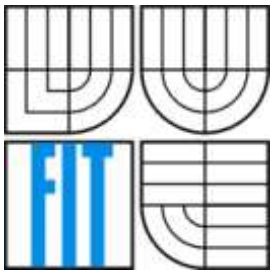
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SINGLE SIGN-ON V J2EE WEBOVÝCH APLIKACÍCH ZALOŽENÉ NA PROTOKOLU SPNEGO/KERBEROS

SINGLE SIGN-ON IN J2EE WEB APPLICATIONS BASED ON SPNEGO/KERBEROS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. Tomáš Nečas
AUTHOR

VEDOUCÍ PRÁCE — Ing. Ondřej Ryšavý, Ph.D.
SUPERVISOR

BRNO 2009

Abstrakt

Diplomová práce se zabývá potřebou, analýzou, popisem a integrací řešení Single Sign-On postaveném na protokolu SPNEGO/Kerberos.
Práce přináší přehled o základních principech a konceptech Single Sign-On a podrobněji se zabývá autentizačním mechanismem Kerberos. Po popisu základů protokolu Kerberos, příslušné terminologie a běžných implementací je pohled zaměřen na služby a nastavení Microsoft implementace Kerbera v prostředí Windows 2000/2003. Demonstrace řešení autentizace je provedena na platformě J2EE prostřednictvím autentizačního filtru a ověřovacího pluginu. Součástí práce je stručný přehled integrace řešení Single Sign-On do různých architektur podnikových informačních systémů a popis procesu implementace takového řešení. Závěr práce obsahuje rozbor použitelnosti řešení Kerberos Single Sign-On v dnešní podnikové sféře.

Klíčová slova

Single Sign-On, Kerberos, autentizace, Domain Controller, Active Directory, J2EE

Abstract

The dissertation deals with requirements, analysis, description and integration of Single Sign-On solution based on SPNEGO/Kerberos protocol.
The thesis provides an overview of the Single Sign-On basic principles and concepts and deals with the Kerberos authentication mechanism in more detail. After introducing the fundaments of the Kerberos protocol, its terminology and common implementations, attention is focused on the services and settings of Microsoft Kerberos implementation in Windows 2000/2003 environment. An authentication solution demonstration is performed on J2EE platform using the authentication filter and plug-in. The thesis also includes a brief overview of integrating the Single Sign-On solution into different architectures of corporate information systems and describes the implementation process of this solution. In conclusion, the usability of Kerberos Single Sign-On solution in today's business sector is analysed.

Keywords

Single Sign-On, Kerberos, authentication, Domain Controller, Active Directory, J2EE

Citace

Nečas Tomáš: Single Sign-On v J2EE webových aplikacích založené na protokolu SPNEGO/Kerberos. Brno, 2009, diplomová práce, FIT VUT v Brně.

Single sign-on v J2EE webových aplikacích založené na protokolu SPNEGO/Kerberos

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Ing. Ondřeje Ryšavého, Ph.D.
Další informace mi poskytli Ing. Boško Manojlovič a Ing. Pavel Novotný.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Na tomto místě bych chtěl poděkovat vedoucímu mé diplomové práce Ing. Ondřeji Ryšavému, Ph.D. za zájem, připomínky a čas, který věnoval mé práci. Dále bych chtěl poděkovat všem, kteří se na vzniku této práce jakkoli podíleli, zejména Ing. Boškovi Manojlovičovi za podnětné konzultace a připomínky a Ing. Janu Kuchařovi za možnost získání praktických zkušeností.
Především bych chtěl ale poděkovat mé manželce a rodičům za trpělivost a velkou podporu.

Obsah

1 Úvod

Pro uživatele počítače není lehké zapamatovat si více autentizačních údajů pro přístup k nejrůznějším službám systému. Přihlašování se tak stává složitou záležitostí zvláště tehdy, když si máme zapamatovat pro každý systém nebo službu v systému jiné uživatelské jméno a heslo. Uživatelé si pak pomáhají způsoby, které jdou obvykle proti zásadám bezpečnosti. Například hesla, která jsou jednoduše zapamatovatelná, lze útočníkem snadněji rozeznat a na druhou stranu opakováním jednoho hesla pro více služeb se zvyšuje riziko jeho odhalení. Bezpečné není ani poznamenání hesla na papír, nebo dokonce do počítače, kde by jej mohl útočník najít a zneužít. A k tomu všemu se často setkáváme s požadavkem administrátorů, abychom svá přístupová hesla často měnili.

Je tedy zřejmé, že IT management přijímá v posledních letech velmi vstřícně myšlenku jednotného přihlašování, díky kterému lze podstatně snížit náklady na správu a podporu uživatelů. Řešení jednotného přihlašování umožní uživatelům disponovat pouze jediným heslem tak, že se přitom nesníží zabezpečení přístupu k požadovaným službám v doméně.

Hlavním motivem diplomové práce byla analýza stávajících možností a řešení na poli Single Sign-On pro intranetové sítě a vytvoření takového řešení pro rozsáhlý CMS[1] systém ve spolupráci se zadavatelem práce – firmou CSC[2].

Jednotné přihlašování Single Sign-On nabízí řešení, které zjednoduší život jak uživatelům, tak administrátorovi.

1.1 Obsah práce

Práce je napsaná tak, aby poskytla čtenáři obecný přehled o principech a možnostech, které poskytuje Kerberos Single Sign-On. Součástí práce jsou kapitoly jak s teoretickým popisem Single Sign-On, tak praktickým přiblížením řešení postaveném na konkrétní implementaci protokolu Kerberos nad J2EE aplikačním serverem.

Text je členěn na kapitoly tak, aby dostal čtenář ucelený obraz problematiky. Následuje přehled obsahu jednotlivých kapitol.

Základy Single Sign-On (kapitola 2)

Kapitola se zabývá úvodem do problematiky Single Sign-on. Na základě tohoto přehledu nachází potřebu a motivaci k řešení Single Sign-On. Součástí kapitoly je výčet základních konceptů Single Sign-On.

[1] CMS - z anglického Content Management System je software zajišťující správu obsahu

[2] CSC - Computer Sciences Corporation. url: http://www.csc.com/

Řešení Kerberos (kapitola 3)

Kapitola představuje základní principy, verze a nastavení protokolu Kerberos a zabývá se podrobněji popisem komunikace, která vede k ověření přístupu ke zdroji. Součástí kapitoly je výčet implementací protokolu Kerberos s podrobnějším popisem služeb a nastavení implementace v prostředí Active Directory.

Autentizace na platformě J2EE (kapitola 4)

Kapitola představuje možnosti platformy J2EE v otázce autentizace SPNEGO/Kerberos a na základě příkladů se zabývá konkrétním řešením pro podnikový informační systém. Kapitola se dále zabývá úrovní podpory protokolu SPNEGO/Kerberos na straně klienta i na straně serveru.

Proces implementace řešení (kapitola 5)

Kapitola se zabývá procesem implementace řešení Single Sign-On do podnikového informačního systému a dále obsahuje doporučení pro implementaci na základě zkušeností autora.

Použitelnost řešení (kapitola 6)

Kapitola je zaměřena na míru použitelnosti a aspekty, které nasazení Single Sign-On na základě protokolu Kerberos do podnikového informačního systému ovlivňují.

Práce je souhrnem informací, které autor získal při řešení různých úloh a implementací, které se týkaly Single Sign-On.

1.2 Pojmy

Active Directory	adresářová služba, která ukládá informace o objektech v síti a poskytuje přístup k těmto informacím [24]
Autentizace	(někdy také autentikace nebo autentifikace) je proces ověření identity subjektu
Autorizace	postup, který omezuje přístup k informacím, funkcím a objektům, přístup mají pouze oprávněné subjekty
Basic Authentication	jednoduchá implementace autentizačního mechanismu se zobrazením výzvy k zadání uživatelského jména a hesla
CIFS	(Common Internet File System) Microsoft implementace protokolu SMB. Open Source implementace má název Samba
Credentials	Identifikační údaje spojené s příslušným účtem v doméně. U uživatele se jedná např. o jméno a heslo
Domain Controller	(doménový řadič) server v síti Windows NT, na němž jsou uloženy všechny informace pro správu systému [24]
Doména	základní adresní jednotkou na Internetu. Překladem domény je IP adresa [24]
GSSAPI	(Generic Security Services Application Program Interface) je aplikační programové rozhraní pro přístup k zabezpečeným službám. RFC 2743, pro Kerberos RFC 1964

JCIFS	open source projekt implementující protokol SMB/CIFS implementovaný v jazyce Java
KDC	(Key Distribution Center) služba, která se stará o distribuci a řízení sdílených a soukromých klíčů pro autentizaci síťových zdrojů a přístupu k aplikacím. V prostředí Windows je implementován jako Domain Controller
Kerberos	síťový autentizační mechanismus vytvořený na MIT (Massachusetts Institute of Technology), který se používá k ověření identity uživatele nebo hostitele
Keytab soubor	soubor, který obsahuje zašifrovaný klíč pro autentizaci vůči autentizačnímu serveru KDC
Lístek	(ticket) obsahuje sadu identifikačních dat pro princip zabezpečení. Je vydáván serverem KDC. Mezi základní typy lístků řadíme TGT (ticket-granting ticket) a lístek služby
LSA	(Local Security Authority) je proces u Microsoft Windows operačních systémech odpovědný za bezpečnostní politiku v systému. Obsahuje informace o zabezpečení systému.
NTLM	(zkratka z NT LAN Manager) autentizační protokol, který je součástí konceptu Windows autentizace
Policy Server	bezpečnostní komponenta v síti (autentizační server), která zajišťuje kompletní služby správy přístupu
Principal	unikátní jméno klienta nebo serveru v rámci domény, které se podílí na síťové komunikaci
SIDs	(Security Identifier) je datová struktura proměnné délky, která identifikuje uživatele, skupinu nebo počítač ve Windows doméně
SMB	(Server Message Block) je protokol pro sdílení souborů, tiskáren a dalších komunikačních abstraktů mezi počítači
SPNEGO	(Simple and Protected GSSAPI Negotiation Mechanism) je GSSAPI pseudo-mechanismus, který určuje dostupný mechanismus pro autentizaci (NTLM, Kerberos)

2 Základy Single Sign-On

2.1 Motivace

Podniková sféra nejrůznějších oborů, mezi které patří mimo jiné bankovní sektor, automobilový a farmaceutický průmysl, využívá ve většině případů informační systémy jako základ úspěšného běhu podniku. Do tohoto základu patří efektivní komunikace, přehledná úložiště, možnosti auditu, výkonné operace a další úkony informačních systémů. S těmito systémy pak pracuje velké množství lidí. Proto je v rámci zajištění efektivity pracovních sil kladen nemalý důraz na účinné využití informačních systémů. Jako příklad je možné uvést firmu, která prodává hardware. Údaje o každém produktu se v této firmě zavádí přes rozhraní informačního systému do databáze. Zde se pak mohou vložená data modifikovat nebo mazat. Vzhledem k tomu, že těchto operací udělá uživatel systému během dne mnoho, je potřeba počítat při návrhu takového systému s maximální optimalizací jednotlivých procesů práce s informačním systémem, jinak dochází ke ztrátám efektivity lidského zdroje. V reálném provozu však k rutinnímu úkolu vkládání položek do systému připadne ještě zapisování odpracovaných hodin, čtení doručené pošty a přístup na firemní informační portál. Všechny tyto aktivity vyžadují přihlášení k patřičnému serveru, který poskytuje danou službu. A zde se dostáváme k činnosti, která se denně mnohokrát opakuje a která je zároveň prioritní z hlediska bezpečnosti dat osobních i dat celé firmy.

Potřeba podnikové sféry v oblasti autentizace uživatelů, ze které pak následně vzešel přístup nazvaný Single Sign-On, je maximalizovat bezpečnost a minimalizovat časové a intelektuální náklady uživatele podnikových systémů. Jak je však z předchozí věty patrné, existuje zde nepřímá úměra. Pokud je snaha o co nejbezpečnější intranetovou síť podle nejrůznějších firemních kritérií, pak se po uživatelích požaduje zapamatování náročných autentizačních údajů a k tomu se ještě vyžaduje jejich častá obměna. A naopak, pokud se snažíme, aby přístup byl pro uživatele řešen co nejjednodušším způsobem - tj. jednoduchá a lehce zapamatovatelná hesla nebo stejné přístupové údaje ke všem službám intranetu, pak se tak děje vždy na úkor bezpečnosti sítě. Optimálním řešením je vyváženost bezpečnosti a možností uživatele. Pokud je v rámci firmy vedena politika bezpečnosti na minimální úrovni, kde například po zapnutí firemního počítače a přihlášení do místní sítě již uživatel není žádán k potvrzení hesla do žádné aplikace, pak je to z hlediska uživatele maximálně výhodné, nicméně z hlediska bezpečnosti se jedná o minimální a nedostačující řešení. Opačným řešením je pak maximalizace bezpečnosti. Maximální bezpečnost si lze představit zadáváním složitých, dlouhých a různorodých hesel do každé z aplikací. V tomto případě se jedná z pohledu uživatele o velmi nesnadný úkol, který je často řešen uložením nebo poznačením hesla pro pozdější

připomenutí. S tímto faktem ale zase vzrůstá možnost odcizení identifikačních údajů útočníkem a možnost napadení systému pod cizí identitou. To stejné platí i pro řešení nejrůznějších PKI (Public Key Infrastructure) klíčů a certifikátů. Ač jsou tato řešení vysoce účinná a bezpečná, vždy je určitá pravděpodobnost, že bude klíč uživateli odcizen a použit k přístupu do systému. Proto je obtížné v rámci klasického způsobu přihlašování docílit maximální bezpečnosti.

Zvyšování efektivity v oblasti autentizace tedy spočívá ve zvýšení bezpečnosti a snížení nároků na uživatele v prostředí mnoha aplikací. Tuto potřebu splňuje řešení Single Sign-On. Pokud politiku přidělování zdrojů a ověřování identity uživatele předáme z uživatele na systém, pak se jedná o výrazné odlehčení požadavků na uživatele. Jediným požadavkem v tomto konceptu zůstává jedno (Single) přihlášení (Sign-On) do systému, které se provádí při přihlášení uživatele do intranetové sítě.

2.2 Architektury systému a autentizace

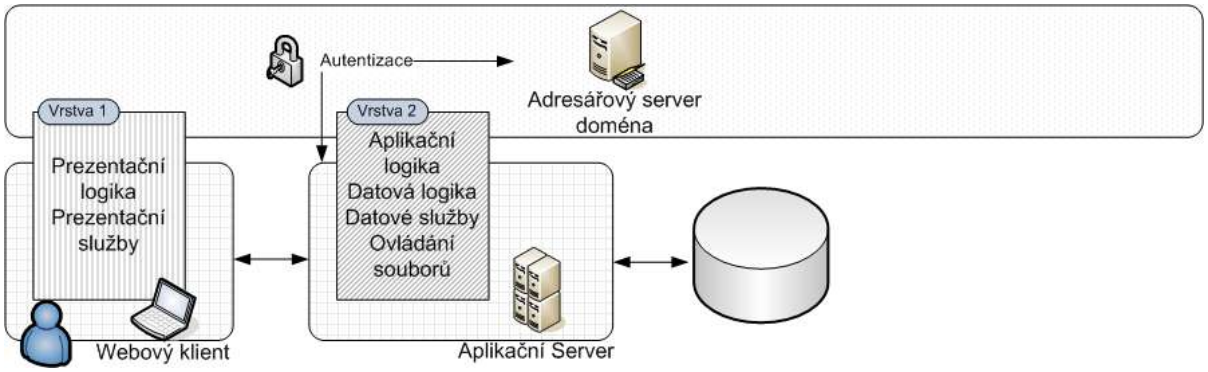
Úvodu k Single Sign-On předchází kapitola, která se zabývá architekturami dnešních informačních systémů s ohledem na způsob autentizace. Jako poslední bude uveden příklad architektury, která bude předlohou pro popisované řešení Single Sign-On.

Způsob autentizace uživatele do webové aplikace se odvíjí od architektury aplikace. K ověřování dochází v takových místech a vrstvách aplikace, aby byla komunikace klient – server bezpečná a dále aby ověření postihlo všechny vrstvy architektury aplikace, ke které má mít daný uživatel přístup. Pokud se tedy jedná o dvouvrstvou architekturu klient – server s aplikačním serverem, který přímo přistupuje pomocí aplikační logiky k databázovému serveru, obsahuje také tento aplikační server logiku k řešení autentizace uživatele v rámci aplikace. Pokud se ale jedná o vícevrstvou architekturu, kde je logika dat a část aplikační logiky přenesena na databázový server, pak je také na tomto serveru, aby řešil úkony spojené s ověřením uživatele.

2.2.1 Typy architektur

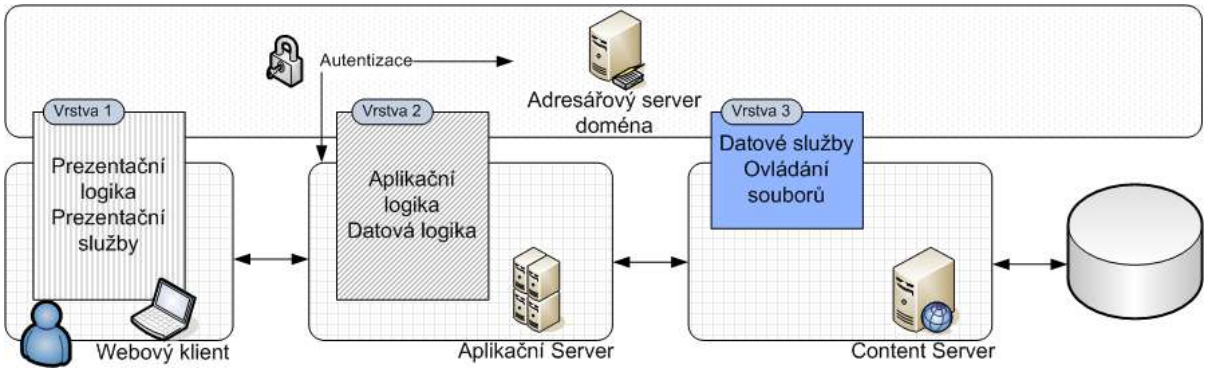
Dvouvrstvá architektura klient – server soustřeďuje logiku aplikace a správu dat na aplikační server. Vzhledem k tomuto stavu je aplikační server vysoce zatížen. Schéma takového typu architektury je na obrázku č. 1. Autentizace je soustředěna na aplikačním serveru. Aplikační server tvoří zároveň jedinou aplikační vrstvu, a proto stačí, pokud dochází k ověření uživatele právě na tomto serveru pomocí aplikační logiky.

Třívrstvá architektura klient – server obsahuje kromě klienta a aplikačního serveru i datovou vrstvu, která poskytuje datové služby a ovládá obsah databáze. Výhoda této architektury spočívá v celkové správě aplikace a rozdělení zátěže mezi aplikační a databázový server.



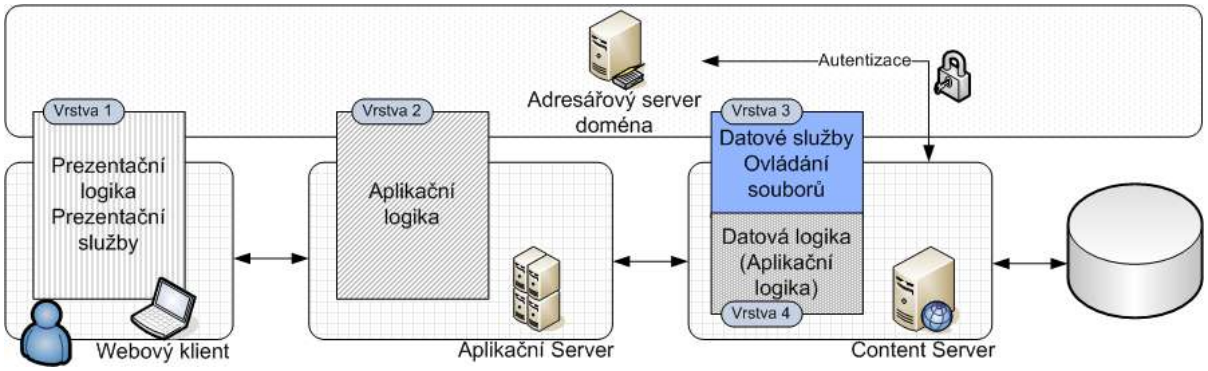
Obrázek 1: Dvouvrstvá architektura klient – server

Autentizace je řešena v rámci třívrstvé architektury podobně jako u modelu klient – server v rámci aplikační logiky na aplikačním serveru.



Obrázek 2: Třívrstvá architektura klient – server

Vícevrstvá architektura může nabývat více možných forem a struktur. Pro příklad implementace Single Sign-On uvažujme architekturu klient – server, která dělí databázový server na Content Server (Content Management) a databázi. V rámci Content Serveru je soustředěno jak ovládání souborů a datových služeb, tak i datová logika. Jedná se o příkladnou strukturu CMS (Content Management System) aplikace.



Obrázek 3: Vícevrstvá architektura klient-server

V určitých případech obsahuje aplikační server pouze služby webového serveru (např. J2EE Apache Tomcat server) a ostatní aplikační logika je přenesena také na Content Server. Autentizace může být v případě vícevrstvé architektury řešena na straně Content Serveru. Jedná se o nejsložitější řešení z pohledu přenosu utajovaných identifikačních údajů v rámci aplikace. Údaje po zadání uživatelem přejdou na aplikační server, kde je třeba tyto údaje z web requestu vyjmout a poslat je ke kontrole na Content Server. Teprve zde pak dochází k ověření těchto údajů a zaslání příslušné odpovědi zpět.

2.2.2 Vícevrstvá architektura

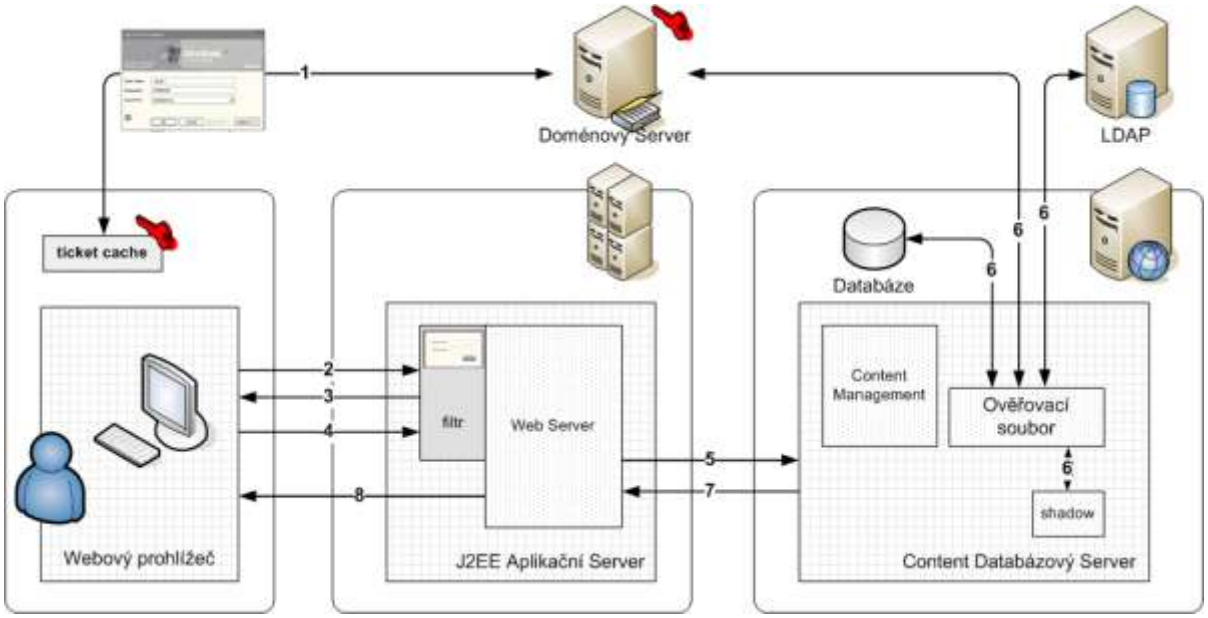
Aplikace, které jsou postaveny na základě vícevrstvé architektury (obrázek č. 3), jsou obvykle rozsáhlé svými možnostmi a zaměřením a důsledně oddělují aplikační server od datové logiky, datových služeb a metod pro práci s obsahem databáze. Jako příklad lze uvést Content Management a Document Management systémy. V následujících kapitolách se bude často v souvislosti se Single Sign-On řešením odkazovat na vícevrstvou architekturu. Z toho důvodu budou představeny jednotlivé komponenty architektury a obvyklý způsob autentizace odpovídající aplikace.

Vícevrstvou architekturu lze rozdělit na webového klienta, aplikační server a databázový server, který bude kvůli logice zprávy dat nazván Content Serverem.

- **Webový klient** – prohlížeč (browser), který komunikuje přes http protokol s webovým serverem. Prohlížeč musí splňovat podporu protokolů k implementaci Single Sign-On.
- **Aplikační Server** – server postavený na J2EE platformě, jejímž základem z pohledu webové aplikace je servlet /JSP kontejner. Příkladem takového řešení může být například server Apache Tomcat. Aplikační server může běžet na jakémkoliv operačním systému. Podpora Single Sign-On zde není závislá na systému, ale na nastavení a instalaci modulu, který přeposílá data určená k autentizaci dále na Content Server.
- **Content Server** – server pro správu obsahu databáze, procesů, workflow dokumentů atd. Součástí serveru je modul, který uskutečňuje autentizaci uživatele. Modulem může být ověřovací soubor (security plugin) vracející boolean hodnotu podle úspěchu ověření.

Autentizaci vícevrstvé architektury zajišťuje Basic Authentication mechanismus, který využívá uživatelské jméno a heslo pro porovnání s údaji, které jsou uloženy v adresáři Active Directory příslušné domény. Autentizace se skládá z několika kroků, do kterých jsou zapojeny všechny základní vrstvy architektury. Jako první vznáší webový klient požadavek (request) na aplikační server. Na aplikačním serveru je tento požadavek transformován do takové podoby, aby jej bylo možné přenést na Content Server pro ověření autentizačních údajů klienta - credentials. Zde bývá rozdíl oproti třívrstvé architektuře, která soustřeďuje autentizační logiku na aplikačním serveru a databázový server slouží jako úložiště dat. Diagram na obrázku č. 4 představuje komunikaci v rámci Basic Authentication autentizace a zároveň ukazuje možnosti ověření klienta na straně Content Serveru. Ověřovací soubor v rámci Content Serveru může mít u různých aplikací různou podobu a

rozdílnou funkčnost. Základem je jednotné rozhraní, které definuje vstupní parametry a návratovou hodnotu. Může se jednat o spustitelný soubor vracející hodnotu true, pokud proběhla autentizace v pořádku nebo hodnotu false. Vstupními parametry mohou být uživatelské jméno, uživatelské heslo a způsob autentizace. Způsob autentizace určuje, jaký mechanismus ověření má plugin použít (závisí na vstupních hodnotách).



Obrázek 4: Autentizace ve vícevrstvé architektuře

Autentizace probíhá v následujících krocích:

1. Přihlášení uživatele do domény
2. HTTP požadavek na zdroj z aplikačního serveru
3. Odpovědí je formulář pro zadání autentizacích údajů
4. Uživatel vyplní login a heslo a požadavek je znovu poslán na aplikační server, který vyjme z dotazu příslušné credentials a upraví jejich tvar pro Content Server
5. Aplikační server pošle credentials na Content Server
6. Content Server předá credentials ověřovacímu souboru s parametrem, dle něhož soubor pozná způsob ověření uživatele – např. Unix shadow, ověření proti LDAP, ověření proti doménovému serveru, ověření proti externí databázi.
7. Pokud je přístup k požadovanému zdroji povolen, oznámí tuto skutečnost Content Server boolean hodnotou na aplikační server
8. Aplikační server zašle uživateli požadovaný obsah na prohlížeč a zároveň zasílá session cookie, která umožní opětovné ověření uživatele v rámci dané session

Během procesu ověřování uživatele je jméno a heslo posláno v rámci vícevrstvé architektury na Content Server. Možností, jak uživatele ověřit, může server poskytovat více (obrázek č. 4). Definice těchto možností je uložena v databázové tabulce uživatelů, kde je u každého záznamu vybrán

administrátorem způsob, jakým má být daný uživatel ověřen. Ověřovací soubor podle způsobu autentizace zavolá metodu autentizace a očekává její výsledek. Podle způsobu autentizace se zároveň předávají metodě parametry. Ověřovací soubor pak může mít následující strukturu:

```
Boolean OverovaciSoubor
{
boolean result = false;
        if (LDAP autentizace) {
                result = overeniProtiAdresarovemuServeru(uživatelJméno,uživatelHeslo,jménoServeru,
                ldap_port,bind_search_dn,atribut);
        }else  if (local autentizace)
                result = overeniProtiLocalBazi(uživatelJméno,uživatelHeslo);
        }else if (domain autentizace) {
                result = overeniProtiDomene(uživatelJméno,uživatelHeslo,doména,pdc,bdc);
        }
        return result;
}
```

2.3 Úvod k Single Sign-On

2.3.1 Definice

Single Sign-On (dále jen SSO) je metoda řízení přístupu, která uživateli umožňuje jednou ověřit autentičnost a získat tak přístup ke zdrojům různých softwarových systémů [8].

Web Single Sign-On je metoda řízení přístupu, která pracuje výhradně s aplikacemi a zdroji, ke kterým se přistupuje prostřednictvím webového prohlížeče [9]. Přístup k webovému zdroji je odchycen buď použitím webového proxy serveru nebo instalací určité komponenty na každý z žádaných webových serverů.

Cílem řešení Single Sign-On je poskytnout organizacím nový přístup k autentizaci uživatelů, který je založen na jednotnosti a zvýšené bezpečnosti ověření identity. Jednotnost je splněna redukcí uživatelských účtů napříč webovými aplikacemi v doméně a s tím související redukce počtu tajných autentizačních údajů uživatele na jeden. Zvýšená bezpečnost je splněna komplexní správou účtů, kterou zabezpečuje šifrovaný přenos zpráv a lístků.

Co to v praxi znamená?

- Uživatel si pamatuje pouze jedno heslo pro přístup do domény
- Ověření pro přístup k zabezpečenému zdroji se provádí na pozadí bez aktivní účasti uživatele
- Bezpečnost síťové komunikace zaručena zasíláním lístků a použitím kryptografie veřejného klíče
- Po síti neputují hesla uživatelů

2.3.2 Základní principy

Základní principy obsahují výčet vlastností systému, který využívá k autentizaci uživatelů Single Sign-On.

Jednotné přihlášení – umožňuje uživateli využít pouze jedno přihlášení do domény k přístupu k zabezpečeným aplikacím, které prvotnímu přihlášení do domény důvěřují

Přehledná správa – administrátor intranetové sítě má možnost spravovat přístup k jednotlivým aplikacím domény z jednoho místa. Přehledná správa souvisí s centralizací bezpečnostní politiky na servery, kterým ostatní servery v doméně důvěřují (trusted server)

Omezení nechtěného přístupu – vzhledem k obecným vlastnostem Single Sign-On je zvýšena bezpečnost intranetové sítě a zabráněno nechtěnému přístupu ze strany útočníka

Volba autentizace – pokud není aktuálně možné využít autentizaci prostřednictvím Single Sign-On, pak má řešení autentizace nabídnout uživateli Basic model autentizace

Využití systémových prostředků – většina implementací Single Sign-On využívá ve větší či menší míře systémové credentials uživatele (jméno a heslo) pro jeho ověření

Úpravy aplikací – každá aplikace nebo služba, na kterou má být aplikována metoda Single Sign-On, musí být upravena. Buď se jedná o úpravu konfigurace aplikace, nebo se může jednat o instalování nových aplikačních modulů

2.4 Základní koncepty Single Sign-On

2.4.1 Koncepty Single Sign-On

Single Sign-On se nejčastěji používá v souvislosti s aplikacemi, ke kterým mají uživatelé přístup prostřednictvím webového rozhraní. V závislosti na implementaci řešení Single Sign-On a typu architektury webové aplikace lze najít různá řešení Single Sign-On. Každé řešení předpokládá určitou formu a vlastnosti architektury, která bude nazvána konceptem.

Následuje výčet tří základních směrů v jednotném přihlašování, jejichž koncepty budou odvozeny od architektur příslušných implementací Single Sign-On. V rámci každého konceptu bude poukázáno na způsob procesu přihlašování do aplikace, dále budou popsány výhody a nevýhody příslušného konceptu a na závěr bude ukázán příklad takového řešení.

Mezi **základní koncepty** lze zařadit:

1. Kerberos Single Sign-On
2. Enterprise Single Sign-On
3. Systémové Single Sign-On

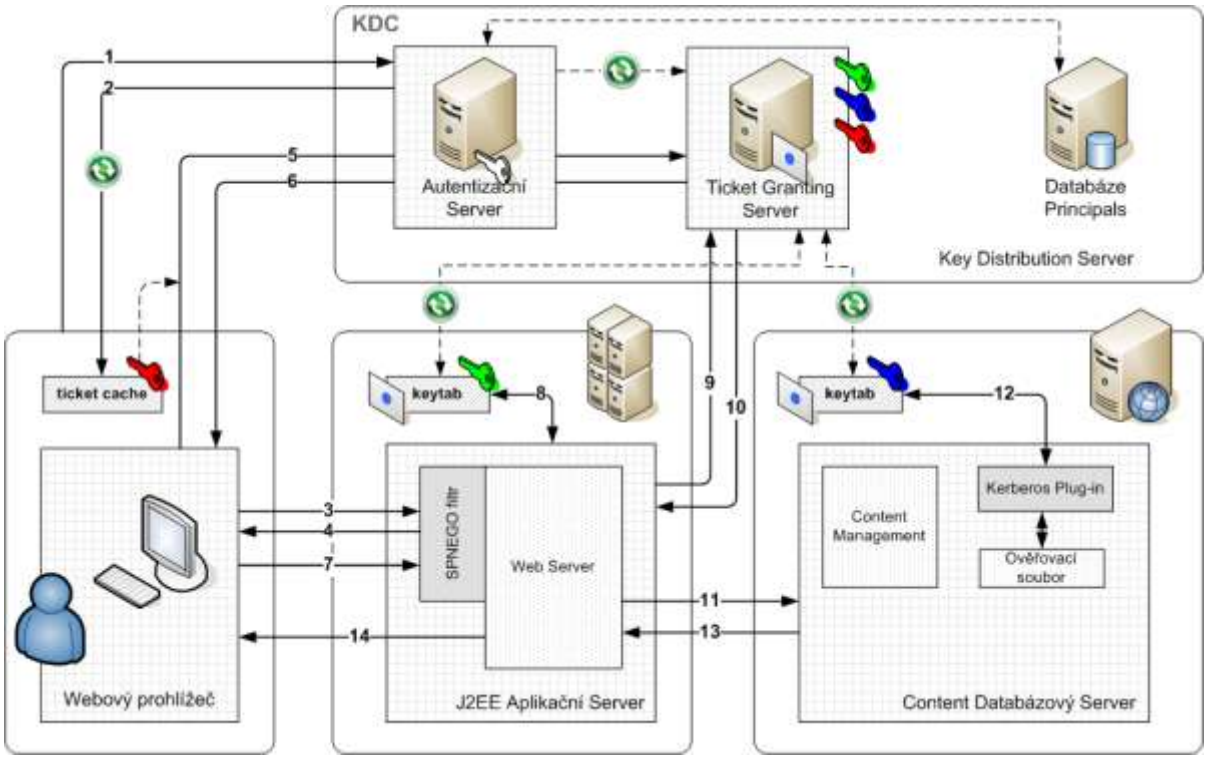
Jednotlivé koncepty budou představeny v prostředí MS Windows Active Directory. Důvodem je četnost použití této adresářové struktury pro řízení domény ve firemních sítích. V následujících kapitolách se poté zaměří pohled na první koncept Single Sign-On Kerberos.

2.4.2 Kerberos Single Sign-On

Koncept Kerberos je založen na autentizaci postavené na zasílání lístků a důvěřování centrálnímu serveru pro přidělování těchto lístků, který je nazýván Key Distribution Center (KDC). V sítích Windows domény se jedná o server Domain Controller. V rámci procesu přihlášení uživatele k doméně obdrží desktop klienta od serveru Domain Controller autentizační lístek TGT (Ticket Granting Ticket). Tento lístek pak slouží pro přihlášení k dalším zdrojům a službám domény.

Kerberos je významný mechanismus díky své použitelnosti a nezávislosti na platformě. Těchto důležitých vlastností je dosaženo systémem serverů, které poskytují služby na základě distribuce lístků a jejich ověřování. Každá služba nebo aplikační server, který chce využívat Kerberos pro přístup metodou Single Sign-On, musí mít přístup k definičnímu souboru keytab, který obsahuje klíč dané služby synchronizovaný s poskytovatelem lístků, a musí se umět dorozumět s klientem a serverem KDC prostřednictvím zásad protokolu Kerberos.

Kerberos je zároveň velmi robustním autentizačním konceptem. Dílčí autentizační utility, metody a programy potřebují znát pouze definované standardizované rozhraní pro příjem a ověření Kerberos lístku. Nezáleží na operačním systému, ani na použitém programovacím jazyce.



Obrázek 5: Kerberos Single Sign-On koncept

Implementace protokolu Kerberos. Pro Unix jsou určeny produkty MIT a Heimdal Kerberos, pro Windows doménu je to Windows Server 2000/2003 Domain Controller implementace Kerbera. V prostředí MS Windows je Kerberos od Windows Server 2000 primárním autentizačním protokolem, který vystřídal dosavadní primární protokol NTLM.

Na obrázku č. 5 je vidět schéma konceptu Kerberos. Schéma se skládá ze 4 komunikujících částí. První je webový prohlížeč, druhou J2EE aplikační server, dále databázový server a server KDC. Součástí obrázku jsou autentizační pluginy SPNEGO filtr a Kerberos plugin. Aplikační i databázový server obsahuje zároveň soubor keytab, který obsahuje klíč služby synchronizovaný se stejným klíčem uloženým v databázi lístků na KDC serveru. Následuje popis komunikace k diagramu.

Popis komunikace u konceptu Kerberos (viz. Obrázek 5.):

1. Během přihlášení uživatele do domény jsou přihlašovací údaje validovány na Autentizačním serveru, kde je vytvořen pro ověřeného uživatele Kerberos TGT lístek.
2. TGT lístek je uložen do cache počítače (např. LSA). Tento lístek se poté používá pro komunikaci s KDC.
3. Request bez autorizace na webový server.
4. SPNEGO filtr odchytí request a odpoví Unauthorized 401 se SPNEGO negotiation požadavkem.
5. Webový prohlížeč vznese požadavek na Ticket Granting Server s použitím TGT lístku o lístek ke službě aplikačního serveru.
6. Ticket Granting Server pošle požadovaný lístek služby zpět webovému prohlížeči.
7. Webový prohlížeč zašle lístek služby aplikačnímu serveru.
8. Aplikační server ověří uživatele na základě souboru keytab, který je synchronizován s Ticket Granting Serverem.
9. Aplikační server zašle dotaz na Ticket Granting Serverem s požadavkem na lístek služby pro autentizaci na databázovém serveru.
10. Ticket Granting Serverem pošle požadovaný lístek služby zpět aplikačnímu serveru.
11. Aplikační server zašle příslušný lístek služby na databázový server pro autentizaci.
12. Databázový server autentizuje uživatele prostřednictvím Kerberos paginu
13. Kerberos plugin ověří shodnost pravost lístku na základě souboru keytab. Pokud je lístek služby validní, povolí databázový server uživateli přístup ke zdroji.
14. Webový server vrací prohlížeči požadovaný zdroj.

Výhody konceptu Kerberos:

- Jedná se o čisté řešení Single Sign-On, které využívá autentizační údaje (credentials) při logování do domény
- Primární autentizační koncept v prostředí Windows 2000/2003

- Je implementovatelný na všechny serverové operační platformy
- Možnost využití v různorodých prostředích (více domén, různé platformy)
- Možnost delegované autentizace, při které se služba vydává za klienta k přístupu k další službě
- Důvěra založená na držení bezpečnostních lístků. Takové řešení pak může být integrováno s použitím SAML rozhraní do SOA, kde jsou autentizační a autorizační funkce dostupné jako webové služby k příslušným aplikacím
- Open source implementace

Nevýhody konceptu Kerberos:

- Nepodporuje Single Sign-On řešení pro tlustého klienta.

Příklad implementace:

Jako příklad systému, který je založen zcela na využití Kerberos protokolu je možné uvést řešení Single Sign-On od firmy Solfit[3]. Řešení je založeno na prostředí MS Active Directory a využívá vestavěné komponenty pro autentizaci na aplikační server a na databázový server. Aplikační server obsahuje Solfit SPNEGO filtr. Na straně databázového serveru (Content Server) je na ověřovací soubor navázán Solfit Plugin, který umí ověřit credentials oproti keytab souboru.

Solfit je typická Kerberos architektura, která je založena na komunikaci SPNEGO. Tu zajišťuje volání filtru aplikačního serveru. Po obdržení Kerberos lístku je tento lístek ověřen, a pokud proběhne ověření v pořádku, pak se aplikační server dotáže KDC serveru na lístek pro Content Server. Tento lístek je předán aplikačnímu serveru a poslán na Content Server. Content Server stejným způsobem ověří zaslaný lístek a pokud se řešení shodují, pak proběhla autentizace v pořádku.

Podpora implementace:

- Aplikační server - BEA WebLogic, Tomcat, WebSphere
- Klient – SPNEGO kompatibilní prohlížeč
- KDC – Active Directory 2000, 2003
- Podpora více domén – ano

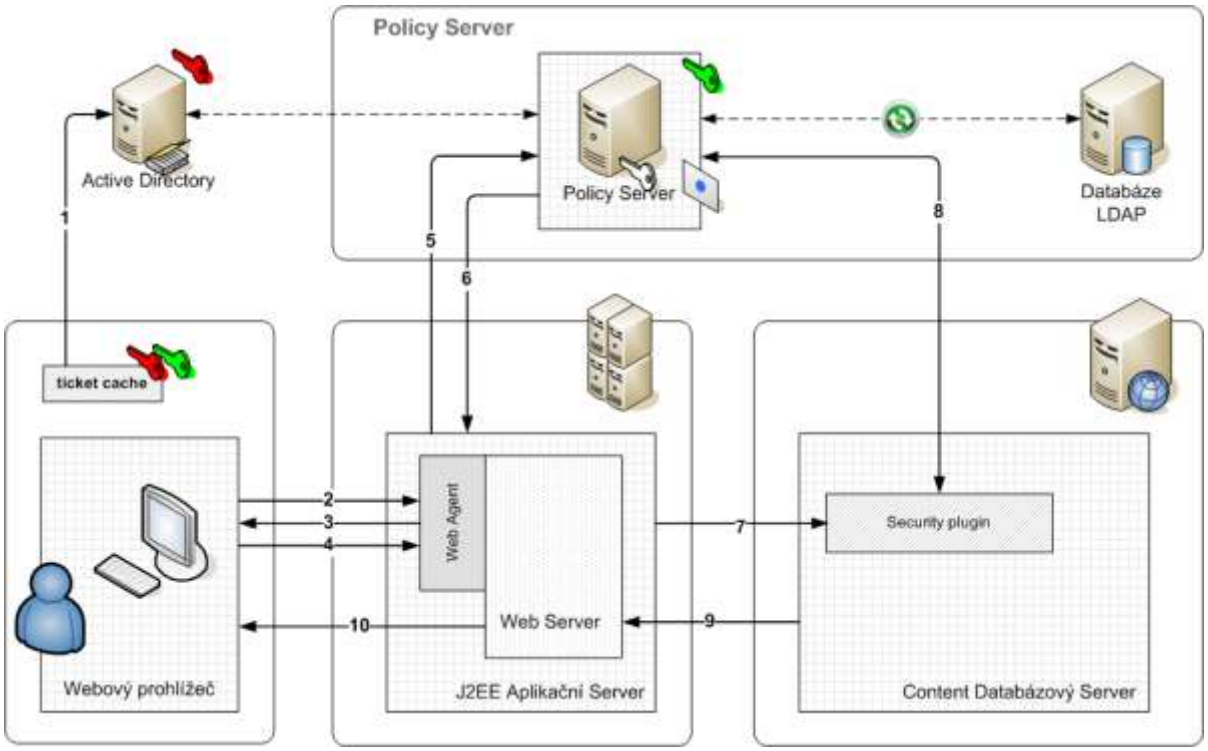
2.4.3 Enterprise Single Sign-On

Do Enterpsise konceptu řešení Single Sign-On je zahrnuta většina komerčních řešení správy, která zajišťuje bezpečný přístup ke zdrojům založeným na webu. Jedná se o koncept s obvyklým centrálním řešením Single Sign-On, jehož základem je Policy server, který řeší

[3] Solfit je Švýcarská firma založená v roce 1997 a poskytující integrační služby v oblasti document management a identity management systémů. Solfit se zaměřuje na řešení SSO pro Oracle, SAP nebo Documentum. (http://www.solfit.ch)

autentizační požadavky od speciálních Single Sign-On pluginů (agentů). Pluginy jsou instalovány na příslušných serverech, které vyžadují ověření přístupu ke zdroji. Pokud takový plugin na aplikačním serveru zachytí neověřený http požadavek na zdroj, tak jej buď na základě konfigurace s Policy serverem obslouží sám, nebo jej na centrální Policy Server přepošle. Řešení může také využívat externě instalovaného agenta na desktop klientovi, který filtruje požadavky přístupu k určitému zdroji v síti a sám je plní autentizačními údaji podle Policy serveru.

V některých případech může být evidentní podobnost s předchozím Kerberos konceptem. Pokud by se nejednalo o KDC nad Active Directory, ale o jinou Kerberos distribuci, např. MIT, pak by tento KDC server měl podobnou úlohu v doméně jako Policy server. Lístky by pak byly jednou z možností ověřování uživatelů.



Obrázek 6: Centralizované Single Sign-On

Popis komunikace u konceptu (viz. Obrázek 6.):

1. Přihlášení uživatele do domény.
2. Http regest na aplikační server.
3. Request klienta je odchycen webovým agentem, který pošle zpět modifikovaný login screen.
4. Uživatel vyplní jméno a heslo (první přístup). Prohlížeč pošle tyto credentials na webový server.
5. Webový agent zavolá Policy Server pro autentizaci požadavku a předá mu autentizací údaje.
6. Policy Server autentizuje uživatele s použitím uživatelského jména a hesla a vygeneruje autentizační lístek, který pošle zpět agentovi.

7. Webový agent zformátuje credentials do tokenu a přesměruje jej na Content Server.
8. Content Server zavolá Secure Plugin pro autentizaci. Plugin pak volá Policy Server, který ověří token a tím ověří uživatele.
9. Pokud proběhlo ověření v pořádku, pošle Content Server boolean hodnotu true na aplikační server
10. Aplikační server pošle požadovaný obsah zpět prohlížeči. Autentizovaná data jsou poslána v session cookie k dalšímu použití.

Výhody centralizovaného konceptu:

- Obvykle robustní řešení Single Sign-On zahrnující rozsáhlé možnosti v oblasti autentizace a autorizace.
- Centralizovaná správa přístupu – Policy server.
- Možnost integrace Single Sign-On do široké škály aplikací.
- Podpora více domén, implementace na různé platformy.

Nevýhody centralizovaného konceptu:

- Potřeba zvláštního agenta na každou aplikaci, která Single Sign-On implementuje.
- Cena řešení.
- Obvykle komerční řešení. Agenti existují jen pro známé webové a aplikační servery.

Příklad implementace:

eTrust Netegrity SiteMinder je access management software od firmy CA[4]. Jedná se o řešení, které poskytuje centralizované bezpečnostní služby pro autentizaci a přístup k webovým aplikacím. Základem řešení je centrální Netegrity SiteMinder Policy server. Základním rysem řešení je podpora Single Sign-On, Strong Authentication Management a centralizované řešení autorizace a auditu.

Autorizační software RSA ClearTrust[5] od bezpečnostní divize RSA korporace EMC je jednotným řešením správy, která zajišťuje bezpečný přístup ke zdrojům založeným na webu. Autorizační software RSA ClearTrust centrálně kontroluje a řídí přístupová práva uživatelů ke zdrojům na webu [12].

Podpora implementace:

- Aplikační server - podle požadavků konkrétní implementace
- Klient prohlížeč – podle požadavků konkrétní implementace
- Podpora více domén – ano

[4] www.ca.com

[5] www.rsa.com

2.4.4 Systémové Single Sign-On

Koncept systémového řešení Single Sign-On je postaven na vlastnostech operačního systému klienta a popř. serveru. Jako příklad lze uvést autentizaci prostřednictvím protokolu NTLM . Pokud webový klient vrátí serveru na základě žádosti o autorizaci potvrzení o možnosti NTLM komunikace, znamená to, že prohlížeč klienta umí komunikovat protokolem NTLM. Komunikace prostřednictvím NTLM dále předpokládá obdržení hashe hesla z počítače uživatele s operačním systémem Windows. Jedná se o implementaci Single Sign-On, která je silně závislá na systému.

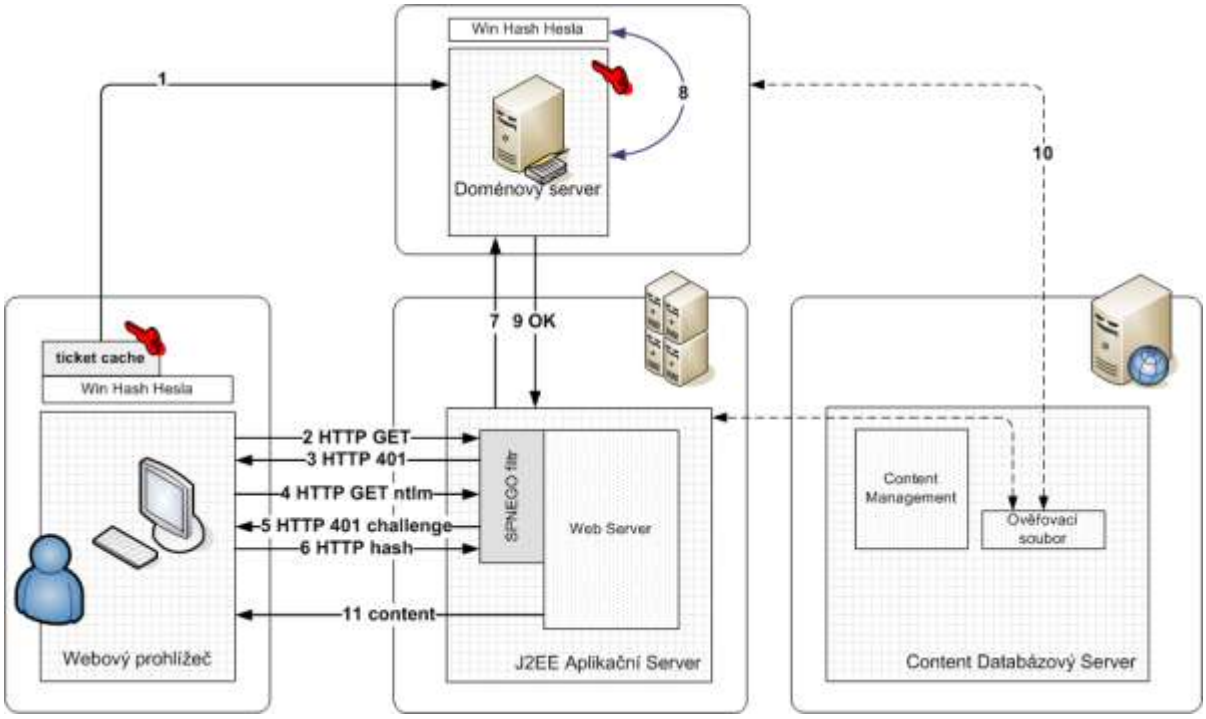
Protokol NTLM je dále silně závislý na platformě aplikace, která chce tento protokol využívat. Z toho důvodu existuje více typů aplikací, které protokol využívají, a zároveň je možné setkat se s různým typem řešení ověřování klienta vůči serveru Domain Controller. Pokud například běží J2EE aplikační server nad operačním systémem Unix, pak je potřeba použít takové řešení pro autentizaci klienta, které umí nejen komunikovat protokolem NTLM, ale umí také ověřit vůči serveru Domain Controller. V tomto případě se často používá CIFS protokol. Jednodušší způsob řešení poskytuje přímo IIS Server, kde existuje řešení vestavěné přímo v rámci homogenní sítě Microsoft.

Výhodou protokolu je jeho přehlednost a jednoduchost. To je způsobeno mimo jiné silnou aplikační závislostí na MS produktech. Jednoduchost i přehlednost spolu souvisejí. Protokol jako takový využívá vlastností Windows domény intranetové sítě. To má za následek možnost využít základní funkčnost Windows systémů bez nutnosti řešení diference operačních systémů. Tímto způsobem odpadá jedna z výkonných aplikačních vrstev systému využívajícího Single Sign-On. Tato výhoda je však nevýhodou v různorodém prostředí operačních systémů, nemluvě o vynucené závislosti na MS Active Directory.

Aplikační server musí komunikovat na dvě strany. První stranou je klient, od kterého očekává poslání NTLM údajů pro ověření. Tato komunikace je udržována prostřednictvím http session. Problémy ale mohou nastat v komunikaci s druhou stranou, kterou je doménový server. Ve standardním řešení se prostřednictvím CIFS vytvoří session prostřednictvím SMB. Vytvoření této session sebou přináší vygenerovanou zprávu nazvanou challenge. Vygenerovaná zpráva challenge je pak závislá na příslušné session.

```
byte[] challenge = SmbSession.getChallenge( dc );
```

Tento stav není problémem, pokud se aplikační server autentizuje přímo vůči doméně. V případě rozsáhlejší architektury, kde je samotné ověření delegováno z aplikačního serveru např. na databázový server, vzniká problém s udržením session a tedy i příslušného challenge. Tento problém lze vyřešit ustanovením SMB serveru, který bude stále udržovat spojení s doménovým serverem pro konkrétního uživatele.



Obrázek 7: Systémové Single Sign-On

Popis komunikace u konceptu (viz. Obrázek 7.):

1. Přihlášení uživatele do domény.
2. Webový prohlížeč posílá http požadavek na webový server.
3. Webový server odesílá zpět http 401 Access Denied. Zároveň nabízí klientovi možnost NTLM autentizace.
4. Klient si vyžádá stránku znovu a uvede jako způsob autentizace NTLM.
5. Webový server odpovídá opět s http 401 Access Denied a doplňuje náhodnou hodnotu pro klienta ("CHALLENGE") na základě SMB protokolu.
6. Klient zakóduje CHALLENGE zaslaný serverem pomocí hashe logon hesla uloženém na klientovi a vytvoří unikátní hash pro tuto transakci. Tento hash pošle serveru přes již otevřené spojení ("Keep alive"). Tím je zaručeno, že jedině uživatel se správným heslem a správnou náhodnou hodnotou mohl vytvořit tento hash.
7. Webový server pošle hash, jméno uživatele a CHALLENGE na Domain Controller.
8. Domain Controller zakóduje CHALLENGE pomocí hash hodnoty uživatelského hesla a pokud se výsledek rovná zaslané hash hodnotě ze strany webového serveru, pak proběhla autentizace v pořádku.
9. Domain Controller posílá výsledek autentizace na aplikační server.
10. Možnost ověření na Content Serveru.
11. Pokud je výsledek autentizace kladný, zasílá webový server klientovi požadovaný obsah.

Výhody systémového konceptu:

- Přehledný koncept protokolu
- Spolehlivost v sítích MS Active Directory

Nevýhody systémového konceptu:

- Systémová závislost (operační systém, Domain Controller)
- Pro jednodušší webové aplikace

Příklad implementace:

Jako příklad implementace systémového konceptu lze ukázat Open Source knihovnu JCIFS[6] [25]. V následujícím pseudo-kódu budou ukázány základy procesu autentizace s ověřením proti Domain Controlleru prostřednictvím NTLM protokolu.

1. Vytvoření session na Domain Controller. Jako základní parametr se udává ip adresa serveru (parametr domainController).

```
dc = UniAddress.getByName( domainController, true );
```

2. Získání challenge od Domain Controlleru. Funkčnost poskytuje třída SmbSession s metodou getChallenge(), které se předá jako parametr identifikátor získané session Domain Controlleru.

3. Získání NTLM struktury na základě zaslání tří zpráv komunikace mezi klientem a aplikačním serverem. Parametry metody authenticate jsou request a response odchycené filtrem a challenge z předchozího kroku. Výsledkem je struktura, která obsahuje mimo jiné LM a NT hash pro identifikaci klienta.

```
ntlm = NtlmSsp.authenticate( req, resp, challenge );
```

4. Autentizace klienta. Metodě logon() se předá identifikace session Domain Controlleru a ntlm struktura. Metoda volá funkce SMB a CIFS protokolu, kterými ověří autenticitu klienta.

```
SmbSession.logon( dc, ntlm );
```

Pokud se provede celý proces ověření bez chyby (vyvolání výjimky), je uživatel považován za oprávněného k obdržení požadovaného zdroje.

[6] JCIFS je Open Source klient knihovna, která implementuje CIFS/SMB síťový protokol v jazyce Java. http://jcifs.samba.org/

3 Řešení Kerberos

3.1 Základy protokolu Kerberos

3.1.1 Kerberos úvodem

Kerberos je protokol zajišťující bezpečné ověření totožnosti přes nezabezpečené sítě. Svého účelu dostává protokol Kerberos v síťových aplikacích, od kterých jsou očekávány následující dva požadavky. Jednoduchost a bezpečnost autentizace. První požadavek je splněn redukcí uživatelských účtů v systému na jeden a s tím související redukce počtu tajných autentizačních údajů na jeden. Druhý požadavek je splněn komplexní správou účtů založenou na šifrování a bezpečném přenosu zpráv, kdy žádná data neputují po síti v textové nezabezpečené podobě. Unikátnost protokolu spočívá především v použití lístků, časově omezených šifrovaných zpráv, které ověřují totožnost uživatele bez toho, aby byly po síti zaslány bezpečnostní důvěrná data. Důvěryhodnost je směřována vůči třetí straně centralizovaného autentizačního serveru KDC.

Kerberos má za sebou coby protokol obdivuhodných 25 let. Po tuto dobu byl vyvíjen a modifikován a dnešní podoba protokolu je označena jako Kerberos v5. Této verzi ale předcházel dlouhý vývoj. Prvními vývojovými verzemi byli verze v1, v2 a v3. Tyto verze měly své omezení a byly spíše ve stavu testování a zavádění do nových prostředí.

Kerberos v4 je prvním kompletním autentizačním balíkem, jehož první vydání bylo uskutečněno 24.ledna 1989. V té době velký projekt Andrew File System přijal koncept Kerbera v4 za svůj vlastní autentizační mechanismus, čímž došlo k jeho masivnímu rozšíření. Základy Kerbera v4 byly zahrnuty již v technickém plánu projektu Athéna MIT. Tímto dílem byla vize projektu Athéna naplněna. Kvůli DES šifrování však organizace mimo USA neměly možnost stahovat tuto verzi Kerbera [2]. Z toho důvodu vývojáři Kerbera vytvořili speciální verzi, která se začala ujímat i za hranicemi USA. Dnes ještě stále několik distribucí Kerbera 4 existuje (např. kth-krb).

Kerberos v5 byl vyvinut proto, aby přidal nové bezpečnostní rysy, které ve verzi 4 chyběly. Jedná se zatím o poslední verzi a je celá dokumentovaná v RFC 1510. Mezi základní nové rysy řadíme například doporučené zasílání a delegování, opakovaná paměť, více flexibilní autentizace napříč domén, rozšířené typy šifrování a další.

Původ slova Kerberos je převzat z řecké mytologie, kde Kerberos jako trojhlavý pes chránil cestu do podsvětí, odkud se duše již nemohly vrátit zpět na zem. Příznačně je jméno síťového protokolu odvozeno právě kvůli těmto vlastnostem, kdy protokol Kerberos ověřuje přístup uživatelů

ke zdrojům sítě. Slovo Kerberos je rozšířeno obecně navzdory původnímu mytologickému jménu Cerberus. Je to způsobeno rozdílným hláskováním téhož jména [3].

Athena projekt byl založen v roce 1983. Cílem projektu bylo vyvinout systém pro integraci počítačů v rámci MIT (Massachusetts Institute of Technology). Od začátku byl projekt koncipován jako síťový systém, založený na bázi klient-server architektury. Základem byl protokol pro síťovou autentizaci. Nový autentizační systém centralizoval důvěrné služby do počítačů, které byly k tomuto účelu úzce specializované, kontrolované a monitorované, a zároveň poskytoval zabezpečený přenos informací mezi těmito centry zabezpečení a ostatními stroji v síti. Protokol Kerberos nebyl jediným produktem, který vzešel z úsilí projektu Athéna. Dalšími významnými produkty byly například X Window System, který je nyní používán jako základ pro Unix GUI. Dalšími síťovými balíčky byly mimo jiné Hesiod, Moira, MUSE, Discuss a Zephyr [4].

3.1.2 Základy konceptu Kerberos

Koncept protokolu Kerberos je založen na třech pilířích, které jsou mezi síťovými profesionály známé jako Autentizace, Autorizace a Audit [2]. Dalším základem je prostředí, ve kterém Kerberos funguje, a tím je úložiště informací. Mezi nejznámější a protokolem Kerberos nejvíce využívané patří zejména úložiště LDAP. Podstatou všech těchto základů protokolu Kerberos je bezpečnost, míra utajení a z toho vyplývající integrita zpráv.

Autentizace je proces ověření identity daného uživatele [3]. Pro ověření identity je uživatel tázán na předem domluvenou informaci. Tato informace může spadat do jedné z následujících kategorií [3]:

1. co uživatel zná – heslo, PIN
2. co uživatel má – USB dongle, smart card, privátní klíč
3. čím uživatel je – otisk prstu, snímek oční zornice

Jak je vidět z uvedených kategorií, heslo nebo pin je jen jednou z možností autentizace. Z tohoto pohledu Kerberos dalece přeskočil dobu, ve které byl vytvořen. Obdobná řešení autentizace Single Sign-On založená například na protokolu NTLM se dají s většími obtížemi (pokud vůbec) přizpůsobit dnešním novým přístupům v ověření totožnosti uživatele. V tom je protokol Kerberos a jeho systém lístků v dnešní době aktuální.

Na systému Kerberos je zároveň cenné to, že není třeba striktně specifikovat, který způsob autentizace by měl být použit. Může se tedy použít jak RSA SecureID, biometrické snímání, tak heslo jako textové pole. Například Microsoft Windows 2000/2003 podporují autentizaci uživatele Kerberem prostřednictvím čipových karet, které poskytují při logování jednodušší postup pro uživatele a zároveň poskytují bezpečnější mechanismus ověřování uživatele [2].

Autorizace je postup, který omezuje přístup k informacím, funkcím a jiným objektům [3]. Základem ověřování přístupu jsou ACL (Access Control List), které asociují jednotlivé uživatele s

danými jim přiřazenými právy. Důležitá je bezpečnostní implikace autorizace. Ta je bezpečná pouze tehdy, pokud je uživatel validně autentizován. Autorizace je zastavena v případě, že autentizační metoda nevytváří důvěrné spojení s autentizační autoritou.

Audit není preventivní část Kerbera, ale jedná se o reaktivní část. Systém logování vestavěný do Kerbera dává dobré možnosti pro ověřování správné funkčnosti autentizace a na ni navazující autorizace. Každá implementace Kerbera poskytuje dobré řešení pro logování aktivit jednotlivých modulů.

Úložiště informací je posledním základem protokolu Kerberos. Obecně je často slyšet názor, že adresáře jako Unix /etc/passwd, NIS, LDAP a další poskytují dobrou autentizaci systému sami o sobě. Není tomu tak. Úložiště mohou obsahovat a popisovat síťové objekty jako tiskárna nebo počítač, které bývají uloženy jako jednoduchý text. LDAP nebo Active Directory obsahují komplexní adresářovou strukturu. Autentizace proti adresáři může mít dva rozdílné způsoby [2]. První možností je, že klientský počítač kontaktuje příslušný adresář a obdrží hash uživatelova hesla, kterou u sebe následně zkontroluje porovnáním. Tím dostane výsledek autentizace. Druhá možnost je vidět například u adresáře LDAP, kdy uživatel ověří svá práva tím, že zkusí přistoupit ke službám adresáře pomocí svých údajů. V tomto případě se používají protokoly jako Samba nebo modul PAM. Použití protokolu Kerberos je lepší v několika ohledech. Zejména použitím lístků může uživatel přistupovat ke všem možným zdrojům na síti bez zasílání uživatelského hesla po síti. Kerberos lístky jsou šifrované zprávy, které jsou validní pouze po určitou dobu (typicky 8-24 hodin). Tímto mechanismem není heslo nikdy posláno po síti jako jednoduchý text.

3.1.3 Šifrování

Šifrování (jako cryptography, cryptos – tajný, graphein – psaní) poskytuje funkčnost pro zašifrování a rozšifrování zpráv posílaných po síti. Kerberos nepoužívá šifrování pouze pro ochranu autentizačních údajů zaslaných a obdržených objekty v síti, ale poskytuje také ochranu proti neoprávněnému podvrhu zprávy zvenku sítě, kdy je snaha vytvořit a podstrčit do sítě zprávu podobnou těm, které v této síti kolují.

Kerberos obsahuje několik možností šifrování dat [2]. Nejrozšířenějšími standardy šifrování Kerbera jsou DES a AES (Advance Encryption Standard). Dalším široce rozšířeným algoritmem pro šifrování je RC4, který je široce rozšířen u Microsoft implementací Kerbera.

Integrita zpráv je zaručena hash funkcemi. Funkce je matematická, jednosměrná a bere rozdílně dlouhý vstup a vytváří z něj výstup o pevně dané velikosti (64-256 bitů). Mezi takové algoritmy patří u Kerbera mimo jiné CRC-32, MD5 a SHA1 (Secure Hash Algorithm) [2].

Windows Server 2003 Kerberos podporuje následující kryptografické algoritmy: RC4-HMAC, DES-CBC-CRC a DES-CBC-MD5. Původním algoritmem je RC4-HMAC. Původní nastavení může být změněno použitím „Use DES encryption types for this account“ v nastavení vlastností u účtu

v doméně. Nastavení umožňuje především kompatibilitu s většinou UNIX implementací Kerbera. Existují dva důvody, proč Microsoft nepoužívá DES jako původní algoritmus:

- Snadný upgrade z NT4 na Windows 2000/2003 server. Klíč, který je použitá pro RC-HMAC může být použitý s Windows NT4 hashem hesla.
- Omezení vývozních práv. V raných fázích vývoje systému Windows 2000 nemohl být 56-bitový DES vyvezen mimo USA. A vzhledem k tomu, že chtěl Microsoft používat stejnou šifrovací technologii kdekoliv na světě, byla vybrána 128-bitová RC4-HMAC alternativa.

Pokud je třeba ověřit v systému algoritmus, jakým je lístek šifrován, lze použít utility kerbtray nebo klist (příloha č. 4), kde je vidět pod položkou Kerb Ticket Encryption Type použitý algoritmus.

```
Server: HTTP/d65.kerberos.local@KERBEROS.LOCAL
  KerbTicket Encryption Type: Kerberos DES-CBC-MD5
  End Time: 1/4/2009 5:44:47
  Renew Time: 1/10/2009 19:44:47
```

Význam jednotlivých položek trojice DES-CBC-MD5 je následující:
DES-CBC značí kryptovací mód, který šifruje data prostřednictvím Data Encryption Standard (DES) za použití Cipher Block Chaining (CBC) módu. Bloková šifra DES je 8 bytová. MD5 značí algoritmus pro kontrolní součet.

3.2 Terminologie

3.2.1 Principal v doméně

Mechanismus Kerberos potřebuje v rámci domény rozeznat jednotlivé služby, servery a uživatele. Z tohoto důvodu má každý server a každá služba svůj vlastní principal. Tato vlastnost Kerberos systému pomáhá zvyšovat bezpečnost celého systému, neboť je možné definovat seznam "povolených" aplikačních serverů, které smí přistupující uživatel požádat o ověření identity [6].

Principal je definován long-term klíčem. Tímto klíčem může být například heslo. Principal má v síti globálně unikátní jméno a je začleněn do hierarchické struktury domény. Každý principal začíná uživatelským jménem nebo názvem služby. Poté následuje volitelná položka. Takto zkonstruovaný principal pak značí unikátní identifikaci v rámci domény. Kerberos dále definuje administrativní kontrolní doménu [2]. Podle konvence je Kerberos doména v rámci pojmenování DNS serverem převedená na velká písmena. Např. DNS doména – kerberos.local, má odpovídající Kerberos doménu KERBEROS.LOCAL. Tato konvence ale není podmínkou.
Jako příklad je uveden Kerberos principal, který je spojen se službou „SERVICE“ poskytovanou aplikačním serverem s názvem „as“ (jako aplikační server):

```
SERVICE/as.kerberos.local@KERBEROS.LOCAL
```

Kerberos principal vlastní i každý uživatel přihlášený do domény. Tvar tohoto principal je odvozen od názvu domény a jména příslušného uživatele v doméně. Jako příklad je uveden Kerberos principal, který je svázán s uživatelem Tomáš Nečas v doméně kerberos.local. Přihlašovací jméno uživatele je tnecas:

```
tnecas@KERBEROS.LOCAL
```

Tento tvar principal je validní ve verzích Kerbera v4 i v5.

3.2.2 Klíče a hesla

Klíče, sloužící k zabezpečené komunikace, jsou sdíleny alespoň dvěma stranami. Jednou stranou je služba, server nebo uživatel a druhou stranou je příslušný KDC server. Kerberos využívá funkci string2key, která má na starosti konverzi uživatelského hesla na kryptovací klíč. Tato funkce aplikuje několik transformací, během kterých se znakově založené uživatelské heslo transformuje na nový řetězec, který vytvoří kryptovací klíč [2].

Nejdůležitější součástí transformace je tzv. „salt“. Obecně řečeno je salt posloupností znaků, která je přidána do hesla před tím, než se z něj udělá hash. Pro Kerberos v5 je salt zadáván ve formě Kerberos domény [7].

```
tnecaskey=string2key(heslo+"tnecas@EXAMPLE.COM")
```

tnecaskey je kryptovací klíč uživatele tnecas a heslo je nešifrované heslo daného uživatele. Pro kompatibilitu s verzí Kerbera v4 je dobré nastavit salt na null.

3.2.3 Key Distribution Center

Autentizační služba v doméně, která je založena na funkci distribuci lístků pro přístup k různým službám domény, se nazývá Key Distribution Center [7]. Služba poskytuje tyto základní služby:

1. Autentizační Server (AS)
2. Ticket Granting Server (TGS)
3. Databáze účtů

I přesto, že jsou tyto tři služby logicky nezávislé, jsou implementovány obvykle na jednom serveru a běží obvykle jako jedna služba. V jedné Kerberos doméně musí existovat alespoň jeden KDC.

Autentizační Server (AS) – služba, která odpovídá na úvodní autentizační žádost od klienta, který ještě není ověřen a který zadá serveru heslo. Jako odpověď na autentizační požadavek obdrží klient od Autentizačního Serveru lístek Ticket Granting Ticket (TGT), který je spojen s daným principal. Pokud uživatel je opravdu ten, za koho se pokládá, může použít TGT lístek k obdržení dalšího lístku služby pro přístup k požadované službě.

Ticket Granting Server (TGS) – komponenta pro distribuci lístků služby klientům s validním lístkem TGT, který garantuje ověřenou identitu pro získání požadované služby. TGS lze považovat za aplikační server, který vydává přístup ke službám a zdrojům.

Databáze účtů (Global Catalog) obsahuje informace pro každý principal v doméně. Jedná se o kontejner položek uživatelů a služeb označených jako principals. V databázi jsou pak uloženy následující informace ke každé položce [26]:

- Jméno principal, se kterým je položka spojena
- Šifrovací klíč a související Key Version Number (kvno – čitač, který zaznamenává počet změn hesla. Aktuální verze čitače identifikuje verzi klíče)
- Maximální doba platnosti pro lístek spojený s daným principal
- Atributy charakterizující chování lístků
- Heslo pro použití po uplynutí data expirace
- Datum expirace, po kterém nebudou lístky vydávány

Windows 2000/2003 ukládá tyto informace v databázi Active Directory. Otevřené implementace jako MIT ukládají data do speciálních bází postavených na KDC souborovém systému [2].

KDC pro doménu ve Windows síti je součástí řadiče domény stejně tak, jako je součástí Windows domény Active Directory. Obě služby jsou automaticky spuštěny prostřednictvím Local Security Authority (LSA) daného řadiče a běží jako součást LSA procesů. Ani jedna služba nemůže být samovolně zastavena.

3.2.4 Lístky

Lístek je kryptovaná datová struktura vydaná KDC serverem a sdílející kryptovaný klíč, který je unikátní pro každou session v rámci Kerberos domény. Lístek dále obsahuje některé další příznaky a pravidla pro pohyb lístku v doméně [7]. Lístek slouží obecně dvěma účelům [2]:

1. potvrzení identity koncového účastníka komunikace
2. ustanovení kryptovacího klíče, který obě strany komunikace sdílí pro bezpečnou komunikaci. Klíč je nazýván jako session key

Každý lístek obsahuje jméno požadujícího principal, jméno principal požadované služby, délku platnosti lístku, seznam IP adres, ze kterých je možné lístek použít, tajný sdílený session klíč pro bezpečnou komunikaci s daným zdrojem [3].

Lístky jsou uchovávány v paměti, která je rozdílná podle typu a implementace Kerbera. Pokud budeme brát v úvahu defaultní úložiště lístků, můžeme si pomocí příkazu klist (příloha č.4) vypsat základní informace o lístcích [7]:

```
$ klist
Ticket cache: FILE:/tmp/krb5cc_500
Default principal: tnecas@KERBEROS.LOCAL
```

První částí výpisu klist je uživatelův principal, který je uložen v souboru /tmp/krb5cc_500. Následují služby a jejich principal s příslušnými informacemi.

Prostředí Windows Server 2003 obsahuje v balíku Resource Kit utilitu kerbtray, která může ukázat obsah cache lístků (LSA). Výstup je podobný jako v případě klist, ale v grafickém režimu.

3.3 Kerberos protokoly

Jak již z textu vyplynulo, existují dvě rozšířené a stabilní verze mechanismu Kerberos v4 a v5. Zatímco koncept a návrh protokolů je u obou verzí velmi podobný, rozdíly jsou na první pohled patrné v základech protokolů a jejich implementaci [2]. Základní operace protokolu Kerberos byly publikovány v roce 1978 pány Needham a Schroeder. Tehdy se tento princip nazýval Needham and Schroeder protokol. Protokol byl navržen jako bezpečnostní autentizační systém. Publikovaná práce pánů Needham a Schroeder, která se stala v roce 1978 základem dalšího vývoje, popisuje dva rozdílné protokoly. První z těchto protokolů popisuje použití privátního kryptovaného klíče a právě tento protokol je základem protokolu Kerberos.

Protokol Needham-Schroeder definuje tři účastníky protokolu [2]: klientský stroj, dotazovaný server a autentizační server. Dotazovaný server je stroj poskytující službu, ke které chce klient přistoupit. Autentizační server je stroj, který obsahuje kopie kryptovacích klíčů uživatelů a služeb na síti. Protokol poskytuje mechanismus pro bezpečnou distribuci šifrovaných klíčů na klienta a na server. Protokol začne kontaktováním autentizačního serveru a zasláním zprávy obsahující identity obou stran a tzv. „nonce“ jako vygenerovanou hodnotu. Autentizační server vrací kopii session klíče, jméno aplikace, původní nonce, aplikační kopii session klíče, jméno klienta, aplikační klíč a uživatelský klíč. Jednotlivé položky odeslané zprávy jsou vnořeny tak, aby se daly dále přeposílat. Po obdržení zprávy klientem ověří klient správnost doručeného balíku a vyjme z něj session klíč, jméno klienta a aplikační klíč. Tyto hodnoty pak pošle aplikačními serveru [17]. Aplikační server potvrdí spojení tím, že zvýší vygenerované nonce klientem o jedničku a spolu s session klíčem přepošle nazpět klientovi.

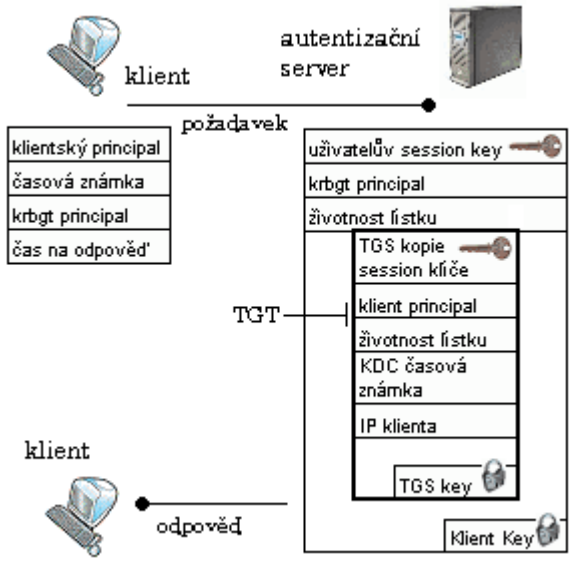
Jak je vidět z popisu protokolu, velkou většinu základů Needham-Schroederova protokolu přebral protokol Kerberos do svých základů.

3.3.1 Kerberos v4

Protokol je založený na základech Needham-Schroeder protokolu se dvěma základními rozdíly [2].

- Prvním rozdílem je omezení množství odesílaných zpráv mezi klientem a autentizačním serverem. Kerberos v4 používá autentizační zprávu, která je vytvořena na základě lokálního času klienta kryptovaného nově vyjednaným session klíčem daného spojení.
- Druhým rozdílem je vytvoření konceptu TGT - Ticket Granting Ticket. Tento koncept umožňuje klientovi autentizovat se vůči většímu počtu aplikačních služeb. To je i hlavní výhodou Kerbera – vytvoření Single Sign-On přístupů ke službám v doméně, aniž by bylo nutné stálé ověřování vůči autentizačnímu serveru. Jako důsledek této politiky se dělí Kerberos služba na část AS (Autentizační Server) a TGS (Ticket Granting Server).

Autentizační server slouží jako primární server pro autentizaci uživatele. V případě přihlášení do domény tedy uživatel kontaktuje nejdříve tento server s žádostí o autentizaci. Základní funkce serveru je tedy obdržet požadavek obsahující uživatelské jméno klienta, který požaduje autentizaci, a vrátit šifrovaný lístek TGT vázaný na daného uživatele [2]. Poté může klient použít lístek TGT pro přihlášení do dalších aplikací a služeb. Celý proces komunikace klienta s autentizačním serverem je vidět na obrázku 8.



Obrázek 8: Autentizační služba AS serveru [2]

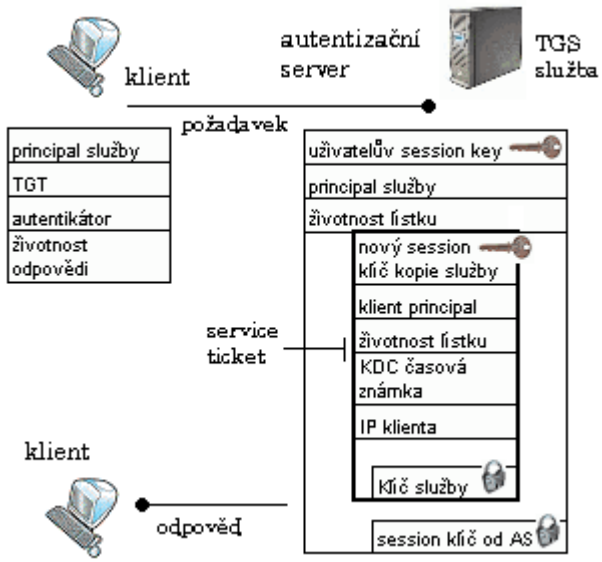
První zpráva putuje od klienta k autentizačnímu serveru a je známa jako AS_REQ. Zpráva je poslána jako jednoduchý text a obsahuje identitu klienta, lokální čas klienta a principal lístku služby. Po obdržení zprávy AS_REQ od klienta AS server ověří, zda zadaný principal existuje a zkontroluje čas na zprávě (toleranci je možno nastavit, obvykle 5 minut). Poté autentizační server generuje session

klíč. Tento klíč bude sdílet klient s TGS serverem. AS server vytvoří dvě kopie session klíče. Jednu kopii pošle v rámci TGT lístku zpět klientovi a druhou kopii pošle TGS serveru.

Odpovědí na AS_REQ je zpráva AS_REP. Tato odpověď se skládá ze dvou částí. První část obsahuje základní data pro zpětné doručení na klienta, který o danou službu požádal. Tato část je uzamknuta pomocí klientského klíče, který sdílí prostřednictvím nastavení administrátora daný klient s autentizačním serverem. Druhou částí je tzv. TGT – Ticket Granting Ticket. Tento lístek obsahuje data potřebná k tomu, aby si klient mohl zažádat o přístup k určité službě v rámci Kerberos domény. Přístup ke službě pak uživatel získá na základě obdrženého session klíče a TGT lístku dotazem na server TGS s žádostí o příslušný lístek relace mezi klientem a požadovanou službou.

Když klient obdrží zprávu, dešifruje ji pomocí klientského klíče (obvykle heslo). Pokud proběhlo dešifrování v pořádku, má klient potřebné náležitosti k tomu, aby zažádal o povolení přístupu k určité službě. Další vlastností TGT lístku je to, že jej nemůže číst klient, který o něj zažádal.

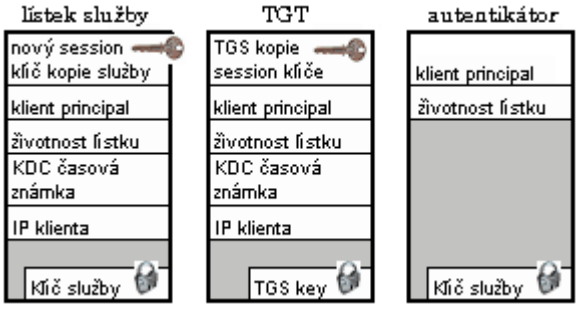
Ticket Granting Server je distributorem lístků pro zabezpečené služby a servery v síti. Pokud uživatel potřebuje přistoupit k určité zabezpečené službě v rámci Kerberos domény, musí k této akci použít příslušný lístek. Ten obdrží od TGS serveru na základě lístku TGT. Klient tedy musí připravit zprávu, která bude obsahovat tyto tři části: TGS požadavek (request), kopii TGT lístku obdrženého od AS serveru a autentikátor.



Obrázek 9: Autentizační služba TGS serveru [2]

Autentikátor se skládá z časové známky, která je kryptovaná session klíčem. Autentikátor zabezpečuje, že každý lístek je unikátní, a ověřuje, že klient zná sdílený session klíč ustanovený AS serverem. Bez autentikátoru by případný útočník mohl poslouchat v rámci sítě a udělat kopii TGT, která by mu pak umožnila případný přístup k serveru TGS nebo jinému zdroji v rámci domény.

Na základě požadavku TGS_REQ je na serveru KDC generována zpráva zvaná TGS_REP [3]. Tato zpráva zahrnuje novou množinu session klíčů, sdílených mezi klientem a aplikačním serverem, o jehož zdroj klient žádá. Klientská kopie session klíče je kryptovaná starší verzí session klíče [2]. Kopie nového session klíče pro komunikaci klienta s požadovanou službou je vestavěna uvnitř lístku služby (service ticket) a je kryptovaná pomocí long-term klíče dané služby. Tento klíč je nastaven administrátorem napevno a je unikátní v dané doméně. Po obdržení těchto informací si klient přidá service ticket a nový session klíč do své paměti pro pozdější použití. Celý systém komunikace s TGS je vidět na obrázku 9.



Obrázek 10: Kerberos lístky [2]

Na obrázku 10 jsou pro přehled vidět všechny tři možné lístky, které kolují mezi KDC serverem a klientem.

Poslední etapou v procesu přístupu k požadované službě je zaslání lístku služby na server služby, která pomocí klíče této služby rozšifruje obsah poslaného lístku, ustanoví session key a zkontroluje časové příznaky pro ověření platnosti příchozího lístku. Tento požadavek je znám jako AP_REQ. Pokud klient požaduje vzájemnou (mutual) autentizaci, pak je serverem poslána zpět odpověď AP_REP jako potvrzení komunikace. Transformace řetězce na 56-bitové klíče, které jsou pak použity pro šifrování a dešifrování zpráv v rámci Kerbera, vykonává string2key funkce. Tato funkce je jednosměrnou funkcí, která generuje hash a pro vstup používá v případě Kerbera v4 heslo. Výstupem je 56-bitový hexadecimální DES klíč [2]. Matematicky je velmi obtížné zpětně získat příslušné heslo. V dnešní době je nicméně právě z tohoto hlediska vhodnější používat Kerberos verze 5 s pokročilejšími technikami jednosměrného šifrování.

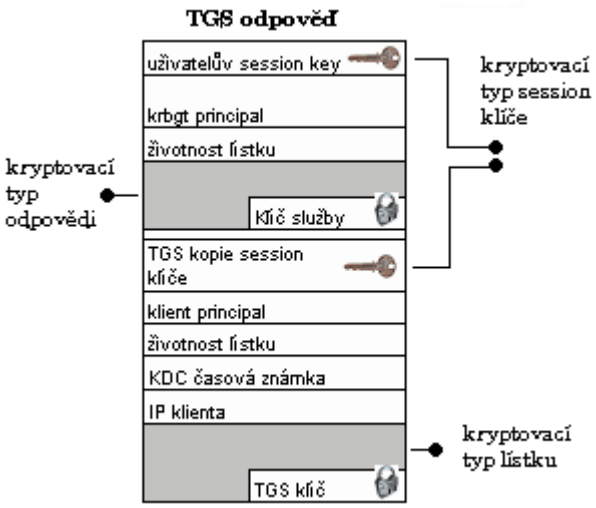
3.3.2 Kerberos v5

Kerberos v5 je evolučním nástupcem Kerbera v4. Kerberos v5 si ponechává veškerou funkcionalitu protokolu Kerberos v4, ale přidává k němu jistá rozšíření. Z hlediska implementace se však jedná o téměř nový protokol.

V první řadě Kerberos 5 posiluje bezpečnost v oblasti jednosměrných funkcí DES. Dalším novým rysem je podpora delegování a přeposílání v rámci Kerberos domén. Přeposílání údajů o

uživateli umožňuje tomuto uživateli přistupovat ke službám na vzdálených serverech. Tento nový jev funguje tím způsobem, že při přihlášení uživatele a komunikaci s AS serverem se TGT lístek bezpečně přepošle i na vzdálené servery. Pro umožnění této funkcionality na bázi komplexního přístupu a multiplatformní charakteristiky vybrali vývojáři Kerbera technologii zvanou ASN.1 k popisu protokolu Kerberos v4. ASN.1 umožňuje vývojářům protokolu vytvářet protokoly na bázi abstraktního jazyka [2]. Co se týče zpětné kompatibility mezi verzemi Kerbera, poskytuje Kerberos v5 transformační službu zvanou krb524, která provádí transformaci lístků Kerbera v5 na lístky Kerbera v4.

Servery AS a TGS prodělaly oproti Kerberos v4 některé změny. Na vrstvě protokolu je změněno několik rysů protokolu, mezi které patří použití ASN.1, dvojité šifrování na straně AS serveru a TGS serveru v rámci KDC. Nové násobné šifrování podporované protokolem verze 5 znamená, že může být použito při transakcích protokolu více šifrovacích typů. Oddělené kryptovací typy mohou být použity v různých zprávách, jakými mohou být lístek, odpověď nebo session klíč. Následující obrázek ukazuje, kde jsou umístěny jednotlivé kryptovací typy (obrázek 11.)

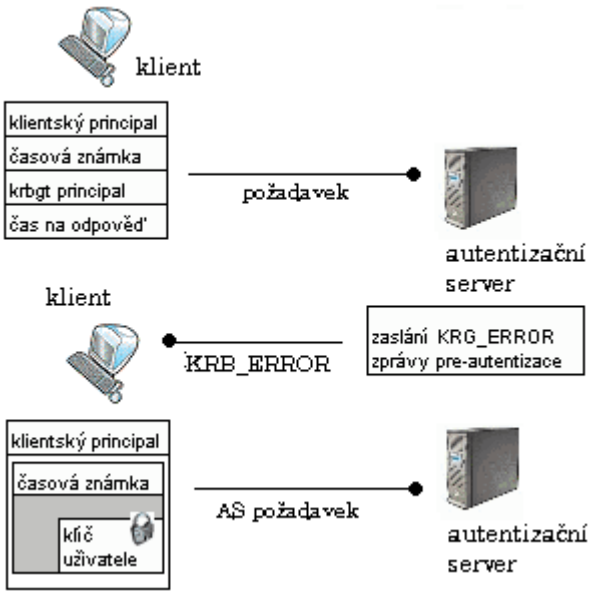


Obrázek 11: Kryptovací typy [2]

Dalším rozdílem oproti Kerberu verze 4 jsou lístky, které mají možnost voleb rozšířených rysů [2]. Tyto příznaky mohou být například následující: možnost předávat lístky, nastavení proxy lístků, obnovitelné vlastnosti lístků a možnost nastavit platnost lístků až po určitém datu (tzv. postdated tickets).

Pre-Autentizace je další z nových rysů protokolu Kerberos v5. Základem vzniku tohoto mechanismu v nové verzi protokolu byly možné útoky v předcházející verzi, které měli za cíl zaslání podvrženého klíče daného principal na KDC server. Aby se tyto útoky znesnadnily, byl zaveden žadatel, který ověřuje identitu ještě před tím, než se zašle důvěrná informace v podobě TGT lístku

zpět klientovi. Existuje několik typů pre-autentizace, nicméně nejrozšířenější je metoda časové známky (timestamp) [2].



Obrázek 12: Proces pre-autentizace [2]

Proces pre-autentizace je popsán nejlépe na obrázku 12. Pokud klient zažádá o TGT lístek a KDC server má nastavenu pre-autentizaci, tak zašle nazpět klientovi KRB_ERROR zprávu místo AS_REP. Klient na to generuje požadovaná data, nejčastěji časovou známku. Pokud jsou data KDC serverem akceptována, pak pošle klientovi požadovaný lístek.

Ucelený přehled Kerberos komunikace je součástí přílohy č.2.

3.3.3 SPNEGO

Výše popsané protokoly slouží k zabezpečení komunikace protokolů a ověření uživatele žádajícího určitou službu. Jakým způsobem ale získáme od klienta potřebné informace pro jeho autentizaci?

Podle RFC 2743 poskytuje GSS-API obecné rozhraní, které může používat rozdílné bezpečnostní mechanismy po domluvě mezi komunikujícími stranami. GSS-API je základem Simple and Protected GSS-API Negotiation (SPNEGO) (RFC 4178), který je pseudo-bezpečnostním protokolem při výběru bezpečnostního mechanismu. SPNEGO tedy ustanoví bezpečnostní kontext pro vybraný běžný bezpečnostní mechanismus. Tato vlastnost je užitečná právě pro aplikace komunikující na základě GSS-API a sdílející více bezpečnostních mechanismů mezi sebou [21].

Negotiation proces (RFC 4178) ustanovení mechanismu pro příslušný kontext je následující:

a. Iniciátor GSS-API kontextu volá GSS_Init_sec_context() metodu s tím, že zadává možnost použití SPNEGO mechanismu. SPNEGO může být výslovně požadován jako výchozí mechanismus.

b. Iniciátor GSS-API kontextu generuje negotiation token, který obsahuje seznam jednoho nebo více bezpečnostních mechanismů, které jsou založené na credentials použitých pro ustanovení kontextu.

c. Iniciátor GSS-API odešle žádost na cílovou aplikaci. V cílové aplikaci je zavolána funkce GSS_Accept_sec_context() a je jí předána žádost (token). Funkce pak provede jednu z následujících možností:
   - Pokud není žádný z navržených mechanismů přijatelný, je kontext ukončen a funkce nastavuje flag GSS_S_BAD_MECH.
   - Pokud iniciátorem preferovaný mechanismus (ten, který je na seznamu mechanismů první) není pro cílovou aplikaci přijatelný, pokračuje funkce GSS_Accept_sec_context() nastavením flagu na GSS_S_CONTINUE_NEEDED.
   - Pokud je pro ustanovení kontextu potřeba alespoň jeden další negotiation token od iniciátora, pak je nastaven flag na GSS_S_CONTINUE_NEEDED a výstup funkce je nastaven na stav neúplný (accept-incomplete).
   - Pokud již nejsou očekávány žádné další tokeny od iniciátora, pak je nastaven flag na GSS_S_COMPLETE a výstup jednání obsahuje stav úplný (accept-complete).

d. Cílová aplikace vrátí negotiation token aplikaci iniciátora. Tím je iniciován bezpečnostní kontext a pokračuje podle standardních GSS-API konvencí pro vybraný mechanismus. Tokeny vybraného mechanismu jsou zapouzdřeny v negotiation zprávách do té doby, dokud není GSS_S_COMPLETE vrácena od obou stran komunikace.

3.4 Kerberos implementace

Existuje několik implementací serveru KDC a protokolu Kerberos. Mezi nejznámější patří například MIT, Heimdal nebo MS Domain Controller.

Původní Kerberos (z laboratoří MIT) měl omezený přístup na evropský trh, protože platila přísná pravidla omezeného exportu šifrovacích algoritmů z USA. Proto vzniká evropský, volně šiřitelný klon s názvem Heimdal. I když má Heimdal jiný kód, podporuje API MIT Kerbera. Heimdal je integrovatelný s více operačními systémy včetně BSD [1]. Windows Domain Controller podporuje pouze Kerberos v5 a neobsahuje na rozdíl od implementací MIT nebo Heimdal žádnou zpětnou kompatibilitu s protokolem verze 4. Domain Controller podporuje pouze RC4 kryptování a starší typy DES.

Kerberos implementace jsou určeny pro různé platformy. Jako příklad je zde uveden seznam implementací a jim odpovídající primární platforma, pro kterou jsou určeny [27]:

MIT Kerberos	NetBSD
CyberSafe TrustBroker	UNIX, MVS, Windows 95, NT4
Sun SEAM	Solaris
DCE Kerberos (IBM)	AIX, OS/390
Computer Associates Kerberos [Platinum (OpenVision)]	Windows 95, 3.1, 3.11
Domain Controller	Windows 2000/2003
Kerberos PAM	Linux, HP-UX
Heimdal	UNIX

MIT distribuce obsahuje server, knihovny pro klientské aplikace a knihovny pro démony. Obsahuje také upravené verze BSD síťových programů jako rsh/rlogin, rcp, telnet, ftp.

Heimdal jevolně dostupná implementace Kerberos v5 pro Unix, která je rozumně kompatibilní s MIT Kerberos v5 API a podporuje GSS-API (RFC1964) . Je vyvíjena mimo území USA, takže na něj neplatí omezení ohledně vývozu šifrovacích algoritmů. Mezi základní rysy implementace patří knihovna libkrb5, pomocí které je možné pracovat s jednoduchými aplikacemi, dále knihovna GSS-API, programy kinit, klist, kdestroy, telnet, telnetd, rsh, rshd, popper, push, ftp, ftpd a další [21].

Domain Controller implementaci Kerbera je věnována samostatná kapitola 3.4.1.

3.4.1 Domain Controller

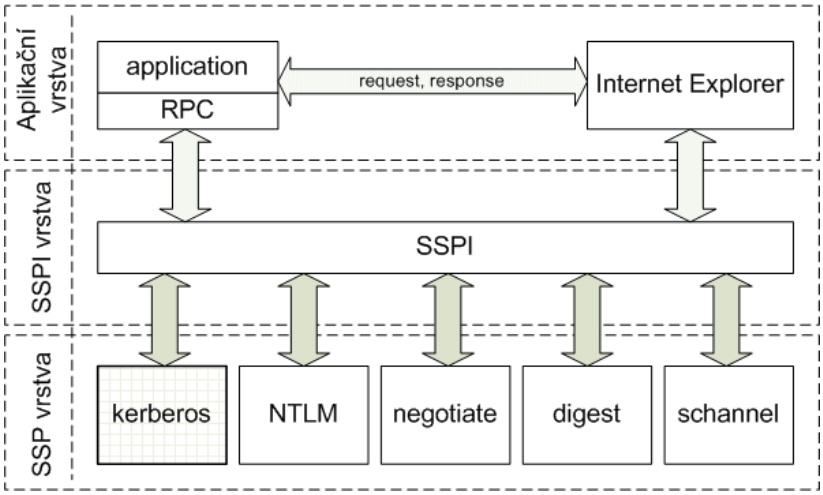
Implementace protokolu Kerberos v prostředí sítě Windows je přítomná na serveru Domain Controller (řadič domény), kde se od MS Windows 2000 využívá jako primární autentizační mechanismus. Dále budou popsány některé základní rysy, které s sebou implementace Kerbera v prostředí MS Windows přináší.

Kerberos a SSP. Windows Server 2000/2003 implementuje autentizační protokol Kerberos v5 jako Security Support Provider (dále SSP). Spolu se SSP pro NTLM jsou načítány prostřednictvím Local Security Authority (LSA) při bootování systému. Tento systém pak může obě SSP požít k autentizaci spojení klient – server. To, jaký mechanismus je použit, záleží na nastavení a možnostech počítače, ze kterého se ověření vůči Active Directory požaduje, nebo nastavením příslušné aplikace. Mezi hlavní SSP patří také Digest (standard pro LDAP a Web autentizaci), Schannel (SSP implementující Secure Socket Layer a Transport Layer Security) a Negotiate.

SSPI je implementace Generic Security Service API (GSSAPI) v prostředí Windows Server 2003 (viz. RFC 2743, RFC 2744) [20]. Jednotlivé vrstvy a jejich komunikace jsou vidět na obrázku č. 13. SSPI poskytuje mechanismus pro přenos autentizačních tokenů přes existující komunikační kanály klienta a serveru.

Kerberos SSP umožňuje, aby všechny služby v rámci domény podporovaly mimo jiné následující úkony [20]:

- Active Directory dotazy používající Lightweight Directory Access Protocol (LDAP)
- Vzdálená správa serveru nebo pracovní stanice použitím RPC volání
- Tiskové služby
- Autentizace klient – server
- Vzdálený přístup k souborům založený na Common Internet System/Server Message Block (CIFS/SMB)
- Distribuované řízení systému souborů
- Certifikované požadavky na certifikované služby pro doménové uživatele a počítače



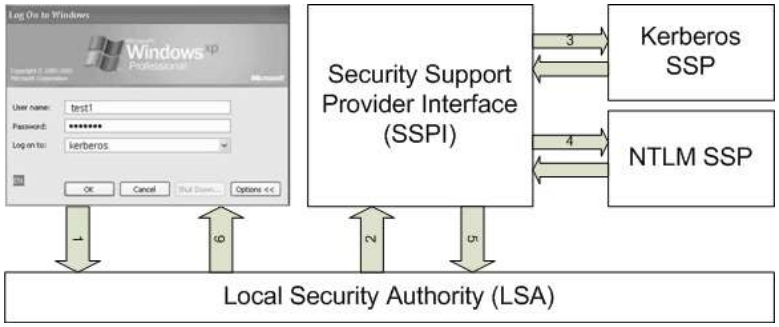
Obrázek 13: Komunikace autentizačních vrstev

Autentizační údaje definují autentizační atributy. Protokol Kerberos (RFC 1510) popisuje mnoho atributů, které definují výměnu autentizačních údajů. Stejně tak je tomu i v implementaci Windows 2003, která však doplňuje několik dalších informací, které se v ostatních implementacích nevyskytují. Jedná se o následující skupiny informací:

- Přihlašovací informace. Informace o lístku a klientovi, jakými jsou Group IDs (RIDs)
- Dodatečné přihlašovací údaje. Vrací služba KDC a umožňuje PKINIT
- Podpisy. Ověřovací data od Microsoftu obsahují dva digitální podpisy, jeden pro Domain Controller a druhý pro server, na kterém běží požadovaná služba
- PAC (Privilege Attribute Certificate). Microsoft ověřovací data jsou zahrnuta v každém před-autentizačním (pre-authenticated) lístku obdrženém po AS požadavku. Klient si nicméně může explicitně vyžádat, zda chce nebo nechce SIDs zahrnout do lístku. Při defaultním nastavení jsou seznamy SIDs mapovány.

Credentials Cache je úložiště na počítačích, kde běží Windows 2000/2003 nebo Windows XP. V Credentials Cache jsou uloženy lístky a klíče, které počítač obdrží od serveru KDC. Jedná se o dočasnou paměť chráněnou LSA. Tato paměť není nikdy mapována na disk a je vymazána pokaždé,

když dojde k odhlášení uživatele od počítače. Credentials cache je řízena Kerberos SSP, který běží v LSA bezpečnostním kontextu. LSA i SSPI jsou základním rozhraním pro proces logování uživatele do domény (viz. obrázek č. 14.).



Obrázek 14: Proces logování do domény

Active Directory obsahuje databázi účtů, kterou KDC server potřebuje pro ukládání autentizačních informací k jednotlivým účtům. Každý principal je reprezentován objektem účtu a kryptovaný klíč, který se používá při komunikaci s uživatelem, počítačem nebo službou, je uložen jako atribut tohoto objektu [20]. Fyzické úložiště dat jednotlivých účtů je řešeno prostřednictvím Directory System Agenta (DSA) – zabezpečeným procesem integrovaným s LSA na Domain Controller serveru. Dále je zaručeno, že klienti účtů nemají žádný přístup k uloženým datům, ale pro přístup k informacím v adresáři je třeba využít rozhraní Active Directory Service Interface ve spojitosti s DSA.

3.5 Služby a nastavení Active Directory

Kapitola obsahuje popis základního nastavení prostředí Active Directory pro možné použití autentizace příslušného uživatele nebo služby pomocí protokolu Kerberos. K tomuto účelu poslouží příklad webové aplikace se jménem HTTP, která poběží na aplikačním serveru „as“ (host name aplikačního serveru). Dále definujeme službu se jménem SERVICE běžící na serveru „se“ (host name serveru služby). Tyto servery budou součástí domény KERBEROS.LOCAL.

Pro autorizovaný přístup k dané službě v doméně je třeba, aby byla tato služba zastoupena uživatelským účtem. Pro službu aplikačního serveru tedy použijeme jméno krb5as (tedy logon name v doméně bude krb5as@kerberos.localhost) a pro službu SERVICE použijeme jméno krb5se (tedy logon name v doméně bude krb5se@kerberos.localhost).

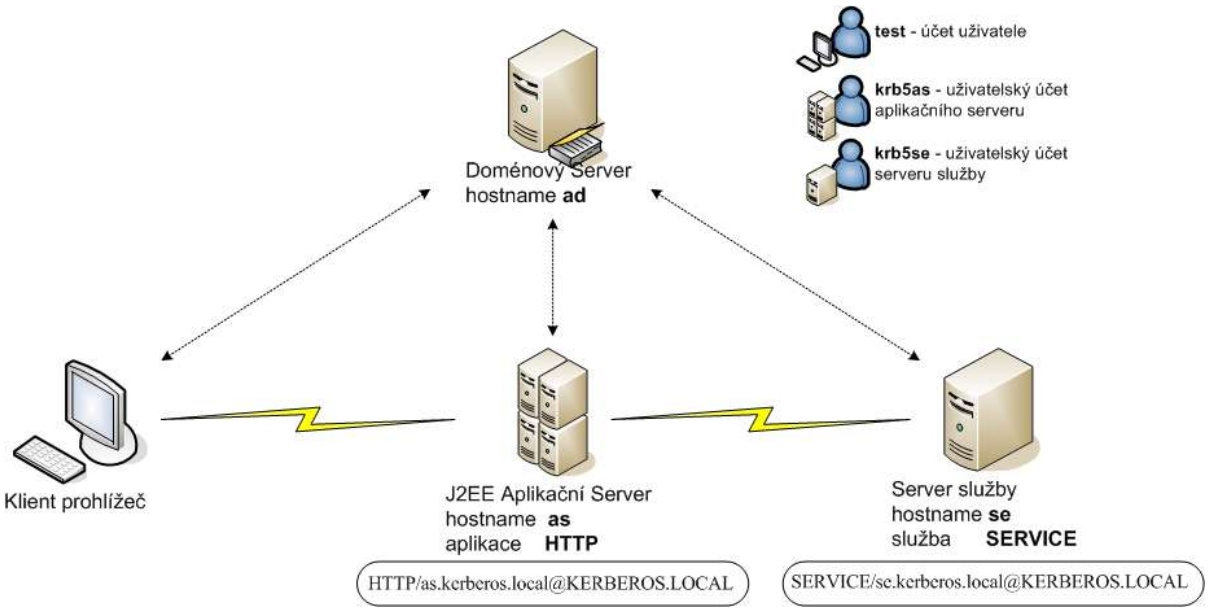
Principal služby aplikačního serveru HTTP mapovaný na uživatele služby krb5as bude řetězec

```
HTTP/as.kerberos.local@KERBEROS.LOCAL
```

Principal služby serveru služby SERVICE mapovaný na uživatelský účet služby krb5se bude řetězec

```
SERVICE/se.kerberos.local@KERBEROS.LOCAL
```

Celkové nastavení prostředí domény je znázorněno na obrázku č. 15.



Obrázek 15: Active Directory prostředí

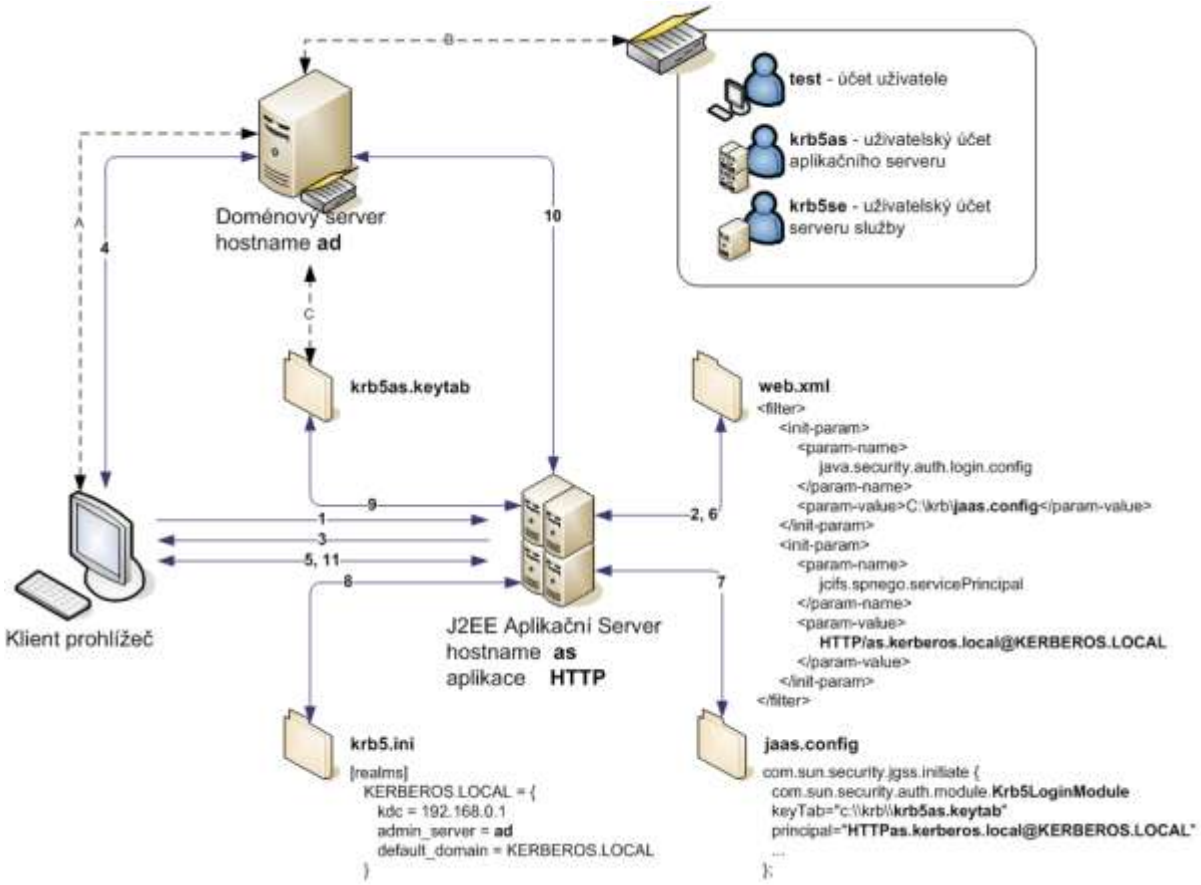
3.5.1 Konfigurace pro J2EE webovou aplikaci

Kerberos komunikace v prostředí Active Directory je založena na základních principech a pravidlech protokolu Kerberos. Každá J2EE webová aplikace, která bude chtít využívat k autentizaci uživatelů protokol Kerberos, vyžaduje několik kroků konfigurace. Konfiguraci pak můžeme dělit na systémovou a aplikační.

a. **Systémová konfigurace** umožňuje na systémové úrovni přihlášení prostřednictvím protokolu Kerberos. Sem patří konfigurace Active Directory, klienta a vytvoření keytab souboru.
   - Konfigurace Active Directory serveru umožňuje vytvořit uživatelské účty ke službám nebo aplikacím v doméně (obrázek 16. B).
   - Nastavení chování klienta tak, aby při prvním přihlášení do domény obdržel a uložil TGT lístek pro další komunikaci s KDC. V prostředí Windows sítě se jedná o modul GINA ukládající informace do LSA (obrázek 16. A).
   - Vytvoření keytab souboru je uskutečněno na serveru KDC. Vstupem je jméno principal dané služby a jméno uživatelského účtu služby. Výstupem je soubor, který je potřeba bezpečně přenést na server služby (obrázek 16. C).

b. **Aplikační konfigurace** se v zásadě skládá z kroků na straně serveru služby. Následně budeme uvažovat aplikační J2EE server, který využívá ke Kerberos autentizaci knihovnu JAAS/JGSS.

- krb5.ini – soubor definuje spojení aplikačního serveru s KDC serverem. Součástí je adresa KDC serveru a konfigurace vlastností komunikace.



Obrázek 16: Webový server v doméně. Čísla udávají pořadí komunikace.

- web.xml – konfigurace chování webové aplikace. Součástí je definice filtru, který provádí autentizaci při přístupu k určitému zdroji na webovém serveru. Zároveň soubor odkazuje na konfigurační soubor J2EE knihovny JAAS jaas.config.
- jaas.config – soubor definuje základní chování autentizačního modulu. Součástí je cesta ke keytab souboru, cesta k modulu provádějícímu autentizaci a jméno ověřovaného principal.

3.5.2 Nastavení prostředí Active Directory

Proces konfigurace systémového prostředí Active Directory za účelem možnosti přistoupit k požadované službě pomocí webového rozhraní se skládá z následujících částí:

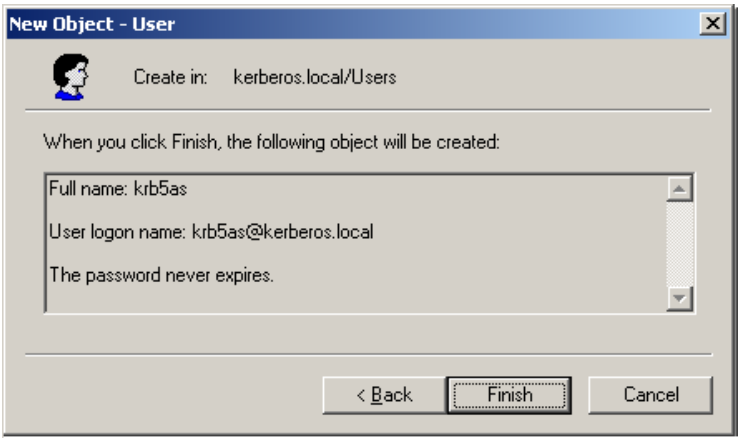
a. Vytvoření uživatelského účtu SPN

b. Vytvoření keytab souboru
c. Nastavení webového serveru
d. Konfigurace klienta

a. Vytvoření uživatelského účtu SPN

Pro každý server (každou službu), ke kterému se bude pomocí protokolu Kerberos přistupovat, je třeba vytvořit speciální uživatelský účet v Active Directory. Pro náš případ bude nazván uživatelský účet krb5as. Postup je následující:

- Administration tools ->Active Directory Users and Computers -> User -> New -> User
- Vyplnění First name a User logon name jménem uživatelského účtu – krb5as, Next
- V následujícím kroku zadat a potvrdit heslo – pro názornou ukázku použijeme psswd_krb5as
- Pokud výsledný uživatelský účet vypadá podobně jako na obrázku č. 17, můžeme potvrdit jeho vytvoření pomocí Finish



Obrázek 8: Vytvoření nového uživatele

- Na závěr je třeba modifikovat uživatelský účet specifikací kryptovacího algoritmu. Otevřít vlastnosti uživatele User -> Properties -> Account.
    - V sekci Account options zatrhnout „Use DES encryption types for this account"
    - V sekci Account options zatrhnout „Account is trusted for delegation"
    - Zkontrolovat, zda není zatržená možnost „Account is sensitive and cannot be delegated"
    - Zkontrolovat, zda není zatržená možnost „Do not require Kerberos preauthentication"

Upozornění: po změně nastavení Account options je třeba resetovat heslo! Pro reset hesla pravým klikem na uživatelský účet a vybrat Reset Password.

b. Vytvoření keytab souboru

K vytvoření keytab souboru se používá utilita ktpass z instalace Windows Resource Kit.

Ktpass program vytvoří vazbu mezi SPN a uživatelským účtem a exportuje důvěrné informace do keytab souboru.

```
C:>ktpass –princ HTTP/as.kerberos.local@KERBEROS.LOCAL -crypto  DES-CBC-MD5  -ptype
KRB5_NT_PRINCIPAL –mapuser krb5as –pass psswd_krb5as –out krb5as.keytab +DesOnly
```

Definice parametrů:

- *princ* parametr specifikuje jméno SPN. To znamená, že host http://as.kerberos.local je URL, které chce využít autentizaci. HTTP je důležitý prefix při použití přes prohlížeč.
- *mapuser* parametr specifikuje uživatelský účet, který se váže na SPN.
- *pass* definuje korespondující heslo k uživatelskému účtu.
- *out* parametr specifikuje jméno výstupního keytab souboru.
- *crypto* des-cbc-md5 parametr specifikuje použitý šifrovací algoritmus.

Pokud byl příkaz proveden správně, pak obsahuje konec výpisu následující sdělení (možné warning před tímto výpisem pouze informují o tom, že již k mapování došlo dříve a nejedná se o chybu):

```
Key created.
Output keytab to webserver.keytab:
Keytab version: 0x502
keysize 59 HTTP/as.kerberos.local@KERBEROS.LOCAL ptype 1 (KRB5_NT_PRINCIPAL) vno -1 etype
0x3 (DES-CBC-MD5) keylength 8 (0xf11f0e1c1cb0a207)
Account has been set for DES-only encryption.
```

c. Nastavení webového serveru

Pokud webový server běží na počítači, který je součástí Windows domény Active Directory, pak je potřeba nastavit důvěru k tomuto počítači:

- Administrative Tools -> Active Directory Users and Computers -> Computers
- Vlastnosti serveru as a záložka General
- Zatrhnout „Trust computer for delegation”

Vstupem dalších kroků je keytab soubor (krb5as.keytab) a jméno principal dané služby (HTTP/as.kerberos.local@KERBEROS.LOCA).

- Bezpečně přenést keytab soubor krb5as.keytab na webový server
- Změna v souboru jaas.config – zadání jména keytab souboru a jména principal. Následuje příklad modulu jgss.initiate:

```
com.sun.security.jgss.initiate {
  com.sun.security.auth.module.Krb5LoginModule
  required
  storeKey=true
  keyTab="c:\\krb5as.keytab"
  doNotPrompt=true
  useKeyTab=true
  realm="KERBEROS.LOCAL"
  principal="HTTP/as.kerberos.local@KERBEROS.LOCAL"
  debug=true;
 };
```

d. Nastavení klienta

Konfigurace prohlížeče musí umožňovat podporu Kerberos protokolu. Single Sign-On pracuje pouze na intranetové úrovni a využívá důvěry daných URL.

Konfigurace prohlížeče Internet Explorer (v5.5 a novější):

- Přidání URL do Local intranet. Tools -> Internet Options, vybrat záložku Securiry -> Local intranet a následně tlačítko Sites. Zde budeme potřebovat přidat SPNEGO Tomcat host do seznamu adres, kterým v rámci intranetové sítě důvěřujeme.
- Konfigurace automatické autentizace v prohlížeči. Tools -> Internet Options, vybrat záložku Securiry -> Local intranet a následně tlačítko Custom Level. V User Authentication -> Logon je třeba, aby byla označena položka „Automatic logon only in Intranet zone“.
- Pokud se jedná o prohlížeč Internet Explorer verze 6 a vyšší, musí se manuálně umožnit SPNEGO Single Sign-On. Tools -> Internet Options, vybrat záložku Advanced a následně si ověřit, zda v sekci Security je nastaveno „Enable Integrated Windows Authentication (requires restart)“.
- Pokud je použitý proxy server, pak je třeba Proxy Settings nastavit tak, aby nedocházelo k zaslání žádosti v rámci intranetu na aplikační server přes proxy servery. Internet Options -> Connections -> LAN settings, Proxy server -> Advanced -> Proxy Settings.

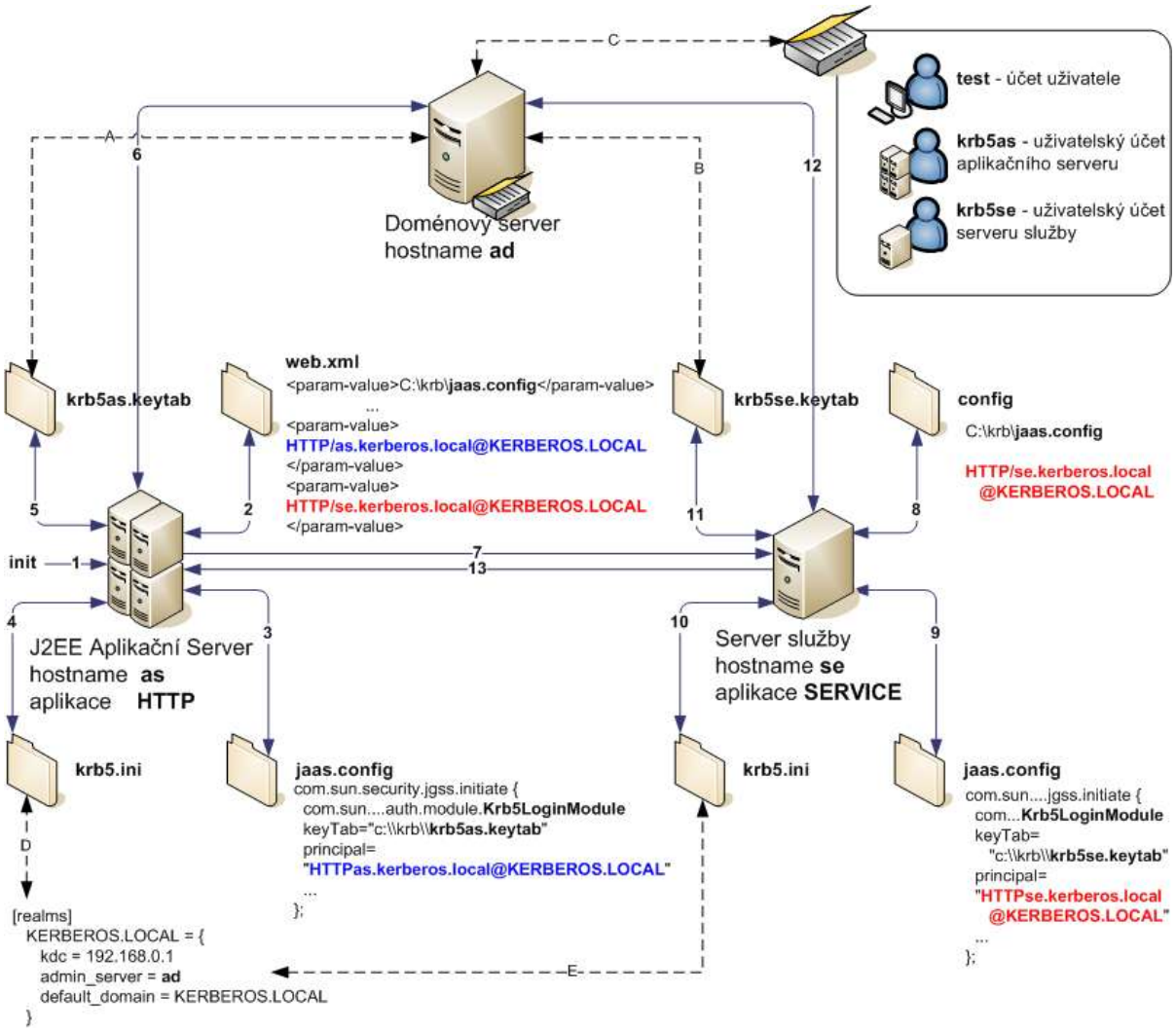
Konfigurace prohlížeče Firefox (v1.0 a novější):

- Nastavení hodnot předvolby network.negotiate-auth.delegation-uris
- Nastavení hodnot předvolby network.negotiate-auth.trusted-uris

3.5.3 Služby v doméně

V případě složitějších architektur webových aplikací je třeba přistoupit ke službě, která stejně jako aplikace vyžaduje autentizaci příchozího požadavku na službu. V tomto případě je situace obdobná, jako u dotazu z webového prohlížeče směrem k serveru. V prvním kroku požádá aplikační server Domain Controller o lístek k požadované službě. Pokud je aplikační server již autorizován, obdrží požadovaný lístek, který pak pošle na server služby.

Obrázek č. 18 obsahuje diagram komunikace aplikačního serveru se serverem služby. Čísla vyjadřují směr a pořadí komunikace. V případě CMS systému se může jednat o J2EE aplikační server, který si na základě ověřeného lístku od webového prohlížeče (klienta) zažádá doménový server o lístek služby pro přístup ke Content Serveru (obrázek 18. 6).



Obrázek 9: Služba v doméně

Poté, co aplikační server lístek služby obdrží, zašle tento lístek na server služby (se), kde dojde k ověření poslaného lístku standardním postupem přes keytab soubor. Je téměř pravidlem, že se přistupuje ke stejnému doménovému KDC serveru. Proto je možné nakonfigurovat soubor krb5.ini stejně na obou serverech.

4 Autentizace na platformě J2EE

4.1 Koncept JAAS/JGSS

4.1.1 Základy Java autentizace

Java platforma poskytuje pro autentizaci bezpečnostní mechanismus, který je postaven na JAAS (Java Authentication and Authorization Service) a Java GSS-API. JAAS se používá k autentizaci uživatelů tak, že se spolehlivě a bezpečně určí, kdo aktuálně provádí Java kód. Prostřednictvím JAAS můžeme zároveň kontrolovat přístupová práva dovolující provést zabezpečené operace. Java GSS-API se používá pro bezpečnou výměnu zpráv mezi komunikujícími aplikacemi [22]. Java GSS-API obsahuje Java vazby pro Generic Security Services Application Program Interface (GSS-API) definované v RFC 2853. GSS-API implementace v jazyce Java (Java 2 Standard Edition) obsahuje podporu Kerberos V5 mechanismu.

JAAS je Java implementace frameworku PAM (Pluggable Authentication Module).
JAAS autentizace je svou strukturou postavena na volání ověřovacího pluginu (modulu). To aplikacím dovoluje zůstat nezávislé na jejich bázových autentizačních technologiích. Autentizační technologie pak mohou být využívány bez požadovaných změn samotné aplikace. Aplikace umožní autentizační proces ustanovením LoginContext objektu, který střídavě odkazuje na Configuration objekt, který určuje technologii autentizace, nebo na LoginModule, který vykonává samotnou autentizaci. Obecně pak LoginModule dotazuje žadatele o službu na přihlašovací údaje (login, heslo, ticket), nebo může číst a ověřovat vzorky hlasu nebo otisky prstu.

K tomu, aby bylo vytvořeno zabezpečené spojení mezi klientem a serverem a byly vytvořeny kryptované klíče této komunikace, potřebuje mít mechanismus GSS-API přístup ke credentials na jednotlivých stranách spojení. V rámci vazby GSS-API se tedy může jednat na straně klienta o Kerberos lístek ke službě a na serveru se obvykle jedná o Kerberos tajný klíč (keytab). Java GSS-API rozhraní požaduje , aby mechanismus obdržel credentials od subjektu (Subject), který je spojován s daným kontextem. K tomu, aby mohl být subjekt s danými credentials spojen, použije aplikace nejprve JAAS autentizaci a její Kerberos modul.

Aplikace klienta i aplikace serveru použijí k ustanovení bezpečnostního kontextu objekt GSSContext. Inicializace tohoto objektu je provedena rozdílně na klientovi a na serveru. Iniciátor kontextu si například může vybrat z více možností nastavení vlastností kontextu. Po těchto nastaveních je volána metoda initSecContext, která produkuje token, který je vstupem do metody

acceptSecContext na serveru služby. Vytvořený kontext tak zajistí bezpečný přenos Kerberos lístků mezi klienty.

4.1.2 Konfigurační soubor jaas

JAAS podporuje vestavěný autentizační framework. To znamená, že jakýkoliv typ autentizačního modulu může být vestavěnou součástí aplikace. Login konfigurace definuje login modul pro použití v konkrétní aplikaci. Původní JAAS implementace od Sun Microsystems požaduje, aby byly konfigurační informace specifikované v konfiguračním souboru [23].

Klient i server mohou používat stejný konfigurační soubor, pokud tento soubor obsahuje dvě položky – klientskou inicializační a serverovou akceptační. Jednoduchý konfigurační soubor jaas.config může vypadat následovně:

```
com.sun.security.jgss.initiate {
  com.sun.security.auth.module.Krb5LoginModule  required;
};

com.sun.security.jgss.accept {
  com.sun.security.auth.module.Krb5LoginModule  required  storeKey = true
};
```

Základem souboru je definice Kerberos login modulu, kterým je v tomto případě Krb5LoginModule. Dvě části souboru odpovídají inicializační straně klienta a akceptační straně serveru. U definice serveru je přidána parametr storeKey a nastaven na true. Parametr označuje, že tajný klíč by měl být vypočítán z hesla poskytnutém při logování a měl by být uložen v privátní cache credentials Subjectu vytvořeném jako výsledek úspěšného logování do domény. Mezi další argumenty je možné zařadit umístění keytab souboru parametrem keyTab, dále možnost použití keytab souboru, definice realm domény, celý název příslušného principal nebo možnost nastavení úrovně debugování.

4.1.3 Výstup login modulu

Standardní výstup login modulu Krb5LoginModule je součástí přílohy č.1. V souvislosti s nastavením prostředí a aplikace se vyskytují čtyři základní chybové výstupy modulu:

1. Podpora šifrování – pokud KDC nepodporuje typ šifrování, který předpokládá server služby a který je definován v konfiguračním souboru krb5.ini, je výstupem následující zpráva:
   GSSException: Failure unspecified at GSS-API level (Mechanism level: KDC has no support for encryption type (14))
2. Rozdíl času v doméně – pokud je mezi komunikujícími stranami časový rozdíl větší, jak nastavená tolerovaná hodnota, je výstupem následující zpráva:

GSSException: Failure unspecified at GSS-API level (Mechanism level: Clock skew too great (37))

3. Chybná konfigurace modulu – výstupem je zpráva:
   GSSException: No valid credentials provided (Mechanism level: Attempt to obtain new ACCEPT credentials failed!)
4. Chyba načtení keytab souboru – výstupem je zpráva:
   GSSException: No valid credentials provided (Mechanism level: Failed to find any Kerberos Key)

4.2 Autentizační filtr

Autentizační filtr dovoluje zpracovat příchozí request od klienta ještě před tím, než dorazí v rámci J2EE aplikace k servletu. Této vlastnosti je použito v rámci autentizace při přístupu k určitému zabezpečenému zdroji na serveru (např. servlet). Koncept ověření uživatele na základě protokolu Kerberos používá autentizační filtr při přístupu uživatele pomocí webového prohlížeče k webové aplikaci. Filtr pak zaručí získání potřebných lístků během komunikace s klientem ještě před tím, než se request dostane do samotné aplikace.

4.2.1 Schéma filtru

Schéma filtru implementuje rozhraní Filtr, které definuje jeho základní části init, destroy a doFilter.

```
public class AuthenticationFilter implements Filter { }
```

Inicializace filtru je provedena pomocí metody init. Prostřednictvím jediného parametru metody FilterConfig je možné se dostat ke konfiguraci filtru, která je uložena v xml souboru aplikace web.xml (příloha č.5). Definice obsahuje jméno filtru a třídu, která rozhraní filtru implementuje. Následují parametry definující jak systémový tak aplikační základ filtru. Systémovým základem filtru je specifikace systémových proměnných. Jako příklad je možné uvést cestu ke konfiguračnímu souboru jaas.config, použití credential cache nastavením useSubjectCredsOnly a povolení debug výpisu u ověřovacího modulu.

```
<init-param>
   <param-name>java.security.auth.login.config</param-name>
   <param-value>C:\krb\jaas.config</param-value>
</init-param>
<init-param>
   <param-name>javax.security.auth.useSubjectCredsOnly</param-name>
   <param-value>false</param-value>
</init-param>
<init-param>
   <param-name>sun.security.krb5.debug</param-name>
   <param-value>true</param-value>
</init-param>
```

Dalším parametrem filtru je jméno Principal, pro které je vyžadován lístek služby Kerbera použitý při vytváření kontextu metodou createAcceptContext.

```
<init-param>
    <param-name>com.krb5.servicePrincipal</param-name>
   <param-value>HTTP/as.kerberos.local@KERBEROS.LOCAL</param-value>
</init-param>
```

Metoda doFilter provádí ověření uživatele na základě http parametrů request a response. Jak je vidět na obrázku č.19, prochází request ve filtru přes několik podmínek, které ověřují tvar http hlavičky odeslané od klienta. Při prvním dotazu klienta na zdroj je kontrolována hlavička Authorization tak, že pokud neobsahuje autorizační údaje Kerberos, Negotiate, NTLM nebo Basic, pak je request a response přesměrován na obslužnou metodu fail(), která příslušným způsobem nastaví odpověď response tak, aby při opětovném dotazu na zdroj byly součástí dotazu i příslušné identifikační údaje.

```
authType = msg.regionMatches(true, 0, "Negotiate ", 0, 10) ?
HTTP_NEGOTIATE : msg.regionMatches(true, 0, "NTLM ", 0, 5) ?
HTTP_NTLM : HTTP_BASIC;
```

Pokud je tedy povolen autentizační mechanismus Kerberos, nastaví se k hlavičce parametru WWW-Authenticate hodnota Negotiate a stav odpovědi je nastaven na unauthorized.

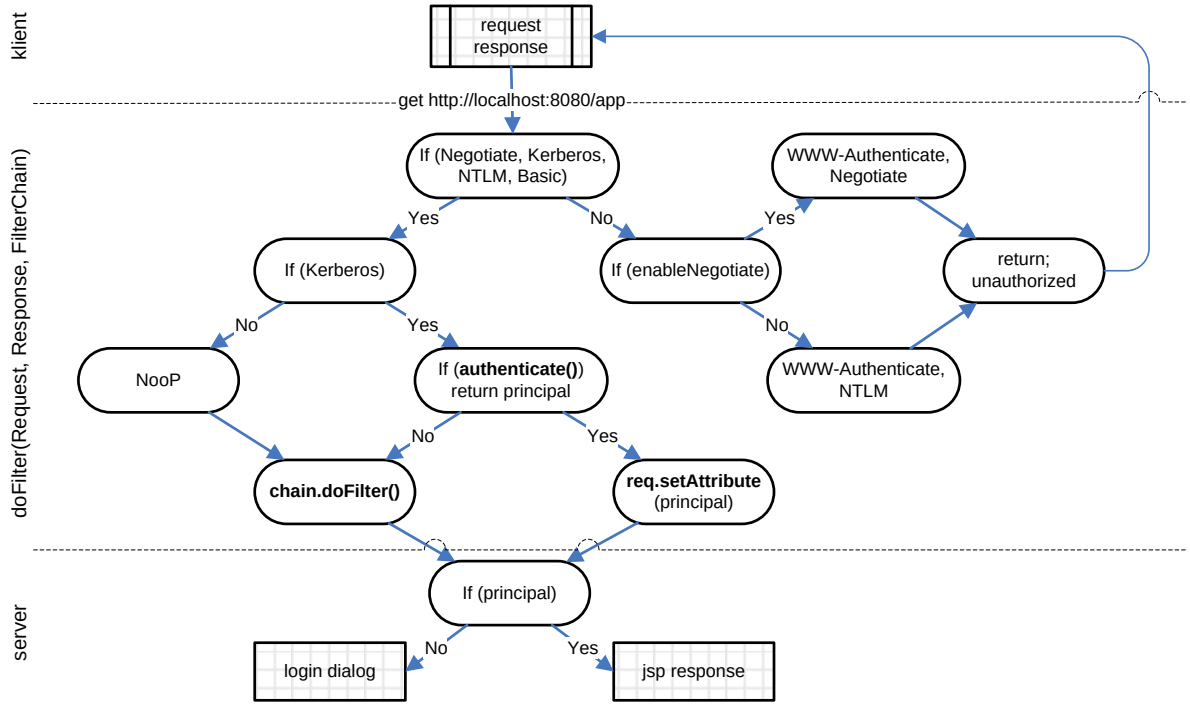
```
if (enableNegotiate) {
   resp.addHeader("WWW-Authenticate", "Negotiate");
}
resp.setStatus(HttpServletResponse.SC_UNAUTHORIZED);
```

V druhém průchodu filtrem již požadavek na zdroj obsahuje v případě úspěchu na klientovi autentizační údaje (lístky) a pokud se jedná o Kerberos lístek, je zavolaná metoda authenticate(), která obdržený lístek ověří.

```
principal = Negotiate.authenticate(req, resp, servicePrincipal, secondPrincipal);
```

Pokud ověření nedopadne dobře, tak je třeba tuto skutečnost určitým způsobem předat dále a patřičně na ni reagovat. To je možné dvěma způsoby. Prvním způsobem je vyvolání výjimky, která při selhání autentizační procedury na straně serveru značí neúspěch. Důsledkem vyvolání výjimky je vyvolání login dialogu. Druhým způsobem je odchycení výjimky a nastavení atributu u requestu, který je dále poslán směrem k servletu (to se děje i u úspěšného ověření). Výhoda druhého způsobu spočívá v tom,

že jako atribut je možné předat lístek Kerbera. Pokud tedy aplikace volá další zabezpečenou službu a součástí autentizačního filtru je inicializační procedura pro získání lístku služby pro tuto službu, pak je možné tento lístek poslat dále do aplikace k případnému zpracování nebo zaslání na server služby.



Obrázek 19: Metoda doFilter

Vstupem do metody doFilter je tedy request a response od klientského prohlížeče a výstupem celého procesu je v případě neúspěšného přihlášení login dialog nebo jsp stránka v případě úspěchu.

4.2.2 Autentizace – server

V pojetí struktury systému klient – server je akceptorem lístku služby server. Ten vytvoří bezpečnostní kontext voláním metody createContext() objektu GSSContext, kterému se jako argument připojuje objekt GSSCredentials.

Metoda doAs (resp. doAsPrivileged) spojuje specifický subject s kontextem AccessControlContext a poté vykoná danou aktivitu navázanou na tento kontext. Tím je dosaženo efektu běhu aktivity pod daným subjektem.

```
doAsPrivileged.invoke(null, new Object[] {
subject, action, null }) : action.run());
```

Potom, co server ustanoví GSSContext a od klienta obdrží Kerberos lístek služby generovaný na straně klienta (např. webový prohlížeč nebo aplikace), je volána metoda acceptSecContext. Metoda acceptSecContext je součástí balíku org.ietf.jgss, který definuje základ rozhraní JGSS-API.

```
acceptSecContext.invoke(Authentication.this.context,
new Object[] { token, new Integer(0),
new Integer(token.length) });
```

4.2.3 Inicializace – klient

V předchozích odstavcích byla popsána komunikace mezi webovým prohlížečem, který plnil funkci klienta, a aplikačním serverem. Pokud se jedná o jednoduchou webovou aplikaci, která provádí ověření na aplikačním serveru a nevolá žádnou další zabezpečenou službu, pak popsaný koncept postačí. Pokud je ale potřeba přistoupit z aplikačního serveru k určité další zabezpečené službě, pak je potřeba iniciovat nové bezpečnostní rozhraní pro předání Kerberos lístků. V rámci autentizačního procesu je pak KDC poptán po Kerberos lístku pro přístup k dané službě, dále je tento lístek přeposlán pomocí rozhraní JGSS-API na server služby, na kterém je proveden stejný mechanismus ověření, jako byl popsán v předchozích kapitolách. Následuje popis základních částí konceptu JGSS klienta.

Vstupem k vytvoření kontextu rozhraní JGSS je Oid (Universal Object Identifier). Oid jsou globálně interpretované identifikátory použité v rámci rozhraní GSS-API k identifikaci bezpečnostního mechanismu. Struktura a kódování identifikátorů Oid je definována v ISOIEC-8824 a ISOIEC-8825 [23]. Obě komunikující strany musejí být na mechanismu domluveny, jinak dojde k výjimce ze strany JGSS. Níže uvedený příklad reprezenruje Oid Microsoft Kerberos mechanismu.

```
Oid krb5Oid = new Oid("1.2.840.48018.1.2.2");
```

Na základě poskytnutého typu mechanismu a na základě objektu GSSManager je vytvořen GSS kontext pro vzájemnou autentizaci se serverem. Vstupem vytvoření kontextu jsou čtyři argumenty. Prvním argumentem je jméno serveru služby, které je zastoupeno GSSName objektem pro service principal reprezentaci serveru. Druhým parametrem je již zmíněný Oid mechanismus. Dalšímu parametru je ponechána hodnota null v případě credentials klienta (default credentials) a poslední parametr určuje délku platnosti kontextu.

```
GSSContext context = manager.createContext(serverName,
        krb5Oid,
        null,
        GSSContext.DEFAULT_LIFETIME);
```

U kontext objektu je možné nastavit několik vlastností. Mezi důležitou vlastnost z hlediska procesu ověření patří možnost vzájemné (mutual) autentizace. Pokud je hodnota nastavena na true, pak se od serveru služby očekává, že dojde k potvrzení výsledku ověření a tím pádem zaslání informace v rámci vytvořeného kontextu zpět na klienta. V případě, že je v rámci aplikace ověření uživatele součástí serveru služby, není potřeba v kontextu GSS zpětná vazba a vlastnost se nastavuje na false.

```
context.requestMutualAuth(false);
```

Další důležitou metodou vytvořeného kontextu je initSecContext. Metodu používá iniciátor kontextu ve chvíli, kdy chce uskutečnit vytvoření tohoto kontextu a vygenerovat token pro server služby. Tento token je výstupem metody a po přenosu na server služby se stává vstupem pro metodu acceptSecContext. Vstupem metody initSecContext může být v případě návaznosti na již aktivní kontext token jako výstup z tohoto kontextu.

```
token = context.initSecContext(token, 0, token.length);
```

4.3 Systémová a aplikační podpora

4.3.1 Úroveň podpory klienta

Podmínkou pro správnou funkčnost protokolu Kerberos na klientském počítači je podporovaná verze prohlížeče. Níže je uvedený seznam základních podporovaných prohlížečů na různých operačních systémech:

MS Windows	Linux	MacOS X
MSIE 5.x a 6 +	Mozilla Suite 1.7.5 +	Mozilla Suite 1.7.5 +
Mozilla Suite 1.7.5 +	Mozilla Firefox 1.0 +	Mozilla Firefox 1.0 +
Mozilla Firefox 1.0 +	Netscape 7.2 +	Netscape 7.2 +
Netscape 8.0 +	Konqueror 3.5 +	Safari OSX 10.4.3

Mezi základní prohlížeče, které doposud neobsahují podporu Kerbera, patří především Opera.
V prostředí MS Windows druhý používaný autentizační protokol NTLM obsahuje na rozdíl od protokolu Kerberos podporu ve více prohlížečích. Oproti prohlížečům, které podporují protokol Kerberos, ještě přibudou prohlížeče Netscape 7.2 +, Opera 9.x + a MSIE 5.x.

4.3.2 Úroveň podpory serveru

Webový Server Apache obsahuje podporu protokolu Kerberos formou přídavného modulu mod_auth_kerb. Použitím Basic Authentication mechanismu obdrží uživatelské jméno a heslo od prohlížeče a ověří jej oproti Kerberos serveru podle toho, jak je ověření nastaveno. Modul dále podporuje Negotiation ověřovací metoda, která zároveň zaručuje plnou podporu Kerberos autentizace založené na zasílání lístků a nepožaduje tak po uživateli zadat heslo do formuláře. Pokud se používá Negotiate metoda, pak je třeba podpory ze strany webového prohlížeče (viz. Úroveň podpory klienta).

Modul podporuje jak Kerberos v4 tak Kerberos v5 protokol. Negotiate mechanismus může být použit pouze s Kerberos v5. Modul podporuje jak 1.x tak i 2.x verzi Apache. Při použití Basic Authentication mechanismu, nepoužívá modul žádné speciální kryptovací techniky. Při použití metody Negotiate je použito kódování Base64. Je tedy jednoduché převést řetězec na jednoduchý text. Abychom tomuto zabránili, je výhodné používat modul mod-ssl nebo Apache-ssl [13].

Podpora protokolu NTLM je zaručena dvěma moduly. Prvním modulem je mod_ntlm. Tento modul však nepodporuje protokol NTLM2, který je už jako jediný podporován v rámci Windows Vista. Podpora NTLM2 je umožněna modulem mod_auth_ntlm_winbind. Tato řešení jsou využívána na serveru Apache bez použití Windows. Pokud je použit OS Windows, je řešením modul mod_auth_sspi [14].

Microsoft Internet Information Services (IIS) podporuje jak Kerberos tak NTLM protokol pro síťovou autentizaci. IIS přímo zasílá Negotiate hlavičku během Windows autentizace. Hlavička tak dá klientovi vybrat, zda si chce zvolit mezi Kerberos a NTLM autentizací. Negotiate proces nevybere Kerberos autentizaci pokud platí alespoň jedna z následujících možností:

- Jeden ze systémů, které jsou zapojeny do autentizace, nemohou použítKerberos autentizaci.
- Volaná aplikace neposkytuje informace pro použití Kerberos protokolu.

Pro umožnění procesu Negotiate k výběru Kerberos protokolu pro síťovou autentizaci, musí klientská aplikace poskytovat Service Principal Name (SPN) a User Principal Name (UPN) nebo NetBIOS jméno účtu jako cílové jméno. V opačném případě Negotiate proces vybere NTLM protokol jako autentizační metodu.

Pro nastavení použití Kerberos a NTLM protokolu je třeba potvrdit, že Negotiate hlavička je nastavena v NTAuthenticationProvider vlastnostech. Tyto vlastnosti se nastavují rozdílně podle verze IIS [15]. Pro nastavení autentizace na Kerberos a NTLM u IIS v6.0 je třeba zjistit aktuální hodnoty NTAuthenticationProvider. To je možné provést následujícím příkazem:

```
cscript adsutil.vbs get w3svc/WebSiteIDnumber/root/NTAuthenticationProviders
```

Adsutils.vbs najdeme v adresáři C:/Inetpub/Adminscripts. Pokud je proces Negotiate podporován, vrátí příkaz následující hodnotu:

```
NTAuthenticationProviders : (STRING) "Negotiate,NTLM"
```

Pokud takovou hodnotu server nevrátí, je třeba zadat následující příkaz pro nastavení autentizace:

```
cscript adsutil.vbs set w3svc/ WebSiteIDnumber /root/NTAuthenticationProviders "Negotiate,NTLM"
```

5 Proces implementace řešení

Implementace řešení Kerberos Single Sign-On do komerčního podnikového prostředí vyžaduje překvapivě mnoho času. Je to způsobeno charakterem celého mechanismu ověřování, který je také důvodem, proč je potřeba věnovat dostatek času analýze celého projektu a zamyšlením se nad několika technickými detaily, které ovlivní pohled na implementaci řešení.

5.1 Systémová implementace

Kerberos je jako takový historicky osvědčeným, robustním a čistým řešením Single Sign-On. Jedná se však o řešení, které je silně závislé na prostředí, ve kterém se nachází. To má za následek jednu výhodu a jednu nevýhodu. Výhoda spočívá v použití Single Sign-On na základě přihlášení k pracovní stanici v doméně a využití systémových zdrojů a vlastností výpočetních strojů k ověření uživatele při přístupu k serverovým službám. Výhoda je to tedy pro klienta. Nevýhoda spočívá v různorodém a často nehomogenním prostředí podnikové sítě, která pak působí značné překážky při konfiguraci systémových prostředků pro běh Single Sign-On řešení. Nevýhoda je to zjevně pro implementátora. To, o jak velký problém se jedná, se často pozná až během samotné implementace a tím se může projekt výrazně prodražit. Z toho důvodu je nutné věnovat dostatek času analýze prostředí zákazníka a hledat dopředu způsoby optimalizace procesu implementace.

Zákazník si může představovat, že nabízené řešení je řešením „krabicovým“. Často má k tomu důvody, protože prezentace marketingu představují až výsledek potenciálního procesu implementace. K iluzi, že se jedná o jednoduchou instalaci, může přispět i vzezření produktu. Může se jednat o složku, ve které je knihovna autentizačního filtru, návod na jeho instalaci a poněkud delší návod na přípravu prostředí zákazníka. Paradox pak spočívá v tom, že zatímco představa zákazníka počítá příkladem s několika hodinami instalace, ve skutečnosti se nad optimalizací sítě stráví až několik týdnů. Co je důvodem?

Základním důvodem instalačních problémů je to, že Kerberos využívá systémových prostředků a systémového nastavení ke své funkčnosti. K těmto nastavením je možné počítat konfigurace Active Directory serveru, optimalizace DNS a SPN záznamů, síťová nastavení jednotlivých uzlů a proxy, systémová konfigurace serverů služeb, systémová nastavení klientů a zejména pak konfigurace prohlížečů. Často se stane, že stejná konfigurace na podobných strojích vyvolává jiné výsledky a naopak. Hledání nefunkčnosti protokolu Kerberos je pak velmi náročné. Všechny doporučované postupy jsou totiž koncipovány na ideální prostředí. Ideálním prostředím je myšlen systém bezprostředně po první instalaci. Jakékoliv následující zásahy do systému mohou narušit čisté nastavení a tím vyvolat změnu, která se projeví odmítnutím uživatele při jeho autentizaci.

Pokud je prostředí zákazníka ve smyslu softwarové různorodosti nehomogenní (a to je případ většiny), může se stát, že například ověřování klienta na daném počítači zafunguje až poté, co se tento počítač jednoduše přeinstaluje. Proces hledání příčin nehomogenity a systémové různorodosti je tedy často velmi náročný.

Po instalaci všech potřebných komponent řešení Single Sign-On a konfiguraci systémového prostředí pro protokol Kerberos dojde v procesu implementace k prvním plošným testům funkčnosti řešení. V následující tabulce jsou uvedeny možné výsledky prvních pokusů autentizace jednotlivých uživatelů webové aplikace. Místo přihlášení k aplikaci bez přihlašovacího dialogu je popsán výsledek pokusu a možná příčina tak, jak ji autor měl možnost při implementacích poznat.

	výsledek	**příčina**
1.	Stránku nelze zobrazit	Problém komunikace mezi klientem a serverem
2.	Bílá stránka	Chyba na straně webového serveru (např. verze Apache Tomcat do 5.5.8) v důsledku velkého počtu skupin SIDs, kterých je uživatel členem. Problém zpracování velkého lístku služby
3.	Přihlašovací dialog	Verze prohlížeče, která nepodporuje protokol SPNEGO/Kerberos. V systému Windows vrací NTLM token
4.	Přihlašovací dialog	Chybná konfigurace prohlížeče (problém zejména u prohlížeče MS Internet Explorer), především nastavení Local Intranet
5.	Přihlašovací dialog	Chybná synchronizace času v doméně
6.	Přihlašovací dialog	Chybná instance prohlížeče MS Internet Explorer na daném stroji. Není možno odinstalovat a nainstalovat, pomůže jen přeinstalace systému
7.	Přihlašovací dialog	Chyby vzniklé chybnou manipulací s keytab soubory. Vytvoření, přenos, mapování na SPN, generování hesla uživatele služby, rozdílné šifrování
8.	Webová aplikace	OK

Mezi výše uvedenými výsledky zkoušek řešení Single Sign-On jsou tři příčiny, které zabraly většinu času na ladění a optimalizaci. Jedná se o problém s počtem skupin (SIDs), problém s nastavením prohlížeče pro Local Intranet a některé odchylky konkrétní instance MS Internet Explorer. V následujících odstavcích bude věnována těmto problémům pozornost.

SIDs (Security Identifier) – často velmi skrytý problém s řešením ve dvou rovinách. V sítích Active Directory je součástí Kerberos ticketu posílána informace o skupinách, ke kterým daný uživatel patří (PAC – Privilage Attribute Certificate). Skupiny jsou reprezentovány unikátní hodnotou proměnné délky tzv. SIDs identifikátory. Pokud daný uživatel náleží ke většímu množství skupin, zvětšuje se tím paralelně velikost autentizačních lístků. Kerberos lístky tedy mají konstantní velikost, která v důsledku limituje velikost PAC. V případě většího množství skupin velikost překročí stanovenou hranici pro přenos a autentizace může selhat. Microsoft tedy umožnil nastavit hodnotu maximální velikosti lístku registrem MaxTokenSize:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Kerberos\Parameters]
"MaxPacketSize"=dword:00004e20
"MaxTokenSize"=dword:00004e20
```

Defaultní hodnota ve Windows 2000 je 8000 bytů, ve Windows 2000 Service Pack 2 a MS Windows Server 2003 je to 12000 bytů. Většina implementací Kerbera, které nepatří do rodiny MS, ignorují PAC pole v obsahu lístků. Během přenosu lístku, který má enormní velikost, přes primárně určené UDP pakety se pak stane, že Windows KDC požádá klienta o komunikaci prostřednictvím TCP.

Druhou rovinou problému velikosti Kerberos lístků může být chyba na straně webových serverů. Příkladem je možné uvést server Apache Tomcat, který až do verze 5.5.8 nemá implementováno rozhraní, které by bylo schopné přijmout lístek větší velikosti. Problém byl od verze Apache Tomcat 6.0 odstraněn přidáním atributu maxPacket do konfiguračního souboru web.xml. Problém se projevuje tak, že response se k webovému serveru nedostane celý a uživatel pak vidí bílou stránku bez žádného obsahu.

Local Intranet – konfigurace atributu v nastavení prohlížeče je často mylným ukazatelem nefunkčnosti řešení. Filtr se může zachovat tak, že i přes zařazení webového serveru do výčtu stránek, které patří do okruhu Local Intranet, ukazuje prohlížeč neúčast serveru ve skupině Local Intranet a tím nedojde k přihlášení Single Sign-On. Vzhledem k tomu, že nastavení prohlížeče MS Internet Explorer je poněkud složitější jak konfigurace aplikace Firefox, je doporučeno vyzkoušet nastavení atributů prohlížeče nejdříve na jiném prohlížeči jak MS Internet Explorer – např. Firefox. Pokud ani Firefox nezafunguje, bude i přes matoucí upozornění na nepřítomnost v Local Intranet chyba pravděpodobně v jiné části systému.

Prohlížeč MS Internet Explorer – většina podniků používá internetový prohlížeč primárně od MS Windows. Vzhledem k tomu, že funkčnost akceptace a provedení autentizace Kerbera na straně prohlížeče úzce souvisí jak s operačním systémem, tak s daným prohlížečem, je možné, že problém nefunkčnosti Kerbera souvisí s určitým chtěným nebo nechtěným nastavením daného prohlížeče. Problém zpravidla vyřeší reinstalace prohlížeče. Tento postup se však nedá aplikovat na produkty MS Windows, kde Internet Explorer je součástí operačního systému. Tato nevýhoda produktu má často za následek reinstalaci celého operačního systému. Pokud na daném počítači administrátor požaduje funkční Kerberos Single Sign-On a pokud současně platí, že

- přihlášení prostřednictvím aplikace Firefox funguje bez problémů,
- přihlášení prostřednictvím aplikace MS Internet Explorer nefunguje pro žádného uživatele a
- nastavení odpovídá doporučovaným postupům,

pak je potřeba buď nainstalovat novější verzi prohlížeče Internet Explorer nebo přeinstalovat operační systém Windows. Systémoví administrátoři pak neslyší příliš rádi, že mají např. 5% firemních počítačů přeinstalovat, aby bylo Single Sign-On s produktem MS Internet Explorer funkční.

5.2 Doporučení pro implementaci

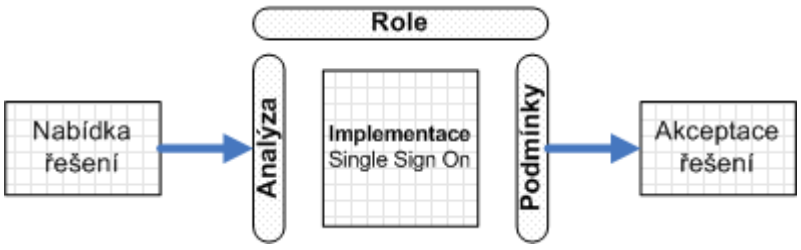
V několika bodech bude uvedeno několik doporučení, která se týkají implementace řešení Single Sign-On tak, aby se předešlo zbytečným nepříjemnostem v rámci projektu, která se pak mohou projevit ve finančních ztrátách dodavatele i zákazníka. Jedná se o zkušenosti a doporučení autora:

1. Dostatečná analýza prostředí zákazníka
2. Definice rolí na projektu
3. Definice podmínek akceptace

Analýza prostředí je základem úspěšné implementace. Vzhledem k systémovému charakteru řešení je nutné znát maximální počet aspektů prostředí zákazníka jak v rovině serverové tak v rovině aplikační a síťové. Součástí nabídky řešení by měl být výčet systémové a aplikační kompatibility řešení. Je potřeba počítat s tím, že navržené řešení nelze přímo implementovat na jakýkoliv systém se stejnými náklady. Analýza prostředí by měla na otázku obtížnosti implementace dát alespoň základní odpověď. Příloha č. 3 obsahuje vzor analýzy prostředí formou dotazníku pro zákazníka.

Role na projektu je potřeba definovat dostatečně dopředu tak, aby zákazník s požadavky souhlasil a počítal s nimi. Kromě koordinátora projektu na straně zákazníka je potřeba počítat s časem síťového specialisty domény zákazníka. Role síťového specialisty je potřebná především během procesu konfigurace Active Directory, proxy serverů a nastavení klientských stanic. Další rolí by měl být správce domény především kvůli zásahům do nastavení Active Directory. Kromě přístupových práv k doméně zná správce domény politiky, které jsou aplikovány na počítače v síti, a může tyto politiky upravovat.

Podmínky akceptace stanovují postačující podmínky pro akceptaci systému zákazníkem. Podmínky jsou důležité proto, že v případě Single Sign-On se nejedná o balíčkové řešení, které stačí nainstalovat a otestovat, ale o systémovou záležitost, která zpravidla vyžaduje určité úpravy. Optimalizace prostředí může být dlouhodobějším procesem a proto je potřeba stanovit body, kterých je třeba dosáhnout, aby bylo řešení považováno za předané a uvedené do provozu. V opačném případě se může jednat o „nekonečný příběh“, který nesedí ani jedné ze zúčastněných stran. Součástí nabídky by tedy měla být i dostatečná rezerva na podporu systému u zákazníka.



Obrázek 20: Základy procesu implementace

V případě implementace Single Sign-On platí, že výrazně jednodušší je implementace řešení pro danou webovou aplikaci za předpokladu, že jiná webová aplikace v rámci podnikové sítě již využívá řešení Single Sign-On na základě protokolu Kerberos. Při implementaci pak není potřeba konfigurovat klientské stanice, protože obecně platí, že pokud funguje ověřování prostřednictvím Kerbera na dané klientské stanici pro jeden webový server, pak bude fungovat i pro další za předpokladu, že jsou obě ve stejné doméně a využívají stejné KDC servery.

6 Použitelnost řešení

Single Sign-On je způsob, jak udělat zabezpečení domény včetně jejich aplikací bezpečnější a použití těchto aplikací pro uživatele jednodušší. Z toho důvodu můžeme určitě tvrdit, že samotná myšlenka Single Sign-On je dobrá a má v současném trendu v informačních technologiích své místo. Otázkou použitých technologií a finančních výdajů pak je, jakým způsobem se podaří cíle Single Sign-On dosáhnout. Kerberos je bezesporu technologie, která zaujímá jedno z prvních míst pomyslného žebříčku dostupných produktů, protokolů a aplikací pro řešení Single Sign-On. Velkou mírou přispívá k rozšířenosti Kerbera jeho dostupnost a spolehlivost. Dostupnost je umožněna prostřednictvím nejrůznějších implementací Kerbera, které jsou často zdarma, a velkou měrou přispívá k rozšíření Kerbera i MS Windows 2000/2003 server, v jehož doméně je Kerberos primárním autentizačním mechanismem. Spolehlivost řešení je zaručena poměrně dlouhou historií existence Kerbera a jeho používání. O kvalitě a jedinečnosti řešení Single Sign-On nad Kerberem svědčí i to, že po celou historii existence Kerbera se tento mechanismus od svého vytvoření měnil jen velmi málo a spíše okolní produkty se snažili přizpůsobit tomuto mechanismu autentizace (viz. přechod z primárního autentizačního mechanismu NTLM na Kerberos u MS Windows 2000/2003) .

Po prvním odstavci přichází otázka, proč není řešení Kerberos masivněji rozšířené? Proč je tak málo firem, které by se mohli „pochlubit“ využíváním výhod autentizace Kerberos? Jaká je použitelnost řešení v podnikovém komerčním prostředí?

Hlavním kladem i problémem řešení Kerberos zároveň je robustnost. Obecně lze říci, že robustní nástroj potřebuje rozsáhlejší instalaci a zkušenou podporu. U protokolu Kerberos je ještě možné dodat potřebu znalosti nastavení prostředí. Všechny tyto požadavky nejsou rozhodně špatné, nicméně jsou jistě náročné na čas a peníze. Z toho pak vyplývá otázka, kterou si jistě položí každá firma před zavedením, zda se vyplatí investovat čas a peníze do Single Sign-On řešení Kerberos. Aby bylo možné vyvrátit určité firmě tuto pochybnost, je dobré zaměřit se na jiný aspekt Kerbera, kterým je homogennost autentizace v doméně. Robustní řešení Kerbera totiž nabízí rozhraní jako standard, který je možné využít nejen pro jednu konkrétní webovou aplikaci, ale pro jakoukoliv aplikaci a službu, která se v doméně nachází. Tuto přidanou hodnotu znají firmy, které do řešení Kerberos investovali a nyní sjednocují autentizační mechanismy intranetových aplikací za účelem využití Single Sign-On Kerberos.

Výhodu pro implementaci protokolu Kerberos mají ta prostředí, která větší měrou obalují požadavky protokolu Kerberos. Pokud tedy bude firma využívat doménový server Domain Controller, aplikační server Windows 2003 Server a 100 pracovních stanic s operačním systémem Windows XP sp2, pak všechny tyto komponenty splňují s drobnými úpravami kompatibilitu s protokolem Kerberos. Implementace řešení Kerberos do takového prostředí bude pro zákazníka

levnější variantou. Co se týče ostatních distribucí Kerbera a prostředí klienta a serverů, pak platí to stejné v závislosti na obecné podpoře protokolu Kerberos.

Na otázku použitelnosti řešení tedy není jednoduchá odpověď. Teoreticky je možné udělat všechno, v praxi je ale situace daleko obtížnější. Situace se pak může jednoduše vyhrotit v případě, že na projektu není udělaná důsledná analýza prostředí, nejsou jasně stanoveny projektové role především ze strany systémových odborníků na prostředí zákazníka a především nejsou přesně definovány akceptační podmínky tak, jak popisuje kapitola ohledně doporučení pro implementaci. To, že teoreticky lze udělat všechno, je u implementace Kerberos Single Sign-On více než aktuální. Během implementace se totiž dá lehce dostat do situací, které mají podle teorie jasné řešení, ale v praxi je z nejrůznějších systémových důvodů nelze uplatnit a řešení začne fungovat až tehdy, když se problémové místo systému popř. celý systém reinstaluje. Tak by ale pochopitelně instalace nové komponenty do stávající struktury intranetu neměla vypadat. V těchto situacích může implementátora napadat myšlenka, že Kerberos je v praxi špatně použitelný a funguje pouze v ideálním prostředí. To je zároveň i možný důvod, proč firmy často sahají po jiných konceptech řešení Single Sign-On, které jsou často implementovány robustními autentizačními servery (Policy Server) a zaštítěny nadnárodními IT společnostmi. Jedná se o drahá řešení přesahující jednoduchou potřebu Single Sign-On přístupu k aplikacím. Na druhou stranu ale existuje stále více firem, které rády zaplatí za čas optimalizace řešení Kerberos, protože je k tomu vedou dva důvody. Prvním důvodem jsou peníze, protože zaplatit si implementaci Kerbera pro své intranetové aplikace není v porovnání s Enterprise autentizačními systémy takový obnos. Druhým důvodem je spoluúčast na Kerberos řešení a následné podpoře systému, která zahrnuje porozumění základním principům autentizace. Pro zákazníka pak není řešení pouze černou skříňkou. Tyto důvody pak nahrávají tomu, že má smysl se řešením Kerberos zabývat a jeho implementace je pro většinu intranetových sítí tím nejlepším řešením Single Sign-On.

7 Závěr

Single Sign-On je řešení, které zajímá v dnešní době nejen teoretiky informačních technologií, ale i širší podnikovou veřejnost a IT management. Možnost doladit aplikaci tak, aby byla bezpečnější a zároveň pro uživatele i administrátora jednodušší, je velice lákavá.

Výsledkem diplomové práce jsou jak zkušenosti teoretické, které se opírají o dvouleté získávání zkušeností na poli různých přístupů k řešení Single Sign-On v podnikové sféře, tak zkušenosti praktické s vývojem řešení Single Sign-On pro vícevrstvou architekturu. Hlavní výzvou pro vývoj takového řešení je aktuální poptávka ze strany velkých společností. A zde je možné položit otázku, zda není pro velkou společnost výhodnější koupit hotové rozšířené řešení od renomovaných společností certifikovaných na bezpečnost v informačních technologiích? Odpověď je rozdílná u různých společností. Často je ale slyšet názor, že je pro firmu výhodnější koupit řešení levnější, méně robustní, které je ale na druhou stranu možné upravit na míru systémům v daném podniku. A to je také hlavní důvod, proč se řešením Single Sign-On zabývat, a výzva ke stálému zlepšování daného řešení. Firmy mají často představu, že řešení Single Sign-On musejí být velmi složitá a nákladná. Hlavní přínos práce vidí autor v osvětě na základě svého vzdělávání tak, aby se firmy seznámily s možnostmi, které jim nabízí jejich intranetové sítě, a tak mohli jednoduše implementovat protokoly k řešení Single Sign-On.

Závěrem této práce o možnostech, které poskytuje řešení Single Sign-On, bych se rád pozastavil nad jednou skutečností, která mě při psaní zaujala. Jak bylo v textu napsáno, protokol Kerberos je protokolem velice starým. Napadá mě tedy otázka jak je možné, že tak starý protokol se stále těší veliké oblibě, jak je možné, že zájem o možnosti Kerberos Single Sign-On stále roste? Myslím, že je to dáno jednoduchou a přitom dokonalou koncepcí tohoto protokolu. A právě taková řešení mají v business sféře své místo i budoucnost.

Literatura

[1] Radkovič,P.: Kerberos a PAM [online]. [cit.10.5.2007]. CZ, Dostupné na URL: http://www.fi.muni.cz/~kas/p090/referaty/2005-jaro/ct/radkovic_KrbPAM.html

[2] Garman,J.: *Kerberos The Definitive Guide* USA, Sebastopol CA Oreilly, 2003, s.6-86. ISBN 0-596-00403-6

[3] Kerberos_(protokol) [online]. Poslední modifikace 12.prosince 2005. [cit.10.5.2007]. USA, Dostupné na URL: http://www.tvwiki.tv/wiki/Kerberos_(protocol)

[4] Kohl,J.: The Kerberos Network Authentication Service [online]. Poslední modifikace září 1993. [cit.11.5.2007]. USA, Dostupné na URL: http://www.ietf.org/rfc/rfc1510.txt

[5] Taylor,A.: *J2EE and Beyond*, Prentice Hall PTR., 2003, ISBN 0-13-141745-2

[6] Glolmus,P.: Web single sign-on řešení pro web [online]. Poslední modifikace 30.listopad 2005. [cit.11.5.2007]. CZ, Dostupné na URL: http://www.cesnet.cz/doc/techzpravy/2005/webiso/

[7] Ricciardi,F.:The Kerberos protokol [online]. Poslední modifikace 26.listopad 2006. [cit.11.5.2007]. IT, Dostupné na URL:http://www.zeroshell.net/eng/kerberos/Kerberos-definitions/#1.3.5

[8] Single Sign-on [online]. Poslední modifikace 27.prosinec 2007. [cit.31.12.2007]. EN, Dostupné na URL: http://en.wikipedia.org/wiki/Single_sign_on

[9] SSO [online]. Poslední modifikace 2005. [cit.11.5.2007]. EN, Dostupné na URL: http://wiki.java.net/bin/view/Javapedia/SSO

[10] Loshin,P.: Konec zapomenutých hesel: Single Sign-on [online]. Computerworld, rok vydání 2001, číslo vydání 44. [cit.12.5.2007]. CZ, Dostupné na URL: http://archiv.computerworld.cz/cwarchiv.nsf/clanky/7DB09143FB84FBB0C1256B480047A838?OpenDocument

[11] Glolmus,P.: Web single sign-on řešení pro web [online]. Poslední modifikace 30.listopad 2005. [cit.11.5.2007]. EN, Dostupné na URL: http://artax.karlin.mff.cuni.cz/~brain/diplomka.pdf

[12] RSA security [online]. Poslední modifikace 2007. [cit.31.12.2007]. EN, Dostupné na URL: http://www.tsoft.cz/index.php?q=node/312/print

[13] Kerberos Module for Apache [online]. [cit.30.12.2007]. CZ, Dostupné na URL: http://modauthkerb.sourceforge.net/

[14] Gal,A.: NTLM auth module for Apache [online]. Poslední modifikace 2000. [cit.29.12.2007]. EN, Dostupné na URL: http://modntlm.sourceforge.net/

[15] How to configure IIS to support Kerberos and NTLM protocol [online]. Poslední modifikace 19.ledna 2007. [cit.29.12.2007]. EN, Dostupné na URL: http://support.microsoft.com/kb/215383

[16] Java Security [online]. Poslední modifikace 2007. [cit.28.11.2007]. EN, Dostupné na URL: http://java.sun.com/javase/6/docs/technotes/guides/security

[17] Needham and Schroeder protokol [online]. Poslední modifikace 24.březen 2007. [cit.12.5.2007]. USA, Dostupné na URL: http://en.wikipedia.org/wiki/Needham-Schroeder

[18] Hunting,G.: Authentication World [online]. [cit.30.12.2007]. EN, Dostupné na URL: http://www.authenticationworld.com/Single-Sign-On-Authentication/

[19] Introduction to SSL [online]. Poslední modifikace 10.9 1998. [cit.5.1.2008]. EN, Dostupné na URL: http://docs.sun.com/source/816-6156-10/contents.htm

[20] How the Kerberos Version 5 Authentication Protocol Works [online]. Poslední modifikace 8.5.2008. [cit. 23.11.2008]. EN, Dostupné na URL: http://technet.microsoft.com/en-us/library/cc772815.aspx

[20] Heimdal [online]. Poslední modifikace 29.1.2008. [cit. 6.11.2008]. EN, Dostupné na URL: http://www.h5l.org/manual/heimdal-1-2-branch/info/heimdal.html#Introduction

[21] The Simple and Protected GSS-API Negotiation Mechanism [online]. Poslední modifikace 10.2005. [cit. 7.10.2008]. EN, Dostupné na URL: http://tools.ietf.org/html/rfc4178

[22] Introduction to JAAS and Java GSS-API Tutorials [online]. [cit. 5.11.2008]. EN, Dostupné na URL: http://java.sun.com/j2se/1.4.2/docs/guide/security/jgss/tutorials/index.html

[23] Use of JavaTM GSS-API for Secure Message Exchanges Without JAAS Programming [online]. [cit. 8.9.2008]. EN, Dostupné na URL: http://java.sun.com/j2se/1.5.0/docs/guide/security/jgss/tutorials/BasicClientServer.html

[24] Výkladový slovník pojmů [online]. Poslední modifikace 23.11. 2008. [cit. 23.11.2008]. EN, Dostupné na URL: http://www.microsoft.cz/Glossary/AZ_list.asp

[25] The Java CIFS Client Library [online]. Poslední modifikace 26.10. 2008. [cit. 20.11.2008]. EN, Dostupné na URL: http://jcifs.samba.org/

[26] Kerberos protocol tutorial [online]. Poslední modifikace 27.11.2007. [cit. 15.1.2009]. EN, Dostupné na URL: http://www.kerberos.org/software/tutorial.html

[27] Windows 2000 Authentication [online]. Poslední modifikace 3.2001. [cit. 16.1.2009]. EN, Dostupné na URL: http://www.windowsitlibrary.com/Content/617/06/7.html

[28] Kerberos & SPNEGO Configuration Issue [online]. [cit. 20.11.2008]. EN, Dostupné na URL: https://support.bea.com/application_content/product_portlets/support_patterns/wls/KerberosSPNEGOConfigurationIssues.html

[29] SPNEGO SSO documentation [online]. Poslední modifikace 25.11.2008. [cit. 26.11.2008]. EN, Dostupné na URL: http://appliedcrypto.com/spnegodoc.do

Seznam příloh

Příloha 1. Výstup modulu Krb5LoginModul

Příloha obsahuje část výstup (log) modulu Krb5LoginModul v rámci implementace Single Sign-On na J2EE aplikačním serveru. O službu HTTP/D65.kerberos.local@KERBEROS.LOCAL na server D65 uživatel žádá prostřednictvím webového klienta.

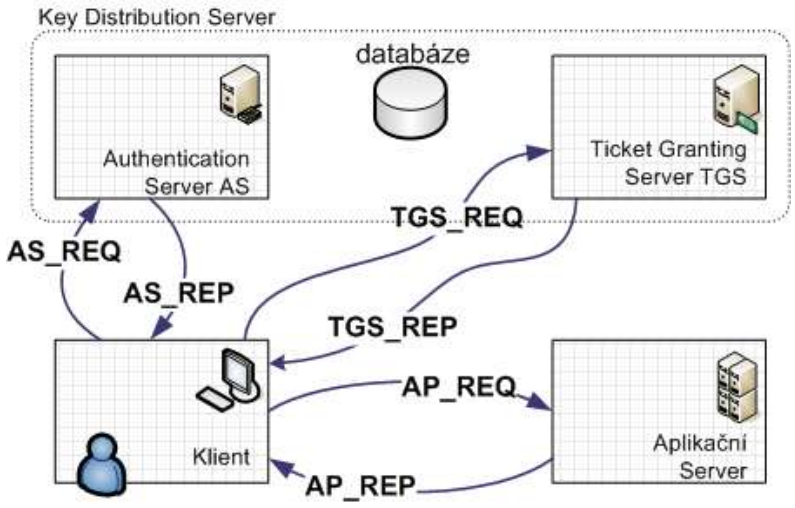
Debug is true storeKey true useTicketCache false useKeyTab true doNotPrompt true ticketCache is null isInitiator true
KeyTab is **c:\krb\krb5as.keytab** refreshKrb5Config is false principal is
HTTP/D65.kerberos.local@KERBEROS.LOCAL ...
>>> KeyTabInputStream, readName(): KERBEROS.LOCAL, HTTP, D65.kerberos.local
>>> KeyTab: load() entry length: 65; type: 3
Added key: 3version: 4
Ordering keys wrt default_tkt_enctypes list
Config name: c:\winnt\krb5.ini
Using builtin default etypes for default_tkt_enctypes
default etypes for default_tkt_enctypes: 3 1 23 16 17.
principal's key obtained from the keytab
Acquire TGT using AS Exchange
Using builtin default etypes for default_tkt_enctypes
>>> KrbAsReq calling createMessage
>>> KrbKdcReq send: kdc=192.168.0.1 UDP:88, timeout=30000, number of retries =3, #bytes=162
>>> KDCCommunication: kdc=192.168.0.1 UDP:88, timeout=30000,Attempt =1, #bytes=162
>>> KrbKdcReq send: #bytes read=240
>>> KDCRep: init() encoding tag is 126 req type is 11
>>>KRBError:
sTime is Sat Jan 03 20:35:43 CET 2009 1231011343000 error Message is Additional pre-authentication required
realm is KERBEROS.LOCAL sname is krbtgt/KERBEROS.LOCAL
>>>**Pre-Authentication Data**:
PA-DATA type = 2 **PA-ENC-TIMESTAMP**
AcquireTGT: PREAUTH FAILED/REQUIRED, re-send AS-REQ
Updated salt from pre-auth = KERBEROS.LOCALHTTPD65.kerberos.local
>>>KrbAsReq salt is KERBEROS.LOCALHTTPD65.kerberos.local
Pre-Authenticaton: find key for etype = 3
AS-REQ: Add PA_ENC_TIMESTAMP now
>>> EType: sun.security.krb5.internal.crypto.DesCbcMd5EType
>>> KrbAsReq calling createMessage
>>> KrbKdcReq send: kdc=192.168.0.1 UDP:88, timeout=30000, number of retries =3, #bytes=246
>>> KDCCommunication: kdc=192.168.0.1 UDP:88, timeout=30000,Attempt =1, #bytes=246
>>> KrbKdcReq send: #bytes read=1306
>>> EType: sun.security.krb5.internal.crypto.DesCbcMd5EType
>>> KrbAsRep cons in KrbAsReq.getReply HTTP/D65.kerberos.local
principal is HTTP/D65.kerberos.local@KERBEROS.LOCAL
EncryptionKey: keyType=3 keyBytes (hex dump)=0000: D9 AB 3E E3 19 AE E3 46
Added server's keyKerberos Principal HTTP/D65.kerberos.local@KERBEROS.LOCALKey Version 4key EncryptionKey: keyType=3 keyBytes (hex dump)=
0000: D9 AB 3E E3 19 AE E3 46

[Krb5LoginModule] added Krb5Principal HTTP/D65.kerberos.local@KERBEROS.LOCAL to Subject
Commit Succeeded

Found key for HTTP/D65.kerberos.local@KERBEROS.LOCAL(3)
Entered Krb5Context.acceptSecContext with state=STATE_NEW
>>> EType: sun.security.krb5.internal.crypto.**DesCbcMd5EType**
Using builtin default etypes for permitted_enctypes
>>> EType: sun.security.krb5.internal.crypto.DesCbcMd5EType
>>> Config reset default kdc KERBEROS.LOCAL
replay cache for testx@KERBEROS.LOCAL is null.
>>> **KrbApReq: authenticate succeed**.
>>>Delegated Creds have pname=testx@KERBEROS.LOCAL

Příloha 2. Kerberos komunikace

Kerberos komunikace se obecně skládá z šesti různých typů požadavků a odpovědí. Na obrázku č.21 je znázorněn tok Kerberos komunikace mezi klientem, aplikačním serverem a KDC serverem.



Obrázek 21: Schéma Kerberos komunikace

Pro detailnější představu o obsahu kerberos zprávy je prostřednictvím sniffovacího nástroje Ethereal zobrazen obsah zprávy TGS_REQ:

```
⊞ Internet Protocol, Src: 192.168.0.7 (192.168.0.7), Dst: 192.168.0.1 (192.168.0.1)
⊞ Transmission Control Protocol, Src Port: 1169 (1169), Dst Port: kerberos (88), Seq: 1261
⊞ [Reassembled TCP Segments (2043 bytes): #42(1260), #43(783)]
⊟ Kerberos TGS-REQ
  ⊞ Record Mark: 2039 bytes
    Pvno: 5
    MSG Type: TGS-REQ (12)
  ⊟ padata: PA-TGS-REQ
    ⊟ Type: PA-TGS-REQ (1)
      ⊟ Value: 6E82074E3082074AA003020105A10302010EA20703050000... AP-REQ
          Pvno: 5
          MSG Type: AP-REQ (14)
          Padding: 0
        ⊞ APOptions: 00000000
        ⊟ Ticket
            Tkt-vno: 5
            Realm: KERBEROS.LOCAL
          ⊞ Server Name (Service and Instance): krbtgt/KERBEROS.LOCAL
          ⊟ enc-part rc4-hmac
              Encryption type: rc4-hmac (23)
              Kvno: 2
              enc-part: 7774ABB423C67B5E9E695B5F1C4A118FB8C9C6FE9D20CD69...
        ⊟ Authenticator des-cbc-md5
            Encryption type: des-cbc-md5 (3)
            Authenticator data: 730C71DE2F5B364C8232D9BC97096C6DFA038D87D416463D...
  ⊟ KDC_REQ_BODY
      Padding: 0
    ⊞ KDCOptions: 60810010 (Forwardable, Forwarded, Renewable, Canonicalize, Renewable OK)
      Realm: KERBEROS.LOCAL
    ⊞ Server Name (Service and Instance): krbtgt/KERBEROS.LOCAL
      till: 2037-09-13 02:48:05 (Z)
      Nonce: 1296965759
    ⊞ Encryption Types: des-cbc-md5 rc4-hmac-old rc4-md4 rc4-hmac des-cbc-crc rc4-hmac-exp
                        rc4-hmac-old-exp
```

Příloha 3. Analýza prostředí – dotazník

Dotazník obsahuje specifikaci systému zákazníkem, na jejímž základě se provádí odhad času a nákladů na implementaci Single Sign-On. Cílem dotazníku je analýza resp. zmapování systémového prostředí s cílem sjednotit resp. aktualizovat aktuální prostředí tak, aby splňovalo požadavky na implementaci Single-Sign On (dále jen SSO).

Dotazník je ve zkrácené verzi a je rozdělen na 4 okruhy podle zaměření. Předlohou je vícevrstvá architektura CMS systému.

1. Doména systému

	Téma	Specifický dotaz	odpověď
1.1	Existence Active Directory a DNS	Obsahuje síťové prostředí servery DNS (Domain Name Server) a AD (Active Directory)?	
1.2	Využití AD a DNS	V jaké míře a za jakými účely se využívají služby Active Directory a DNS serverů?	
1.3	Doména prostředí	Jakou strukturu má doména, jejíž je CMS součástí? Specifikujte strukturu domény, zařazení CMS v rámci této domény, výjimky v zařazení pracovních stanic do Active Directory (tam, kde je využíván webový klient)	
1.4	Uživatelé domény	Jakým způsobem probíhá přihlášení uživatele k pracovní stanici? Specifikujte proces přihlášení a využití Active Directory v rámci procesu.	
1.5	Uživatelé CMS aplikace	Jakým způsobem probíhá aktuálně přihlášení uživatelů k CMS v rámci webového klienta? Specifikujte autentizační mechanismus a nastavení mechanismu u uživatelů v rámci CMS a existenci výjimek mezi uživateli	
1.6	Účty uživatelů aplikace CMS	V rámci Single Sign-On implementace, která je založena na protokolu Kerberos, je nutnou podmínkou totožnost přihlašovacích údajů (credentials) k pracovní stanici Windows (tedy přihlášení do domény) a přihlašovacích údajů do CMS. A to bez výjimky pro ty, kteří budou chtít k CMS přistupovat pomocí webového klienta. Je tato podmínka splněna?	
1.7	Service Principal Name	Jaký tvar bude mít název služby (např. služba HTTP) v rámci domény Active Directory? Příklad `HTTP/as.kerberos.local@KERBEROS.LOCAL` tak, že HTTP je název služby, documentums je název stroje, na kterém služba běží, KERBEROS.LOCAL je název domény	
1.8	Support Tools	V rámci instalace bude třeba použít nástroje Support Tools pro vytvoření SPN (Service Principal Name). Obsahuje MS Windows Server, na kterém běží Active Directory instalovaný balík Support Tools?	
1.9	Uživatel služby	Pro instalaci SSO je třeba vytvořit uživatele vázaného na SPN v rámci Active Directory. Je možné uživatele v rámci domény vytvořit?	

2. Pracovní stanice

	Téma	Specifický dotaz	odpověď
2.1	Operační systém pracovních stanic	Jaký operační systém je využíván na pracovních stanicích, ze kterých se bude přistupovat k CMS přes webového klienta?	
2.2	Webový prohlížeč	Jaké webové prohlížeče jsou používány k web přístupu k CMS? Specifikujte výčet všech prohlížečů a jejich verzí používaných na klientských pracovních stanicích.	
2.3	Funkčnost mechanismů Active Directory	Řešení SSO předpokládá, že při přihlášení uživatele do domény obdrží klientský počítač (Windows) do LSA cache autentizační ticket. Jedná se o standardní funkčnost Windows domény Active Directory. Je tato funkčnost aktivní? (ověřit lze pomocí kerbtray utility)	

3. Content Server

	Téma	Specifický dotaz	odpověď
3.1	CMS	Detailní přehled instalace a verze Content Serveru CMS systému	
3.2	Hostname	Je instalace serveru Aplikačního a Content Serveru provedena fyzicky na jednom hostu (stejný hostname) nebo se jedná o instalace na dvou různých strojích?	
3.3	Operační systém	Jaký operační systém je instalován na hostu, kde běží Content Server?	
3.4	Vytvoření souboru krb.ini	V rámci instalace je třeba v adresáři C:\winnt\ vytvořit soubor krb.ini. Bude možné soubor v adresáři vytvořit?	
3.5	Upravení systémové proměnné Path	Je možné v rámci instalace upravit systémovou proměnnou Path na Content Serveru doplněním dvou cest?	

4. Aplikační server

	Téma	Specifický dotaz	odpověď
4.1	Aplikační Server	Detailní přehled instalace a verze aplikačního serveru	
4.2	Verze Javy	Jakou verzi Javy využívá Tomcat na aplikačním serveru?	
4.3	Verze Tomcat	Jaká verze Tomcat je instalována na aplikačním serveru?	
4.4	Operační systém	Jaký operační systém je instalován na stroji, kde běží aplikační server? (pokud aplikační server běží na jiném hostu)	
4.5	Instalace	V rámci instalace bude potřeba administrátorská práva pro následující úkony: - Nastavení a konfigurace Active Directory (vytvoření SPN, uživatele, generování keytab) - Vytváření souborů na Aplikačním Serveru a Content Serveru - Úprava webového klienta v rámci CMS - Úprava autentizačního mechanismu na Content Serveru	

Příloha 4. Klist a kerbtray utility

Utilita klist – je nástroj (součást Windows Resource Kit Tool), který umožňuje vidět položky v lokální credentials cache. Utilita umožňuje vypsat držené lístky na daném stroji.

klist -tgt

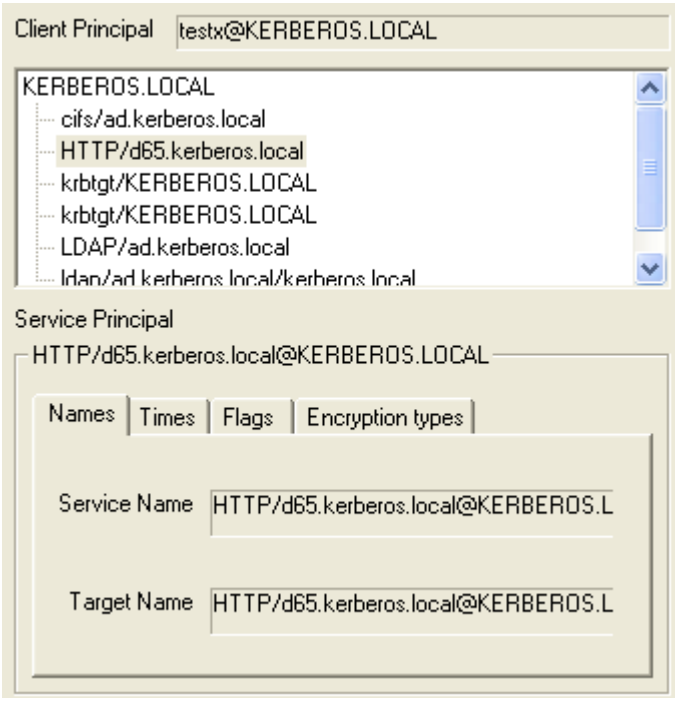
```
Cached TGT:

ServiceName: krbtgt
TargetName: krbtgt
FullServiceName: testx
DomainName: KERBEROS.LOCAL
TargetDomainName: KERBEROS.LOCAL
AltTargetDomainName: KERBEROS.LOCAL
TicketFlags: 0x40e00000
KeyExpirationTime: 1/1/1601 1:00:00
StartTime: 1/3/2009 19:44:47
EndTime: 1/4/2009 5:44:47
RenewUntil: 1/10/2009 19:44:47
TimeSkew: 1/1/1601 1:06:26
```

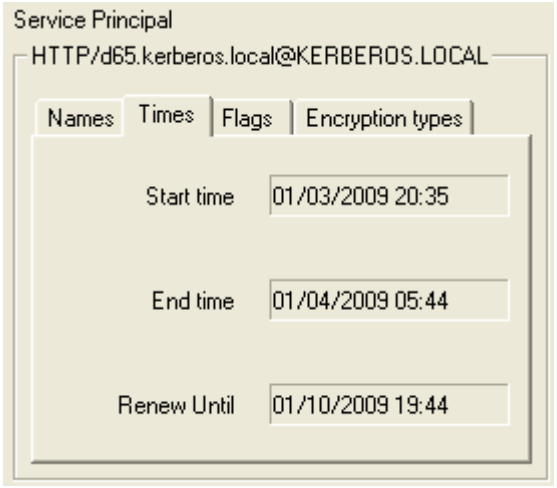
klist -tickets

```
Cached Tickets: (7)

   Server: krbtgt/KERBEROS.LOCAL@KERBEROS.LOCAL
      KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
      End Time: 1/4/2009 5:44:47
      Renew Time: 1/10/2009 19:44:47

   Server: HTTP/d65.kerberos.local@KERBEROS.LOCAL
      KerbTicket Encryption Type: Kerberos DES-CBC-MD5
      End Time: 1/4/2009 5:44:47
      Renew Time: 1/10/2009 19:44:47

   Server: LDAP/ad.kerberos.local@KERBEROS.LOCAL
      KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
      End Time: 1/4/2009 5:44:47
      Renew Time: 1/10/2009 19:44:47

   Server: host/dctm_ws_xp.kerberos.local@KERBEROS.LOCAL
      KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
      End Time: 12/22/2008 23:11:05
      Renew Time: 12/29/2008 13:11:05
```

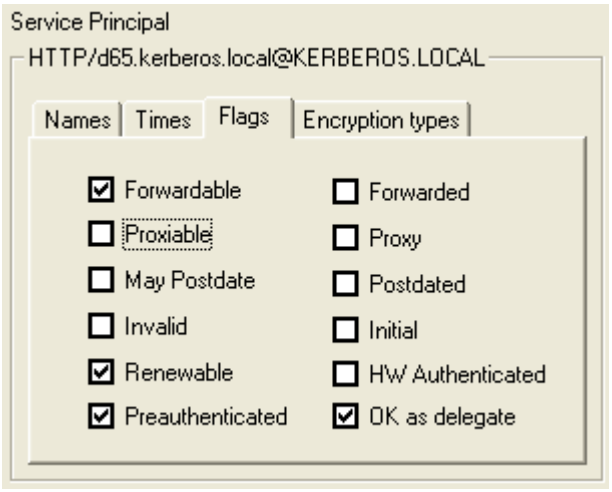
U výpisu všech Kerberos lístků je zvýrazněn lístek služby HTTP, který se vytvoří po první autentizaci vůči HTTP službě na stroji klienta.

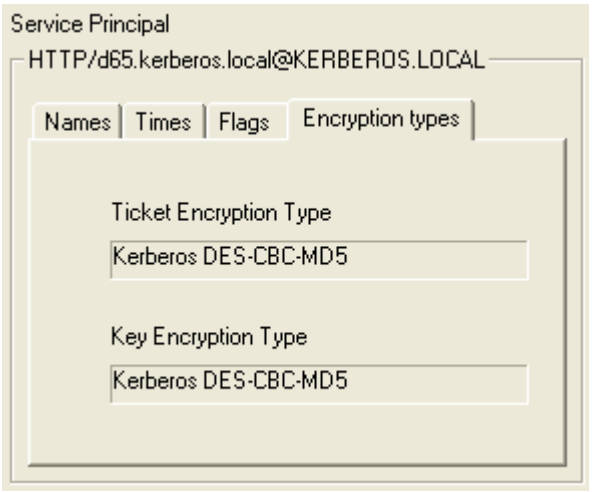
Utilita kerbtray – je nástroj pro grafickou reprezentací lístků (součást Windows Resource Kit Tool)





Obsah grafického nástroje kerbtray je podobný, jako u nástroje klist. Na záložce Times je počáteční a koncový čas platnosti lístku a záložka Flags obsahuje základní nastavení, která se k danému lístku vážou. Jako příklad lze uvést zatržení volby Preauthenticated pro ověření časové známky (timestamp).





Příloha 5. Konfigurace serveru

Konfigurace základních systémových komponent aplikačního serveru web.xml a app.xml.

web.xml

```
<web-app>
  <filter>
     <filter-name>auth</filter-name>
     <filter-class>com.krb5.http.AuthenticationFilter</filter-class>
     <init-param>
        <param-name>java.security.auth.login.config</param-name>
        <param-value>C:\krb\jaas.config</param-value>
     </init-param>
     <init-param>
        <param-name>sun.security.krb5.debug</param-name>
        <param-value>true</param-value>
     </init-param>
     <init-param>
          <param-name>com.krb5.applicationPrincipal</param-name>
        <param-value>HTTP/as.kerberos.local@KERBEROS.LOCAL</param-value>
     </init-param>
     <init-param>
          <param-name>com.krb5.servicePrincipal</param-name>
        <param-value>CSERVER/cs.kerberos.local@KERBEROS.LOCAL</param-value>
     </init-param>
     <init-param>
        <param-name>com.krb5.loggingLevel</param-name>
        <param-value>5</param-value>
     </init-param>
     <init-param>
        <param-name>com.krb5.csvLog</param-name>
        <param-value>C:\krb\csvLogger.csv</param-value>
     </init-param>
  </filter>
  <filter-mapping>
     <filter-name>auth</filter-name>
     <url-pattern>/component/main/*</url-pattern>
  </filter-mapping>
</web-app>
```

app.xml

```
<config><scope>
    <application extends="webappl/app.xml"><authentication>
        <!-- Default doména a databáze -->
        <domain>domainName</domain>
        <docbase>docbaseName</docbase>

        <!-- Single Sign-On konfigurace -->
        <sso_config>
          <plugin>Kerbauth</plugin>
          <ticket>SMSESSION</ticket>
          <header>SMUSERHEADER</header>
        </sso_config>
    </authentication></application>
</scope></config>
```

Příloha 6. CD

Součástí přílohy elektronického nosiče CD je struktura výsledného řešení Single Sign-On[7], obecný popis instalace řešení a další materiál, jehož výčet je součástí souboru README.txt na CD.

Příloha 7. Prohlášení zadavatele

Řešení Single Sign-On postavené na základě protokolu Kerberos splnilo naše očekávání a je už nyní použito u zákazníka.

Ing. JAN KUCHAŘ
Delivery Director
CSC

[7] Programové řešení Single Sign-On je zastoupeno strukturou výsledného řešení. Zdrojový kód je chráněn autorským právem firmy CSC.