Vysoká škola ekonomická v Praze
Fakulta informatiky a statistiky

Analýza testovacího prostředí Shibboleth IdP pro potřeby CI VŠE

BAKALÁŘSKÁ PRÁCE

Studijní program: Aplikovaná informatika
Studijní obor: Aplikovaná informatika

Autor: Václav Pleskač
Vedoucí práce: Ing. Ondřej Vadinský, Ph.D.

Praha, červen 2019

Prohlášení

Prohlašuji, že jsem bakalářskou práci *Analýza testovacího prostředí Shibboleth IdP pro potřeby CI VŠE* vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne 27. června 2019 .....

Podpis studenta

Poděkování

Děkuji panu Ing. Ondřeji Vadinskému, Ph.D. za věcné rady a odborné vedení této práce.

Abstrakt

Shibboleth je aplikace pro webový Single sign-on, která se zakládá na standardu SAML. Ten umožňuje přenášet bezpečnostní informace vyměňované při autentizaci a autorizaci. Single sign-on (SSO) je typ autentizace, kdy po prvním přihlášení je uživateli umožněn přístup k vícero službám bez potřeby vkládat znovu přihlašovací údaje. Ve webové aplikaci umožňuje SSO takzvaný Poskytovatel služby (SP), jehož protistranou je Poskytovatel identity (IdP), který provádí autentizaci uživatele a zásobuje službu daty, která jsou k autorizaci nutná.

Technologie Shibboleth je hojně využívána jak v korporátním prostředí, tak na univerzitách. Vysoká škole ekonomická využívá Shibboleth k přihlašování na mnoho serverů. Tato bakalářská práce řeší problémy nedostatečné dokumentace Shibboleth IdP a chybějícího testovacího IdP prostředí v rámci VŠE, vůči kterému by mohly být ověřovány testované aplikace.

Cílem práce je vysvětlit fungování technologie Shibboleth a standardu SAML, shromáždit požadavky Centra informatiky Vysoké školy ekonomické na testovací IdP prostředí a podle těchto požadavků navrhnout a implementovat testovací IdP server a ověřit jeho funkčnost.

V práci provedená rešerše řeší problémy nedostatečné dokumentace – jejím výsledkem je nový, komplexní zdroj informací o technologii Shibboleth. Požadavky na testovací IdP byly shromážděny metodou rozhovorů se čtyřmi správci SP a brainstormingem v rámci Oddělení systémové podpory VŠE. Byly získány následující požadavky: Server musí konfigurací IdP odpovídat již fungujícím IdP serverům, na serveru má být použit Shibboleth IdP verze 3.4, vůči serveru se lze ověřovat různými testovacími identitami, kterým je možné měnit role a atributy, poskytovatelé služeb mají možnost vytvářet identity dle svých přání, testovací IdP server bude mít vlastní web s popisem svých funkcí a s návody, databáze serveru se bude každý den obnovovat podle produkčního serveru, server bude využívat BTRFS Snapshots, bude založena testovací federace pro správu metadat, správci SP na požádání získají přístup do relevantních logů serveru a server má mít volnou politiku uvolňování atributů. Testovací IdP musí dále podle vytvořeného diagramu případů užití dokázat komunikovat s testovacím SP. Testovací uživatel musí získat přístup k chráněným zdrojům umístěným v aplikaci Poskytovatele služby přihlášením k Poskytovateli identity. K implementaci byly zvoleny požadavky na konfiguraci co nejpodobnější provozním IdP serverům, na použití poslední verze Shibboleth IdP, na vytvoření upravitelných testovacích identit a na využití technologie BTRFS Snapshots kvůli snadným obnovám systému. Zbytek požadavků bude implementován v následujících měsících během zkušebního provozu.

Server byl se všemi potřebnými komponentami nainstalován a nakonfigurován – byl nainstalován linuxový server distribuce Debian Stretch se souborovým systémem BTRFS, připraven webový server, Apache Tomcat servlet, LDAP server, Shibboleth IdP verze 3.4 a bylo zařízeno zálohování, monitorování a logování serveru. Byli vytvořeni tři testovací uživatelé – po vzoru typického učitele, studenta a zaměstnance. Testování podle diagramu případů užití prokázalo, že testovací uživatel je ověřován Poskytovatelem identity a je mu umožněn přístup k aplikaci u Poskytovatele služby. Také bylo ověřeno vydání a úspěšné předání všech

uživatelských atributů. Výsledkem je testovací IdP server, který byl po otestování zařazen do zkušebního provozu v rámci infrastruktury CI VŠE.

Klíčová slova

Poskytovatel identity, poskytovatel služby, řízení přístupu, SAML standard, Shibboleth, Single sign-on, XML.

Abstract

Shibboleth is an application for web-based Single Sign-on based on SAML Standard. SAML allows for security information to be transferred during authentication and authorization. Singe Sign-on (SSO) is a type of authentication where, after first log in, user is allowed to access multiple services without the need of entering credentials again. In web applications SSO is enabled by so called Service Provider (SP), whose counterpart is Identity Provider (IdP), which authenticates users and supplies data used for authorization to SP.

Shibboleth is widely used in corporate environment but also on universities. University of Economics uses Shibboleth for authentication on many of its servers. This bachelor thesis solves problems of unsatisfying documentation of Shibboleth IdP and also of missing IdP environment for testing on VŠE, against which could be tested applications evaluated.

This thesis aims for explaining how Shibboleth and SAML works, gather requirements of Center of Informatics VŠE for test IdP environment and design, implement and test the new test IdP server.

Research conducted in this thesis solved the problem of unsatisfying documentation – its result is a new, complex source of information about Shibboleth technology. Requirements for test IdP were gathered by the Interview method with four SPs and by brainstorming in System Support Department of VŠE. Following requirements were gathered: Configuration of the test server must be as similar as possible with the production IdP servers, Shibboleth IdP of version 3.4.3 has to be used, testing identities has to be created for authentication against test IdP and those have to be modifiable, SPs have the option of creating identities at will, test IdP will have its own web page with description of its functions and with manuals, server database will be overwritten by a production database every day, server will use BTRFS Snapshots, identity federation id-test will be created for metadata management, SPs will gain access to relevant server logs (if they ask) and the test IdP will have a loose attribute policy. Test IdP will also, according to created Use Case diagram, be able to communicate with testing SP. Test user has to be able to gain access to a protected source placed in an application of SP by logging in to IdP. The requirements for configuration similar to the

production IdP servers, for the use of the last version of Shibboleth IdP, for creation of modifiable test identities and for the use of BTRFS Snapshots technology were chosen to be implemented primarily. The remaining requirements will be implemented in the upcoming months during trial run of test IdP.

Keywords

Access Control, Identity Provider, SAML Standard, Service Provider, Shibboleth, Single Sign-on, XML.

Obsah

Seznam obrázků

Seznam použitých zkratek

SSO Single sign-on
XML eXtensible Markup Language
SAML Security Assertion Markup Language
HTML Hypertext Markup Language
CPU Central Processing Unit
BTRFS B-tree file system
SOAP Simple Object Access Protocol
URI Uniform Resource Identifier
URL Uniform Resource Locator
XML eXtended Markup Language
DS Discovery Service
WAYF Where Are You From
LDAP Lightweight Directory Access Protocol.

Úvod

Tématem této bakalářské práce je Analýza testovacího prostředí Shibboleth IdP pro potřeby Centra informatiky Vysoké školy ekonomické. Shibboleth je Single sign-on systém, který umožňuje uživatelům přistupovat k vícero službám za použití jedné sady přihlašovacích údajů.[1] Technologie Shibboleth, která stojí na standardu SAML je hojně využívána jak v korporátním prostředí, tak na univerzitách. Vysoká škole ekonomická v Praze využívá této technologie k přihlašování na mnoho svých serverů. Ačkoliv je tato technologie na VŠE používána již několik let, kvůli nedostatečné projektové dokumentaci Shibboleth IdP a nedostatku expertů v této oblasti na naší škole má tato práce potenciál shromáždit cenné informace o používané technologii jak pro nové zaměstnance CI, tak pro ty stávající. V současné době také VŠE nemá testovací prostředí IdP, které by se dalo využít pro testování aplikací. Zároveň je Shibboleth technologií, která se neustále rozvíjí a je potřeba sledovat změny a nové možnosti, které přicházejí s novou verzí Shibboleth IdP 3.4.

Tato práce má tři cíle. Za prvé chce čtenáři vysvětlit fungování technologie Shibboleth a standardu SAML, ze kterého Shibboleth vychází. Protože se jedná o komplexní témata, seznámí práce čtenáře i s principy řízení přístupu ve webových technologiích, které tvoří nutný teoretický základ k pochopení Shibbolethu a SAMLu. Dalším cílem práce je shromáždit a analyzovat požadavky CI VŠE a správců aplikací, které využívají Shibboleth v rámci naší školy, a navrhnout podle těchto požadavků testovací prostředí Shibboleth IdP. Třetím cílem práce je navržené testovací prostředí zprovoznit a ověřit.

Prvního cíle dosáhnu prostřednictvím rešerše dostupné literatury[2]. Požadavky pro splnění druhého cíle získám jednak metodou rozhovorů[3] se správci služeb v rámci VŠE, a druhotně pomocí brainstormingu v rámci Oddělení systémové podpory CI VŠE. Tyto požadavky analyzuji a určím jim prioritu splnění. Také vytvořím diagram případů užití serveru. Na základě vytvořeného diagramu případů užití otestuji funkčnost prostředí a testovací server bude zařazen do provozu, čímž bude dosaženo třetího cíle. Toto testovací prostředí bude sloužit správcům služeb k efektivnějšímu a snazšímu testování aplikací, které budou následně Shibboleth využívat.

Shibboleth je komplexní technologií. Přes svůj potenciál ale postrádá srozumitelný popis. Přínosem této práce proto bude za prvé komplexní příručka, která bude sloužit především zaměstnancům Centra informatiky VŠE jako zdroj informací dostatečný k pochopení technologie Shibboleth a souvisejících standardů a pojmů. Druhým, ještě významnějším přínosem této práce bude funkční testovací IdP server, vůči kterému lze ověřovat testované aplikace. Protože sladění některých specifických aplikací s IdP může být složité, testovací IdP server výrazně zvýší komfort správců služeb využívajících Shibboleth na naší škole. Testovací prostředí bude i po obhájení bakalářské práce dále rozvíjeno, aby co nejvíce pomáhalo jak správcům aplikací tak pracovníkům Centra informatiky Vysoké školy ekonomické.

První tři kapitoly práce jsou věnovány teoretickému základu nutnému k pochopení Shibbo-

lethu. Kapitola 1 se věnuje principům řízení přístupu ve webových službách, zejména autentizaci, autorizaci a pojmu Single sign-on. kapitola 2 se zaměřuje na teoretické základy technologie Shibboleth, tedy zejména na jazyku XML (eXtensible Markup Language) a na souvisejícími pojmy jako XML schéma nebo XML jmenné prostory, a také na standard SAML. Třetí kapitola této práce (3) se zaměřuje na samotný Shibboleth, jeho jednotlivé části a pojmy s nimi související. Druhá polovina práce je věnována konkrétnímu testovacímu prostředí pro VŠE. V kapitole 4 jsou analyzovány požadavky Centra informatiky a správců služeb a je sestaven diagram případů užití testovacího serveru. kapitola 5 se věnuje důležitým detailům implementace navrženého řešení a podrobnostem splnění vybraných požadavků. kapitola 6 popisuje testování Shibboleth IdP a funkcí podle diagramu případů užití ze čtvrté kapitoly.

1. Řízení přístupu ve webových službách

Než se dostaneme k samotnému jádru práce, je třeba rozumět základním pojmům a bezpečnostním konceptům, které se vyskytují kolem webových služeb. Řízení přístupu je bezpečnostní prvek, který je klíčový pro zabezpečení jakékoliv webové aplikace. Do řízení přístupu spadají všechny procesy a mechanizmy, které limitují či detekují přístup ke kritickým informačním zdrojům, ať už s pomocí software nebo biometrických zařízení.[4] Tato kapitola si klade za cíl vysvětlit jednak úplně základní a běžné pojmy autentizace a autorizace, v třetí sekci potom pojem Single sign-on, spolu s výhodami a nevýhodami jeho využití v praxi. Do tématu Řízení přístupu ve webových službách spadají i další pojmy, které však nepotřebují vlastní sekci v této práci a budou vysvětleny v průběhu práce.

1.1 Autentizace

NIST definuje pojem Digitální autentizace jako určování, zda subjekt pokoušející se přistoupit k digitální službě kontroluje jeden nebo více platných autentikátorů spojených s digitální identitou subjektu.[5] Autentizace je pomyslným prvním krokem v interakci se zabezpečenou webovou službou. Úkolem uživatele služby je prokázat, že je tím, za koho se vydává. Podle J. Rosemberga a D. Remyho se zpravidla jedná o porovnání poskytnutých informací („výzva“) s něčím, co je již o jednotlivci známo.[6] Pro služby typu Shibboleth znamená úspěšná autentizace také ujištění, že vracející se subjekt přistupující ke službě je shodný se subjektem, který přistupoval ke službě předtím. [5]

Podle Rosemberga a Remyho[6] je autentizace klasicky dělena do tří typů: něco co *znáte*, něco co *máte*, nebo něco co *jste*:

- **Něco co znáte** – pin, heslo, pass phrase, sdílené tajemství
- **Něco co máte** – klíč, karta, token[7]
- **Něco co jste** – Biometrie, například otisky prstů, sken rohovky, hlasový otisk, otisk dlaně

Obecně se správcům služeb nabízejí dvě možnosti. Použít pouze jeden z předešlých typů autentizace, což je nejjednodušší varianta. Nejčastější volbou je heslo, které je ale často lehké odcizit, či dokonce uhodnout. V takovém případě navíc oběť nemusí vůbec tušit, že někdo přistupuje ke službě jejím jménem. V posledních letech se také stále častěji objevují bezheslové přihlašovací systémy, tedy systémy, které k autentizaci nevyžadují heslo. Podle J. Killorana je nejčastější metodou autentizace v tomto případě autentizace pomocí e-mailu, druhou nejčastější metodou je biometrie.[8] Druhou možností je vícefaktorová autentizace. V praxi se, snad jen s výjimkou armádních zařízení, uživatel se silnější než dvoufaktorovou autentizací nese-

tká. Dvoufaktorová autentizace je výrazně silnější z hlediska bezpečnosti a je považována za standard zejména v bankovnictví a dalších oblastech, kde je bezpečnost autentizace klíčová. Podle J. Rosemberga a D. Remyho se nejčastěji jeden způsob přihlášení z kategorií *něco co máte* nebo *něco co jste* přidává k něčemu z kategorie *něco co znáte*. K posílení dvoufaktorové autentizace je třeba posílit proces použitý k vytváření jednotlivých faktorů.[6]

1.2 Autorizace

Zatímco autentizace je tedy ověření identity uživatele, autorizaci definuje M. Wasson ze společnosti Microsoft jako „rozhodování jestli je uživatel oprávněn provést akci.“[9] Podle standardu NISTIR 7316 je autorizací proces přidělování nebo odmítání přístupových práv uživateli, programu či procesu.[4]Jinými slovy, po autentizaci uživatele je třeba určit, co má uživatel oprávnění provést.

Jeden způsob autorizace spočívá v přidělení množiny pověření subjektu, který potom pověření prezentuje. Tato pověření jsou potom namapována ke zpřístupnění určitých míst s omezeným přístupem.[6] Alternativně mohou být práva přidělena k omezenému obsahu a tato práva jsou poté mapována k identitám spolu s oprávněními která dostanou k tomuto obsahu.[6]

Mezi další pojmy související s bezpečností webových služeb patří také integrita, důvěrnost a nepopiratelnost. Integritou se rozumí, že zpráva nebyla od okamžiku svého vytvoření změněna. Nepopiratelnost zprávy znamená, že příjemce nemůže popřít obdržení zprávy. V případě zájmu o tyto a další pojmy z této oblasti lze doporučit první kapitolu knihy J. Rosemberga a D. Remyho - [6].

1.3 Single sign-on

Při běžném přihlašování se děje několik věcí:

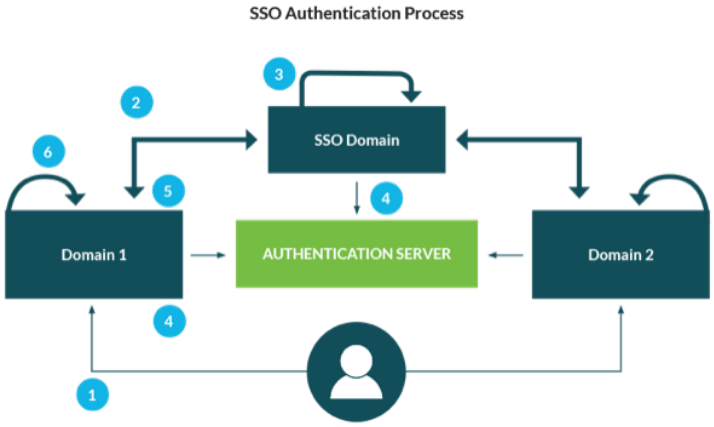
- Systém porovnává vložené údaje se *všemi* daty v příslušné databázi, dokud nenajde shodu.
- Když je detekována shoda, informace a oprávnění připojená k příslušnému účtu jsou poslána zpět do webového rozhraní.
- Webová stránka udělí pouze danému účtu přístup.

Klasické metody autentizace po uživateli vyžadují, aby si pamatoval neustále narůstající množství přihlašovacích údajů pro různé webové služby. To s sebou nese riziko ztráty údajů, také to vede k praktikám, které oslabují bezpečnost údajů - používání stejných hesel a podobně. Single sign-on (dále jen SSO) tvoří centralizovaný přihlašovací systém. Ten umožňuje uživatelům získat přístup k vícero službám bez potřeby pokaždé vkládat přihlašovací údaje. To umožňuje uživatelům starat se pouze o jeden účet, oproti standardnímu modelu s mnoha účty. SSO tak šetří čas, snižuje nároky na uživatele a výrazně zlepšuje pohodlí uživatelů.[8]

Jak znázorňuje obrázek 1.1, autentizační proce při použití SSO lze popsat v pěti krocích:

1. Uživatel přistoupí k cílové službě, ke které se chce přihlásit.
2. Služba automaticky přesměruje uživatele na centrální SSO službu k přihlášení - pokud uživatel nemá aktivní sezení.
3. Uživatel vloží přihlašovací údaje.
4. Centrální SSO stránka ověří údaje u autentizačního serveru.
5. Když jsou uživatelovy údaje ověřeny, centrální SSO služba přesměruje uživatele zpět na původní stránku a připojí k němu token s garancí autentizace.

Od této chvíle je token uložen a uživatel může bezpečně přistupovat ke kterékoliv stránce sdílející stejnou centrální SSO doménu.[8]



Obrázek 1.1: SSO autentizační proces.[10]

SSO autentizační nástroje spadají do dvou kategorií. Interní SSO autentizace má za úkol obstarat bezpečný a pohodlný přístup zaměstnanců ke všem digitálním účtům, kolaborativním nástrojům a management platformám v rámci organizace. Veškeré online služby či databáze, které jsou chráněny přihlašovacím oknem mohou být integrovány do interního SSO systému. Externě zaměřená SSO autentizace má za cíl zjednodušit uživateli přístup k produktům a službám. Nejjasnějším příkladem externě zaměřené SSO autentizace je společnost Google - kterýkoliv uživatelem s Gmail účtem má snadný přístup ke všem volně poskytovaným Google produktům, bez nutnosti se opakovaně přihlašovat.[8]

Klady a zápory SSO autentizace

SSO nabízí nesporné zlepšení bezpečnosti. Uživatel si s větší pravděpodobností vytvoří silné heslo, když si jich nemusí pamatovat tucet. V kontextu interně zaměřené SSO autentizace, kam lze zařadit i Vysokou školu ekonomickou, ačkoliv částečně spadá i do kategorie externě zaměřených SSO autentizací, se výrazně zvyšuje pohodlí uživatelů při využívání desítek různých webových služeb. Zároveň také snížená potřeba poskytovatelů služeb uchovávat hesla snižuje riziko úniku údajů v případě výskytu poskytovatele se slabým zabezpečením. Za zápor SSO lze považovat fakt, že systém provádějící autentizaci uživatelů má přístup k datům o webové aktivitě uživatelů. Při klasické autentizaci jsou taková data rozprostřena mezi různé

domény, v případě SSO jsou shromážděna na jednom místě. To nemusí být problém, pokud správcem autentizace je důvěryhodná autorita. Co se týče bezpečnosti při implementaci SSO autentizace se autentizující server stává kritickým bodem, jehož případná porucha ohrožuje chod všech poskytovatelů služeb na něm závisejících.[8] Předávat informace bezpečně mezi autoritou a uživatelem umožňují například jazyky XACML a SAML. Jazyku SAML se věnuje kapitola 2.2.

2. Stavební kameny Shibbolethu

Ještě něž se dostaneme k samotnému Shibbolethu, je nutné rozumět konceptům, ze kterých vychází. Shibboleth implementuje standard SAML, který je založen na jazyku XML. Jelikož je Shibboleth aplikace, která pracuje s osobními daty, je důležité vymezit způsoby zajištění bezpečnosti v XML.

2.1 Jazyk XML

XML je zkratkou pro eXtensible Markup Language. Jak název napovídá, je to značkovací jazyk, ve kterém lze vytvářet vlastní značky („tagy“). „Prosadilo se především díky své jednoduchosti a flexibilitě, která spočívá v možnosti snadného zachycení běžných datových struktur do podoby XML zprávy.“[11]Bylo vytvořeno kvůli narůstající potřebě překonat limity HTML, které je sice dobré v popisování, jak se elementy mají zobrazit, ale není dobré v popisování dat. Stejně jako HTML i XML je založeno na SGML – Standard Generalized Markup Language.[12] XML je „strukturovaná, sebepopisující cesta jak reprezentovat data, naprosto nezávislá na aplikaci, protokolu, slovní zásobě, operačním systému a dokonce programovacím jazyku.“[6] XML je často nazýváno univerzálním jazykem byznysu, protože je tak často používáno napříč odvětvími k přenosu byznys dat.[6] V kontextu této práce tvoří XML jeden ze základních kamenů, na kterých dále staví jazyk SAML i Shibboleth IdP.

2.1.1 XML schéma

Ačkoliv XML dokumenty mohou mít teoreticky libovolnou strukturu a mohou využívat libovolné značky, v praxi je potřeba udržet nějakou předem definovanou podobu vyměňovaných dokumentů. K tomu slouží jazyky pro popis schématu XML. Mezi tyto jazyky patří například jazyk DTD, který je podporován velkým počtem aplikací, nepodporuje však jmenné prostory a datové typy. Dalším příkladem jsou jazyky XML schéma a Relax NG. Jelikož vývojové nástroje komerčních firem (Microsoft, IBM, Oracle) podporují jazyk XML schéma, je tento jazyk často upřednostňován.[11] I SAML, potažmo i Shibboleth používají jazyk XML schéma. „Tyto jazyky umožňují definovat jaké elementy a atributy můžeme v XML dokumentu používat, jak je lze vzájemně kombinovat, co mohou obsahovat. Schéma dokumentu XML vlastně definuje nový značkovací jazyk, který má syntaxi XML, ale používá námi vytvořené značky. (...) Primární význam schémat leží v jejich použití pro *formální definici* značkovacích jazyků, nebo chcete-li datových formátů, založených na XML.“[11]Formalizované definice znemožňují různé interpretace, na rozdíl od definic popsaných v přirozeném jazyce. Jasná podoba dokumentu odpovídajícímu danému schématu je velmi užitečná při validaci, tedy při ověřování, zda konkrétní dokument XML vyhovuje všem omezením definovaným ve schématu. Můžeme si tak například ověřit, zda nám zaslaný dokument vyhovuje předem domluvenému formátu.

Velký význam mají schémata také při vývoji aplikací, kde díky validaci dokumentu už je při kontrole správného vstupu většina chyb odhalena.

2.1.2 XML jmenné prostory

Aby se předešlo konfliktům jmen v různých dokumentech, XML jmenné prostory poskytují jednoduchou metodu pro udržení jedinečnosti jmen elementů v XML dokumentu.[13] Fungují na podobném principu jako balíčky v objektových programovacích jazycích (Java, C++) - ty brání jménům lokálních dat či metod v kolizi se jmény z ostatních tříd. Stejně tak jmenné prostory umožňují vytvoření vlastních elementů a atributů bez obav z kolize, která by mohla nastat v případě, že bude potřeba použít shodná jména ve stejné xml instanci.[6] Jmenné prostory mají podobu URI a vypadají například takto:

```
xmlns="urn:oasis:names:tc:SAML:2.0:metadata
```

2.1.3 XML Signature

XML Signature je standard pro přidání elektronický podpisu k XML dokumentu, nebo jeho části metodou asociace klíče a příslušných dat (oktetů). [14] „Cílem elektronických podpisů obecně je poskytnout mechanismus pro integritu zprávy (nikdo zprávu nezměnil od doby co byla vytvořena) a nepopiratelnost (žádná ze stran nemůže tvrdit, že k výměně zprávy nedošlo. XML Signature umožňuje zakódovat elektronický podpis do XML.“[6] Základní funkcí XML Signature je schopnost podepsat pouze specifické části XML dokumentu. To je užitečné v případě, že XML dokument je postupem času upravován různými stranami. V takovém případě každá ze zúčastněných stran podepíše pouze elementy pro ní relevantní. Dalším případem užití této funkce může být dokument, ve kterém je důležité zajistit integritu určitých částí, zatímco jiné části jsou otevřeny pro uživatele k doplnění. Kdyby byl podepsán celý XML formulář, jakákoliv změna provedená uživatelem by zneplatnila elektronický podpis.[15]

2.1.4 XML Encryption

Druhou úrovní zabezpečení XML dokumentů je XML Encryption. „XML šifrování je specifikací vytvořenou W3 konsorciem v roce 2002 a obsahuje kroky jak zašifrovat data, jak dešifrovat data, XML syntaxi pro reprezentaci zašifrovaných dat, informace k použití pro dešifrování těchto dat a výčet šifrovacích algoritmů, jako jsou DES, AES a RSA.“[16] Stejně jako XML Signature vychází i XML Encryption z vyspělých kryptografických technologií - v tomto případě jde o technologii šifrování sdíleným tajným klíčem. Základním požadavkem na XML Encryption je, že musí dokázat zašifrovat XML zprávu o libovolné velikosti a musí to udělat efektivně. Proto bylo autory zvoleno právě šifrování sdíleným tajným (symetrickým) klíčem jako základ pro XML Encryption. Šifrování zajišťuje, že zpráva zůstane důvěrná, tedy že si ji nepřečte nikdo jiný, než adresát, i v případě, že zpráva zpráva prochází vícero místy na

cestě k cíli. Zpráva také musí zůstat důvěrná i po uložení v cílové destinaci. To je označováno jako *persistent confidentiality*, tedy v překladu do češtiny, trvalá důvěrnost. Stejně jako XML Signature, i XML Encryption lze aplikovat selektivně na vybrané části dokumentu[6]

2.2 SAML Standard

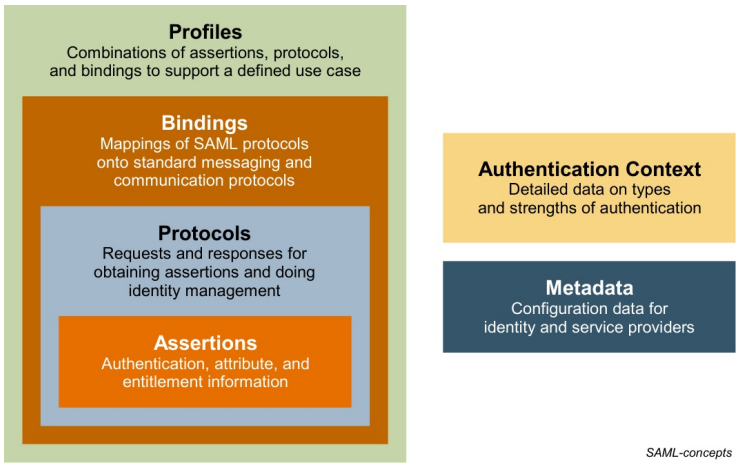
Security Assertion Markup Language (dále jen SAML) je jazyk, nebo také standard, který umožňuje přenášet bezpečnostní informace vyměňované při autorizaci a autentizaci. Jeho hlavní význam je v tom, že umožňuje realizovat SSO. Se SAML standardem se úzce pojí pojmy poskytovatel identity (nebo také IdP) a poskytovatel služby (SP). Pro účely této kapitoly stačí chápat poskytovatele identity jako službu, která uchovává identity uživatelů a která je zodpovědná za jejich autentizaci pro poskytovatele služeb. Detailnější rozbor těchto pojmů je obsažen v kapitole 3. SAML využívá XML pro standardizovanou komunikaci mezi takzvaným poskytovatelem identity a poskytovatelem služby. [17] Může „prosadit“, že jde o určitého uživatele s určitým oprávněním. SAML je neutrální vzhledem k platformě, díky využití XML. SAML je vyvíjen neziskovým konsorciem OASIS. Již v roce 2002 byla schválena verze 1.0, následovala verze 1.1 v roce 2003 a nakonec, v roce 2005 vyšel SAML V2.0.[18] Verze 2.0 se od verze 1.1 liší natolik, že jsou nekompatibilní.[17].

SAML se skládá ze čtyř základních, na XML založených, mechanizmů:

1. Tvrzení (assertions) – balík informací, které obsahují prohlášení vydaná SAML autoritou - viz. 2.2.1.
2. Protokoly (protocols) – „definuje protokoly žádostí a odpovědí mezi žádající a vydávající stranou pro výměnu bezpečnostních informací. Tedy protokol, kterým je možno takové tvrzení ověřit u důvěryhodné autority.“[19]
3. Vazby (bindings) – pravidla na použití tvrzení se standardy pro transport a posílání zpráv. V podstatě jde o napojení na přenosový protokol (HTTP, SMTP apod.)[19]
4. Profily (profiles) – kombinují předchozí tři prvky, jejich omezení či rozšíření, pro použití v konkrétní aplikaci. Existují také profily atributů, které ale neodkazují k žádné protokolové zprávě ani vazbě, ale definují jak vyměňovat informace obsažené v atributech za použití **tvrzení**[20].

Dva další koncepty viditelné na obrázku 2.1 jsou užitečné k vytvoření SAML prostředí:

- **Metadata** definují způsob, jakým dojde k vyjádření a sdílení konfigurace mezi zúčastněnými stranami. Tedy například podporované SAML vazby, role, podporované atributy, nebo informace o šifrování a podepisování. Metadata jsou definována vlastním XML schématem.[20]
- **Autentizační kontext** – „V mnoha případech potřebuje mít poskytovatel služby detailní informace ohledně typu a síly autentizace, kterou uživatel použil když se autentizoval u poskytovatele identity.“[20]**Autentizační kontext** je používán (nebo je na něj odkazováno) v *tvrzení* obsahujícím autentizační prohlášení k přenosu této informace.



Obrázek 2.1: Základní koncepty SAMLu.[20]

2.2.1 Tvrzení - assertions

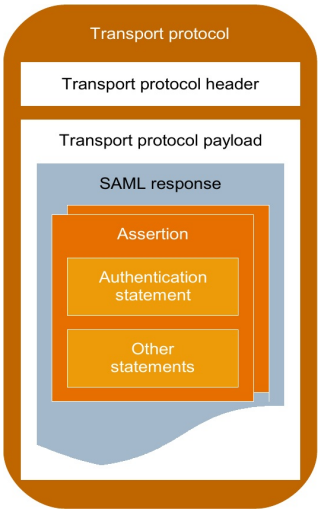
SAML slouží především jako „přenašeč identit“, vytváří vlastně jakousi „přenosnou důvěru“.[6] Důležitou roli v tom hrají SAML assertions, dále jen **tvrzení**. „Tvrzení je balík informací, které obsahují nula nebo více prohlášení vydaných SAML autoritou (poskytovatelem identit); SAML autority jsou někdy označovány jako **prohlašující strany** v diskusi o generování tvrzení a jejich výměně, a systémové entity které používají obdržená tvrzení jako **spoléhající strany** (poskytovatelé služeb).“[21] Tvrzení SAMLu neprovádí samotnou autentizaci, ani nedefinují žádný nový způsob autentizace či autorizace. SAML slouží k obalení, nebo také zapouzdření procesu autentizace a jeho přenosu. Specifikace SAML 2.0 definuje tři různé typy tvrzení, které mohou být vytvořeny SAML autoritou.

Všechna SAML tvrzení jsou vždy spojena se subjektem. SAML rozlišuje následující typy tvrzení:

- Autentizace – prohlašuje, že určitá autorita ověřila subjekt tvrzení určitým procesem, v určitý čas, s platností na určitou dobu.
- Autorizační rozhodnutí – oznamuje, zda žádost subjektu o přistoupení k určitému zdroji byla zamítnuta či nikoliv.
- Atribut – subjekt je spojován s připojenými atributy.

[21]

Vnější struktura tvrzení je generická, poskytuje informace společné všem prohlášením uvnitř. Uvnitř tvrzení se nachází sada elementů, které popisují autentizaci, autorizační rozhodnutí, atributy, nebo uživatelem definovaná oznámení. SAML assertion schéma povoluje uživatelem definované dodatky k tvrzením, stejně jako umožňuje definovat nové druhy tvrzení a oznámení.[21]



Obrázek 2.2: Vztahy mezi komponenty SAMLu.[20]

Obrázek 2.2 ukazuje typický příklad SAML zprávy: SAML tvrzení obsahující sérii prohlášení, je zabaleno uvnitř SAML odpovědi, která je přenášena nějakým druhem transportního protokolu.[20]

SAML standard má v současné době desítky SSO implementací, ať už komerčních (například LoginRadius nebo Numina Application Framework), nebo open source(OpenSSO, OpenSAML, simpleSAMLphp, nebo Shibboleth). Tato práce se zabývá především implementací Shibboleth. Její rozbor je obsahem následující kapitoly.

3. Shibboleth jako implementace SAML

„Shibboleth je na standardech založená open source aplikace pro webový Single sign-on, napříč organizací či mimo ni. Umožňuje stránkám provádět informovaná rozhodnutí o individuálním přístupu k chráněným online zdrojům způsobem, který zároveň chrání soukromí.“[1] Shibboleth implementuje SAML standard k poskytování federovaného SSO a rámce pro výměnu atributů. Uživatel se autentizuje svými, nebo organizačními přihlašovacími údaji a poskytovatel identity (IdP) přepošle nutné minimum informací poskytovateli služby, který potom rozhodne o autorizaci. Shibboleth také umožňuje uživateli kontrolovat atributy rozesílané konkrétním aplikacím, čímž napomáhá ochraně osobních údajů.[1]

Projekt Shibboleth začal již v roce 2000, jako projekt komunity Internet2. Ve stejném roce došlo k propojení s prací OASIS SAML Working Group a v roce 2003 vyšla první verze Shibbolethu. Verze 2.0 vyšla v roce 2006 a od roku 2014 je dostupná verze 3.x Shibboleth IdP a Shibboleth SP, která se zaměřuje na pokročilé způsoby autentizace a rozšířené možnosti přizpůsobení.[1] V době vzniku této práce je aktuální verze Shibboleth IdP 3.4.3.

Jak funguje Shibboleth?

Shibboleth je v jádru podobný ostatním webovým SSO systémům (například systémům zmíněným na konci předchozí kapitoly). Odlišuje se svým důrazem na standardy a svou schopností poskytnout SSO podporu i mimo uživatelovu organizaci, zatímco stále chrání jeho soukromí.[22]

Jako každý webový SSO systém, i Shibboleth se skládá z několika elementů[22]:

- Webový prohlížeč – slouží jako reprezentant uživatele v SSO procesu
- Zdroj – obsah, ke kterému se snaží uživatel dostat, ale který má omezený přístup
- Poskytovatel identity (IdP) – ověřuje uživatele
- Poskytovatel služby (SP) – provádí SSO proces pro zdroj

V některých případech je součástí Shibbolethu i takzvaný Discovery Service (DS), který pomáhá poskytovateli služeb odhalit uživatelova poskytovatele identity. DS se může nacházet kdekoliv na webu a není nutnou součástí Shibbolethu.[23]

3.1 Poskytovatel služeb (SP)

Service Provider, poskytovatel služeb, nebo jednoduše SP je jednou ze stran SSO autentizace. Umožňuje webovým aplikacím napsaným v jakémkoliv jazyce provozovat SSO, funguje na populárních webových serverech, jako jsou Apache nebo IIS. Při správně zvolené integrační

strategii umožňuje podporu SAMLu, výměnu atributů bez potřeby výrazně měnit aplikaci či používat proprietární rozhraní. Zpravidla se nachází na stejném serveru jako zdroj, ke kterému se snaží uživatel dostat.[24] Jeho hlavní úlohou v procesu SSO je autorizace uživatele.

Podle Shibboleth Konsorcia[24] je úkolem SP:

- Zamezit přístupu k chráněnému zdroji, nebo vstupu do aplikace
- Zjistit, vůči kterému IdP se uživatel chce ověřit
- Vytvořit a předat SAML autentizační žádost zvolenému poskytovateli identity
- Zpracovat odpověď poskytovatele identity a extrahovat z ní informace o uživateli
- Aplikovat lokální politiky a shromáždit dodatečná data
- Předat informace o uživateli cílové aplikaci nebo zdroji

3.2 Poskytovatel identity (IdP)

Identity Provider, poskytovatel identity, nebo také IdP je nejdůležitější součástí SSO procesu. Provádí samotnou autentizaci uživatele a zásobuje služby daty, která jsou nutná pro informované autorizační rozhodnutí. Kromě prosté odpovědi ano/ne na otázku zda je uživatel ten, za koho se vydává, může IdP službě poskytnout i množství uživatelských informací. Tyto informace mohou pomoci poskytovateli služeb poskytnout více personalizovaný zážitek a ušetřit uživateli čas, který by jinak strávil zadáváním informací, které SP vyžaduje. Tato data jsou také obnovována pokaždé když se uživatel přihlásí.[25]

Úkolem IdP je:

- Přijmout SAML autentizační požadavek od SP ke kterému uživatel požaduje přístup
- Autentizovat uživatele porovnáním poskytnutých informací s existující databází organizace - například s LDAP
- Shromáždit data o uživateli z databází organizace
- Aplikovat politiku svázanou s konkrétním SP, tedy jaká konkrétní data budou uvolněna danému SP
- Bezpečně přenést shromážděné informace k SP

[25]

V klasickém scénáři se IdP nachází v domovské organizaci, která spravuje uživatelský účet.[23] Poskytovatel identity se skládá z pěti komponent, konkrétně z autentizační autority, atributové autority, mezistránkové přenosové služby (Intersite Transport Service), SSO služby a služby na rozlišování artefaktů.[26] Autentizační autorita vydává pro poskytovatele služeb autentizační tvrzení, atributová autorita vydává atributová tvrzení (typy tvrzení viz. 2.2.1). SSO služba iniciuje autentizační proces, na jehož závěru přesměrovává prohlížeč na mezistránkovou přenosovou službu. Mezistránková přenosová služba poté na základě interakce s autentizační autoritou (IdP) vytvoří HTTP odpověď pro cílový prohlížeč. Služba na rozlišování artefaktů dostává od poskytovatele služeb žádosti na rozluštění SAML artefaktů

do příslušných tvrzení podle zvoleného SAML profilu. Vztah komponent IdP je zobrazen na obrázku 3.1.

Obrázek 3.1: Struktura komponent IdP (Zdroj: autor podle [6]).

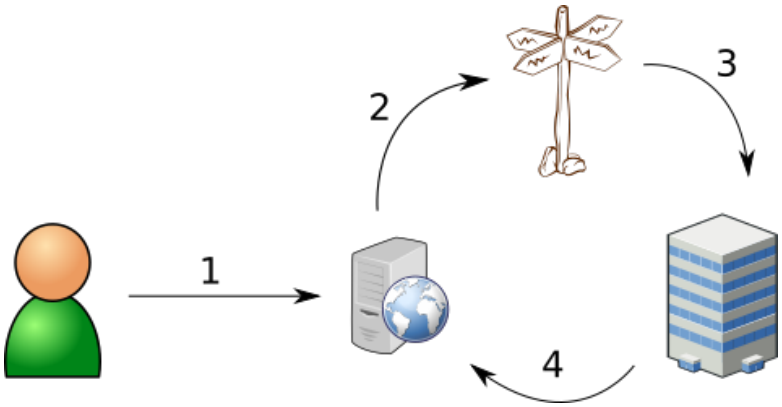
3.3 Federace identit

S pojmem Shibboleth se pojí také pojem federace, nebo také federační SSO. Všechny SSO systémy mají společné principy popsané v kapitole 1.3, některé z nich jsou ale vybaveny pouze pro práci uvnitř jedné organizace. Ostatní jsou navržené tak, aby fungovaly bez ohledu na to, zda se SP i IdP nacházejí ve stejné organizaci, či nikoliv. Takové systémy implementují **federační SSO**. Federací nazýváme skupinu poskytovatelů identity a poskytovatelů služeb, kteří se domluví na spolupráci. Je běžné, že si určitý poskytovatel služeb přeje pracovat s více než jedním poskytovatelem identity (například komerční služby s více zákazníky, nebo zdroje používané k výzkumu vícero organizacemi), a stejně tak poskytovatel identity si může přát pracovat s více poskytovateli služeb z různých organizací.[22]

3.3.1 Federované přihlášení

„Chce-li uživatel využít nějakou službu, jde na odpovídající adresu služby (krok 1). Tam se mu po zvolení přihlášení zobrazí „rozcestník“ s výpisem organizací (krok 2), z něhož vybere svou domovskou organizaci a dojde k přesměrování na přihlašovací stránku dotyčné organizace (krok 3). Po zadání správného uživatelského jména a hesla je uživatel přesměrován zpět

na službu jako přihlášený uživatel a může službu začít používat (krok 4).“[27] Tato služba není nezbytná, avšak poskytovatel služeb často nemá způsob jakým zjistit uživatelova poskytovatele identity. Tato centralizovaná služba umožňuje interaktivně zjistit poskytovatele identity bez potřeby získávání informací uložených v prohlížeči.[26] U starších verzí Shibbolethu zprostředkovává rozcestník služba WAYF, u aktuálního Shibbolethu pak Discovery Service (DS).



Obrázek 3.2: Proces federovaného přihlášení.[28]

3.3.2 Výhody a nevýhody federovaného přístupu

Při aplikování federovaného přístupu se výrazně zvýší uživatelský komfort. Uživatel vždy ví, vůči které organizaci se ověřuje a stačí mu jedna sada přihlašovacích údajů. Uživatelské údaje neopouští domovskou organizaci, takže případná napadená služba je nemůže získat z přihlašovacího formuláře. Pokud uživatel o heslo přijde, stačí si heslo změnit v domovské organizaci, nebo na stejném místě účet zablokovat. Pro správce služeb má federovaný přístup také spoustu výhod. Ti nemusí řešit ukládání uživatelských informací a nemusí se starat o bezpečnou správu uživatelských hesel, protože ta přes jejich službu nikdy neprojdou. Nemusí se také starat o aktuálnost údajů. Správcům stačí definovat, jaký typ uživatelů smí službu používat a implementaci již řeší domovské organizace uživatelů.[27] Nevýhodou je celkem složitá implementace ze strany IdP. Federovaný přístup také klade nároky na technické zázemí - zejména adresářový server s uživatelskými informacemi (LDAP, Active Directory) na straně IdP. IdP a SP se také mezi sebou musí dohodnout na politikách uvolňování atributů, což poté musí IdP zajistit. Bez toho není možné odlišit od sebe uživatele, nabídnout uživatelské profily, historii hledání, ukládání dat a podobně.[27]

3.4 Metadata a jejich správa

Metadata jsou v kontextu Shibbolethu (a SAMLu) konfigurační data potřebná pro ustanovení komunikace mezi SP a IdP. V Shibbolethu jsou metadata jedinou cestou jak vyřešit problém důvěry. V modelu SSO přestává SP nést odpovědnost za uživatele aplikace a přenáší tuto

odpovědnost na IdP (nebo federaci). Tato důvěra je ustanovena právě metadaty. Metadata jsou zpravidla v podobě XML, kvůli snadné výměně.[29] Metadatový soubor Shibbolethu musí obsahovat nadřízený element `<md:EntitiesDescriptor>` s jedním podřízeným elementem `<EntityDescriptor>` pro každého Poskytovatele identity u metadat SP a steně tak pro každého IdP pro metadata SP.[29] V Shibbolethu jsou metadata SP nahrána u IdP a naopak. To umožňuje snadnou komunikaci - zúčastněné strany navzájem znají svá URL, entityID - identifikátor, kryptografické informace o vytváření zpráv, názvy a popisy.[30]

V praxi je tedy potřeba, aby IdP server uchovával aktuální metadata všech poskytovatelů služeb, kteří vůči němu chtějí autentizovat uživatele. To lze zajistit dvěma způsoby – buď klasickou výměnou metadat, nebo využitím federace.

Klasická výměna metadat probíhá následovně:

- Správce SP vygeneruje XML soubor s metadaty a pošle ho správci IdP spolu s požadavky na uvolňované atributy
- Správce IdP zkontroluje správnost elementů XML a vyhodnotí validitu požadavků na uvolňované atributy.
- Pokud je vše v pořádku, zapracuje správce IdP metadata do serveru a nastaví uvolňování dohodnutých atributů.
- Správce IdP otestuje funkčnost IdP s novými atributy, funkčnost přihlášení na nového SP a informuje správce SP o zpracování požadavků.
- Při změně metadat nebo potřebě nových atributů probíhá proces znovu

Tento způsob výměny metadat tedy vyžaduje komunikaci správce IdP a správce SP, která často trvá několik dní. Také validace metadat správcem je složitá a úspěšnost často není stoprocentní. Většina úkonů při zapracovávání metadat padá na správce IdP.

Druhý způsob, tedy využití federace, je mnohem méně náročný jak na správce IdP, tak i na čas správců IdP i SP. Při využití federace stačí IdP serveru nastavit synchronizaci federačních metadat, která se skládají z metadat všech SP pod federací. Federace také ručí za funkčnost metadat, takže šance na nedostupnost služby kvůli chybě metadat je minimální. Jednou z možností jak zpracovávat metadata ve federaci je aplikace Jagger Management System od skupiny CESNET. Ta umožňuje správci SP vyplnit žádost o připojení do federace, nahrání metadat, která jsou automaticky zkontrolována a vybrání atributů, které správce považuje za nezbytné. Správce federace poté už jenom zkontroluje požadavky na uvolňované atributy a v případě nestandardních, ale odůvodněných požadavků informuje správce IdP o nutnosti nastavit individuální politiku uvolňování atributů pro daného SP. Při změně metadat stačí nová metadata opět nahrát do systému Jagger a správce IdP je o změně informován. Metadata jsou tak na serverech federace stále aktuální a požadavky jsou vyřizovány o poznání rychleji.

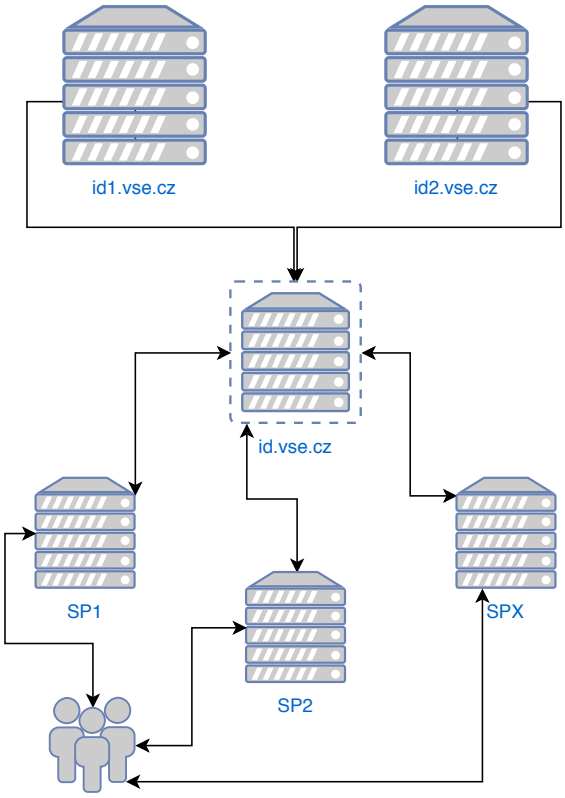
4. Analýza požadavků na testovací prostředí IdP

V předchozích kapitolách byl shromážděn teoretický základ nutný ke splnění cílů práce. S důrazem na technologie a standardy související s implementací Shibbolethu bylo prezentováno vše, čemu je nutné rozumět před samotnou implementací testovacího prostředí. V této kapitole je provedena analýza požadavků na testovací prostředí Shibboleth IdP. Jedná se o požadavky Poskytovatelů služeb, kteří budou uživateli testovacího prostředí po jeho dokončení. Na základě jejich námětů a přání je tedy vhodné vytvořit model, podle kterého bude testovací server nakonfigurován.

4.1 Současná infrastruktura Shibboleth na VŠE

Vysoká škola ekonomická v současné době využívá autentizaci pomocí Shibbolethu na mnoha serverech. Jedná se například o library.vse.cz, eduroam.vse.cz nebo telefony.vse.cz. Všichni tito Poskytovatelé služeb sdílí metadata s Poskytovatelem identity, kterým je v případě VŠE doména id.vse.cz. V současné době fungují dva IdP servery – jeden jako provozní, jeden jako záložní. Tyto servery je v případě potřeby možné zaměňovat. Za správu IdP serverů zodpovídá Oddělení systémové podpory, které je součástí Centra informatiky VŠE. Za stávajících podmínek tedy neexistuje server, vůči kterému by Poskytovatel služeb mohl zkoušet různá nastavení Shibboleth autentizace.

Pro správu metadat byla vytvořena federace VŠE-internal. Díky aplikaci Jagger Management System stačí tedy na IdP serverech synchronizovat metadata federace a nastavit pravidla pro uvolňování atributů členům federace a všichni SP spadající pod VŠE, kteří jsou správně nakonfigurovaní, se mohou ověřovat vůči právě nasazenému IdP serveru.



Obrázek 4.1: Infrastruktura Shibbolethu na VŠE (Zdroj: autor).

4.2 Proces shromažďování požadavků

Protože chceme, aby testovací IdP sloužilo co nejlépe svému účelu a splňovalo v rámci možností očekávání Poskytovatelů služeb, shromažďování požadavků je v rámci projektu velmi důležité. Vzhledem k malému počtu správců Poskytovatelů služeb fungujících v současné době na naší škole byla zvolena metoda rozhovorů[3]. Osloveno bylo osm správců SP, čtyři z nich byli ochotni přispět svými podněty k fungování serveru. Rozhovory byly vedeny jak verbální tak neverbální formou. Verbální rozhovory byly vedeny formou osobních schůzek autora práce se správci SP, neverbální rozhovory byly vedeny formou e-mailových konverzací. Obě formy pomohly utvořit finální diagram případů užití, podle kterého bude testovací IdP sestaveno.

Druhým zdrojem požadavků na testovací prostředí byl brainstorming v rámci Oddělení systémové podpory (OSP) Centra Informatiky VŠE. Pohled z druhé strany, tedy ze strany správců IdP byl klíčový pro tvorbu diagramu. Díky hluboké znalosti technologií souvisejících s IdP byly požadavky správců SP porovnány s možnostmi CI a byl vytvořen finální model pro testovací prostředí Shibboleth IdP.

4.3 Shromážděné požadavky a jejich vyhodnocení

Ze zdrojů zmíněných v předchozí sekci byl shromážděn seznam požadavků na testovací prostředí. Ne všechny požadavky jsou klíčové pro užitečnost testovacího prostředí. Tato práce se z důvodu časového omezení soustředí na implementaci požadavků s nejvyšší prioritou. Nejvyšší priorita byla přiřazena těm požadavkům, které nejvíce adresují potřeby správců SP. Po dokončení této práce dojde ke splnění cílů se střední a nízkou prioritou.

4.3.1 Tabulka požadavků

Následuje tabulka shromážděných požadavků s datem pořízení, popisem požadavku, způsobem jeho získání a přiřazenou prioritou.

4.3.2 Rozbor shromážděných požadavků a návrh řešení

Prvním požadavkem na testovací ID server je, aby jeho konfigurace odpovídala stávajícím ID serverům. To je velmi důležité z hlediska využitelnosti serveru, neboť hlavním cílem připravovaného testovacího serveru je poskytnout správcům SP možnost testovat své aplikace v neprovozním, avšak reálném prostředí. Z tohoto důvodu jsem přiřadil tomuto požadavku vysokou prioritu. Konfigurace IdP bude tedy v rámci možností nové verze Shibboleth IdP (a kromě nutných odlišností, jako je jiná politika uvolňování atributů, nebo EntityID) podobná provozním IdP serverům. Toho bude dosaženo importem (a důkladnou kontrolou) klíčových částí IdP konfigurace z provozních IdP serverů.

Dalším klíčovým požadavkem, kterému jsem přidělil vysokou prioritu, je použití poslední verze Shibboleth IdP. Přechod mezi verzemi bývá často složitý kvůli změnám v podporovaných funkcích a potřebě mít neustále dva funkční IdP servery. Nasazením nové verze Shibboleth IdP na testovací server tak můžeme ověřit kompatibilitu stávající konfigurace a aplikací s novou verzí a usnadnit tak budoucí přechod provozních IdP serverů na novou verzi. Na testovací server bude nasazena poslední verze Shibboleth IdP, tedy verze 3.4.3.

Shibboleth umožňuje různé úrovně oprávnění pro různé skupiny uživatelů. Aby bylo možné různé způsoby autorizace v aplikacích testovat, jedním z požadavků na testovací IdP server je i možnost ověřování se vůči serveru různými testovacími identitami. Tomuto požadavku byla přiřazena vysoká priorita – testovací identity zvýší možnosti testování a zamezí situaci kdy se skutečný uživatel omylem přihlásí k aplikaci přes testovací IdP. Provozní IdP servery čerpají informace o uživatelích z univerzitního LDAP serveru. To je server, který pomocí protokolu LDAP udržuje informace o uživatelích, jejich skupinách a přihlašovacích údajích.[31] Protože pro důkladné testování autorizace aplikace je vhodné použít identity s různými parametry, součástí testovacího ID serveru bude i LDAP server se sadou typických testovacích identit – tedy identit s parametry podobnými typickým uživatelům. K tomu se váže další požadavek – možnost měnit role a atributy testovacích identit. Role v LDAPu slouží k podpoře auto-

Datum	Popis požadavku	Způsob získání požadavku	Priorita
10.03.2019	Server musí konfigurací IdP odpovídat již fungujícím IdP serverům	Brainstorming	Vysoká
10.03.2019	Na serveru bude použita poslední verze Shibboleth IdP	Brainstorming	Vysoká
02.04.2019	Vůči serveru se lze ověřovat různými testovacími identitami	Konverzace se správci	Vysoká
02.04.2019	Testovacím identitám lze měnit role a atributy	Konverzace se správci	Vysoká
02.04.2019	Poskytovatelé služeb mají možnost vytvářet sami identity dle svých přání	Konverzace se správci	Střední
10.03.2019	Testovací IdP server bude mít vlastní web s popisem svých funkcí a s návody	Brainstorming	Nízká
10.03.2019	Databáze serveru se bude každý den obnovovat podle aktuálně nasazeného serveru	Brainstorming	Nízká
10.03.2019	Server bude využívat technologie BTRFS Snapshots pro snadné obnovy	Brainstorming	Vysoká
10.03.2019	Bude založena nová federace id-test pro snadnější správu metadat	Brainstorming	Nízká
10.03.2019	Každý Poskytovatel služby má přístup do logů serveru, pokud o něj požádá	Konverzace se správci	Střední
10.03.2019	Na serveru bude volná politika uvolňování atributů, všem SP budou uvolňovány všechny atributy o které požádají	Konverzace se správci	Střední

Obrázek 4.2: Tabulka požadavků na testovací server (Zdroj: autor).

rizačního rozhodnutí - na základě role může být uživateli určena úroveň přístupu. Dle mého názoru zvýší tato funkce využitelnost testovacího IdP. Proto bude pro specifické potřeby aplikací možné měnit role a atributy testovacích identit, tedy přidávat uživatele do dalších skupin či přidávat hodnoty atributům. Po důkladném zvážení výhod a rizik jsem se rozhodl, že do budoucna bude správcům SP umožněno vytvářet si vlastní testovací identity dle svých potřeb. Výhodou takového rozhodnutí bude, že se sníží nároky na práci správce IdP, rizikem bude možnost neúmyslného poškození adresářové struktury LDAP serveru správci SP. Z tohoto důvodu jsem tomuto požadavku přiřadil střední prioritu a tato možnost bude zavedena do praxe až po dokončení příslušných návodů pro správce SP.

Pro zvýšení využitelnosti testovacího IdP bude do budoucna vytvořena webová stránka s popisem funkcí serveru a návody na jeho využití. Vytvoření webu není klíčové pro tento projekt, protože naprostá většina správců SP má dostatečné znalosti k využívání serveru bez potřeby návodů, avšak pro nově nastupující správce bude web vhodný. Proto jsem tomuto požadavku přiřadil střední prioritu. V případě nejasností bude před nasazením webové stránky k dispozici správce testovacího IdP.

Dalším požadavkem je skutečná podoba databáze serveru. Provozní servery id1 a id2 si mezi sebou synchronizují databázi, která uchovává údaje o Shibboleth instancích, uživatelských souhlasech a hodnotách některých atributů. Testovací server bude jednou denně synchronizovat databázi s provozními ID servery, což umožní v případě potřeby přepnout autentizaci z testovacího LDAPu na provozní LDAP server. Pokud se tedy v budoucnu rozhodneme testovat aplikace vůči skutečným uživatelům, bude přechod snadný. Protože se ale jedná o funkci, kterou server nepotřebuje ke svému fungování, přiřadil jsem tomuto požadavku nízkou prioritu.

Jelikož se jedná o testovací server, je potřeba počítat s chybami ze strany správců. Pro snadné obnovy byla kromě běžných záloh navíc zvolena technologie BTRFS Snapshots, která bude na serveru implementována. Tato technologie umožňuje snadné obnovování do vybraného bodu a její výhodou je malá náročnost na využití diskového prostoru. Snadné obnovování funkcí serveru má pro server vysokou prioritu, proto jsem i tento požadavek označil prioritou "Vysoká".

Pro správu metadat bude do budoucna vhodné vytvořit testovací metadatovou federaci. To bude mít i další výhody spojené s testováním neprovozní federace, jako je například využití již zmíněné služby DS3.3.1, díky čemuž nebude třeba významně zasahovat do nastavení Shibboleth SP – při nastavení aplikace na ověřování se vůči dvěma IdP je potřeba výrazně více změn v konfiguraci SP, než při zavedení služby DS - více na [32]. Proces tvorby federace je však časově náročný a federace není klíčová pro úspěšnost tohoto projektu – proto jsem určil nízkou prioritu požadavku.

Pro snadné hledání příčin problémů slouží logování. Všechny důležité procesy na testovacím ID serveru budou logovány a do budoucna bude umožněn přístup správců SP do serverových logů Shibboleth IdP, Tomcatu a Apache.

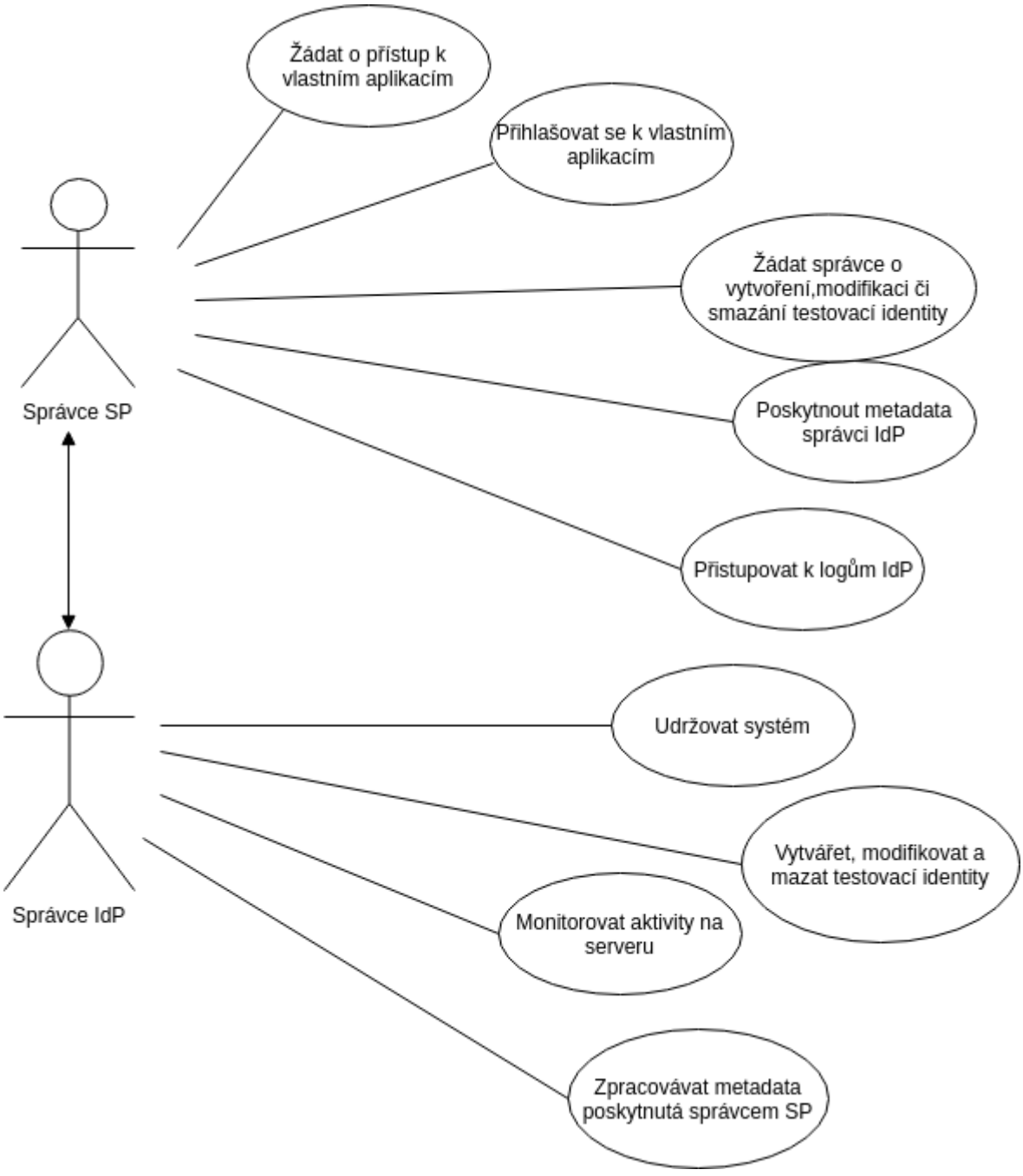
Běžná politika IdP serverů je striktní – poskytovatel služby dostane jen takové uživatelské atributy, které má důvod vyžadovat. Z důvodu testování aplikací bude na testovacím IdP serveru politika volná – všem SP budou uvolňovány všechny uživatelské atributy, o které si v metadatech požádají. Tento požadavek sice není klíčový pro využitelnost serveru avšak výrazně pomůže při testování. Proto jsem určil střední prioritu.

4.4 Diagram případů užití

Následující diagram případů užití ukazuje interakci správců SP a správců IdP s ostatními entitami systému, zatímco zároveň ilustruje typické scénáře užití testovacího serveru. Server má dva typy uživatelů – administrátor SP a administrátor IdP. Administrátor SP žádá s pomocí testovací identity server o přístup k vlastním aplikacím, poskytuje metadata svých aplikací správci IdP, žádá správce IdP o vytváření, modifikaci a smazání testovacích identit a přistupuje k logům IdP. Administrátor IdP udržuje systém, monitoruje aktivity na serveru, vytváří, modifikuje a maže testovací identity a zpracovává metadata poskytnutá správcem SP.

Běžný postup použití serveru tedy bude následovný:

- Správce SP kontaktuje správce IdP s žádostí o využívání testovacího IdP pro svou aplikaci. Zároveň dodá správci IdP metadata své aplikace k validaci a zpracování. Dále sdělí očekávanou dobu využívání testovacího serveru, tedy dobu po kterou je potřeba držet na serveru metadata aplikace.
- Správce IdP provede validaci metadat, při které zkontroluje úplnost a správnost jednotlivých XML elementů a pokud jsou validní, nahraje metadata do IdP a vytvoří o nich příslušný záznam. Aplikaci budou automaticky uvolněny všechny atributy. Poté pošle správce IdP metadata IdP serveru, údaje a doplňující informace o dostupných testovacích identitách správci SP a informuje ho o zapracování metadat. Správci SP je také správcem IdP umožněn přístup do logů IdP.
- Správce SP může od této chvíle testovat Shibboleth autorizaci s připojením na testovací IdP. V případě potřeby může nahlížet do logů IdP, například kvůli řešení chyb.
- V případě zvláštních požadavků na testovací identity kontaktuje správce SP správce IdP, který provede přidání či modifikaci testovacích identit.
- Správce IdP udržuje na IdP aktuální systém a kontroluje s pomocí monitoringu jednotlivé funkce. Monitoruje aktivity na serveru a v případě potřeby řeší problémy. Pokud je kontaktován správcem, který si přeje ukončit využívání testovacího IdP, vymaže správce IdP metadata ze serveru a informuje správce SP. Jednou za půl roku kontroluje správce IdP, zda všechna metadata zapracovaná v IdP stále slouží k testování aplikací.



Obrázek 4.3: Use case diagram užití serveru (Zdroj: autor).

5. Instalace IdP a jeho součástí

Cílem této kapitoly je demonstrovat zprovoznění navrženého testovacího IdP pro potřeby Centra informatiky VŠE a splnění požadavků analyzovaných v kapitole 4. Protože lze na internetu nalézt mnoho návodů na instalaci IdP, soustředí se tato kapitola především na specifika konkrétní implementace testovacího IdP pro VŠE. Na některé návody bude v průběhu kapitoly odkazováno, ale kapitola se zaměřuje na odlišnosti od běžných instalací a na splnění požadavků na testovací server.

Celý proces instalace, konfigurace a nasazení testovacího IdP serveru má několik částí. K fungujícímu Shibboleth IdP prostředí je potřeba za prvé fungující server. Na tomto serveru musí být nasazen webový server, v tomto případě Apache HTTP Server – tento webový server bude zprostředkovávat přihlašovací portál uživatelům testovaných aplikací. Tento server musí být správně nastaven a zabezpečen – zejména musí používat https spojení. Samotná aplikace Shibboleth IdP bude být umístěna v takzvaném servletu, o kterém blíže pojednává kapitola (5.2). Pro ověřování testovacích uživatelů bude být dle analyzovaných požadavků použit LDAP server, který je taktéž potřeba nakonfigurovat. Aplikace Shibboleth IdP bude být nastavena tak, aby dokázala úspěšně komunikovat se Správci služeb pomocí metadat. Také je potřeba nastavit politiku uvolňování atributů a nastavit metadata IdP. Podrobný postup při konfiguraci IdP lze nalézt na stránkách federace eduID [33].Po úspěšném nastavení uvedených komponent bude nastaveno monitorování a logování důležitých částí systému. Jednotlivé části procesu jsou dále podrobněji popsány.

5.1 Příprava serveru

Testovací IdP, jehož zprovoznění a analýza je cílem této práce, je umístěn ve vlastním virtuálním serveru. Aby mohl IdP správně fungovat musí být splněno několik požadavků na hardware a software. V době vzniku této práce je poslední verzí Shibboleth IdP verze 3.4 a pro tu jsou, podle oficiální dokumentace Shibboleth IdP[34], požadavky následující:

- Oracle Java nebo OpenJDK verze 8 nebo 11
- Servletový kontejner implementující Servlet API 3.0, oficiálně podporované jsou Tomcat verze 8 a vyšší a Jetty verze 9.2 a vyšší
- Shibboleth nemá specifické požadavky na operační systém, autoři pouze doporučují použít Linux, Windows nebo OS X

Na hardware nejsou stanoveny konkrétní požadavky. Při volbě hardware pro funkční IdP server je ale nutné vzít v potaz potřebu záloh, potřebu rychlých startů a dalších služeb, které budou na serveru s IdP koexistovat – například webový server, LDAP server a podobně. Pro testovací IdP server, jehož příprava je popsána v této kapitole, jsem zvolil varianta virtualizace. Server má alokovaný diskový prostor 40 GB, operační paměť 4 GB a jeden

dvoujádrový procesor. Server je také připojen do školní sítě virtuálním síťovým adaptérem.

Softwarová architektura této konkrétní implementace Shibboleth IdP zahrnuje několik aplikací, každá z nich hraje důležitou roli v celém procesu. Chyba jednotlivých elementů může způsobit nefunkčnost IdP. Následuje výčet softwaru použitého při instalaci testovacího IdP.

1. Linuxový operační systém Debian verze 9.7 Stretch
2. OpenJDK verze 8 jako Java kompilátor
3. Apache2 jako webový server
4. Apache Tomcat 8.5 jako Java servlet
5. Shibboleth Identity Provider verze 3.4
6. OpenLDAP server verze 2.4 jako adresářový server pro registraci a uchovávání uživatelů, která se připojí k IdP pro autentizaci
7. PostgreSQL verze 9.6 jako systém řízení báze dat

Pro snazší správu, zálohování, splnění požadavků z kapitoly 4 a další podpůrné činnosti jsem použil další, nekritický software. Následuje výčet důležitějších balíčků z této kategorie:

- btrfs-progs – obsahuje nástroje pro správu použitého BTRFS souborového systému
- htop – přehledně vypisuje běžící procesy, jejich vlastníky, využívané CPU a podobně
- network-manager – umožňuje snadnou správu sítě
- iptables-persistent – ukládá trvale nastavení firewallu, bez kterého by se neměl IdP obejít
- snapper – umožňuje vytváření snapshotů, jejichž princip je popsán níže
- tail – nástroj pro sledování logů v reálném čase
- ccze – nástroj pro zpřehlednění logových zpráv pro snazší debugging
- dnsutils – balíček několika nástrojů užitečných při zjišťování chyb sítě
- xymon-client – balíček umožňující monitoring serveru
- ldap-utils – nástroje pro ovládání LDAPu

5.2 Nastavení servletu

Ačkoliv je autory Shibbolethu doporučován spíše servlet Jetty[34], pro testovací IdP jsem zvolil Apache Tomcat. K tomuto rozhodnutí vedl mimo jiné fakt, že Tomcat je v současné době nejpoužívanější servletový kontejner, což znamená zejména snazší debugging – je velká šance, že pokud narazím na problém, někdo už ho řešil přede mnou. Tomcat je také využíván na produkčních IdP serverech, zachováváme tím tedy jednotnost prostředí. Apache Tomcat je open source implementace Java Servlet kontejneru vyvíjená Apache Software Foundation.[35] Ve zkratce funguje tak, že administrátor vytvoří WAR (zkratka pro Webový ARchiv) a umístí ho do správného adresáře v Tomcatu. Tomcat poté zařídí zbytek a pokud je WAR soubor v pořádku, aplikace běží na HTTP serveru. Pro funkční IdP postačí balíček `tomcat8-user`. Po jeho instalaci běžným způsobem je třeba vytvořit uživatelskou instanci - za uživatele, který bude zodpovědný za správu Shibboleth IdP, není rozumné vytvářet instanci jako uživatel

root. Je potřeba jen specifikovat instalační adresář (z důvodu přehlednosti byl zvolen adresář `/opt/tomcat8-shibboleth`), HTTP port (výchozí je 8080), a cestu k java instalaci. Příkaz `tomcat8-instance-create /opt/tomcat8-shibboleth/` vytvoří instanci na zvoleném místě, poté už stačí instanci spustit pomocí scriptu `./bin/startup.sh`, který se nachází v adresáři instance. Pro automatické spouštění, snadné restartování a další manipulace s Tomcat instancí je dobré vytvořit `system.d` jednotku, která může vypadat například takto:

```
[Unit]
Description=Tomcat 8.5 servlet container for shibboleth IdP
After=network.target

[Service]
Type=forking
PIDFile=/opt/tomcat8-shibboleth/temp/tomcat.pid

User=shibboleth-idp
Group=shibboleth-idp

Environment="JAVA_HOME=/usr/lib/jvm/default-java"
Environment="JAVA_OPTS=-Djava.awt.headless=true"

Environment="CATALINA_BASE=/opt/tomcat8-shibboleth"
Environment="CATALINA_HOME=/usr/share/tomcat8"
Environment="CATALINA_PID=/opt/tomcat8-shibboleth/temp/tomcat.pid"
Environment="CATALINA_OPTS=-Xms512M -Xmx1556M -server"

ExecStart=/opt/tomcat8-shibboleth/bin/startup.sh
ExecStop=/opt/tomcat8-shibboleth/bin/shutdown.sh

[Install]
WantedBy=multi-user.target
```

Úspěšnou instalaci lze ověřit na `http://localhost:8080` - pokud vše funguje správně, zobrazí se Tomcat domovská stránka.

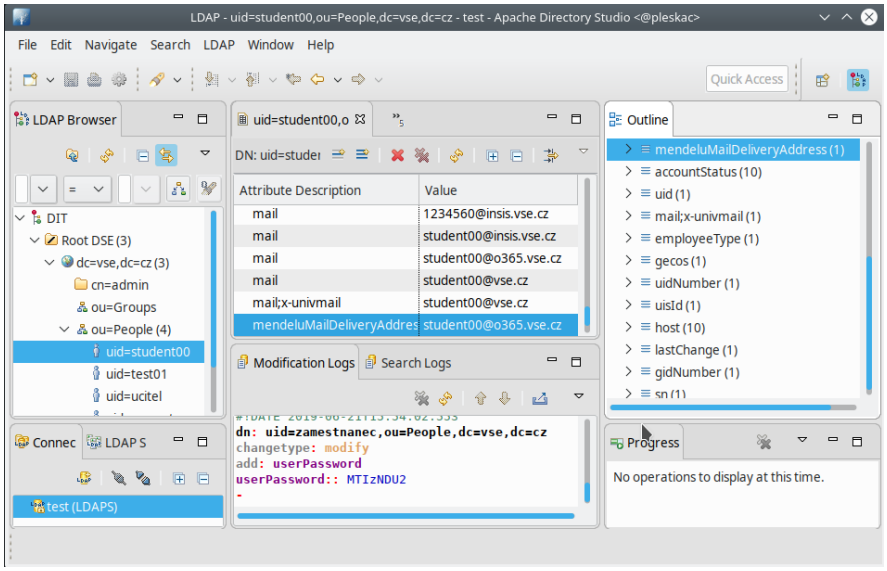
5.3 Nastavení LDAP

Samotný Shibboleth neautentizuje uživatele. Používá k tomu externí autentizační systémy. V kapitole 4 bylo určeno, že server má mít vlastní LDAP server na správu testovacích uživatelů a tato kapitola popisuje jeho nasazení.

LDAP je zkratkou pro Lightweight Directory Access Protocol. Tento protokol slouží k ukládání a přístupu k datům na LDAP serveru, který funguje na principu stromově uspořádaného adresáře. Podle jednoznačného identifikátoru, který musí každý záznam obsahovat, v něm lze snadno najít informace o uživatelích.[36]

Pro testovací IdP jsem zvolil implementaci OpenLDAP. Jedná se o open souce implementaci LDAPu, která obsahuje LDAP server a knihovny potřebné k jeho používání. Jedním z důvodů, proč jsem se rozhodl pro OpenLDAP je fakt, že i provozní školní LDAP server používá tuto implementaci. Právě skripty, kterými byla vystavěna struktura školního LDAPu byly použity k vytvoření struktury LDAPu testovacího serveru. Díky stejné struktuře bude testování aplikací testovacími uživateli zaručovat stejné chování v případě přihlášení skutečného uživatele. Po nainstalování `slapd` (OpenLDAP) a `ldap-utils` ještě nainstalujeme balíček `python-ldap3`, tedy klientskou LDAP knihovnu pro jazyk Python. Poté je potřeba nastavit korektně použití certifikátů a spustit připravené skripty, které vybudují strukturu LDAP adresáře. LDAP jsem nakonfiguroval na používání portu 636, který je LDAPem používán pro zabezpečené SSL spojení. Příslušnému portu jsem následně i povolil průchod přes firewall po školní síti.

Pro správu LDAPu jsem zvolil nástroj `Apache Directory Studio`. Ukázku úspěšného připojení LDAPu a zmíněného nástroje zobrazuje obrázek 5.1

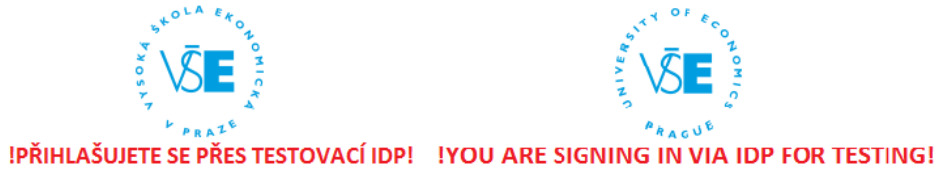


Obrázek 5.1: Apache Directory Studio rozhraní (Zdroj: autor).

5.4 Instalace a konfigurace IdP

Samotná instalace IdP se ukázala být velmi snadná. Stačí stáhnout instalační balíček s poslední verzí IdP, ten pomocí příkazu `tar -xzvf` rozbalit a spustit instalační skript, který se nachází v rozbaleném adresáři (`/bin/install.sh`). Instalační skript se ptá na cílový adresář instalace – zvolil jsem adresář `/opt/shibboleth-idp`, na hostname pro náš IdP server – ser-

veru bylo přiděleno jméno `id-test.vse.cz` a `entityID`, tedy označení samotného IdP. Dále je potřeba zadat název domény pod kterou server spadá, což je v našem případě `vse.cz` a heslo pro klíč používaný pro šifrování cookies. Zbytek důležitých úprav IdP, jako je nahrání knihoven, nebo vynucení SSL spojení se nijak neliší od běžné instalace IdP, proto postačí odkázat na použitý návod – [37]. Velká část konfiguračních souborů byla pro požadovanou podobnost nastavení testovacího prostředí s provozními IdP servery převzata ze serverů Id1 a Id2. Po doladění drobností podle zmíněného návodu je třeba zařídit českou lokalizaci. Shibboleth IdP nemá zabudovaný český překlad, proto jsem do adresáře `/opt/shibboleth-idp/messages/` umístil soubor pro českou lokalizaci, pojmenovaný `messages_cs.properties`, který jsem převzal z produkčního IdP serveru. Tento soubor zároveň poskytuje informace pro přihlašovací rozhraní Shibboleth IdP. Součástí rozhraní je i identifikace organizace, ale protože se jedná o testovací server VŠE, pouhé logo školy by mohlo být matoucí. Vytvořil jsem tedy českou a anglickou variantu obrázku, který obsahuje jednak oficiální logo školy v příslušném jazyce a jednak upozornění, že se jedná o testovací IdP prostředí. Výsledek je vidět na obrázku 5.2.



Obrázek 5.2: Upravený identifikátor organizace (Zdroj: autor).

Po tomto kroku jsem vytvořil WAR archiv, zmíněný v 5.2, pomocí skriptu `build.sh`, který je součástí instalace Shibboleth IdP.

5.4.1 Nastavení politiky uvolňování atributů.

Shibboleth IdP, jakožto server starající se o bezpečí uživatelských informací, má běžně striktní politiku. V případě provozních serverů VŠE (Id1 a Id2) se aplikacím uvolňují pouze ty atributy, které má aplikace rozumný důvod požadovat. Důvody posuzuje správce IdP. V případě testovacího IdP jsem však zvolil volnou politiku uvolňování atributů. Jednak to byl jeden z požadavků popsaných v kapitole 4, jednak není důvod v případě testovacích identit dbát na bezpečnost uživatelských údajů.

Politika se v Shibboleth IdP nastavuje zejména v souboru `attribute-filter.xml`, který lze nalézt v adresáři `conf` Shibboleth instalace. U většiny atributů jsem zvolil způsob vydávání "per request", tedy každá aplikace dostane takové atributy, o jaké požádá v metadatech. Výpis všech vydávaných atributů, které VŠE podporuje, lze nalézt na stránkách Id serveru – `https://id.vse.cz/atributy.html`. Politiku jsem u většiny atributů nastavil následujícím způsobem:

```
<AttributeRule attributeID="givenName">
      <PermitValueRule xsi:type="MappedAttributeInMetadata"
      onlyIfRequired="true"
```

```
        matchIfMetadataSilent="false"/>
    </AttributeRule>}
```

Po změnách v souboru `attribute-filter.xml` jsem spustil následující skript s příslušným parametrem pro opětovné načtení konfigurace politik:
`./bin/reload-service.sh -id shibboleth.attributeFilterService`.

5.4.2 Tvorba metadat

Při korektně provedené instalaci Shibboleth IdP jsou metadata IdP vygenerována automaticky. I přesto je potřeba udělat několik úprav. Metadata IdP se nacházejí v souboru `metadata/idp-metadata.xml`. Klíčovými elementy metadat jsou entityID, název domény, šifrovací a podepisovací certifikát, popis služby a nastavení *vazeb* (popsaných v kapitole 2.2). Metadata také mohou obsahovat doplňující informace o službě, o organizaci která ji provozuje a kontaktní informace na správce. Tyto informace jsem do metadat doplnil. Metadata jsou následně dostupná ke stažení na URL adrese `https://id-test.vse.cz/idp/shibboleth`. Tato metadata nahrát na příslušné umístění dle individuální Shibboleth SP konfigurace každá aplikace, která chce testovací IdP používat.

5.5 Zálohování

Důležitým tématem v souvislosti s IdP je zálohování. Jednak kvůli obnovám při výpadcích, při chybách administrátorů, nebo jiných neočekávaných událostech, i testovací server by měl být nějakým způsobem zálohován. Stejně jako všechny důležité univerzitní servery i testovací IdP server je kompletně zálohován na zálohovací server školy. Kromě toho jsem zde k zálohování použil technologie BTRFS Snapshots, kterou umožňuje souborový systém BTRFS. „Jedná se o moderní CoW (copy on write) linuxový souborový systém, jehož cílem je implementace nejmodernějších funkcionalit, zatímco se soustředí na snadnou administraci, toleranci chyb a jejich opravy.“[38]

BTRFS Snapshot je v podstatě speciální typ podsvazku, takzvaného BTRFS Subvolume. BTRFS subvolume je nezávislý, připojitelný podsvazek (nejedná se ale o blokové zařízení), který sdílí svá data (a metadata) s nějakým dalším podsvazkem, za použití CoW možností BTRFS.[39] Jedná se o kopii podsvazku, bez zahrnutí nadřazených či podřazených podsvazků. Jakmile je vytvořen snapshot, není žádný rozdíl mezi stavem originálního podsvazku a nového snapshot podsvazku. Snapshot ale nekopíruje soubory, pouze sdílí data a metadata originálního podsvazku, což je zároveň prostorově nenáročné a extrémně rychlé. Funguje nezávisle na originálním podsvazku, takže přidáním souborů do snapshotu neovlivníme originální podsvazek. Lze také vytvářet snapshoty snapshotů, každý bude fungovat jako samostatná jednotka.[40] Tato práce nezachází do detailů nastavování serveru, více informací o implementaci BTRFS Snapshots lze nalézt na [39]. BTRFS Snapshots tedy umožní snadno a rychle obno-

vit server do stavu před chybou, bez velkých nároků na paměť. V případě větších problémů, například selhání fyzického serveru, bude využita záloha ze zálohovacího serveru.

5.6 Monitorování a logování

Monitorování spočívá v neustálé kontrole důležitých částí systému. Jedná se o velmi důležitou činnost u serverů, které mají samostatně a dobře sloužit svému účelu. I testovací IdP server je tedy třeba monitorovat. K monitorování je na VŠE používán nástroj Xymon, jehož hlavní výhodou je přehledné rozhraní, možnost grafů a reportování a také se jedná o open-source software.

Monitoring provádí vzdálený server, na který je potřeba dodat z testovacího IdP potřebné informace. K tomu slouží již zmíněný balíček `xymon-client`. Po jeho instalaci stačí zadat IP adresu monitorovacího serveru a data jsou automaticky odesílána. Pro rozšířené možnosti monitorování byl nainstalován ještě balíček `hobbit-plugins`. V tuto chvíli sice testovací server informace posílá, je však ještě potřeba informovat správce monitorovacího serveru a požádat ho o nastavení monitorování podle následujících požadavků. Ty jsou derivací monitorovacích požadavků produkčních IdP serverů. Rozdílem je potřeba monitorovat fungování LDAPu a menší množství aplikací na serveru.

Monitorovací požadavky jsem stanovil následovně:

1. Požadavky na monitorování síťových portů
    - 22 musí poslouchat na všech rozhraních – port 22 používá SSH, tedy protokol, přes který je spravován server
    - 80 musí poslouchat na všech rozhraních – port pro http protokol
    - 443 musí poslouchat na všech rozhraních – port pro https protokol
    - 5432 musí poslouchat místní smyčce – tento port je využíván databází PostgreSQL
    - 636 musí poslouchat na všech rozhraních – přes tento port je spravován LDAP server
2. Požadavky na monitorování procesů serveru
    - sshd – proces protokolu SSH
    - postgres – proces databáze
    - apache2 – webový server
    - rsyslogd – proces zajišťující logování
    - systemd-timesyncd – proces, který aktualizuje čas serveru
    - tomcat8-shibboleth – java servlet, ve kterém se nachází Shibboleth IdP
    - nsrexecd – proces posílající data zálohovacímu serveru
    - slapd – LDAP server
3. Monitorování souborů
    - `/var/log/messages/` – tento soubor musí existovat a mít nenulovou velikost

Dále je ještě monitorován stav aktuálnosti serveru, fungování LDAP serveru (kontroluje se

pravidelným vyhledáváním v LDAP databázi), stav ssl certifikátu a fungování SSH. V případě, že některá z uvedených sledovaných věcí není v pořádku, zasílá Xymon e-mail zaměstnancům příslušného oddělení se zprávou o problému.

Logování je řešeno nástrojem rsyslog. Ten dokáže při správném nastavení konvertovat textový vstup do formátu `syslog message` a výsledný soubor posílat na centrální logovací server VŠE. Nastavení jsem provedl v hlavní konfiguraci nástroje, v souboru `/etc/rsyslog.conf`. Tam jsem nahrál modul `imfile`, který zajistí převod textu na logový soubor a nastavil jsem odesílání na centrální logovací server.

6. Testování

Předchozí kapitola se věnovala implementační části projektu. Jejím výstupem byl testovací IdP server s funkčním Shibboleth IdP a LDAP serverem. Cílem této kapitoly je otestovat funkce testovacího IdP serveru jako celku a prokázat jeho připravenost k nasazení do provozu v rámci infrastruktury CI VŠE.

6.1 Testovací strategie

Protože práce pracuje s již hotovým softwarem, vycházíme z předpokladu, že jednotkové testování již není potřeba. Zvolenou testovací strategií je tedy Systémové testování, jinak také Testování černé skříňky. Tento typ testování je používán v případech kdy není k dispozici přístup k programovému kódu.[41] Nebudeme tedy testovat samotný Shibboleth IdP, ale zaměříme se na otestování zda jednotlivé systémové komponenty testovacího serveru, jako jsou Shibboleth IdP, LDAP, Tomcat a podobně, byly nainstalovány, implementovány a nakonfigurovány úspěšně. Testování má prokázat, že systém splňuje požadavky z kapitoly 4. Hlavní částí testování je ověření, zda je server připraven fungovat podle diagramu případů užití.

6.2 Plán testování a jeho provedení

Aby server vykonával požadované funkce, je třeba otestovat následující aplikace:

- Shibboleth IdP
- PosgreSQL
- Apache2
- Tomcat

Tyto aplikace byly úspěšně nakonfigurovány v kapitole 5 a tedy předpokládáme, že byly nainstalovány a nakonfigurovány úspěšně. Jejich fungování můžeme kontrolovat v rozhraní monitorovací aplikace `xymon`.

Druhá, důležitější část testování má ověřit, zda celá implementace pracuje správně, či nikoliv. Zprovoznění Shibbolethu je více o správném jednotlivých spolupracujících částí, než o pouhé instalaci aplikace. Všechny části musí být připraveny komunikovat správným způsobem mezi sebou.

V rámci testování jsem v adresářové struktuře LDAP vytvořil vzorové uživatele, jednoho podle vzoru typického vyučujícího VŠE, jednoho po vzoru typického studenta a jednoho po vzoru nevyučujícího zaměstnance. Uživatele jsem vytvořil exportováním skutečných uživatelů z provozního LDAP serveru a jejich následnou anonymizací. Uživatelé tedy mají atributy

typické pro své zařazení. Cílem testu je úspěšné přihlášení testovacích uživatelů k aplikaci a poskytnutí všech dostupných atributů. Jako testovací aplikaci jsem použil Prohlížeč atributů na serveru `atributy.vse.cz`. Tento SP jsem upravil tak, aby si uživatel mohl vybrat mezi ověřením přes testovací IdP a provozní IdP.

Scénář je následující: Proces předpokládá, že proběhla simulovaná komunikace mezi SP a správcem testovacího IdP a byla úspěšně zapracována validní metadata jak na straně IdP (metadata SP), tak na straně SP (metadata IdP). Dalším předpokladem je, že uživatel se přihlašuje poprvé. Cílem uživatele je získání přístupu k následujícím zdrojům:

- index.html (rozcestník aplikace)
- printenv.cgi (výpis atributů a dalších informací v jazyce CGI)
- printenv.php (výpis atributů a dalších informací v jazyce PHP)

Úspěšně autorizovaný uživatel musí získat přístup k výše vypsaným zdrojům a jeho uvolněné atributy by měly být zobrazeny při spuštění skriptů v souborech `printenv.cgi` a `printenv.php`

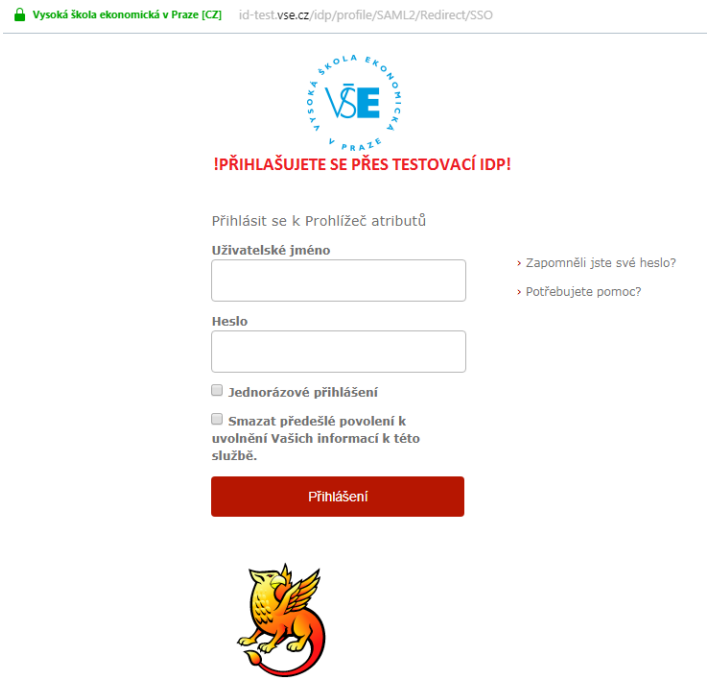
Následuje příklad atributů vytvořeného testovacího uživatele. Uživatel Ben Student má získat přístup k výše vypsaným chráněným souborům a mají mu být na požádání zobrazeny jeho atributy. Kompletní podoba uživatelů je v příloze A.

```
cn: Ben Student
gidNumber: 1000
homeDirectory: /home/student01
sn: Student
uid: student00
uidNumber: 1001
userPassword: 123456
```

Uživatel Ben Student se s pomocí libovolného internetového prohlížeče pokusí vyžádat přístup k chráněným zdrojům (výše zmíněným aplikacím). Uživatel spustí žádost přistoupením k url `https://atributy.vse.cz/secure2`. Poskytovatel služby přesměruje uživatelskou žádost na testovací IdP server k ověření zda již existuje uživatelské sezení s atributy uživatele, pokud ano, je uživateli udělen přístup k chráněným zdrojům. Pokud ne, Poskytovatel služby pošle Poskytovateli identity autentizační stránku, kterou SP pošle k zobrazení prohlížeči uživatele. Ověřovací stránka je zobrazena na obrázku 6.1.

Uživatel vloží své přihlašovací údaje (`student00 a 123456}`. IdP porovná údaje s LDAP databází – ověří, zda existuje uživatel s `uid= student00` a zda k uživateli patří zadané heslo. Při vložení nesprávného hesla je uživateli zamítnut přístup a uživatel je informován o nesprávnosti hesla, jak lze vidět na obrázku 6.2.

Pokud uživatel vloží správné údaje, Poskytovatel služby obdrží od IdP potvrzení o úspěšné autentizaci uživatele. Uživatel je následně informován o uvolňovaných atributech a jejich



Obrázek 6.1: IdP autentizační stránka (Zdroj: autor).

Obrázek 6.2: Zamítnutý přístup IdP (Zdroj: autor).

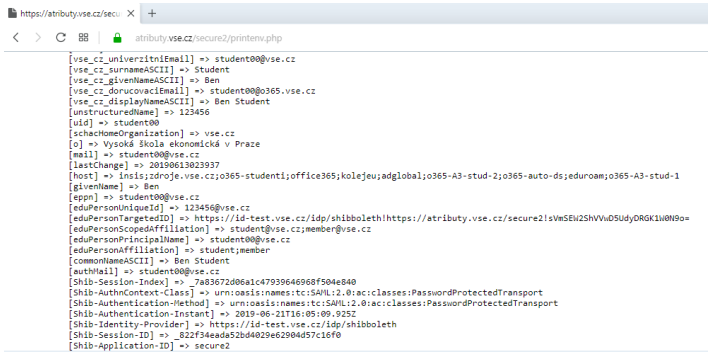
hodnotách. Pokud dá uživatel souhlas s uvolněním atributů, je přesměrován zpět na SP, který podle poskytnutých atributů posoudí úroveň přístupu. Po autorizaci získává uživatel přístup k chráněným zdrojům.

Obrázek 6.3: Úspěšné přihlášení (Zdroj: autor).

Obrázek 6.3 je výsledkem úspěšného testu přihlášení testovacího uživatele. Rozkliknutím prvního odkazu na zobrazené stránce získá uživatel přístup ke zdroji `printenv.cgi`, rozkliknutím druhého odkazu pak ke zdroji `printenv.php`. Uživatel se nyní může pohybovat po stránce bez potřeby vkládat znovu přihlašovací údaje.

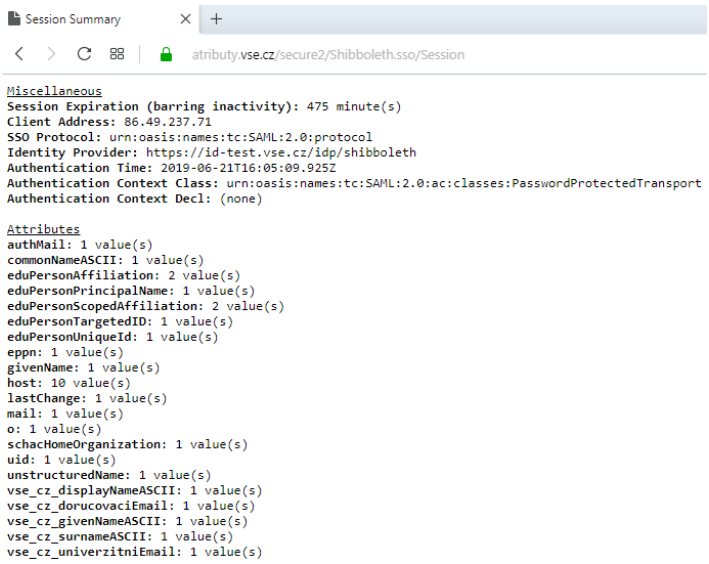
Obrázek 6.4: Přístup ke zdroji printenv.cgi (Zdroj: autor).

Na obrázku 6.4 je vidět část výpisu skriptu `printenv.cgi`, ke kterému uživatel přistoupil. Výpis obsahuje výčet atributů, které aplikace obdržela díky IdP.



Obrázek 6.5: Přístup ke zdroji printenv.php (Zdroj: autor).

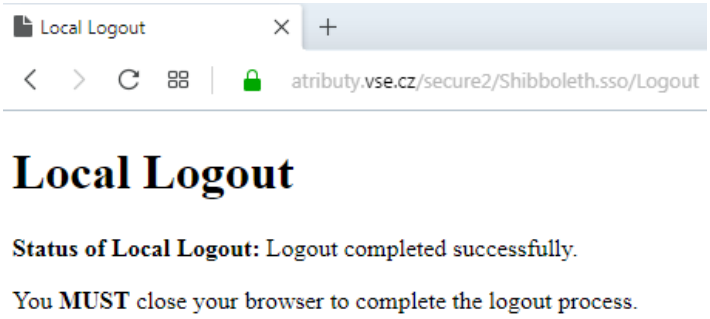
Obrázek 6.5 zobrazuje úspěšné vypsání atributů, tentokrát v jazyce PHP. Ověřili jsme tedy přístup k druhému chráněnému zdroji – `printenv.php`.



Obrázek 6.6: Výpis obsahu Shibboleth sezení (Zdroj: autor).

Na obrázku výše (6.6) lze vidět obsah Shibboleth sezení, který je dostupný (po úspěšném SSO přihlášení) na URL adrese `https://atributy.vse.cz/secure2/Shibboleth.sso/Session`. Lze z něho vyčíst například dobu po kterou bude sezení drženo v paměti, adresu IdP a množství hodnot jednotlivých získaných atributů.

Nakonec bylo otestováno odhlášení od zdroje. Kliknutím na poslední odkaz v testovací aplikaci byl uživatel odhlášen a sezení skončilo. Po úspěšném odhlášení je ale třeba zavřít prohlížeč. Ukončení sezení totiž pouze zašle žádost Poskytovateli služby, aby přestal udržovat všechna otevřená sezení. Jediný mechanizmus jak se úplně odhlásit z aplikace je tedy zavření prohlížeče. Následující obrázek manifestuje úspěšné odhlášení.



Obrázek 6.7: Úspěšné odhlášení z aplikace (Zdroj: autor).

Výše uvedený testovací scénář ukazuje úspěch implementace testovacího serveru s Shibboleth IdP prostředím. Došlo k úspěšnému ověření fungování Shibboleth IdP, jeho interakci s testovacím SP a databází LDAP. Testování dokázalo připravenost testovacího serveru na nasazení do provozu v rámci infrastruktury CI VŠE.

Závěr

Tématem této práce byla Analýza testovacího prostředí Shibboleth IdP pro potřeby Centra informatiky Vysoké školy ekonomické. Kvůli nedostatečné a hlavně nesrozumitelné dokumentaci Single sign-on aplikace Shibboleth jsem se rozhodl technologii analyzovat, srozumitelně popsat a na základě zjištěných poznatků vytvořit testovací IdP server, který má sloužit potřebám CI VŠE a zejména správcům aplikací, kteří v době započetí tohoto projektu neměli způsob jak pohodlně testovat přihlašování do svých aplikací.

V první části práce jsem se zabýval teoretickou stránkou projektu. V kapitole 1 jsem s pomocí rešerše literatury vysvětlil principy řízení přístupu ve webových službách. Klíčovým bodem této kapitoly vysvětlení pojmu Single sign-on, tedy autentizační metody, která umožňuje uživatelům získat přístup k vícero službám bez potřeby pokaždé vkládat přihlašovací údaje. V kapitole 2 jsem se soustředil na stavební kameny Shibbolethu, tedy koncepty, ze kterých Shibboleth vychází. Těmito koncepty je zejména jazyk XML – značkovací jazyk, který se stal populárním zejména kvůli nezávislosti na platformě, jednoduchosti a flexibilitě, dále pak SAML standard, který umožňuje přenášet bezpečnostní informace vyměňované při autentizaci a autorizaci. Díky zmiňované rešerši jsem v této kapitole srozumitelně popsal XML mechanizmy tvořící SAML standard, jako jsou *tvrzení* – balíky informací, které obsahují prohlášení vydaná SAML autoritou, nebo *vazby* – pravidla na použití tvrzení se standardy pro transport a posílání zpráv. V kapitole 3 jsem se věnoval samotnému Shibbolethu. Mimo jiné jsem popsal, že Shibboleth je na standardech založená open source aplikace pro webový Single sign-on a že se jako každý webový SSO systém skládá z webového prohlížeče, zdroje, ke kterému uživatel přistupuje, Poskytovatele identity a Poskytovatele služby. Popsal jsem, jak Shibboleth funguje a jak mezi sebou tyto části interagují, aby vytvořily bezpečný SSO přihlašovací mechanizmus.

V druhé části jsem se zaměřil na praktickou aplikaci nově nabytých znalostí. Nejprve jsem popsal současnou infrastrukturu Shibbolethu na VŠE. Poté jsem metodou rozhovorů[3] a s pomocí brainstormingu shromáždil požadavky na testovací IdP prostředí. Jako nejdůležitější jsem vyhodnotil například požadavky na přítomnost testovacích identit, kterými se lze ověřovat vůči testovacímu serveru, a také na co největší podobnost konfigurace testovacího IdP s konfiguracemi na provozních IdP serverech. Na základě shromážděných požadavků jsem vytvořil návrh, jehož plánované použití jsem znázornil diagramem případů užití. V kapitole 5 jsem se zaměřil na popis implementace návrhu z předcházející kapitoly, tedy zejména na přípravu serveru, instalaci IdP a na splnění požadavků, tedy například na tvorbu testovacích uživatelů. V poslední, 6. kapitole jsem nejprve zvolil testovací strategii – vybral jsem Systémové testování, a následně jsem otestoval fungování Shibbolethu prostřednictvím přihlášení testovací identity na testovací aplikaci Prohlížeč atributů. Testování ukázalo, že testovací IdP server bez problémů identifikuje uživatele, ověří ho vůči databází LDAP a uživatel je schopen se dostat k chráněným zdrojům v aplikaci.

Na počátku práce jsem si stanovil následující cíle:

- Na příkladu technologie Shibboleth a standardu SAML vysvětlit principy řízení přístupu ve webových aplikacích.
- Na základě analýzy požadavků CI VŠE navrhnout vhodnou konfiguraci testovacího prostředí Shibboleth IdP.
- Navržené řešení testovacího prostředí zprovoznit a ověřit.

První cíl jsem splnil rešerší dostupné literatury v první části práce. Výsledkem splnění prvního cíle je komplexní příručka, která poskytuje čtenáři dostatek informací k pochopení principů technologie Shibboleth a souvisejících pojmů, ve stravitelné formě. Druhá, praktická část práce popisuje splnění zbylých dvou cílů. Požadavky byly shromážděny a analyzovány a na jejich základě vzniklo testovací Shibboleth IdP prostředí, které jsem zprovoznil a ověřil. Výsledkem je funkční testovací IdP, které bude v následujících letech sloužit zejména správcům aplikací a Poskytovatelům služeb ke snazšímu testování aplikací. Přínosem práce je pak i vytvoření testovacích identit, které budou správcům SP k dispozici.

Po odevzdání této práce bude projekt testovacího IdP prostředí nadále pokračovat. V následujících měsících se budu soustředit zejména na splnění cílů, kterým jsem nepřiřadil vysokou prioritu – tedy například na vytvoření webového portálu s návody, jak testovací server používat a jak postupovat při žádosti o využívání testovacího IdP. Testovací IdP bude bezprostředně po odevzdání této práce uvedeno do zkušebního provozu – správci služeb v rámci VŠE budou informováni o jeho nasazení, o jeho funkcích a možnostech, které jim testovací IdP nabízí. Teoretická práce bude sloužit jako příručka zejména pro zaměstnance Centra informatiky, kteří chtějí proniknout do principů technologie Shibbolethu a SAMLu.

Seznam použité literatury

1. *What's Shibboleth.* Dostupné také z: https://www.shibboleth.net/index/. [Online; accessed 05-May-2019].
2. FINK, Arlene. *Conducting research literature reviews : from the internet to paper.* Thousand Oaks, California: SAGE, 2014. ISBN 9781452259499. [Online; accessed 03-May-2019].
3. ŠVARCOVÁ-SLABINOVÁ, Iva. *Základy pedagogiky.* Vydavatelství VŠCHT, 2005. ISBN 978-80-7080-690-6.
4. VINCENT HU David Ferraiolo, Richard Kuhn. *Assessment of Access Control Systems.* 2006. Dostupné také z: https://csrc.nist.gov/publications/detail/nistir/7316/final. [Online; accessed 10-March-2019].
5. GRASSI; PAUL; GARCIA; MICHAEL; FENTON; JAMES. *Digital Identity Guidelines.* 2017. Dostupné také z: https://csrc.nist.gov/publications/detail/sp/800-63/3/final. [Online; accessed 10-March-2019].
6. ROSENBERG, Jonathan. *Securing Web services with WS-Security.* Indianapolis, IN: SAMS, 2004. ISBN 06-723-2651-5.
7. *Token.* Dostupné také z: https://it-slovnik.cz/pojem/token/?utm_source=cp&utm_medium=link&utm_campaign=cp. [Online; accessed 03-March-2019].
8. KILLORAN, John. *What Is Single-Sign-On Authentication.* 2018. Dostupné také z: https://swoopnow.com/sso-authentication/.
9. WASSON, Mike. *Authentication and Authorization in ASP.NET Web API.* 2012. Dostupné také z: https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/authentication-and-authorization-in-aspnet-web-api. [Online; accessed 13-March-2019].
10. KILLORAN, John. *SSO Authentication Process.* 2017. Dostupné také z: https://swoopnow.com/wp-content/uploads/2017/11/sso-authentication-process.png.
11. KOSEK, Jiří. *XML schémata.* 2014. Dostupné také z: https://www.kosek.cz/xml/schema/. [Online; accessed 02-March-2019].
12. TIDWELL, Doug. *Introduction to XML.* 2002. Dostupné také z: https://www.ibm.com/developerworks/xml/tutorials/xmlintro/xmlintro.html. [Online; accessed 03-March-2019].
13. BRAY, Tim; HOLLANDER, Dave; LAYMAN, Andrew; TOBIN, Richard; THOMPSON, Henry S. 2009. Dostupné také z: https://www.w3.org/TR/xml-names/#concepts. [Online; accessed 15-March-2019].
14. BARTEL, Mark; BOYER, John; FOX, Barb; LAMACHIA, Brian; SIMON, Ed. 2013. Dostupné také z: https://www.w3.org/TR/xmldsig-core/#sec-Introduction. [Online; accessed 03-March-2019].
15. SIMON, Ed; MADSEN, Paul; ADAMS, Carlisle. 2001. Dostupné také z: https://www.xml.com/pub/a/2001/08/08/xmldsig.html#sigs. [Online; accessed 05-March-2019].

16. *XML Encryption*. Dostupné také z: https://www.ibm.com/support/knowledgecenter/en/SS7JFU_8.5.5/com.ibm.websphere.express.doc/ae/cwbs_encryptv6.html. [Online; accessed 15-March-2019].

17. PETTERS, Jeff. *What is SAML and How Does it Work?* 2018. Dostupné také z: https://www.varonis.com/blog/what-is-saml/. [Online; accessed 09-March-2019].

18. *SAML wiki*. Dostupné také z: https://wiki.oasis-open.org/security/FrontPage. [Online; accessed 03-March-2019].

19. BRECHLEROVÁ, Dagmar. SAML a XACML jako nová cesta pro Identity management. In: 200. Dostupné také z: http://invenio.nusl.cz/record/37836. [Online; accessed 03-March-2019].

20. RAGOUZIS, Nick; HUGHES, John; PHILPOTT, Rob; MALER. *Security Assertion Markup Language (SAML) V2.0 Technical Overview*. 2008. Dostupné také z: https://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0-cd-02.pdf. [Online; accessed 11-March-2019].

21. CANTOR, Scott; KEMP, John; PHILPOTT, Rob; MALER, Eve. *Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0*. 2005. Dostupné také z: https://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf. [Online; accessed 06-May-2019].

22. *How Shibboleth Works: Basic Concepts*. Dostupné také z: https://www.shibboleth.net/index/basic/. [Online; accessed 25-March-2019].

23. SCOTT CANTOR T. Zeller, I. YOUNG. *Shibboleth Concepts*. Dostupné také z: https://wiki.shibboleth.net/confluence/display/CONCEPT. [Online; accessed 03-February-2019].

24. *Service Provider*. Dostupné také z: https://www.shibboleth.net/products/service-provider/. [Online; accessed 13-February-2019].

25. *Identity Provider*. Dostupné také z: https://www.shibboleth.net/products/identity-provider/. [Online; accessed 03-May-2019].

26. CANTOR, Scott. *Shibboleth Architecture: Protocols and Profiles*. 2005. Dostupné také z: https://www.immagic.com/eLibrary/ARCHIVES/TECH/INTRNET2/I050920C.pdf. [Online; accessed 03-February-2019].

27. *Co je federace identit?* Dostupné také z: https://www.eduid.cz/cs/about. [Online; accessed 08-February-2019].

28. *Proces federacniho prihlaseni*. Dostupné také z: https://www.eduid.cz/_media/cs/aai-simple.png. [Online; accessed 10-May-2019].

29. CANTOR, Scott. *Metadata*. Dostupné také z: https://wiki.shibboleth.net/confluence/display/CONCEPT/Metadata. [Online; accessed 03-Jun-2019].

30. BADOUET, Gilles Rubens. *Investigation and implementation of Shibboleth SSO authentication mechanism through a specific scenario*. 2013. Dostupné také z: https://play.google.com/store/books/details?id=AeBaAQAAQBAJ.

31. *Learn About LDAP*. 2019. Dostupné také z: https://ldap.com/learn-about-ldap/. [Online; accessed 02-Jun-2019].

32. *eduID.cz*. Dostupné také z: https://www.eduid.cz/cs/tech/sp/shibboleth. [Online; accessed 03-March-2019].

33. *Shibboleth IdP*. Dostupné také z: https://www.eduid.cz/cs/tech/idp/shibboleth. [Online; accessed 23-March-2019].

34. CANTOR, Scott. *SystemRequirements*. Dostupné také z: https://wiki.shibboleth.net/confluence/display/IDP30/SystemRequirements. [Online; accessed 01-March-2019].

35. *Apache Tomcat*. Dostupné také z: http://tomcat.apache.org/. [Online; accessed 03-March-2019].

36. *What is LDAP?* 2014. Dostupné také z: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.1.0/com.ibm.zos.v2r1.glpa200/tivwhats.htm. [Online; accessed 15-March-2019].

37. MALAVOLTI, Martin. *HOWTO Install and Configure a Shibboleth IdP v3.3.2 on Ubuntu Linux LTS 16.04 with Apache2 + Tomcat8*. 2018. Dostupné také z: https://github.com/malavolti/HOWTO-Install-and-Configure-Shibboleth-Identity-Provider/blob/master/HOWTO%20Install%20and%20Configure%20a%20Shibboleth%20IdP%20v3.3.2%20on%20Ubuntu%20Linux%20LTS%2016.04%20with%20Apache2%20Tomcat8.md#install-shibboleth-identity-provider-v332. [Online; accessed 05-May-2019].

38. *Main Page*. Dostupné také z: https://btrfs.wiki.kernel.org/index.php/Main_Page. [Online; accessed 03-March-2019].

39. *Btrfs*. Dostupné také z: https://wiki.archlinux.org/index.php/Btrfs. [Online; accessed 01-March-2019].

40. SCHRODER, Carla. *How to Create and Manage Btrfs Snapshots and Rollbacks on Linux (part 2)*. 2014. Dostupné také z: https://www.linux.com/learn/how-create-and-manage-btrfs-snapshots-and-rollbacks-linux-part-2. [Online; accessed 10-March-2019].

41. HLAVA, Tomáš. *Testování bílé a černé skříňky (white box, black box, grey box)*. 2011. Dostupné také z: http://testovanisoftwaru.cz/tag/black-box/. [Online; accessed 03-Jun-2019].

Přílohy

A. Vytvořené testovací identity

A.1 Uživatel „student00“

```
dn: uid=student00,ou=People,dc=vse,dc=cz
uid: student00
objectClass: account
objectClass: posixAccount
objectClass: top
objectClass: sambasamaccount
objectClass: inetOrgPerson
objectClass: person
objectClass: organizationalPerson
objectClass: mendeluPerson
objectClass: radiusProfile
objectClass: eduPerson
loginShell: /bin/bash
homeDirectory: /home/student00
cn: Ben Student
uidNumber: 1001
gidNumber: 1000
gecos: Ben Student
host: adglobal
host: eduroam
host: insis
host: zdroje.vse.cz
host: o365-auto-ds
host: o365-studenti
host: office365
host: kolejeu
host: o365-A3-stud-1
host: o365-A3-stud-2
mail: 1234560@insis.vse.cz
mail: student00@vse.cz
mail: student00@o365.vse.cz
mail: student00@insis.vse.cz
sn: Student
givenName: Ben
uisId: 123456
accountStatus: o365-auto-ds:active
accountStatus: insis:active
```

```
accountStatus: zdroje.vse.cz:active
accountStatus: office365:active
accountStatus: eduroam:active
accountStatus: adglobal:active
accountStatus: kolejeu:active
accountStatus: o365-studenti:pending,05.10.2019
accountStatus: o365-A3-stud-2:active
accountStatus: o365-A3-stud-1:active
givenName;lang-utf8: Ben
mail;x-univmail: student00@vse.cz
employeeType: student
mendeluMailDeliveryAddress: student00@o365.vse.cz
eduPersonAffiliation: member
eduPersonAffiliation: student
lastChange: 20190613023937
```

A.2 Uživatel „zamestnanec“

```
dn: uid=zamestnanec,ou=People,dc=vse,dc=cz
uid: zamestnanec
host: eduroam
host: netadmin
host: adglobal
host: office365
host: eis
host: o365-zam
host: insis
host: zdroje.vse.cz
host: o365-auto-ds
host: hobbit
host: o365-A3-zam
host: o365-A3-zam-2
accountStatus: eduroam:active
accountStatus: adglobal:active
accountStatus: office365:active
accountStatus: eis:active
accountStatus: insis:active
accountStatus: zdroje.vse.cz:active
accountStatus: o365-auto-ds:active
accountStatus: netadmin:active
accountStatus: hobbit:active
accountStatus: o365-A3-zam:active
```

```
accountStatus: o365-zam:pending,05.10.2019
accountStatus: o365-A3-zam-2:active
cn: Ing. Alfons Ucitel
gecos: Alfons Ucitel
sn: Ucitel
givenName: Alfons
uisId: 5678
employeeType: staff
uidNumber: 1001
gidNumber: 1000
homeDirectory: /home/zamestnanec
loginShell: /bin/bash
mail: zam01@vse.cz
mail: zamestnanec@vse.cz
mail: alfons.zamestnanec@vse.cz
mail: zamestnanec@o365.vse.cz
mail: zam01@o365.vse.cz
mail: alfons.zamestnanec@o365.vse.cz
mail: zam01@insis.vse.cz
mail: zamestnanec@insis.vse.cz
mail: 5678@insis.vse.cz
mail: alfons.zamestnanec@insis.vse.cz
objectClass: posixAccount
objectClass: organizationalPerson
objectClass: account
objectClass: inetOrgPerson
objectClass: mendeluPerson
objectClass: radiusProfile
objectClass: top
objectClass: sambasamaccount
objectClass: person
objectClass: eduPerson
mendeluNetwareDn: .Zamestnanec.ZAM.VSE
faculty: 31180,.5
mail;x-univmail: zamestnanec@vse.cz
mendeluMailDeliveryAddress: zamestnanec@o365.vse.cz
eduPersonAffiliation: member
eduPersonAffiliation: employee
eduPersonAffiliation: staff
lastChange: 20190616023006
```

A.3 Uživatel „ucitel“

```
dn: uid=ucitel,ou=People,dc=vse,dc=cz
uid: ucitel
host: eduroam
host: adglobal
host: office365
host: eduroam2
host: o365-zam
host: insis
host: zdroje.vse.cz
host: o365-auto-ds
host: o365-A3-zam
host: o365-A3-zam-2
accountStatus: eduroam:active
accountStatus: adglobal:active
accountStatus: office365:active
accountStatus: eduroam2:active
accountStatus: eis:active
accountStatus: insis:active
accountStatus: zdroje.vse.cz:active
accountStatus: o365-auto-ds:active
accountStatus: o365-A3-zam:active
accountStatus: o365-zam:pending,05.10.2019
accountStatus: o365-A3-zam-2:active
gecos: Karel Ucitel
sn: Ucitel
givenName: Karel
uisId: 57999
employeeType: staff
uidNumber: 1002
gidNumber: 1000
homeDirectory: /home/ucitel
loginShell: /bin/bash
mail: karel.ucitel@vse.cz
mail: ucitel@vse.cz
mail: karel.ucitel@o365.vse.cz
mail: ucitel@o365.vse.cz
mail: 57999@insis.vse.cz
mail: ucitel@insis.vse.cz
mail: karel.ucitel@insis.vse.cz
objectClass: posixAccount
objectClass: organizationalPerson
```

```
objectClass: account
objectClass: inetOrgPerson
objectClass: mendeluPerson
objectClass: radiusProfile
objectClass: top
objectClass: sambasamaccount
objectClass: person
objectClass: eduPerson
mendeluNetwareDn: .Ucitel.ZAM.VSE
givenName;lang-utf8: Karel
mail;x-univmail: ucitel@vse.cz
mendeluMailDeliveryAddress: ucitel@o365.vse.cz
faculty: 31140,1
cn: Ing. Karel Ucitel, Ph.D.
eduPersonAffiliation: member
eduPersonAffiliation: employee
eduPersonAffiliation: faculty
lastChange: 20190616023053
```