014-07-19_06.03.00--2m    1.84G    -  6.88G  -
zpool/jail/snort@weekly-2014-07-26_05.57.00--2m    1.74G    -  7.11G  -
```

```
zpool/jail/snort@weekly-2014-08-02_06.26.00--2m    1.78G    -  7.12G  -
zpool/jail/snort@weekly-2014-08-09_06.39.00--2m    1.68G    -  7.12G  -
zpool/jail/snort@weekly-2014-08-16_06.13.00--2m    1.70G    -  7.13G  -
zpool/jail/snort@weekly-2014-08-23_06.30.00--2m    2.05G    -  7.13G  -
zpool/jail/snort@weekly-2014-08-30_06.24.00--2m    1.26G    -  7.03G  -
zpool/jail/snort@Wed                                  0    -  6.98G  -
...
```

Ze snapshotu lze vytvořit proud dat, který můžeme uložit do souboru pro účely zálohování. Snapshot lze také odeslat na vzdálený systém, kde může být připraven jako záloha pro případ výpadku.

```
[root ~]# zfs send zpool/jail/snort@Wed | ssh backup zfs recv spool/backup
```

Systém implementuje Self-Healing vlastnost použitím techniky RAID-Z. Jedná se o RAID5 s COW transakcemi. Kdykoliv ZFS zjistí poškození dat nebo metadat špatným kontrolním součtem, může blok opravit pomocí správného bloku na redundantním disku.

Pro ušetření místa na disku můžeme zapnout vlastnost deduplikace. Pokud systém při zápisu na disk zjistí, že zapisovaný blok již existuje, neuloží ho, ale vytvoří pouze ukazatel na existující blok.

V dokumentaci k ZFS na FreeBSD stojí, že některé vlastnosti ZFS jsou velice náročné na paměť RAM. Obecně se doporučuje 1 GB RAM na 1 TB storage. Pokud použijeme vlastnost deduplikace doporučuje se až 5 GB RAM na 1 TB dat.

Komprese je ve výchozím nastavení pouze pro metadata. Typ komprese můžeme vybrat z možností lzjb, gzip, gzip-[1-9], zle, lz4. Šifrování je v beta stádiu.

3 Vlastnosti a správa BTRFS

BTRFS je mladším kolegou ZFS. Stejně jako ZFS se snaží o implementování rozšířených vlastností s důrazem na toleranci chyb, opravy a jednoduchou administraci. Název může mít několik významů: BetterFS, BtreeFS, ButterFS, ButterFace. Nemá tak vysoké cíle jako ZFS. Chce být stejně tak dobré nebo dokonce lepší než stávající Linuxové systémy souborů (ext4, ReiserFS, XFS). Důraz klade na škálovatelnost a výkon. Používá pevnou velikost bloků, které seskupuje do extentů. Extenty zajistí nižší fragmentaci dat, ale mají i své nevýhody. Například při kompresi je nutné přečíst celý extent. Pro ukládání datových struktur se používají optimalizované b+stromy. Transakce v BTRFS porušují některé ACID pravidla (atomicity, consistency). Oproti ZFS se snaží co nejvíce využít funkcí jádra (io scheduler, cache, komprese). Na stránkách BTRFS se můžeme dočíst, že návrh systému již není nestabilní, ale kód prochází překotným vývojem. Není doporučeno jeho použití pro produkční účely. BTRFS je licencován GPL, tedy

stejně jako Linux, což je možná jeden z důvodů, proč je vůbec tento systém vyvíjen.

V názvosloví BTRFS se skupina zařízení pro ukládání dat označuje jako filesystem. Do filesystemu můžeme přidávat a odebírat zařízení online. Ve filesystemu vytváříme subvolumy.

Aby systém věděl, kde se BTRFS filesystemy nacházejí, použije se skanování.

```
/mnt/btrfs # btrfs device scan
Scanning for Btrfs filesystems
```

Pro vytvoření filesystemu použijeme standardní utilitu mkfs. Při použití více zařízení, jsou v základním nastavení metadata v RAID1 a data RAID0. Rozložení dat a metadat se může nastavit parametry mkfs při vytváření, nebo později změnit příkazem btrfs. Při vytváření systému může být v parametru mkfs.btrfs více zařízení.

```
/ # mkfs.btrfs -L btrfs_data -d raid1 -m raid1 /dev/sda3 /dev/sdb2 /dev/sda2
/ # mount /dev/sda3 /mnt/btrfs
```

Vypsání dostupných BTRFS filesystemů provedeme příkazem btrfs.

```
/mnt/btrfs # btrfs filesystem show
Label: 'btrfs_data'  uuid: b4793685-b88a-4480-8869-437f71554959
        Total devices 3 FS bytes used 267.15GiB
        devid    2 size 74.61GiB used 2.00GiB path /dev/sda3
        devid    4 size 461.95GiB used 293.03GiB path /dev/sdb2
        devid    5 size 387.34GiB used 291.00GiB path /dev/sda2

Btrfs v3.14.2
```

Vidíme systém s názvem btrfs_data, obsahující 3 diskové oddíly.

Rozložení dat ve filesystemu.

```
/mnt/btrfs # btrfs fi df .
Data, RAID1: total=291.00GiB, used=266.10GiB
System, single: total=32.00MiB, used=52.00KiB
Metadata, RAID1: total=2.00GiB, used=1.05GiB
unknown, single: total=360.00MiB, used=0.00
/mnt/btrfs # df -h .
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       924G  535G  243G  69% /mnt/btrfs
```

Pro ukládání dat a metadat se použije systém RAID1. Příkaz df zobrazuje volné místo na disku netradičně, oproti zvyklostem. Velikost je součet všech zařízení ve filesystemu. Volné resp. použité místo se mění v závislosti na použitém RAID modelu.

Zařízení odebereme příkazem btrfs.

```
/mnt/btrfs # btrfs dev delete /dev/sda3 .
/mnt/btrfs # btrfs fi sh
Label: 'btrfs_data'  uuid: b4793685-b88a-4480-8869-437f71554959
        Total devices 2 FS bytes used 272.93GiB
        devid    4 size 461.95GiB used 293.03GiB path /dev/sdb2
        devid    5 size 387.34GiB used 293.00GiB path /dev/sda2

Btrfs v3.14.2
```

Data se automaticky rozloží na zbylé disky.

Vytvoření subvolumu provedeme příkazem btrfs.

```
/mnt/btrfs # btrfs subvolume create tmp
Create subvolume './tmp'
/mnt/btrfs # btrfs su list .
ID 256 gen 692121 top level 365 path var/cache/pacman
ID 257 gen 692148 top level 365 path var/lib/mysql
ID 270 gen 699161 top level 5 path usr
ID 317 gen 691332 top level 5 path opt
ID 365 gen 699174 top level 5 path var
ID 497 gen 699174 top level 5 path home
ID 499 gen 699144 top level 497 path home/virts
ID 1099 gen 699173 top level 5 path tmp
```

Pokud připojujeme BTRFS filesystem, je jedno, které zařízení použijeme jako parametr pro příkaz mount. Nový systém obsahuje pouze jeden subvolume (subvolid=5). Ten je implicitně nastaven jako výchozí pro mount. Výchozí subvolume se nastaví příkazem btrfs. Pro připojení subvolumu na jiné místo, než jaké má ve struktuře filesystemu přidáme příkazu mount parametr subvol, nebo subvolid.

```
/mnt/btrfs # mount /dev/sda3 /usr/home -o subvol=home
```

Kvóty jsou v BTRFS implicitně vypnuty. Jejich zapnutí se provede příkazy btrfs.

```
/mnt/btrfs # btrfs quota enable .
/mnt/btrfs # btrfs qu rescan .
```

Každý subvolume je zařazen do qgroup. Limit lze nastavit přímo nad subvolumem.

```
/mnt/btrfs # btrfs qgroup limit 1g opt
```

Snapshot vytvoříme analogicky jako subvolume.

```
/mnt/btrfs # btrfs su snapshot opt/ opt2
Create a~snapshot of 'opt/' in './opt2'
/mnt/btrfs # ls
home  opt  opt2  tmp  usr  var
```

Snapshoty jsou v BTRFS prosté subvolumy, které sdílí společná data s jiným subvolumem. Defaultně jsou read-write.

Výpis qgroup pak vypadá následovně.

```
/mnt/btrfs # btrfs su li .
...
ID 1105 gen 712160 top level 5 path opt2
/mnt/btrfs # btrfs qg sh -r .
qgroupid rfer         excl
-------- ----         ----
...
0/317    889212928    4096         1073741824
...
0/1105   889212928    4096         0
```

Ve sloupcích je název qgroup, zabrané místo odkazované a vlastní, limit.

Pokud jsme vytvořili systém pouze na jednom zařízení, není zajištěna redundance. Toho docílíme přidáním dalšího zařízení. Samotná operace přidání ale nestačí k tomu, aby se data na nově přidané zařízení rozprostřela. Pro tento účel existuje příkaz balance. Další jeho funkcí je konvertovat rozložení dat (single, RAIDx).

```
/mnt/btrfs # btrfs dev add /dev/sda3 .
/mnt/btrfs # btrfs balance start -dconvert=raid0 -mconvert=raid1 .
```

Oproti ZFS je podporováno online změna velikosti file systému. Nároky na použití jsou také nižší. Doporučuje se 1 GB místa na disku a 128 MB RAM.

Pro kontrolu dat na disku má btrfs příkaz scrub. Ten přečte všechna data a metadata a použije kontrolní součty pro odhalení chyb. Pro opravy použije redundantní kopie z RAID zařízení.

```
/mnt/btrfs # btrfs scrub start .
/mnt/btrfs # btrfs sc status .
scrub status for b4793685-b88a-4480-8869-437f71554959
  scrub started at Thu Sep 4 16:28:51 2014 and finished after 276 seconds
  total bytes scrubbed: 31.91GiB with 0 errors
```

Kompresní algoritmus můžeme volit lzo, rychlý a lepší pro SSD, nebo zlib, pomalejší s lepší kompresí.

4 V praxi...

ZFS používám již asi 2 roky na několika serverech. Během této doby jsem se nesetkal s žádným problémem. Jeho nasazením jsem získal možnost inkrementálního zálohování a jednoduše nastavitelných kvót.

ZFS jsem testoval také na portu Debian GNU/kFreeBSD. Ani zde se neobjevily žádné problémy.

BTRFS používám pouze na své pracovní stanici. Při standardních operacích jako přidávání, odebírání subvolumu, pořizování snapshotu proběhlo vždy vše v pořádku. Problém nastal, když jsem se snažil rozšířit filesystem o další zařízení. Při operaci balance vzniknul kernel panic. Po prohledání zdrojových kódů a krátkou komunikací s vývojaři se zjistila příčina problému. Commit, který chybu opravoval nebyl včas začleněn do hlavní větve linuxového kernelu, což bylo napraveno v následující verzi. Poté proběhl balance již v pořádku.

Provedl jsem také testy zápisu do obrazu virtuálního stroje (kvm). Pro porovnání jsem testoval obraz umístěn na ext4 a btrfs. Zapisovány byly soubory velikosti 20 MB až 2 GB. Průměrná rychlost zápisu v případě ext4 byla 63 MB/s, v připadě BTRFS 73 MB/s. Měření bylo provedeno na hostu s jádrem 3.16.1-1-ARCH, Btrfs v3.14.2.

Zátěžové testování aneb „Sakra, kdo to bude platit“

Ján Hrabik

E-mail: jhrabik@trask.cz

Zátěžové testování je jednou z velmi důležitých částí procesu testování. Bohužel ale také nejčastěji opomíjenou. V rámci IT o zátežovém testování panuje velké množství bludů a omylů, které pak ovlivňují rozhodování odpovědných manažerů o provedení zátěžového testu. Nejčastěji se jedná o informace spojené s náklady na provedení ZT a informace spojené s finančními dopady provedení, případně neprovedení, zátěžového testu.

Zátěžové testování

Zátěžové testování je specifickým oborem testování, podobně jako unitové testování, integrační testování nebo bezpečnostní testování. Jako specifická forma testování je opředena mýty a polopravdami, které mohou mít kardinální dopad na vykonání zátěžového testu jako takového, případně na projekt samotný. Mezi nejčastější mýty o ZT patří tvrzení, že „ZT je drahý“. Dalším velmi rozšířeným omylem je názor: „Na ZT potřebuji specialisty, kteří jsou drazí“ a jeho protipólem je skoro stejně nebezpečný mýtus „Proč bych měl za ZT platit, když se to dá udělat zadarmo“. Velmi často je z projektové praxe zmiňován i mýtus „Tady nikdo ZT nepotřebuje, nevidím důvod ho dělat“ a jeho opačný extrém „ZT musíme udělat stůj co stůj, i když nám na systém přistupuje jen jeden uživatel za týden“. Pojďmě si tyto mýty rozebrat podrobněji.

„ZT je drahý“

Velmi často u PM převládá názor, že ZT je drahý, a že jelikož je drahý, tak je ZT velmi často vyřazen ze scope projektu, protože na něj většinou není rozpočet a tím se ušetří peníze na „něco užitečnějšího, například na programátory“. Základem tohoto mýtu je zažitá představa, že velké ZT byly v počátcích informatizace opravdu velmi nákladné a toto přesvědčení přežilo do současnosti. Je potřeba uvést několik zásadních informací:

a) ZT z principu být drahý nemusí

b) ZT vyžaduje přípravu, která se může prodražit

c) ZT většinou vyžaduje nástroj, který se může prodražit

d) ZT většinou potřebuje specialisty, kteří se mohou prodražit

e) ZT potřebuje testovací prostředí, které se také může prodražit

To, co opravdu může výrazně zátěžový test prodražit, je v drtivé většině případů špatná příprava ZT. Lidé, kteří jsou za ZT zodpovědní , jsou velmi často v oblasti ZT neproškolení a nemají potřebné penzum informací, znalostí a zkušeností k tomu, , aby mohli učinit správné rozhodnutí o ZT vzhledem k okolnostem projektu. Velmi často je prováděním ZT pověřen programátor, který nemá žádné znalosti v oblasti designu ZT, byl kolegy nabriefován, že existují nějaká sharewarová řešení pro ZT a v rámci svého programátorskéo zápalu dojde k přesvědční, že to je„triviální“ a že když zvládá Javu, tak bez problémů zvládne jakýkoliv shareware na ZT. V tomto případě dojde k podcenění přípravy, která pak má přímé dopady do výběrů nástroje, potřeb na specialisty a přípravy prostředí. Tyto nechtěné chyby pak mohou relativně jednoduchý a levný ZT prodražit do rozměrů, kdy může hrozit až selhání ZT, navíc spojené s velkými náklady. Druhým extrémem pak bývá manažer, který o ZT ví pouze to, že existuje a který je ochotný uvěřit čemukoliv, ale pouze v případě, že to bude dostatečně levné, nebo ideílně zadarmo. Z tohoto pohledu je pak projektovým zázrakem sledovat proškoleného PM, který se dokáže orientovat v problematice ZT a který deleguje provedení ZT na specialistu s dostatečnou znalostí problematiky. V tomto případě dokáže pak ZT dosáhnout dobrých výsledků za použití minimálních prostředků.

„ZT není potřeba/ZT musí být“

Výrazně horším případem projektového mýtu o ZT je situace „ZT není potřeba/ZT musí být“. Celý mýtus je postavený na direktivním určení PM o provedení činnosti, případně její neprovedení bez ohledu na skutečnost, zda je ZT pro projekt přínosný, nebo ne. Nejčastějším argumentem PM pro „ZT není potřeba“ je věta „děláme přece věc, kterou bude používat jen několik málo lidí, tak proč dělat ZT“. Tato logika vychází ze zažitého stereotypu, že když je něco pomalé, tak se to nasadí na rychlejší HW, nebo se zvedne počet serverů, které aplikaci/web obsluhují a „ono“ se to zrychlí. Z praxe lze uvést mnoho příkladů, kdy povýšení HW nejen že nepřineslo zrychlení aplikace, ale dokonce ad absurdum vedlo ke zhoršení chování. Základem tohoto omylu je neexistence ověřitelné analýzy pro ZT na projektu. V rámci analýzy by totž PM zjistil, zda ZT potřebuje, jak velký, jak obsáhlý a hlavně by musel definovat akceptační kritéria

vyhodnocení výsledků ZT. Je s podivem, že většina PM má pro své projekty akceptační kritéria na kde co, ale akceptační kritéria pro ZT většinou neexistují, nebo jsou tak vágně zadaná, že jsou nejasná, zkreslená, nebo přímo technologicky nesplnitelná. Z pohledu PM je ale provedení analýzy otázka cca dvou MD pro specialistu – analytika ZT. Protikladem nepotřebnosti ZT je vynucování i zátěžového testu v případech jeho naprosté zbytečnosti. Zde je většinou ze strany PM argumentováno, že má ZT v akceptačních kritériích, tak ho prostě udělat musí. PM si v tomto okamžiku neuvědomuje, že je chycen do podobné pasti, jako v případu „ZT není potřeba". I zde má pravděpodobně velmi špatně definovaní akceptační kritéria pro ZT. V tomto případě je PM navíc „nucen" projektem k vykonání činností podle nesmyslného zadání. Řešením tohoto problému je opět provedení test analýzy nad ZT. Výsledek pak může být pro PM i vcelku šokující – například zjištění, že ZT opravdu nepotřebuje, nebo že hodnoty dané v akceptačních kritériích může splnit například už v SIT (system integration tests) a nemusí konstruovat speciální ZT.

Mezi nejčastěji nesprávně zadaná akceptační kriteria pro ZT patří následující:

- požadovaná odezva aplikace je delší než 10 vteřin (toto je hranice, kdy by měla aplikace uživatele informovat, že načítání bude trvat déle a zobrazit mu informační lištu s progresem načítání – toto může být paradaxně velmi velký požadavek do vývoje s dopadem na pracnost programování).

- požadovaná odezva do 1 sekundy (v realné třívrstvé architektuře jsou časy pod 1 s pro weby buď zcela nedosažitelná , nebo těžce dosažitelná kvůli velikosti přenášeného obsahu)

- požadovaný počet současně obsluhovaných konexí větší než 100 tisíc (pro globální řešení možná, ne však prostředí ČR, zde byl jeden z největších požadavků za posledních 10 let 50 000 konexí. ZT s takovou dostupností se již řeší velmi výrazně odlišně a jsou HW oříškem, nemluvě o možnostech propustnosti linek v ČR)

- požadovaný počet zároveň obsluhovaných konexí je stonásobně větší, než jaký je reálný počet konexí do DB (není výjimečné, že některé DB jsou nastaveny na 5, případně 20 konexí, a je od nich očekáváno, že se budou chovat jako výrazně dražší DB, které mohou poskytnout 500 nebo 1 000 konexí).

Z předchozích řádek je patrné, že většina problémů se ZT pramení z nedostatečné zkušenosti lidí odpovědných za provádění ZT a jejich pocitu, že to je „přece naprosto triviální" alternativně „je to raketová věda, kterou nikdy nepochopím". Z reálné praxe mohu říct, že pokud si PM nechá poradit, nebývá realizace ZT problém, případně je ZT zařazen do seznamu projektových rizik a je s ním jako takovým pak příslušně procesně nakládáno.

SQL včera a dnes

Petr Jiroušek

E-mail: petr@civ.zcu.cz

Informace (z lat. in-formatio, utváření, ztvárnění) je velmi široký, mnohoznačný pojem, který se užívá v různých významech. V nejobecnějším smyslu je informace chápána jako údaj o prostředí, jeho stavu a procesech v něm probíhajících. Informace snižuje nebo odstraňuje neurčitost (entropii) systému (např. příjemce/uživatele informace). Množství informace lze charakterizovat tím, jak se jejím přijetím změnila míra neurčitosti přijímajícího systému
Databáze (neboli **datová základna**) je určitá uspořádaná množina informací (dat), uložená na paměťovém médiu. V širším smyslu jsou součástí databáze i softwarové prostředky, které umožňují manipulaci s uloženými daty a přístup k nim. Tento software se v české odborné literatuře nazývá systém řízení báze dat (SŘBD). Běžně se označením databáze – v závislosti na kontextu – myslí jak uložená data, tak i software (SŘBD).
SQL (někdy vyslovováno anglicky es-kjů-el, někdy též síkvl) je standardizovaný dotazovací jazyk používaný pro práci s daty v relačních databázích. SQL je zkratka anglických slov Structured Query Language (strukturovaný dotazovací jazyk).

Zdroj: http://cs.wikipedia.org/

Na úvod

Život je plný nejrůznějších informací. A tak dříve nebo později každý pocítí potřebu informace uchovávat, ale hlavně umět v nich vyhledávat a to ještě pokud možno rychle a přesně. Lidská paměť postupem doby na nával informací přestala stačit a tak přišla ke slovu různá „umělá“ úložiště. Nastaly doby kamenných destiček, papyrových svitků či knih. Ale pojďme dále, nechme uplynout pár tisíciletí a nechme lidstvo vynalézt první počítač – ideální přístroj na uchovávání informací. Vejdou se jich tam kilobajty, co kilogajty, megabajty, gigabajty, terabajty ... vznikají nám první databáze. Ale jak teď rychle najít to, co potřebuji? Je potřeba najít nějaký jazyk, kterým se s počítačem domluvím, který bude jednoduchý na pochopení, jednoduchý na zpracování a vždy mi rychle přihraje přesně tu informaci, kterou právě teď potřebuji. A světe div se, takový jazyk se opravdu našel...

Historie databází

Historii databází asi můžeme začít psát s letopočtem 1890. V tomto roce byl poprvé komerčně použit první elektromechanický stroj, který dokázal zpracovávat děrné štítky. Byl vytvořen kvůli prvnímu sčítání lidu v USA a jeho autorem byl Herman Hollerith. Díky úspěchu byly jeho stroje hned v následujícím roce použiti pro sčítání v Kanadě, Norsku a Rakousku. Další využití našly u řady dopravních společností. Elektromechanické stroje v různé podobě se pak využívaly pro účely zpracování dat další půlstoletí.

V roce 1911 po spojení firmy Hermana Hollerita se třemi dalšími vzniká firma dnes známá pod názvem International Bussines Machines. Tato firma byla i u dalšího velkého projektu, o kterém by se dalo říci, že byl databázový, a to u zakázky vlády USA na evidenci Social Security Act (obdoby našeho rodného čísla) tehdejších obyvatel USA v roce 1935. IBM vytvořila pro zpracování podobných úloh nové zařízení. Byl to první digitální počítač pro komerční využití, tzv. UNIVAC I., založený na státem podporovaném projektu Electronic Discrete Variable Automatic Computer (EDVAC) z University of Pennsylvania.

Za další mezník bychom asi mohli považovat představení prvního datového skladu Charlesem Bachmanem z General Electric v roce 1961. Byl nazván IDS (Integrated Data Store), obsahoval virtuální paměťovou databázi s příkazy, které bychom podle dnešních měřítek již mohli označit za příkazy dnešního SQL jazyka. Pro objasnění funkčnosti tohoto skladu používal Bachman struktury, které bychom dnes přirovnali k datovému modelu. Na počátku šedesátých let byl Bachman též u založení seskupení Codasyl, které postupně začalo publikovat základní specifikace pro programovací jazyky. V roce 1965 publikovalo specifikaci jazyka COBOL a roce 1968 pak jeho rozšíření právě pro datové báze. COBOL se po mnoho dalších let stal nejrozšířenějším jazykem pro hromadné zpracování dat. V roce 1969 pak uskupení publikovalo první specifikaci jazyka pro tzv. síťový databázový model známý pod označením Codasyl Data Model. De facto se jednalo o první definici dvou skupin příkazů: DDL pro definování schématu databáze a DML pro manipulaci s daty. DML příkazy se staly přímo součástí jazyka COBOL. V roce 1971 se vývoj jazyka COBOL a DML příkazů odděluje a pro každou větev vzniká nové uskupení, které řídí její další vývoj.

Na bázi Codasyl Data Model začínají postupně vznikat první implementace databázových modelů např. počítače Univac od firem Eckert-Mauchly Computer Corporation, Honeywell Incorporated, Siemens AG a pro mikropočítače od Digital Equipment Corporation (DEC) a Prime Computer. Ve stejné době byly vyvíjeny i hierarchické databáze. Jednou z prvních implementací hierarchických databází byl systém IMS, který byl vyvinut firmou IBM pro program letu na Měsíc – Apollo. Pracoval na počítači System/360. Ale vraťme se opět k tzv. relačním databázím a jazyku SQL.

Jako autora jazyka pro práci s databázemi můžeme asi označit Edgara Franka „Teda“ Codda. Ten byl v této době zaměstnancem IBM a přišel s návrhem začít

pro práci s daty používat srozumitelné výrazy pocházející z běžné angličtiny, tedy ne relativně nesrozumitelné zkratky běžné u tehdejších programovacích jazyků. Jeho koncepce vycházela z ukládání dat do tabulek a tím i nezávislosti na konkrétním použitém hardware a konkrétním fyzickém způsobu uložení dat. Pro vlastní přístup k datům předpokládal neprocedurální jazyk s možností pracovat nejen s jedním konkrétním záznamem, ale s celou definovanou množinou dat. Tento koncept pak v roce 1970 představil v článku „A Relational Model of Data for Large Shared Data Banks“. Odtud patrně pochází termín relační databáze.

Zpočátku Ted Codd neměl u IBM příliš pochopení, IBM viděla tento projekt jako příliš „intelektuální“ a hlavně to v té době nekorespondovalo s jejich hlavním produktem – IMS. Codd proto publikoval svůj návrh i mimo IBM. Nakonec vznikly dva projekty relačních databází, a to System-R v rámci IBM a od roku 1972 projekt Ingres na University of California at Berkeley (UC-Berkeley) s podporou armády a National Science Foundation (NSF).

Historie SQL

Počátky SQL jazyka se datují někde kolem roku 1974. Jeho první implementací byl právě Systém-R. Dostupné prameny neříkají zcela jasně, proč se na první verzi SQL nepodílel Tedd Codd, ale pravděpodobně v té době nebyl u vedení IBM právě v oblibě a proto jsou jako autoři první verze SQL uváděni Donald D. Chamberlin a Raymond F. Boyce. První verze se jmenovala SEQUEL (Structured English Query Language). IBM zpočátku relační databáze a SQL příliš nepreferovala, stále se zaměřovala spíše na svou vlajkovou loď – IMS. Průlom nastal až koncem sedmdesátých let, kdy se hybatelem věcí stala konkurence. V té době firma Relational Software, Inc. (dnes Oracle Corporation) využila potenciálu konceptu SQL popsaného Coddem, Chamberlinem a Boycem a vyvinula vlastní relační databázový systém na bázi SQL. V roce 1979, Relational Software, Inc. Uvedlo na trh první vlastní implementaci SQL – OracleV2 (Version2) pro počítače VAX. Teprve následně uvedlo svou vlastní implementaci SQL na trh i IBM. Jednotlivé verze byly pojmenovány System/38, SQL/DS a DB2. Jednotlivé verze se dostaly na trh v letech 1979, 1981 a 1983. Dalšími známými SQL systémy byly např. Ingres (dnes Progres), Informix a SyBase. Ve všech těchto systémech se používala varianta jazyka SEQUEL, který byl později přejmenován na SQL. Údajně to bylo zejména z toho důvodu, že ochrannou známku SEQUEL měla zaregistrovanou letecká společnost UK-based Hawker Siddeley.

SQL standardy

V roce 1986 ISO a ANSI publikují počáteční standard SQL, v roce 1989 ISO publikuje dodatek Integrity Enhancement Feature a začíná se hovořit o SQL89. Obsahem jsou zejména všechny příkazy a výrazy jazyka SQL. V roce 1992 do-

chází k první úpravě tohoto ISO standardu, který je pak znám jako SQL2 nebo SQL92. Tato verze SQL standardu je dnes naprostým základem pro všechny databáze, a pokud je jakýkoliv projekt prezentován jako SQL, pak je prakticky vyloučené, aby tomuto standardu neodpovídal. Tento standard opět obsahuje zejména jednotlivé SQL příkazy a výrazy. Postupně byl rozšířen o další části, zejména o SQL interface a v roce 1999 vzniká standard SQL3. Tento standard pak doznal ještě dalších rozšíření (v roce 2006 XML). Dnes je aktuálně nejnovější standard z roku 2011. SQL standardy sice podporuje prakticky každá relační databáze, ale obvykle nejsou implementovány vždy všechny požadavky normy. A naopak, každá z nich obsahuje prvky a konstrukce, které nejsou ve standardech obsaženy. Přenositelnost SQL dotazů mezi jednotlivými databázemi je proto velmi omezená.

V tab. 1 je vidět přehled SQL standardů od roku 1986 dodnes.

Tab. 1

Rok	**Název**	**Alias**	**Poznámka**
1986	SQL-86	SQL-87	První formalizace ANSI.
1989	SQL-89	FIPS127-1	Přidána integritní omezení
1992	SQL-92	SQL2, FIPS 127-2	Zásadní celková revize, vzniká ISO 9075 (existuje dodnes),
1999	SQL:1999	SQL3	Regulární výrazy, rekurzivní dotazy, triggery, procedurální rozšíření, objektově orientované funkčnosti, java
2003	SQL:2003	SQL 2003	SQL/XML, sekvence, sloupce s automaticky generovanou hodnotou
2006	SQL:2006	SQL 2006	XML – import, datové typy, funkce pro práci s nimi, XQuery
2008	SQL:2008	SQL 2008	Rozšíření order by, instead of triggery, truncate
2011	SQL:2011		„Časová“ (temporal) data, delete v merge, nové analytické funkce

V tab. 2 pak rozdělení standardu na jednotlivé části. Aktuálně je SQL standard označován jak ISO/IEC 9075 standard. Za pomlčkou následuje číslo příslušné části a za dvojtečkou pak rok poslední revize, např. ISO/IEC 9075-1:2011 je poslední verze (z roku 2011) části pojednávající o části 1 – SQL/Framework.

Dalším standardem, který na výše uvedený ISO/IEC 9075 přímo navazuje, je ISO/IEC 13249 *SQL Multimedia and Application Packages*. Tento standard definuje speciální datové typy a jejich návaznosti na SQL:

- ISO/IEC 13249-1 Part 1: *Framework*
- ISO/IEC 13249-2 Part 2: *Full-Text*
- ISO/IEC 13249-3 Part 3: *Spatial*

Tab. 2

Jednotlivé části SQL standardu	
část	**Obsah**
Part 1 – SQL/Framework	Základní model, informace společné všem částem (jediný je public)
Part 2 – SQL/Foundation	Jednotlivé elementy jazyka SQL
SQL/OLAP	Funkce a operace pro OLAP (Online Analytical Processing)
Part 3 – SQL/CLI	SQL interface (ODBC)
Part 4 – SQL/PSM	Uložené procedury, externí procedury, procedurální rozšíření
Part 5 – SQL/Bindings	Embedded SQL
Part 6 – SQL/Transaction	XA (distribuované) transakce
Part 7 – SQL/Temporal	Rozšíření SQL o časově orientované typy a funkce.
Part 8 – SQL/Objects Extended Objects	Objektový model
Part 9 – SQL/MED	Zpracování externích dat
Part 10 – SQL/OLB	Vazba SQL na Javu, JDBC
Part 11 – SQL/Schemata	Definice SQL schémat
Part 12 – SQL/Replication	Replikace SQL dat – replikační schémata, pravidla, řešení konfliktů
Part 13 – SQL/JRT	Java procedury a typy (Persistent Stored SQLJ)
Part 14 – SQL/XML	SQL a XML

- ISO/IEC 13249-5 Part 5: *Still image*
- ISO/IEC 13249-6 Part 6: *Data mining*
- ISO/IEC 13249-8 Part 8: *Metadata registries (MDR)* (ve fázi návrhu)

Postupný vývoj jednotlivých částí je pak naznačen v tab. 3.

Na závěr

Svět je dnes plný databází. S trochou fantazie můžeme databází nazvat seznam kontraktů v našem mobilním telefonu, seznam kanálů v naší televizi, seznam skladeb v našem mp3 přehrávači. Databáze jsou dnes doslova na každém kroku a tak se nelze divit, že potřeba jednotného jazyka, kterým se s databází domluvíme, je veliká. Výše uvedené standardy tedy odráží potřebu dnešní doby a i na jejich vývoji lze vypozorovat jisté trendy a náznaky, kam asi se databázový svět ubírá a v budoucnu ubírat bude.

Tab. 3

Část	**SQL 1992**	**SQL 1999**	**SQL 2003**	**SQL 2008**	**SQL 2011**
1. SQL/Framework		√	√	√	√
2. SQL/Foundation	√	√	√	√	√
SQL/OLAP		Příloha k SQL/ Foundation	Sloučeno se SQL/ Foundation		
3. SQL/CLI	√ (1995)	√	√	√	
4. SQL/PSM	√ (1996)	√	√	√	√
5. SQL/Bindings		√	Sloučeno se SQL/ Foundation		
6. SQL/Transactions		Projekt zrušen pro nepotřebnost			
7.SQL/Temporal			Projekt zrušen pro nezájem		
8. SQL/Objects		Sloučen se SOL/ Foundation			
9. SQL/MED		√	√	√	
10. SQL/OLB	Samostatný standard	√	√	√	
11. SQL/Schemata			√	√	√
12. SQL/Replication			Projekt zrušen pro nezájem		
13. SQL/JRT		Samostatný standard	√	√	
14. SQL/XML			√ (2006)	√	√

Seznam použitých zdrojů

http://en.wikipedia.org/wiki/SQL

http://www.jcc.com/resources/sql-standards

http://www.root.cz/clanky/historie-relacnich-databazi

http://www.fi.muni.cz/usr/jkucera/pv109/2004/xbukovsk.htm