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

51. konference
Sborník příspěvků

U Jeňoura, Nechory – Prušánky
8.–11.října 2017

Sborník příspěvků z 51. konference EurOpen.CZ, 8.–11. 10. 2017

Plzeň 2017. První vydání.
Editor: Zdeněk Šustr
Sazba a grafická úprava: Zdeněk Šustr, vydavatelství, Praha
e-mail: `vydavatelstvi@sustr.net`

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

Příspěvky neprošly redakční ani jazykovou úpravou.

ISBN 978-80-86583-32-7

Obsah

Distribuovaná DC/VI infrastruktura CESNETu

Martin Pustka

E-mail: Martin.Pustka@vsb.cz

Geograficky distribuovaná DC/VI infrastruktura CESNETu byla budována jako technický celek, který se skládá z jednotlivých komponent – LAN i SAN sítě, serverových systémů, diskových úložišť virtualizační technologie.

Byla budována tak, aby do ní mohly být zapojeny fyzické systémy, virtualizované technologie, a byla připravena na možná rozšíření, která vyžadují propojení různých sítí a aplikací. V současné době je provozována ve dvou lokalitách – Praze a Brně. Podporovány jsou i živé migrace virtualizovaných systémů mezi oběma lokalitami a to bez nutnosti jakýchkoliv změn na straně provozovaných systémů.

Co DC/VI zajišťuje

Na první pohled je nejčastěji využívanou službou klasická virtualizace serverových systémů. Tato služba je využívána zejména pro interní potřeby CESNETu, v menším rozsahu je využívána externími subjekty, členy nebo zákazníky CESNETu.

Z provozního pohledu je naším primárním cílem stabilita celého provozu, zejména proto, že na této infrastruktuře běží provozní služby, které jsou využívány větším množstvím uživatelů. Další poskytované služby jsou:

- připojení fyzických serverů (standalone nebo v UCS blade server),
- provoz virtuálního serveru, vč. diskových kapacit,
- zálohování, monitoring a notifikace,
- anycastové služby.

Jak jsme začínali . . .

Při realizaci v květnu roku 2016 jsme byli omezeni použitím v minulosti již pořízených technických prvků. Tyto technické prostředky jsme převzali, nicméně díky ne úplně optimální konfiguraci jsme museli započít s budováním téměř od počátku.

Na finální řešení jsme měli tyto technicko-koncepční požadavky:

- převzetí správy všech prvků infrastruktury,
- využití 10GE technologií, s výhledem na nasazení ještě vyšších kapacit (25/40GE),
- virtualizace serverové infrastruktury,
- integrace fyzických serverů,
- zajištění minimálně redundance N+1 v každé technické vrstvě,
- využití diskových úložišť přes IP,
- požadavek na technicko-geografické rozdělení DC technologií do min. dvou center,
- vyčlenit DC/VI z páteřní sítě CESNETu, udělat z infrastruktury „zákazníka“ páteřní sítě,
- rozšiřování kapacit DC bez výpadku provozovaných technologií,
- umožnit poskytování služeb i v interních sítích zákazníků CESNETu,
- snaha využít v maximální možné míře stávající HW,
- maximalizovat možnosti vzdálené správy, i pro případy výpadků,
- schopnost spustit VM v druhé lokalitě v případě výpadku jedné lokality,
- podpora IPv4 i IPv6, vč. anycastových služeb.

Při zmapování skutečného stavu jsme dospěli k rozhodnutí, že stávající infrastruktura se musí postavit v podstatě zcela znovu, ovšem s využitím stávajícího HW. Proto jsme se rozhodli veškerý provoz přesunout z Brna

do Prahy a zrekonstruovat uzel v Brně. Po úspěšném zprovoznění Brněnského uzlu jsme přesunuli veškerý provoz z Prahy do Brna a provedli rekonstrukci v Praze. Následně jsme virtualizované systémy rozložili mezi obě centra.

Je třeba ještě poznamenat, že přesuny systémů mezi oběma lokalitami proběhly „za živa“ – tedy bez výpadku provozovaných služeb i jejich dostupnosti v počítačové síti.

Technické řešení

První uzel v Praze je umístěn v datovém sále CESNETu. Druhý uzel je umístěn na Masarykově univerzitě v Brně. Oba uzly mají tu výhodu, že v dané lokalitě jsou dostupné dva POPy CESNETu, je možno poměrně dobře realizovat zálohované připojení na dva oddělené prvky páteřní sítě CESNET.

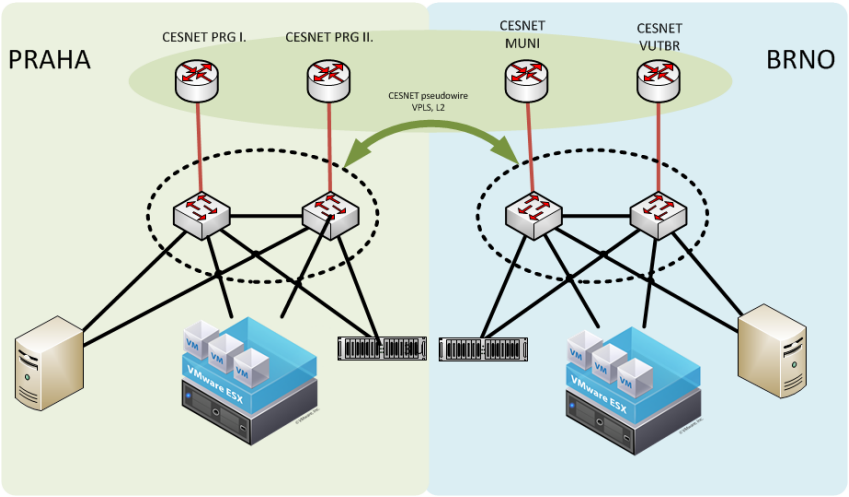
Celá infrastruktura je složena z několika částí, které lze v budoucnu obměnit tak, aby služby celku byly poskytovány bez významnějších dopadů do jiných částí. Infrastruktura je rozdělena do následujících částí – vrstev:

- směrovače a přepínače počítačové sítě,
- disková pole,
- fyzická serverová infrastruktura,
- virtualizační infrastruktura.

Směrovače a přepínače počítačové sítě

Směrovače jsou tvořeny v každé lokalitě dvěma L3 přepínači Cisco Nexus řady 9000. Oba prvky jsou v režimu active-active. U L2 vrstvy je použita technologie vPC, u L3 jsou využívány protokoly FHR (HSRP).

Oba prvky zajišťují IPv4/6 konektivitu do páteřní sítě CESNET a také L2 služby propojení s druhým datovým centrem, popř. propojení do dalších sítí. Propojení obou prvků je ukončeno na dvou nezávislých prvcích páteřní sítě CESNETu a v případě výpadku dojde ke konvergenci sítě v řádech sekund.



Disková pole

Úložnou kapacitu zajišťuje v každé lokalitě dvojice diskových polí Dell EqualLogic, která jsou z důvodu rozložení zátěže a zvýšení propustnosti spojena do jedné skupiny. Data jsou proti selhání disku jištěna kombinací RAID 6 a 10.

Každé diskové pole disponuje dvěma diskovými řadiči pracujícími v režimu Active-Standby. Každý řadič má dva 10Gbps síťové porty, což nám umožňuje redundantní připojení do síťové infrastruktury datacentra.

Výhodou diskových polí jedné řady (výrobce) je jejich vzájemná kompatibilita. Diskové pole spolu umí spolupracovat a provádět replikaci (zálohy, přesuny volumu) mezi sebou. Této funkcionality využíváme pro křížovou asynchronní replikaci záloh mezi datacentry.

Disková pole jsou použita nejen pro ukládání dat virtuálních serverů, ale hostitelé je využívají i jako svůj systémový disk, ze kterého bootují. Koncový server díky tomu nemusí obsahovat lokální disky a usnadňuje to reinstalaci v případě výměny serveru (např. reklamace) a v kombinaci s platformou Cisco UCS zejména možnost vzdáleného zásahu v případě poruchy.

Fyzická serverová infrastruktura

V pozici serverových systémů je použito řešení Cisco UCS. Velkou výhodou tohoto serverového řešení je jednotná a vzdálená správa serverových systémů. V praxi se osvědčila zejména pak existence serverových profilů, díky které lze realizovat vzdáleně velmi jednoduše a efektně změny logického i fyzického charakteru.

Celkem máme v obou lokalitách k dispozici dvě blade šasi s celkem devíti blade servery. Bootování systémů probíhá z diskových polí s využitím protokolu iSCSI.

Virtualizační infrastruktura

Jako základ virtualizační infrastruktury používáme nejnovější verzi VMware vSphere 6.5 v edici Enterprise plus, provozujeme celkem 9 hypervizorů.

Nad clusterem v každé z lokalit je aktivní služba DRS (Distributed Resource Scheduler), která dynamicky balancuje virtuální servery mezi hostiteli tak, aby byly jednotliví hostitelé rovnoměrně zatíženi a bylo dostatek výkonu pro náročnější VM. Díky možnosti živé migrace mezi datacentry jsme schopni zajistit běh kritických strojů i v případě odstávky v jedné z lokalit.

Pro denní zálohování je použito řešení od VMware, vSphere Data Protection (VDP). Každá lokalita je zálohována separátně a zálohy jsou dále křížem replikovány. Pro případ naprostého „disaster recovery/zničení/katastrofy“ odléváme týdenní zálohu externě do datových úložišť CESNETu.

Praktické zkušenosti z provozu

Provoz mezi lokalitami

Provoz i migrace VM mezi oběma lokalitami je v podstatě bezproblémová. V tomto případě jsme museli původně plochou síť pro ESX a disková úložiště nahradit směrovanou sítí. Důvodem je rychlost konvergence při výpadku některé z tras. V případě nasazení ploché sítě byla konvergence několik desítek sekund až jednotek minut. Každá změna nebo výpadek, a to i mimo síť DC/VI, tak měla fatální dopady do provozu VI. V případě použití dynamických směrovacích protokolů se dostáváme ke konvergencím v řádech jednotek sekund bez dopadu do provozu.

V celé síti DC/VI jsme od počátku podporovali tzv. jumbo rámce, MTU jsme sjednotili v celé síti na velikost 9000 B. Toto nastavení by se mělo lehce pozitivně projevit v interním provozu zvýšením výkonu, zejména práce s diskem a migracích mezi hosty.

Pro vzdálenou správu virtualizačního prostředí i pro komunikaci jednotlivých hostů VMware vSphere 6.5 používáme pouze IPv6, se kterým jsme nezaznamenali žádné problémy.

Disková úložiště

Diskové kapacity máme připojeny protokolem iSCSI. Pro tuto volbu jsme měli dva důvody. Prvním byla nutnost využití stávající hardware. Druhým důvodem je finanční náročnost – investice do oddělené SAN infrastruktury by v tomto relativně malém prostředí byla příliš veliká. S IP úložištěm počítáme i do budoucna, jen zvážíme nahrazení iSCSI protokolem NFS.

Optimalizace směrování

Pro interní dynamické směrování je použit protokol OSPFv3, který zajišťuje poměrně rychlou konvergenci vnitřní sítě. Dále je použit protokol BGP pro propagaci IP rozsahů do páteřní sítě CESNET a také pro podporu anycastových služeb. Pro oddělení a zjednodušení celého řešení jsou v roli BGP route reflectorů použity virtuální směrovače, které jsou provozovány v každé z lokalit.

Podpora anycastových služeb

V rámci celého DC/VI, bez ohledu na to, zda se jedná o fyzický nebo virtuální systém, podporujeme služby anycastu. Lze provozovat anycastovou službu ve dvou lokalitách (Praha, Brno), popř. doplněnou o další anycastový uzel v síti mimo DC/VI, např. v mateřské síti univerzity zapojené v síti CESNET. Služba využívá k propagaci IPv4/6 adresy protokol BGP a je zajištěna i její propagace do páteřní sítě CESNETu.

Anycastové uzly mohou být propagovány se stejnou prioritou, kdy provoz je směrován na všechny uzly, nebo s různou prioritou, kdy provoz je primárně odbavován jedním z uzlů a až v případě jeho nedostupnosti je směrován na další z uzlů.

Tato služba je využívaná např. pro web server instituce, který je primárně provozován v mateřské síti instituce. V případě odstávky sítě nebo

serveru je automaticky provoz směrován na záložní web server hostovaný v DC/VI CESNETu.

Závěr

V současné době je DC/VI CESNETu provozována ve dvou geograficky oddělených lokalitách – v Praze a v Brně, kde obě centra jsou redundantně propojena prostřednictvím páteřní sítě CESNETu.

Provoz celé infrastruktury je stabilizován a v současné době probíhají instalace některých projektů – např. knihovní systémy KOHA, disková úložiště pro ESA, systémy pro podporu IoT apod.

DC/VI je dnes platformou, nad kterou jsme schopni velmi pružně budovat další služby, komplexní virtualizované celky, ale i integrovat fyzickou serverovou infrastrukturu. V nejbližší době počítáme s ověřením konceptů, popř. i testovacím provozem technologií propojení VXLAN a dalších IP VPN.

Tím, že datové centrum je složeno z několika částí s jasně definovaným účelem a propojením, tak lze při dalším rozvoji v budoucnu řešit postupnou obnovu nebo rozvoj jednotlivých částí. Změnou jedné z části není narušen koncept celého datového centra a není nutno realizovat celkovou obnovu, čímž lze v čase rozložit investice.

Architektura a zkušenosti s resortními IS OVŠ MŠMT

Ivan Burian

E-mail: burian@ics.muni.cz

Abstrakt

Představení sedmi resortních informačních systémů Odboru vysokých škol (OVŠ) MŠMT. Seznámení s jejich architekturou, postupem vývoje systémů a zkušenostmi se zadavateli z MŠMT a uživateli z jednotlivých veřejných a soukromých vysokých škol v ČR.

Z historie informačních systémů OVŠ MŠMT

Smluvní spolupráce s OVŠ má na MUNI mnohaletou tradici. V roce 1999 byla uvedena do provozu první verze celostátní matriky vysokoškolských studentů, se zkratkou SIMS (Sdružené Informace Matrik Studentů). Informační systém SIMS tehdy MUNI nejen vyvíjela a podporovala, ale i provozovala na své infrastruktuře, a úspěšně započatá spolupráce trvala 15 let.

Změna přišla na jaře 2015, kdy byl provoz matriky převeden na infrastrukturu MŠMT, a bezprostředně na to navázalo výběrové řízení na 1leté zajišťování provozu a rozvoje nejen matriky, ale i dalších 4 informačních systémů OVŠ, s nimiž MUNI do té doby neměla žádnou autorskou či provozní spojitost. Ve výběrovém řízení MUNI uspěla, a dostala tím do smluvní péče kromě matriky také Registr docentů a profesorů (REDOP), Informační systém Akreditační komise (ISACC), Registr žádostí o uznání zahraničního vysokoškolského vzdělání a kvalifikace – nostrifikací (NVS) a Registr žádostí o posouzení postavení studia na vysoké škole v cizině

(PPSVS). Prakticky co systém, to jiná aplikační oblast, architektura a technologie – ale o tom podrobněji až dále.

Situace se zopakovala o rok později, na jaře 2016, tentokrát s výhledem na další dva roky. Smluvní péče MUNI o resortní informační systémy se rozšířila na další dva – Registr žádostí o uznávání odborných kvalifikací (UOK) a Registr uměleckých výstupů (RUV). Opět každý architekturně i technologicky jiný než všechny předchozí.

Bezprostředně nato vstoupila v platnost novela Zákona o vysokých školách (ZoVŠ) a prováděcí vyhlášky, které se nějakým způsobem dotkly téměř všech jmenovaných systémů. Novela též přesněji stanovila výčet provozovaných IS v gesci MŠMT a vyhlášky detailněji stanovily strukturu datových vět, četnost sběrů a metodické pokyny k evidenci požadovaných údajů. Pojďme se nyní blíže podívat na některé zajímavější momenty.

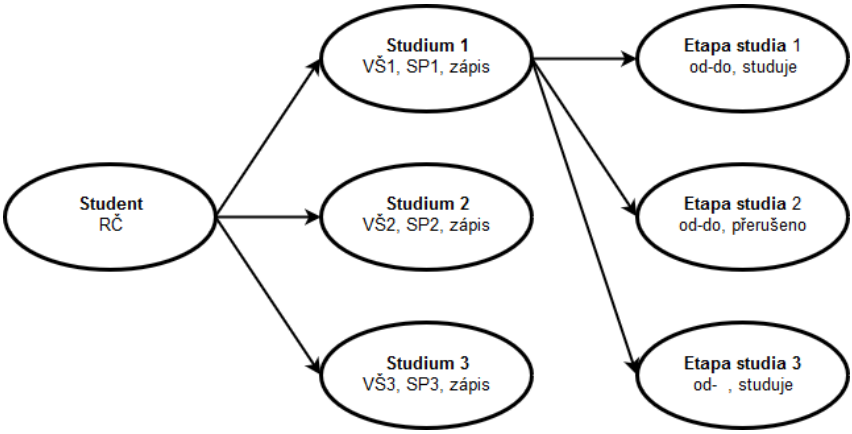
Matrika studentů – IS SIMS

Matrika studentů obsahuje údaje o všech studentech veřejných i soukromých vysokých škol, zapsaných k prezenčnímu, kombinovanému a distančnímu studiu v bakalářských, magisterských a doktorských studijních programech, včetně studentů, kteří studium přerušili či ukončili, a slouží k evidenčním, statistickým a rozpočtovým účelům.

Databázová vrstva je tvořena MS SQL Serverem s použitím T-SQL a podporou stored procedur. Ty zajišťují výpočetně náročnější operace s daty (přepočty odstudovaných dob studia a přidaných položek) a podklady pro některé složitější výstupy prezentační vrstvy. Aplikační a prezentační vrstva je postavena na MS .Net frameworku s využitím C# v kombinaci s ASP.net. Pro uživatele je systém přístupný jako webová aplikace.

Základní datová věta je tvořena osobními údaji studenta (jméno, příjmení, tituly, adresa, informace o středoškolském vzdělání) s identifikací přes rodné číslo (problém u cizinců). Dále pak informacemi o jeho studiích na vysoké škole (vysoká škola, studijní program, datum zápisu, délka studia, předchozí vzdělání, ukončení studia) a informacemi o etapách těchto studií (státní příslušnost, jazyk výuky, pobyt v ČR, studijní obory, aprobace, místo a forma studia, způsob financování, informace o přerušení a časové platnosti etapy od a do).

Uvedená logika ukládání dat i rozsah datové věty je v podstatě stejný od začátku fungování matriky. Během doby se některé položky rušily (ná-



Obrázek 1 Schéma evidence základních dat v matrice studentů

rodnost), některé naopak přidávaly (registrace přestupů, sociální a ubytovací stipendia, stáže, čísla diplomů atd.) Též docházelo ke změnám v používaných číselnících (UIR-ADR->RUIAN, číselníky studijních programů/oborů), což si vynutilo převod uložených dat a vazeb na tyto nové číselníky.

Počáteční architektura matriky byla také třívrstvá a postavena nad MS SQL Server databází s podporou stored procedur. Aplikační vrstva však byla tvořena ASP, VisualBasic a Perlovými skripty. Perlový skript byl použit na zpracování a kontrolu vstupních souborů, které byly v počátcích v CSV formátu (textový soubor s jednotlivými položkami oddělenými svislítkem „|“, kdy jeden řádek reprezentoval jednu etapu studia). Posílané údaje se v jednotlivých sběrech vkládaly do tzv. „sběrových“ tabulek v DB a po uzavření sběru dat se hromadně převedly do hlavních provozních tabulek. Aktuálně se zpracovávané soubory vkládají již přímo do provozní DB.

V rámci jednotlivých sběrů dat dochází k doplňování (aktualizaci) již vložených údajů. Jednotlivé záznamy se jednoznačně identifikují přes pětici klíčových položek – rodné číslo, kód vysoké školy, kód studijního programu, datum zápisu a začátek platnosti dané etapy studia. Sběry dat probíhají čtvrtletně za uplynulé období a vkládání a aktualizace údajů je povolena pouze na základě vstupních datových souborů v XML. Vstupní soubory jsou šifrovány pomocí PGP klíčů.

Aktuální velikost základní databáze je cca 20,6 GB a archivní databáze (kopie tabulek k 31.10. /pro financování/ a 31.12. /pro výročí zprávy/) je 25,7 GB. Evidováno je celkem cca 1 470 tis. záznamů o studentech, 2 640 tis. záznamů o jejich studiích v cca 3 720 tis. etapách studiích. Evidovány jsou záznamy z 85 subjektů – veřejných a soukromých vysokých škol v ČR.

S příchodem novely ZoVŠ vyvstaly pro matriku následující implementační změny:

- Akceptace okamžitých změn bez ohledu na termín sběru.
- Rozšíření datové věty o evidenci čísla vysokoškolského diplomu a čísla dodatku diplomu, byl-li vydán.
- Evidence studentů poskytovatelů zahraničního vzdělání na území České republiky. S touto evidencí je spojena změna struktury, rozsahu a četnosti sbíraných informací: pro tyto poskytovatele bude mít matriční věta jinou skladbu, jinak (nestandardně) budou tvořeny kódy těchto subjektů (RID), kódy studijních programů (oborů) a s tím spojené „zahraniční“ tituly.

Ačkoliv funguje matrika již více jak 17 let, stále se ze strany uživatelů potýkáme s nepochopením metodiky, vkládáním chybných údajů a chybných návazností. Problémem je také velká fluktuace uživatelů na soukromých vysokých školách, což způsobuje nepravidelnost ve sběrech, nutnost předávání zkušeností a metodiky mezi zaměstnanci.

Registr docentů a profesorů – IS REDOP

Registr REDOP obsahuje údaje o docentech, profesorech a mimořádných profesorech zaměstnaných na veřejných a soukromých vysokých školách. Slouží k evidenčním účelům a využívá se při akreditačních řízeních.

Podobně jako matrika je REDOP realizován v databázovém prostředí MS SQL Serveru. Aplikační a prezentační vrstva však využívá vývojové prostředí DevExpress (C#), což přineslo vývojovému a provoznímu týmu MUNI dříve neznámé zážitky a výzvy, například nutnost zapomenout na databázové standardy a normální formy.

V základě se eviduje část osobních údajů zaměstnanců (docentů a profesorů) a k nim informace o jmenování docentem/profesorem – kdy, v jakém oboru a na jaké vysoké škole. Dále pak informace o získaném vzdělání, pracovních úvazcích a garanci studijních programů a oborů.

Vkládání dat do registru je možné skrze webové formuláře nebo pomocí importovacích souborů v XML. Aktualizace údajů v registru probíhá měsíčně - vždy za uplynulé období (měsíc). Do konce roku 2016 probíhaly sběry dat čtvrtletně.

Na rozdíl o matriky studentů, kdy se průběžně aktualizují uložené údaje o studentech a jejich studiích, dochází zde v rámci každého sběru dat v podstatě k opakovanému ukládání totožných údajů – uložení „kopie“ části personálních údajů, platných k určitému datu. V podstatě takové časové snímky. Tato „jednodušší logika“ evidence s sebou ovšem přináší problém při snaze získat ucelený přehled o průběhu zaměstnání a jednotlivých úvazcích.

Celková velikost databáze registru je cca 6,2 GB a eviduje údaje o cca 10 300 zaměstnancích – docentech a profesorech.

Hlavní požadavky na úpravu registru, opět vyplývající z novely ZoVŠ, se týkaly evidence mimořádných profesorů (týká se pouze vysokých škol s institucionální akreditací) a evidence zaměstnanců poskytovatelů zahraničního vzdělání na území ČR. Podobně jako u SIMS i zde dochází u poskytovatelů zahraničního vzdělání k modifikaci počtu a obsahu sbíraných informací v porovnání s českými vysokými školami. Další změny přinesla až vyhláška č. 276/2016 Sb., která detailně definuje formát a strukturu datové zprávy a technické podmínky a lhůty předávání údajů. Největší změnou bylo zavedení evidence dohod (DPP/DPČ) a přechod na v podstatě průběžnou aktualizaci údajů – nejpozději do 10. dne po uplynutí kalendářního měsíce (tzn. měsíční sběry dat).

Neméně zásadní byl i zásah do počtu, obsahu a významu evidovaných položek, například zrušení rodného čísla coby jednoznačné identifikace osoby (nahrazeno rokem narození), zrušení kompletní adresy trvalého pobytu (nahrazeno pouze obcí), doplnění pohlaví a hlášeného pobytu u cizinců, doplnění kompletních dat (datum) získání jednotlivých titulů, u jmenování docenty či profesory doplnění informace, na jaké vysoké škole jmenování probíhalo, a v neposlední řadě doplnění detailnějších informací u probíhajících dohod (DPP/DPČ).

Největší problém ze strany uživatelů je nevyjasněná metodika ohledně obsahu datové věty (dohody) a neexistence některých „povinných“ údajů v personálních databázích vysokých škol. Taktéž fluktuace odpovědných pracovníků (uživatelů) na soukromých VŠ. Dále pak snaha vkládat údaje bez ohledu na datové typy jednotlivých položek.

Informační systém Akreditační komise – ISACC

Informační systém Akreditační komise (nově Národního akreditačního úřadu – NAU) slouží k zadávání žádostí vysokých škol o akreditace studijních programů, jejich evidenci a záznamu průběhu akreditačního procesu. Je využíván i k evidenci studijních programů a evidenci oborů habilitačních řízení a řízení ke jmenování profesorem.

ISACC je technologicky shodný s REDOPem (MS SQL Server + DevExpress), a přinesl obdobné vývojové a provozní zážitky a výzvy. Navíc je rozdělen do několika „funkčních a prezentačních" modulů napojených na jednu databázi: lokální desktopová aplikace pro pracovníky sekretariátu NAU (plus off-line verze pro zasedání konaná mimo MŠMT); webová aplikace pro zadávání žádostí referenty VŠ; webová aplikace pro přístup a vyjádření hodnotitelů z jednotlivých hodnotících a pracovních skupin a komisí; veřejná webová aplikace s přehledem všech akreditovaných studijních programů/oborů na jednotlivých VŠ a lokální serverová služba starající se o rozesílání e-mailů a provádění pravidelných naplánovaných akcí. Z tohoto členění plyne fakt, že pokud je požadována změna v jednom modulu (např. desktopová aplikace), v drtivé většině případů je potřeba „sáhnout" i do všech ostatních modulů (minimálně z důvodu povýšení verze).

Celková velikost databáze registru ISACC je cca 41,8 GB a eviduje údaje o cca 24 200 akreditačních žádostech. Vkládání a aktualizace dat v registru probíhá pracovníky NAU nepřetržitě.

Aktuálně požadované úpravy přímo vychází z novely ZoVŠ a dopadly na ISACC dramatičtěji než na ostatní systémy v tom smyslu, že Akreditační komisi nahradil Národní akreditační úřad. Nový akreditační orgán si chce v co nejkratším čase vybudovat zbrusu nový informační systém, takže všechny úpravy do ISACCu čeká jepičí život. Aktuálně požadované úpravy se kromě přejmenování všech výskytů „AK" na „NAU" a „Komise" na „Zasedání" týkají doplnění oblasti vzdělávání a institucionální akreditace, a v podstatě kompletního přepracování systému hodnotitelů a pracovních skupin.

Aktuálně největším problémem je fakt, že nově požadované změny již naráží na technologické hranice stávajícího modelu registru. Již nejde udělat jednoduchou úpravu či doplnění, ale musela by se změnit půlka celé aplikace včetně logiky evidence a ukládání údajů (nové položky, nové či změněné číselníky, nové vazby a závislosti).

Registr nostrifikací – NVS

Registr žádostí o uznání zahraničního vysokoškolského vzdělání a kvalifikace je určen ke sběru a správě žádostí, vydávání rozhodnutí, a v neposlední řadě k zamezení duplicit způsobených vkládáním a vyřizováním stejných žádostí různými uznávacími orgány. Jednotlivými uznávacími orgány (UO) jsou veřejné vysoké školy, MŠMT, MV a MO (pro státní vysoké školy).

Registr NVS je opět z jiného soudku než předchozí. Stále jsme v prostředí MS, ale tentokrát v SharePointu (SP). Zážitky a výzvy spojené s vývojem v SP předčily i zážitky s prostředím DevExpress – například problém s vytvářením uživatelských účtů (mimo AD) a vůbec s nastavením adekvátních přístupových práv pro jednotlivé role uživatelů a pro potřebné programové celky aplikace. Nové a nové překážky na cestě k pokoření technologie si nakonec vynutily, že správci prostředí SP na MŠMT nasdíleli týmu MUNI nejvyšší uživatelské oprávnění na celou SP farmu.

Veškeré požadované úpravy vycházely z novely ZoVŠ a z vyhlášky č. 278/2016 Sb. Novela zákona stanovuje MŠMT povinnost registr vést, a prováděcí vyhláška stanovuje formát a strukturu datové zprávy, technické podmínky a lhůty předávání. NVS nebyl před převzetím servisní smlouvou veřejně provozován a nebyl veřejně přístupný, takže se mohl jevit jako ideální adept na „stavbu na zelené louce“. Smluvně však bylo určeno vybudovat jej nad interní verzí již provozovanou na MŠMT v prostředí SP, se všemi klady a zápory s tím spojenými. Zmíněná louka měla tudíž poněkud nestabilní a podmáčený podklad, a stavbu významně pozdrželo i vyjasňování metodiky evidence jednotlivých žádostí a struktury datové věty.

Vlastní aplikace je tvořena 29 dílčími tabulkami pro jednotlivé UO a jednou společnou tabulkou přístupnou pro všechny uživatele sloužící pro náhled a ověření, zda již není žádost evidovaná na jiném UO. Vkládané žádosti se z dílčích tabulek do společné kopírují pomocí globálního skriptu. Vstup dat je možný skrze webové formuláře dílčích tabulek nebo pomocí importovacího souboru v XML formátu. Údaje do registru vkládají a aktualizují jednotlivé UO nepřetržitě.

Aktuálně je registr již v provozu (od 1. 7. 2017) a eviduje se v něm cca 940 nostrifikačních žádostí.

Registr žádostí o posouzení postavení studia na vysoké škole v cizině – PPSVS

Registr PPSVS slouží k vydávání rozhodnutí MŠMT o postavení studia na vysokých školách v cizině na roveň studia na vysokých školách v ČR pro sociální účely a účely zdravotního pojištění.

PPSVS je technologicky opět kombinace MS SQL Serveru a DevExpressu, a má část veřejnou a neveřejnou. První je určena žadatelům k elektronickému vyplnění a podání žádosti, druhá slouží pracovníkům MŠMT ke správě žádostí a vydávání rozhodnutí. Vkládání a aktualizace údajů probíhá nepřetržitě.

Velikost databáze je cca 450 MB a je v ní evidováno cca 35 700 žádostí.

Aktuálně nejsou k tomuto registru požadovány žádné úpravy a i provoz je zatím bezproblémový.

Registr žádostí o uznávání odborné kvalifikace – UOK

Registr OUK slouží k evidenčním, procesním a statistickým účelům v oblasti uznávání odborné kvalifikace. Jeho součástí je evidence uznávacích orgánů, informace o regulovaných profesích, evidence přeshraničních poskytovatelů služeb, a evidence osob vykonávajících regulovanou profesi na území ČR, s vazbou na evropskou databázi regulovaných profesí.

UOK je opět jiná technologická výzva: stále nad MS SQL Serverem, ale tentokrát jsou aplikačními technologiemi PHP a XML/XSLT. Většina aplikační logiky a generátorů výstupních sestav je obsažena ve stored procedurách databázového serveru. Webová aplikace se stará o proces přihlašování, zadávání vstupních dat a zobrazování výstupů.

Veřejná část aplikace slouží široké veřejnosti k náhledům na evidované žádosti a oznámení, neveřejná část slouží pro jednotlivé uznávací orgány k zadávání, správě a vytváření statistických výstupů. Aktualizace dat probíhá uznávacími orgány nepřetržitě.

Velikost databáze je cca 10,3 GB a je v ní evidováno cca 820 oznámení a cca 8 700 žádostí v 330 regulovaných profesí. Do autorizované části registru přistupuje cca 80 uživatelů z 24 uznávacích orgánů (ministerstva, Advokátní komora, Komora architektů, Energetický regulační úřad atd.).

Tento registr nespadá přímo pod OVŠ, ale do servisní smlouvy byl za-

hrnut. Požadované úpravy jsou vynuceny změnami v zákonech (č. 18/2004 Sb. a č. 126/2016 Sb.) vycházejících z evropských směrnic (89/48/EHS, 92/51/EHS a 1999/42/ES). Aktuálně je UOK rozšiřován o evidenci „Evropského profesního průkazu“, s níž jsou spojeny změny ve vykazování a statistických výstupech.

Opět provoz tohoto registru je více méně bezproblémový a stabilizovaný, až na někdy zmatená zadání na provedení požadovaných změn.

Registr uměleckých výstupů – RUV

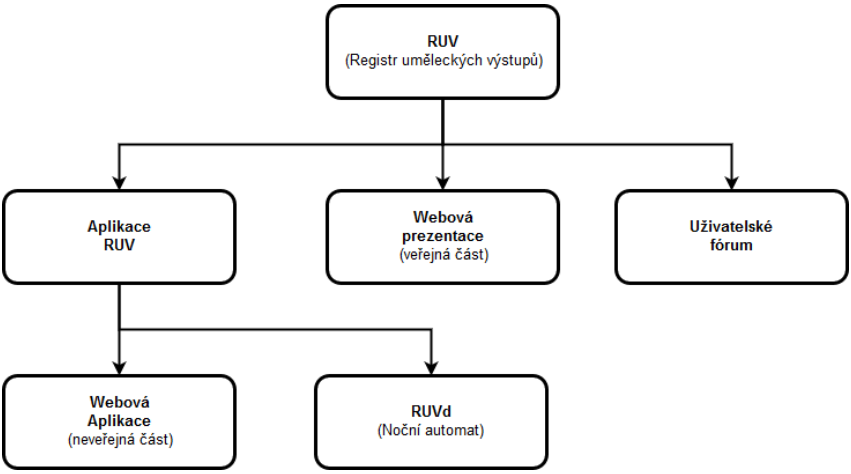
Registr RUV je mezi ostatními výjimečný tím, že byl dříve vyvíjen jako vlastní systém umělecky orientovaných vysokých škol, financovaný z rozvojových projektů a provozovaný na infrastruktuře ČVUT. Jádro registru vyvinulo ČVUT, minoritní webová prezentace byla vytvořena komerční firmou. Novelou ZoVŠ byl registr převeden do správy MŠMT, takže koncem loňského roku proběhlo totéž, co počátkem roku 2015 s matrikou studentů – přenos z hostitelské infrastruktury vysoké školy do infrastruktury MŠMT. V případě RUVu šlo o komplikovanější přenos díky tomu, že na rozdíl od všech ostatních systémů OVŠ není postaven na databázi a technologiích MS, ale na DB Oracle a Javě. Nutné bylo také řešit přenos a postoupení autorských práv jak k vlastní aplikaci, tak i k webovým doménám, na kterých je aplikace přístupna.

Jak již bylo napsáno, aplikace využívá databázi Oracle a je napsaná v Javě s využitím aplikačního serveru Tomcat a s podporou technologií Spring, Freemaker, Bootstrap, Mybatis, Maven a phpBB forum. Základní bloková struktura celého registru je zhruba následující

Veřejná část aplikace v podstatě slouží k prezentaci základních informací, harmonogramu, metodických pokynů a stručným statistickým přehledům. Hlavní část registru je neveřejná, přístupná po registraci a přihlášení a slouží autorům k zadávání jednotlivých uměleckých výstupů, gestorům a certifikátorům k hodnocení výstupů a správcům ke správě a celkovému vyhodnocování sběrů. Sběry probíhají jednou ročně na jaře za uplynulý kalendářní rok.

Velikost databáze registru je cca 4,3 GB a eviduje se v ní cca 27 300 uměleckých výstupů od cca 6 400 autorů.

RUV byl před migrací plně funkční a z důvodu plánovaného sběru dat a certifikace uměleckých výstupů od 1. 1. 2017 se do něj ani po migraci zatím nijak nezasahovalo. Nicméně během aktuálního sběru určité problémy



Obrázek 2 Základní blokové schéma registru uměleckých výstupů

vyvstaly, a z minulých sběrů jsou evidovány připomínky a požadavky na úpravy a rozšíření některých funkcí.

Aktuálně řešíme problémy způsobené proběhlou migrací a ladíme prostředí a i vlastní aplikace.

Závěrečné shrnutí

Spravované informační systémy OVŠ MŠMT jsou architekturně i technologicky hodně roztříštěné, což velice ztěžuje efektivní správu a jejich další rozvoj a úpravu. Většinu systémů převzalo MŠMT do své správy bez jakékoliv koordinace jejich předchozího vývoje – architektura a vývojové prostředí. Jediným spojujícím článkem je resortem používaná MS SQL databáze (a ani toto není v případě jednoho převzatého systému splněno). Chybí zde prvky vzájemné provazby systémů, přejímání dat, centrálně spravovaná a využívaná databáze společných základních číselníků atd.

U řady systémů není dosud vyjasněná procesní metodika – postupy řešení nestandardních případů, rozsahy a obsah sbíraných informací. Některé informace, požadované novelou ZoVŠ školy ve svých lokálních systémech vůbec nemají a velice složitě je doplňuji.

I přes delší dobu fungování registrů, uživatelé (vysoké školy) stále posílají chybné a „neodladěné“ údaje. Což s sebou přináší potřebu dodatečných kontrol a vývoj dalších a dalších kontrolních mechanismů.

Změny definované novelou ZoVŠ jsou v některých případech tak rozsáhlé a odlišné od prvotního zadání, že při jejich implementaci narážíme u některých systémů na jejich architekturní a logické hranice. Požadované změny se tak implementují za cenu mnohem větší složitosti a volby nestandardních řešení a postupů.

Literatura

[1] Novela zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění zákona č. 137/2016 Sb.

[2] Vyhláška č 277/2016 Sb., o předávání statistických údajů vysokými školami

[3] Vyhláška č. 276/2016 Sb., o předávání údajů do registru docentů, profesorů a mimořádných profesorů vysokých škol

[4] Vyhláška č. 278/2016 Sb., o předávání údajů do registru řízení o žádostech o uznání zahraničního vysokoškolského vzdělání a kvalifikace

[5] Zákon č. 18/2004 Sb. o uznávání odborné kvalifikace a jiné způsobilosti státních příslušníků členských států Evropské unie a některých příslušníků jiných států a o změně některých zákonů (zákon o uznávání odborné kvalifikace)

[6] Zákon č. 126/2016 Sb., kterým se mění zákon č. 18/2004 Sb., o uznávání odborné kvalifikace a jiné způsobilosti státních příslušníků členských států Evropské unie a některých příslušníků jiných států a o změně některých zákonů (zákon o uznávání odborné kvalifikace), ve znění pozdějších předpisů, a další související zákony

Vývoj softwaru se studenty a zvládání fluktuace v rámci týmu

Zdeněk Strmiska, Slávek Licehammer

E-mail: zdenek.strm@ics.muni.cz, slavek@ics.muni.cz

Kvalitní organizace práce je základem efektivního využití zdrojů v každé instituci a v širším kontextu je předpokladem pro správně fungující ekonomiku i společnost jako celek. Moderní doba si žádá vyšší flexibilitu pracovní doby jak na straně zaměstnanců, tak i na straně zaměstnavatelů.

Flexibilní organizace práce spočívající například ve spolupráci na dálku, home-office, pružné pracovní době nebo v podobě zkrácených úvazků je stále populárnější. V některých společnostech také z různých důvodů dochází k vyšší fluktuaci zaměstnanců, což klade nároky na zaškolování nových kolegů a jejich integraci do fungujících týmů.

Perun

Perun je systém pro správu uživatelských identit a zahrnuje řízení celého souvisejícího ekosystému včetně skupin, zdrojů a služeb. Perun se užívá především pro správu uživatelů v rámci organizací či samostatných projektů a k řízení přístupových práv ke službám. Do budoucna se plánuje jeho další rozšíření.

Při vývoji systému Perun dlouhodobě využíváme výše zmíněných pracovních postupů. Poměrně velká část týmu je tvořena studenty, u kterých musíme počítat s významnější fluktuací, která je způsobená tím, že délka studia je omezená a také tím, že se v průběhu studia účastní různých stáží nebo zahraničních pobytů. Dále někteří přicházejí například jen kvůli vypracování bakalářské práce a následně tým opouští. Je pochopitelné, že v případě studentů je dalším důležitým aspektem flexibilita pracovního vytížení v průběhu týdne i v rámci celého semestru. Typicky lze očekávat, že ve zkouškovém období mají studenti méně času na práci, než v průběhu semestru, i když to vždy závisí na konkrétní situaci. Mohou také žádat

o volno kvůli testům a zápočtovým zkouškám během semestru, nebo na dokončování školních projektů.

Pro správné fungování týmu je nezbytné tuto situaci reflektovat a používat nástroje, procesy a konvence, které umožňují hladké zapojení nových členů do týmu a nekladou nadbytečné nároky na členy stávající.

V praxi se ukazuje, že klíčové otázky každého nového zaměstnance bývají zpravidla podobné:

1. Jak mám nastavit vývojové prostředí a kde najdu zdrojové kódy?

2. Jaká j e moje pracovní náplň a jakým konkrétním úkolům se mám věnovat?

3. Jak můžu kód bezpečně upravovat, abych omylem nepokazil něco, co funguje?

Naší snahou je vyřešit problémy naznačené v uvedených otázkách tak, aby tyto otázky vůbec nemusely zaznít.

Nastavení vývojového prostředí a zdrojové kódy

Prvním krokem každého nově příchozího vývojáře je nainstalování a nastavení všech nástrojů nezbytných pro práci na projektu. Tato činnost se v našem prostředí vzhledem k fluktuaci zaměstnanců často opakuje a zatímco ve větších společnostech mohou mít přímo specializované pracovníky, kteří nově příchozím pomáhají, v menších týmech, jako je ten náš, pro tyto účely bývají zpravidla připravené různé manuály a příručky.

Vzhledem k relativně velkému poměru nově příchozích, především z řad studentů, ke stávajícím pracovníkům, bylo účelné připravit poměrně podrobný návod, který umožní nastupujícím pohodlné nastavení vývojového prostředí bez nutnosti asistence dalších kolegů. Udržujeme proto wiki stránku, na které jsou uvedeny jak návody na instalaci a nastavení jednotlivých nástrojů, tak i konvence a obecné postupy pro práci s kódem.

Jelikož pracovat na Perunu často začínají studenti bez předchozích zkušeností, do návodů jsou zahrnuty i obecné tipy např. pro práci s GitHubem resp. gitem obecně. V praxi se tento systém osvědčil, a nově příchozí studenti si zpravidla bez větších obtíží nastaví vývojové prostředí

a naučí se používat základní nástroje sami, respektive jen s menší pomocí od stálých zaměstnanců.

Vývoj Peruna se snažíme udržet co nejvíce transparentní. Kód je opensource, a prakticky každý může podat pull request s požadavkem na jeho změnu. Jak naznačeno výše, jako verzovací nástroj používáme při vývoji git. Celý zdrojový kód Peruna je pak umístěn na veřejně přístupném repozitáři GitHubu (https://github.com/CESNET/perun).

Plánování úkolů

Pro organizaci práce používáme Redmine. Jedná se o webovou aplikaci určenou pro projektový management, kterou kromě zadávání úkolů využíváme v celé řadě dalších případů.

Na Redminu jsou jednak umístěny výše zmíněné návody a další obecné postupy, ale i zápisy ze schůzek, dokumentace, plány apod. Klíčovou funkcionalitou je zadávání a sledování úkolů. Redmine umožňuje přehlednou kontrolu úkolů i stavu jejich plnění. Úkoly studentům zadávají kmenoví zaměstnanci, kteří jim v průběhu práce pomáhají a následně jejich výstupy kontrolují.

Upozornění na plnění úkolů je dotčeným pracovníkům standardně zasíláno na email. V Redminu je možné postupně doplňovat, jaká část je již hotova a po splnění úkol uzavřít. Dodržujeme pravidlo, že úkol uzavírá ten, kdo jej zadal, nikoli ten, kdo jej splnil. Tím se nejlépe ověří, že splněný úkol opravdu odpovídá požadavkům zadavatele.

git

již bylo zmíněno, že projekt Perun je opensource veřejně zpřístupněný na GitHubu. protože se do vývoje kontinuálně zapojuje poměrně velké množství nováčků, je důležité, aby se mohli do práce zapojit bez obav, že jejich úpravy způsobí nechtěné důsledky v některých částech systému.

aby nedocházelo k zásahům do hlavní vývojové větve má každý z vývojářů vlastní fork celého zdrojového kódu. pro jednotlivé dílčí úkoly si pak připravuje nové větve. jakmile je úkol hotový a připravený na začlenění (merge) do hlavní vývojové větve (master), podá vývojář žádost o modifikaci hlavní vývojové větvě (pull request). pochopitelně nejdříve musí vyřešit případné konflikty, pokud se relevantní části kódu mezitím změ-

nily. jelikož jsou úkoly zadávány tak, aby se netýkaly stejné části kódu, zpravidla k nijak zásadním konfliktům nedochází.

v zásadě udržujeme dvě hlavní větve: vývojovou větev (master) a větev, která je nasazená v produkčním prostředí (production). systém Perun nemá žádné verze. spolehlivě otestované změny na větvi master jsou postupně promítány na větev production. větev production je tedy vždy o určitou část „pozadu“ za větví master. právě do masteru jsou po schválení začleňovány změny provedené jednotlivými vývojáři. Průběžná integrace je částečně automatizovaná pomocí serveru Jenkins, který usnadňuje kompilaci, testování i nasazení softwaru. Jenkins nám významným způsobem pomáhá při vývoji, protože kód s libovolnou změnou se po podání pull requestu pokusí zkompilovat i otestovat. Může tak efektivně upozornit vývojáře na chyby ještě před tím, než příspěvek začne revidovat další člen týmu.

Tento způsob vývoje používáme mezi kmenovými zaměstnanci a osvědčil se i při spolupráci se studenty. Je zajištěno, že prakticky každý update vidí někdo jiný, čímž může odhalit chyby, které vývojář mohl přehlédnout (princip čtyř očí). Navíc umožňuje pohodlnou kontrolu úkolů vypracovaných studenty. Výhodou je, že přímo na GitHubu vidí zpětnou vazbu nejen autor kódu, ale i ostatní. Ti se eventuálně mohou přidat s dalšími relevantními komentáři. Vzhledem k tomu, že pull requesty a komentáře k nim jsou přístupné všem, mohou studentům sloužit i jako určitý zdroj inspirace a náhled do zajímavých technických řešení.

Poměrně extenzivně také využíváme unit testy, kdy se snažíme mít minimálně jeden test na každou důležitou metodu. Unit testy většinou píše každý sám a ihned pomocí nich svůj příspěvek kontroluje. Navíc při přidání nového pull requestu nebo při jeho úpravě detekuje změnu Jenkins, který nechá proběhnout i jednotkové testy a jejich výsledek uloží jako součást pull requestu. Kontrolující tak již nemusí spouštět testy manuálně.

Závěr

Při práci na projektu Perun se snažíme prostředí nastavit tak, abychom mohli prakticky ihned do vývoje zapojit studenty, z nichž velká část získává svoje první zkušenosti s programováním dílčích částí většího systému. Snažíme se přizpůsobit požadavkům na flexibilitu pracovní doby, jak v rámci pracovního týdne, tak v rámci celého semestru, respektive studia.

Zároveň musíme být připraveni na rychlé zapojení nově příchozích do týmu. Důležitý je jednoduchý návod k nastavení vývojového prostředí, díky kterému si student může připravit svůj stroj prakticky sám. Následně studenti pracují na jednodušších úkolech, které začínajícím vývojářům ukážou aspekty práce na systému Perun.

K organizaci práce používáme především Redmine a GitHub. Úpravy kódu probíhají odděleně od hlavní vývojové větve a před začleněním do ní jsou zpravidla kontrolovány ostatními vývojáři. K ověření správného fungování kódu využíváme také unit testy, které se při požadavku o změnu hlavní vývojové větve spouští automaticky. Díky těmto postupům se noví členové týmu nemusí obávat provádět změny a zároveň je zaručena vysoká kvalita výsledného kódu.

Vývoj Peruna se snažíme udržet co nejvíce transparentní. Kód je open-source, a prakticky každý může podat pull request s požadavkem na jeho změnu. Komentáře k pull requestům jsou zpřístupněny všem ostatním a mohou tak sloužit i jako určitá inspirace, především pro začínající studenty.

OpenID Connect v praxi

Pavel Vyskočil

E-mail: vyskocilpavel@muni.cz

1 Úvod

Na internetu v dnešní době existuje velké množství služeb, ke kterým je nutné se autentizovat. Uživatelé musí často řešit problém různých typů autentizací k různým službám. Řešení, které by konsolidovalo tyto různé typy by zajistilo uživatelsky přívětivý přístup a zvýšilo zabezpečení.

Předpokladem pro provedení autentizace je registrace uživatele a jeho přístupových údajů u služby. Tuto operaci většinou provádí uživatelé při prvním přístupu ke službě. Při každém dalším přístupu se již provede pouze kontrola přístupových údajů, které uživatel zadává (nejčastěji uživatelské jméno a heslo) s údaji uloženými během registrace. Na základě této kontroly je poté uživatel autentizován. Nevýhodou tohoto řešení je separátní registrace uživatele pro každou službu zvlášť. Používá-li uživatel stejné přístupové údaje u více služeb, pak jsou tak zabezpečeny, jak je zabezpečena nejslabší služba. Správa různých přístupových údajů pro různé služby je velice náročná na správu.

Na výše uvedené problémy se snaží najít odpověď metoda Single Sign-On (SSO). Jde o metodu, kdy je uživatel autentizován při přístupu k první službě. Autentizaci však neprovádí služba samotná, ale důvěryhodná třetí strana, která uživateli vydá pověření ve formě tokenu (sada dat, která má v sobě zakódovanou identitu uživatele). Pokud následně uživatel přistupuje k dalším službám, nemusí znovu zadávat přístupové údaje, ale je autentizován tokenem. Výhodou je možnost přístupu k více službám za využití jedné sady přihlašovacích údajů zadaných pouze jednou.

Metoda Single Sign-On může být realizována za pomoci různých protokolů. Níže je uvedeno pět vybraných protokolů, které jsou a nebo byly využívány v akademickém prostředí.

1. *SAML 2.0 – Security Assertion Markup Language*[1]

   Jedná se o standard vyvinutý organizací OASIS,[2] který popisuje formát zpráv pro autentizaci a autorizaci mezi poskytovatelem identit a poskytovatelem služeb. Formát zpráv je ve formě XML dokumentů.

2. *Kerberos*[3]

   Jedná se o síťový autentizační protokol podporující Single Sign-On. Uživatel se autentizuje s autentizačním serverem (KDC) tak, že nedochází k přenosu 2hesla po síti. Výsledkem je autentizační token. Pomocí tohoto tokenu se uživatel autentizuje ke službám.

3. *OpenID 2.0*

   Jedná se o standard popisující decentralizovaný způsob autentizace uživatelů. Uživatel je identifikován pomocí unikátního URL. V dnešní době se již nevyužívá (např. společnosti Google i Seznam.cz ukončily podporu v roce 2015).

4. OAuth 2.0[4]

   Jedná se o otevřený síťový protokol umožnující bezpečnou autentizaci i autorizaci v bezpečné a standardní metodě nejen z webových, ale i mobilních a desktopových aplikací.

5. *OpenID Connect*[5]

   Jedná se o protokol určený pro autentizaci a autentizaci vzniklý spojením protokolů OAuth 2.0 a OpenID.

Dále se budeme již zabývat pouze protokoly OAuth 2.0 a OpenID Connect, jelikož mají vhodné vlastnosti pro použití ve webovém i newebovém prostředí.

[1] http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0.html
[2] https://www.oasis-open.org/
[3] https://web.mit.edu/kerberos/
[4] https://oauth.net/2/
[5] http://openid.net/connect/

2 OAuth 2.0

Na rozdíl od protokolů SAML 2.0 a Kerberos, které umožňují jen autentizaci, protokol OAuth 2.0 umožňuje delegovat uživatelská oprávnění, aby aplikace mohla jednat jeho jménem. Tedy autorizuje cizí aplikaci k provádění operací nad uživatelskými daty. Protokol OAuth 2.0 definuje čtyři role: resource owner, resource server, client a authorization server.

Resource owner (uživatel) – Vlastník chráněného zdroje (dat). Umožňuje přidělit nebo odepřít přístup k danému zdroji.

Resource server (služba) – Server spravující data o uživateli a také operace nad těmito daty. Přístup k datům je řízen pomocí tzv. access tokens.

Client (klient) – Aplikace, která přistupuje k resource serveru s oprávněním uživatele.

Authorization server – Server vydávající access tokens po úspěšném ověření uživatele.

Dále jsou definovány následující pojmy:

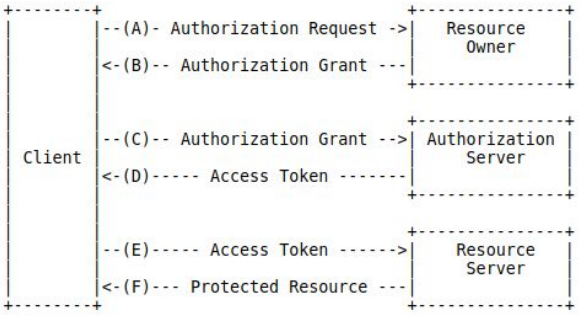
Access token – Slouží k přístupu k chráněným prostředkům. Představuje jistou abstraktní vrstvu, která slouží k nahrazení jiných autorizačních způsobů jediným tokenem, který je poté resource server schopen zpracovat. Access tokeny mohou mít různé formáty, struktury a způsoby využití založené na požadavcích na zabezpečení serveru.

Authorization grant – Souhlas uživatele s vydáním access tokenu klientovi.

Refresh token – Slouží k opakovanému vydávání access tokenu po vypršení jeho platnosti. Vydávání refresh tokenu je nepovinné.

Interakce mezi rolemi probíhají v tomto pořadí:

1. Client požaduje autentizaci od resource ownera. To lze provést přímo, jak je znázorněno na schématu, nebo nepřímo prostřednictvím authorization serveru jako zprostředkovatele.



Obrázek 1 Popis interakcí mezi rolemi

2. Client obdrží od uživatele authorization grant.
3. Client požaduje access token ověřením se authorization serveru a předložením authorization grant.
4. Authorization server ověří clienta a ověří authorization grant. Pokud jsou obě ověření úspěšná, vydá klientovi access token.
5. Client požaduje protected resource od resource serveru, předkládá access token.
6. Resource server ověří access token a pokud je platný, předá zpět protected resource.

Access token neopravňuje klienta k libovolným operacím s uživatelovými daty, ale obsahuje seznam tzv. scopes, která představují oprávnění ke konkrétním operacím. Uživatel může kdykoliv odvolat některá oprávnění.

3 OpenID Connect

OpenID Connect je autentizační vrstva nad protokolem OAuth 2.0 Jde formálně o nástupce protokolů OpenID a OpenID 2.0, ale s těmito protokoly není zpětně kompatibilní. Naopak je plně kompatibilní s OAuth 2.0.

OAuth 2.0 obecně autorizuje libovolné operace, zatímco OpenID Connect standardizoval operace pro získání údajů o uživateli. Seznam možných údajů o uživateli byl inspirován protokolem OpenID.

Ten umožňuje poskytovatelům služeb zajistit jednotnou autentizaci uživatelů bez ohledu na to, jakým způsobem se ověřují u autorizačního serveru. Na rozdíl od jiných autentizačních protokolů přináší dynamickou registraci klientů (popsána v bodu 3.4).

Je vyvíjený organizací OpenID Foundation.[6] Jedná se o skupinu nezávislých bezpečnostních odborníků a specialistů z několika kontinentů ve firmách včetně Microsoft, Google, salesforce.com a Deutsche Telekom. V současné době bylo více než 20 implementací testováno na interoperabilitu s protokolem OpenID Connect.

Na začátek je potřeba si objasnit pár pojmů, které jsou pro OpenID Connect stěžejní, a které nám pomohou pochopit jak probíhá autentizace pomocí tohoto protokolu:

Claims – Informace o daném uživateli, např. jméno, emailová adresa, atd.

Relying Party (RP) – Koncová aplikace, která požaduje ověření uživatele a také požaduje claims. V OAuth 2.0 definován jako klient.

OpenID Provider (OP) – Autorizační server, který je schopen ověřit uživatele a poskytnout RP požadované informace o uživateli.

EndUser – Koncový uživatel, odpovídá pojmu „resource owner“ v protokolu OAuth 2.0.

ID Token – Bezpečnostní token obsahující údaje o uživateli.

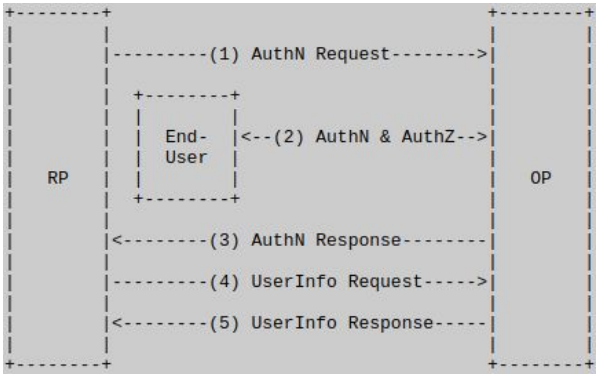
Authentication Request – Požadavek na ověření koncového uživatele u OP.

UserInfo endpoint – URL pro vyžádání dat o uživateli

3.1 Obecný průběh autentizace pomocí OpenID Connect

1. Relying Party odešle požadavek k OpenID Provideru se seznamem požadovaných oprávnění (scopes) pro přístup k požadovaným informacím mezi nimiž musí být speciální scope `openid` odlišující OpenID Connect od OAuth 2.

[6] http://openid.net/foundation/



Obrázek 2 Obecný průběh autentizace

2. OpenID Provider provede autentizaci uživatele a získá od něj souhlas se všemi nebo jen některými scopes.

3. OpenID Provider vrátí ID Token a obvykle i access token na stranu Relying Party.

4. Relying Party může pomocí access tokenu na UserInfo endpointu požadovat další informace o uživateli.

5. OpenID Provider předá informace o uživateli zpět na stranu Relying Party.

Userinfo endpoint vydává jen claims (údaje o uživateli) odpovídající scopes, která povolil uživatel. Jeden scope povoluje přístup k více claims, například scope profile povoluje přístup ke claims name, picture, gender, locale a dalším. Standardní scopes jsou profile, email, address, phone a speciální scope openid, který odlišuje OpenID Connect od OAuth 2.0.

ID Token musí obsahovat claims o autentizaci uživatele, jmenovitě `sub` (subject - identifikátor uživatele), `iss` (issuer – identifikátor OP), `aud` (audience – identifikátor RP), `exp` (expiration time – čas vypršení platnosti ID Tokenu) a `iat` (issued at – čas vydání ID Tokenu). Dále ID Token může (ale nemusí) obsahovat další claims, pokud je neobsahuje, klient je získá voláním Userinfo endpointu.

3.2 Autentizace pomocí OpenID Connect

OpenID Connect provádí ověřování k autentizaci uživatele, anebo ke zjištění, že daný uživatel byl již autentizován. Výsledek autentizace je poté vrácen klientovi bezpečným způsobem pomocí ID Tokenu. Tento proces autentizace může probíhat dle následujících tří způsobů: *Authorization Code Flow*, *Implicit Flow* a *Hybrid Flow*. Výběr daného toku určuje, jak bude klientovi vrácen ID Token a access token.

Authorization Code Flow je vhodný pro klienty, které mohou bezpečně udržovat sdílené heslo (client secret) s autorizačním serverem. Je tedy vhodný pro aplikace běžící na serveru Relaying Party.

Implicit Flow je vhodné pro klienty, u kterých z bezpečnostních důvodů nechceme mít v dané aplikaci uloženo sdílené heslo. Jedná se o aplikace psané v JavaScriptu, nebo pro aplikace běžící v prohlížeči uživatele.

Hybrid Flow kombinuje aspekty Authorization Code Flow a Implicit Flow. Klient musí bezpečně udržovat sdílené heslo s autorizačním serverem. Je vhodný především pro aplikace, jejichž funkčnost je rozdělena na část serverovou a na část běžící na straně uživatele.

3.3 ID Token

Jedná se o bezpečnostní token obsahující údaje týkající se ověřování uživatele autorizačním serverem a požadované claims. Je podepsán autorizačním serverem a předáván ve formátu JSON Web Token (JWT) aplikaci. JSON Web Token je kompaktní a bezpečný prostředek pro reprezentaci claims, které mají být přenášeny mezi dvěma stranami.

Příklad ID tokenu:

```
{
"iss": "https://server.example.com",
"sub": "24400320",
"aud": "s6BhdRkqt3",
"nonce": "n-0S6_WzA2Mj",
"exp": 1311281970,
"iat": 1311280970,
"auth_time": 1311280969,
"name": "Jane Doe",
"given_name": "Jane",
"family_name": "Doe",
"gender": "female",
```

```
"birthdate": "0000-10-31",
"email": "janedoe@example.com"
}
```

3.4 Registrace klientů

Před přihlašováním uživatelů k Relying Party je nutné danou Relying Party zaregistrovat u OpenID Providera. Protokol OpenID Connect umožňuje dynamickou registraci klientů, a tudíž danou registraci může zařídit provozovatel dané aplikace. Tato registrace může pracovat ve dvou režimech v závislosti na konfiguraci OpenID Connect serveru.

Prvním režimem je tzv. otevřená registrace, kdy se aplikace mohou volně registrovat pro běžně užívaná oprávnění. Otevřenou registraci může administrátor aplikace provést ručně, nebo automaticky. Druhým režimem je registrace za pomoci administrátora serveru.

Průběh registrace klienta:

1. Klient odešle svoje metadata HTTP POST zprávou na registrační endpoint OpenID Providera. Metadata si při registraci zvolí sám klient. OpenID Provider přiřadí takto registrovanému klientovi jedinečné ID, které může být doplněno heslem. Následně přiřadí k danému ID metadata, které obdržel v žádosti.

2. Po úspěšné registraci tento endpoint vrátí nově vygenerované ID clienta, případně i heslo, spolu se všemi registrovanými metadaty daného klienta. OpenID Provider může odmítnout nebo nahradit vhodnými hodnotami jakýkoliv přijatý parametr.

3.5 OpenID Connect vs. SAML 2.0

Protokol OpenID Connect je určen nejen pro webové aplikace, ale i pro aplikace mobilní nebo desktopové. Naproti tomu SAML 2.0 podporuje pouze aplikace webové s výjimkou Enhanced Client or Proxy (ECP). ECP zajišťuje podporu přihlášení i pro aplikace mobilní a desktopové. Není však příliš rozšířen a porušuje princip federovaného přihlášení. Dalším značným rozdílem mezi protokoly je použitý datový formát. OpenID Connect využívá JSON, na rozdíl od SAML 2.0 používajícího XML. Datový formát JSON je oproti XML znatelně menší a lépe zpracovatelný.

Uživatelům poskytuje protokol OpenID Connect plnou kontrolu nad vydávanými oprávněními. Uživatel může vybrat libovolnou podmnožinu

z požadovaných scopes. U protokolu SAML 2.0 uživatel musí buď souhlasit s vydáním všech údajů, nebo odmítnout přihlášení.

Vývojářům dále protokol OpenID Connect přináší snadné nasazení do svých služeb a velké množství implementací jak serverů, tak i klientů. Další výhodou pro OpenID Connect je také již zmíněná dynamická registrace klientů.

3.6 Implementace protokolu OpenID Connect

Existuje celá řada nejen knihoven pro OpenID Connect, ale i již kompletních řešení, a to jak pro Relaying Party, tak i pro OpenID Provider. Na stránkách protokolu OpenID Connect je dostupný seznam knihoven a implementací daného protokolu.[7] Jednotlivé implementace jsou rozděleny na certifikované a necertifikované, přičemž proces certifikace pomáhá ke zvýšení interoperability mezi jednotlivými implementacemi.

4 OpenID Connect v praxi

Využití protokolu OpenID Connect budeme demonstrovat na využití ve výzkumné infrastruktuře ELIXIR.[8]

ELIXIR (European life-sciences Infrastructure for biological Information) je evropská mezivládní organizace sdružující zdroje pro life sciences (biologie, medicína, a další vědy o živých organismech) napříč Evropou. Cílem této organizace je koordinovat tyto zdroje tak, aby tvořily jednotnou infrastrukturu. Daná organizace zahrnuje 21 členů a více než 180 organizací. Byla založena v roce 2014.

4.1 ELIXIR AAI

ELIXIR Authentication and Authorization Infrastructure (AAI) poskytuje služby pro identifikaci uživatelů a správu přístupu k jednotlivým službám.

Uživatelé, kteří chtějí využívat služeb ELIXIR, musí mít nejprve účet. Identita uživatele v infrastruktuře ELIXIR je určena dvěma atributy. Prvním atributem je eduPersonUniqueId a jedná se o neměnný identifikátor

[7] http://openid.net/developers/libraries/
[8] https://www.elixir-europe.org

daného uživatele. Ten je uživateli vygenerován při registraci do infrastruktury. Druhým atributem je ELIXIR username. Při registraci do infrastruktury je uživatel vyzván ke zvolení svého uživatelského jména. Uživatel má právo kdykoli toto uživatelské jméno změnit, to však již nesmí být použito pro žádného jiného uživatele.

Obdobná věc platí, pokud chceme naši aplikaci zpřístupnit členům infrastruktury. Danou aplikaci musíme nejdříve zaregistrovat.

V infrastruktuře ELIXIR jsou pro autentizaci uživatelů a správu přístupu k jednotlivým službám použity protokoly SAML 2.0 i OpenID Connect. Pro protokol SAML 2.0 je použita implementace simpleSAMLphp,[9] pro OpenID Connect implementace MITREid Connect. Protokol SAML 2.0 je také použit jako autentizační metoda pro autorizační server MITREid Connect. Jelikož protokol SAML 2.0 přesahuje rámec tohoto příspěvku, je již dále popsána pouze implementace protokolu OIDC.

Vhodná implementace autorizačního serveru pro užití v infrastruktuře ELIXIR byla vybírána z již hotových řešení. Daná implementace musela splňovat kritéria. Prvním kritériem je implementace napsaná v jazyce Java, kvalitní dokumentace a také jednoduchost použití. Jazyk Java byl zvolen kvůli kompatibilitě se systémem Perun,[10] který je používán pro správu uživatelů, služeb a řízení přístupu uživatelů ke službám. Poslední nutnou podmínkou pro výběr vhodné implementace byla možnost upravit komponentu pro autentizaci uživatele, aby dokázala přijímat identitu od webového serveru [1].

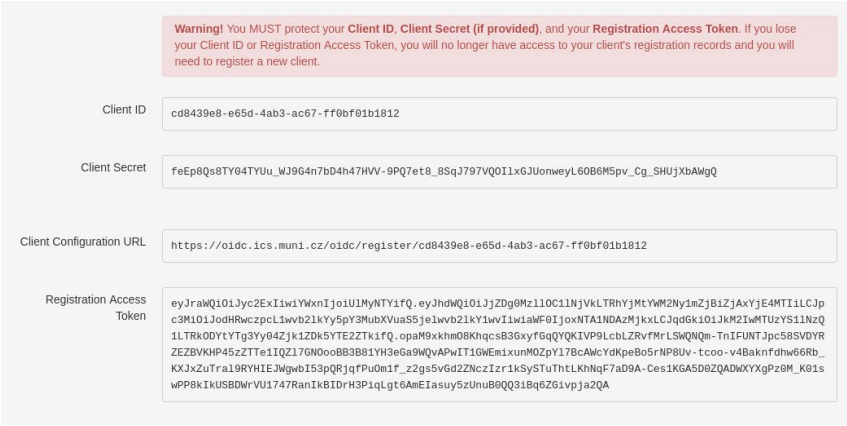
4.2 MITREid Connect

Jedná se o implementaci protokolu OpenID Connect. Poskytuje implementaci jak serverové, tak i klientské části s dobrou dokumentací. Podporuje dynamickou registraci klienta pro standardní OpenID Connect scope: openid, address, email, profile, phone, and offline_access. Další scopes a funkce jsou dostupné registrací pomocí správce systému.

Je implementována v Javě a na platformě Spring. Je dostupná na serveru GitHub[11] pod licencí Apache 2.0.

[9] https://simplesamlphp.org
[10] https://perun.cesnet.cz/
[11] https://github.com/mitreid-connect/OpenID-Connect-Java-Spring-Server

Obrázek 3 Registrace nového klienta

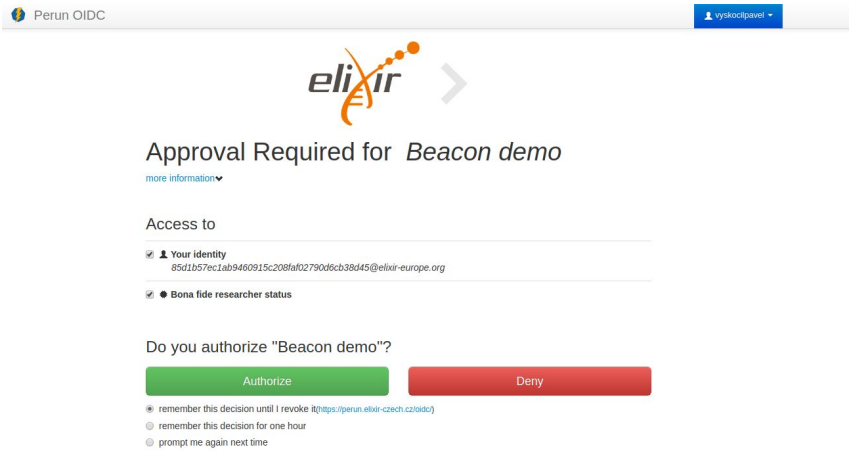


Obrázek 4 Vygenerované údaje

4.3 Ukázka registrace aplikace k ELIXIR AAI

Zaregistrovat aplikaci postavenou na protokolu OIDC k ELIXIR AAI je velice snadné, a to díky dynamické registraci, kterou tento protokol podporuje. V případě, že uživatel chce zaregistrovat aplikaci, musí být registrován do infrastruktury ELIXIR. Poté se již stačí daným účtem přihlásit na adresu *https://perun.elixir-czech.cz/oidc/manage*, kde je možné provést registraci nového klienta, a nebo editovat již existujícího klienta.

Uživatel pokračuje na formulář pro registraci nového klienta. Jakmile daný formulář potvrdí a uloží, jsou mu vygenerovány následující údaje: client_id, client_secret a registration_token. Tyto údaje je nutné si uložit, jelikož je možné dodatečně údaje editovat.



Obrázek 5 Autentizace pomocí OIDC v infrastruktuře ELIXIR

4.4 Ukázka autentizace ke službě v ELIXIR AAI

Autentizace k dané službě v infrastruktuře ELIXIR probíhá podle Authorization Code Flow, jak je popsáno výše. Při přístupu uživatele k dané službě je uživatel vyzván k autentizaci, pokud tak již neučinil při přístupu k jiné službě v infrastruktuře ELIXIR. Následně je dotázán na povolení předat uvedené atributy dané službě. Pokud uživatel toto povolení udělí, je následně autorizován k přístupu k dané službě.

5 Závěr

I když je protokol OpenID Connect relativně mladý, přináší podporu nejen webových aplikací, ale i newebových. To je v dnešní době, vzhledem k rozšíření chytrých mobilních telefonů, nutnou podmínkou. Spolu s dynamickou registrací a plnou kontrolou uživatele nad vydávanými daty může jít o rozhodující atribut při výběru autentizačního a autorizačního protokolu.

Použitá literatura

[1] VELÍŠEK, Ondřej. Nasazení OAuth 2.0 pro systém Perun [online]. Brno, 2016 [cit. 2017-09-22]. Dostupné z: <http://is.muni.cz/th/422680/fi_b/>. Bakalářská práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce Michal Procházka.