

PKCS standardy pro potřeby zdravotnické dokumentace

Josef Špidlen, Miroslav Nagy

*EuroMISE centrum,
Ústav Informatiky AV ČR, Praha, Česká republika*

Abstrakt:

V rámci výzkumu oddělení medicínské informatiky ÚI AV ČR se zabýváme reprezentací medicínských informací a vývojem tzv. elektronického zdravotního záznamu. Ve snaze uvést myšlenky a vyvinuté přístupy do praxe ve zdravotnictví byla navázána spolupráce s několika výrobci nemocničních informačních systémů. Společně s nimi pracujeme na projektu „Informační technologie pro rozvoj kontinuální sdílené péče o zdraví“, ve kterém je třeba řešit zabezpečení citlivých medicínských dat a další bezpečnostní otázky v souladu s platnou legislativou. Dílčími úlohami jsou spolehlivá autentifikace každého účastníka řetězce poskytování zdravotní péče, bezpečné podepsání elektronické zdravotní dokumentace zaručeným elektronickým podpisem a zpětné jednoznačné ověření původního autora dokumentace. Pro řešení těchto úloh je vhodné využít externí kryptografické zařízení. Výhoda kryptografického zařízení je např. v tom, že soukromá část asymetrického klíče je vygenerována přímo zařízením a nikdy jej neopustí, což napomáhá bezpečné identifikaci autora elektronického podpisu. Abychom předešli úzké vazbě na konkrétní kryptografické zařízení, je vhodné využít existujících standardů pro komunikaci s těmito zařízeními. Představiteli těchto standardů jsou například rodina standardů PKCS či MS CAPI. Text článku se převážně zabývá právě úvodem do kryptografického standardu PKCS #11 a možnostmi jeho využití pro účely projektu.

Klíčová slova: kryptografické standardy, zdravotnická dokumentace, PKCS

Úvod

V rámci aplikovaného výzkumu oddělení medicínské informatiky Ústavu Informatiky AV ČR se skupina informatiků v úzké spolupráci s lékaři zabývá reprezentací medicínských informací a vývojem univerzálního elektronického zdravotního záznamu (EHR). Převážně s ohledem na potřeby lékařské komunity a na základě standardů CEN TC251 zde vznikla pilotní třívrstvá aplikace elektronického zdravotního záznamu „MULTimediální DistRibuovaný zdravotní záznam“ (MUDR) [1,2,3,4]. Ve snaze uvést myšlenky a nové přístupy EHR MUDR do reálné praxe ve zdravotnictví v naší republice byla navázána spolupráce s několika výrobci nemocničních a klinických informačních systémů. Tato spolupráce vyústila ve společný projekt oddělení

medicínské informatiky ÚI AV ČR se dvěma MSP: Medical software, s.r.o. a EuroMISE, s.r.o.

Společný projekt „Informační technologie pro rozvoj kontinuální sdílené péče o zdraví“ programu „Informační společnost“ tématického programu II Národního programu výzkumu je zaměřen na návrh a evaluaci vhodné infrastruktury pro účely vedení elektronické zdravotní dokumentace v prostředí českého zdravotnictví a implementaci systému pro kontinuální sdílenou péči o zdraví občanů mezi nemocnicemi a praktickými lékaři. Vzhledem k citlivosti medicínských dat je jedním z hlavních cílů projektu navrhnout infrastrukturu patřičně robustně a bezpečně v souladu s platnou legislativou České republiky, resp. Evropské Unie. Z hlediska české legislativy je pro nás závazný hlavně zákon o ochraně osobních údajů číslo 101/2000 Sb., který byl několikrát novelizován a jeho další novelizace je připravována k 1.1.2005. Z dalších platných předpisů je třeba zohlednit zákon číslo 227/2000 Sb. o elektronickém podpisu a zákon č. 260/2001 Sb., který nemocnicím i lékařům dává zákonný podklad pro vedení zdravotní dokumentace v elektronické podobě, aniž by museli pořizovat její listinné kopie. Přes faktickou nemožnost uplatnit okamžitě výše uvedené předpisy k vedení čistě elektronické zdravotnické dokumentace (díky chybějícím prováděcím předpisům a dalším nedostatkům v zákonném rámci) nám toto dává představu, jakým způsobem bude třeba zabezpečit citlivé informace ve zdravotnické dokumentaci. Již nyní je jasné, že bude třeba řešit následující problematické oblasti:

- autentifikaci a autorizaci aktérů poskytovatele zdravotní péče,
- řízení přístupu,
- identifikaci příjemce zdravotní péče (pacienta),
- ověřování a potvrzování původu informací obsažených ve zdravotnické dokumentaci.

Musíme tedy například umět ověřit, že aktér je tím, za kterého se vydává, dále pak zda jednou identifikovaný aktér má právo určitého přístupu k určitému zdroji (informaci). Musíme dokázat bezpečně identifikovat a ověřit autora elektronicky podepsané části zdravotnické dokumentace včetně ověření časového razítka udávajícího, kdy byl příslušný záznam pořízen. Musíme umět zabezpečit, aby komunikaci mezi distribuovanými moduly zdravotního záznamu obsahující citlivá data nebylo možné odposlechnout neoprávněným subjektem, apod.

Součástí projektu by ale neměla být tvorba zcela nových technologií a metod, nýbrž zmapování a použití (případně mírné upravení) existujících kryptografických standardů v uvedených oblastech. Různé části výše uvedené bezpečnostní problematiky jsou řešeny převážně ve dvou množinách standardů: „Public-Key Cryptography Standards“ (PKCS) a „Microsoft Crypto API“ (MS CAPI).

Standard PKCS

Rodina standardů PKCS je vyvíjena RSA Laboratořemi [5] ve spolupráci s neformálním konsorciem původně založeným společnostmi Apple, Microsoft, DEC, Lotus, Sun a MIT. V dnešní době na vývoji PKCS standardů spolupracují zainteresovaní reprezentanti průmyslu, americké akademické veřejnosti i americké vlády.

Standardy PKCS uvažují implementace jak definovanými kryptografickými metodami tak nezávisle na konkrétní zvolené metodě. Podporováno je mnoho kryptografických metod, ale detailně specifikovány jsou pouze dvě: RSA [6] a Diffie-Hellman [7]. Rodina standardů PKCS sestává aktuálně z následujících částí:

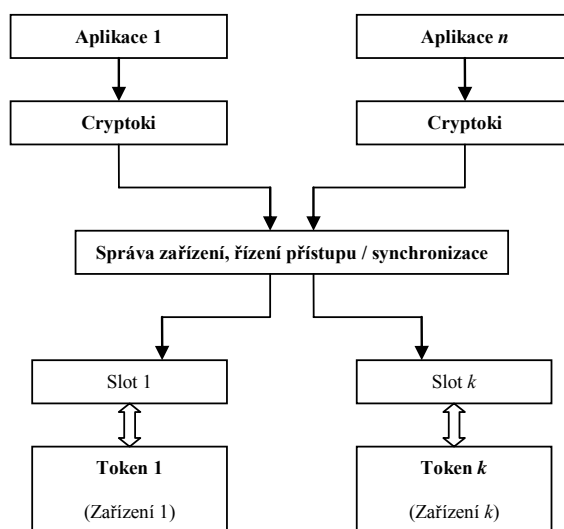
- PKCS #1 Definice mechanismu šifrování a podepisování dat pomocí kryptografického systému založenému na RSA veřejných klíčích.
- PKCS #3 Defínuje protokol pro odsouhlasení Diffie-Hellmanových klíčů mezi komunikujícími entitami.
- PKCS #5 Popisuje metodu zakódování řetězce tajným klíčem odvozeným od hesla.
- PKCS #6 Přestává být používán, je nahrazován standardem X.509 verze 3.
- PKCS #7 Defínuje obecnou syntaxi zprávy obsahující kryptografická rozšíření jako například digitální podpisy či kódování.
- PKCS #8 Defínuje formát soukromého klíče k určitému veřejnému klíči a asymetrickému algoritmu (volitelně s množinou dalších parametrů a atributů).
- PKCS #9 Defínuje typy vybraných atributů pro použití v dalších PKCS standardech.
- PKCS #10 Defínuje syntaxi požadavku o certifikaci.
- PKCS #11 Defínuje programové rozhraní nezávislé na technologii nazývané Cryptoki určené pro kryptografická zařízení jakými jsou například čipové procesorové karty.
- PKCS #12 Specifikuje přenosný formát pro ukládání a transport privátních klíčů a certifikátů uživatele a dalších tajných informací.
- PKCS #13 Je zamýšlen k definici mechanismu kódování a podepisování dat s využitím kryptografie eliptické křivky.
- PKCS #14 Je stále ve vývoji, zabývá se standardizací generování pseudonáhodných čísel.
- PKCS #15 Je doplňkem PKCS #11 udávajícím standardní formát kryptografických certifikátů uložených na kryptografických tokenech.

Vzhledem k možnému využití pro potřeby projektu je nejdůležitější standard PKCS #11 [8]. Dále využitelné jsou například PKCS #12 [9] či PKCS #15 [10].

Standard PKCS #11

Účelem PKCS #11 standardu je specifikovat aplikační rozhraní pro přístup ke kryptografickým informacím a funkcím nezávisle na fyzickém zařízení, které tyto informace a funkce poskytuje. Toto rozhraní nazývané Cryptoki je založené na jednoduchém objektovém přístupu s důrazem na sdílení zdrojů tak, aby více aplikací mohlo přistupovat k více kryptografickým zařízením. Cryptoki poskytuje jednotný logický pohled na každé kryptografické zařízení. Aplikace, které chtějí kryptografické služby využít, pracují pouze s tímto logickým modelem – tzv. „kryptografickým tokenem“. Standard tedy specifikuje API rozhraní ve tvaru datových typů a funkcí, které může využít každá aplikace naprogramovaná v ANSI C jazyce. Tyto funkce a datové typy jsou zpravidla zpřístupněny pomocí standardních ANSI C hlavičkových souborů.

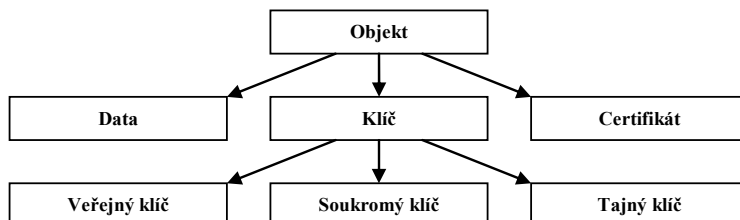
Zařízení jako čipové procesorové karty, USB tokeny či PCMCIA karty jsou ideálními zařízeními pro implementaci asymetrické kryptografie. Tato zařízení umožňují bezpečné uložení soukromé části klíče tak, aby tato nikdy neopustila zařízení a byla tím stoprocentně ve vlastnictví daného uživatele. S takovýmto zařízením není vhodné, aby uživatelské aplikace sami implementovali algoritmy asymetrické kryptografie. Aplikace tedy raději požádá zařízení, například pomocí Cryptoki, o poskytnutí kryptografické služby. Obecný model, jak aplikace přistupují k zařízením poskytujícím kryptografické služby je zobrazen na obr. 1.



Obr. 1. Obecný Cryptoki model.

Objekty Cryptoki

Token představuje libovolné zařízení umožňující ukládání objektů a poskytující kryptografické funkce. Cryptoki definuje tři typy objektů: uživatelská data, klíče a certifikáty. Klíče existují tří druhů: veřejné, soukromé a tajné. Tato hierarchie je zobrazena obr. 2.



Obr. 2. Hierarchie objektů.

Objekty lze dále klasifikovat dle jejich životnosti a viditelnosti. Objekty tokenu (Token Objects) jsou viditelné všem aplikacím připojeným k tokenu, pakliže mají zároveň dostatečná oprávnění. Takovéto objekty zůstávají na tokenu uchovány i po ukončení aktivního spojení (session) s tokenem (vyjmutí zařízení, ukončení příslušné aplikace apod.). Objekty aktivního spojení (Session Objects) jsou viditelné pouze v rámci aplikace, která toto spojení vytvořila. Má-li aplikace více aktivních spojení s jedním tokenem, jsou tyto objekty viditelné přes všechna tato spojení. Objekt aktivního spojení zaniká okamžitě po ukončení aktivního spojení (uzavření session), ve které vznikl.

Další klasifikaci lze provést podle přístupnosti objektu. Objekty rozlišujeme veřejné, které jsou přístupné aplikacím bez nutnosti dalšího přihlášení, a objekty soukromé, kde je nutné tzv. přihlášení uživatele k tokenu. Toto přihlášení je obvykle prováděno tzv. PINem, ale může být provedeno i jinou metodou závislou na tokenu / druhu zařízení (tj. například biometricky).

Každý token může vytvářet, hledat a rušit objekty či s nimi různě manipulovat a provádět kryptografické funkce. Ovšem ne každé kryptografické zařízení musí nutně umět všechny kryptografické funkce. Jednodušší zařízení mohou mít třeba jen malé pevné místo na několik klíčů a implementovat pár algoritmů, jimiž provádějí omezenou množinu operací. Ne všechna zařízení musí nutně podporovat všechny typy objektů. Do budoucna je však předpokládána definice několika tzv. profilů, kde bude řečeno, co je třeba minimálně implementovat a podporovat, aby zařízení daný profil splňovalo.

Uživatelé Cryptoki

Cryptoki rozlišuje 2 typy uživatelů: administrátora (Security Office, SO) a normálního uživatele. Jen normálnímu uživateli je dovolen přístup k soukromým objektům tokenu. Tento přístup je poskytnut pouze po autentifikaci uživatele (některé tokeny mohou autentifikaci vyžadovat před každou kryptografickou operací neohledně na to, zda se dotýká soukromých objektů). Role administrátora je hlavně v nastavení PINu (nebo jiného způsobu autentifikace) normálního uživatele. Bez tohoto prvotního nastavení není možné normálního uživatele autentifikovat. Administrátorovi je rovněž umožněna manipulace s veřejnými objekty tokenu.

Z pohledu Cryptoki je PIN pouze řetězec libovolné délky. V některých případech je PIN vkládán jinak než prostřednictvím aplikace (speciální PIN klávesnice, biometrická zařízení apod.). Na toto je v Cryptoki pamatováno a je vytvořen mechanismus, jakým je možné na zadání takového PINu počkat.

Práva a možnosti přístupu k objektům

Jak jsme se již zmínili, uživatele rozlišujeme na normálního a administrátora a objekty na objekty tokenu a objekty aktivního spojení. Další dělení objektů je na soukromé a veřejné. Dále je třeba uvést, že aktivní spojení rozlišujeme s právem pouze čtení (read/only, R/O) a s právem čtení i zápisu (read/write, R/W). Toto právo se ale ve skutečnosti týká jen objektů tokenu, k objektům aktivního spojení máme vždy i právo zápisu. Není možné, aby se administrátor přihlásil pouze s právem ke čtení. Všechna práva a možnosti přístupu k jednotlivým typům objektů shrnuje tab. 1.

Typ objektu	Typ aktivního spojení				
	R/O bez přihlášení	R/W bez přihlášení	R/O uživatel	R/W uživatel	R/W administrátor
veřejný session objekt	R/W	R/W	R/W	R/W	R/W
soukromý session objekt	x	x	R/W	R/W	x
veřejný token objekt	R/O	R/W	R/O	R/W	R/W
soukromý token objekt	x	x	R/O	R/W	x

Tab. 1. Pravomoci a typ přístupu k různým druhům objektů.

Základní práce s Cryptoki z pohledu aplikace

Každá aplikace, která chce pracovat s Cryptoki, musí všechna volání tohoto API uzavřít mezi volání `C_Initialize()` a `C_Finalize()`. Inicializaci stačí provést jednou a tato je platná pro všechna vlákna aplikace. Je-li ale počítáno s voláním Cryptoki API ve více vláknech jedné aplikace, je potřeba při inicializaci oznámit, jakým způsobem se má provádět synchronizace, tj. zda využít synchronizační primitiva daného operačního systému nebo zda použít specializovaná volání vlastní aplikace. Zde je zároveň doporučováno, aby si každé vlákno využívající Cryptoki otevřelo vlastní aktivní připojení. V případě typicky Unixového přístupu klonování

procesů pomocí příkazu `fork()` je třeba provést inicializaci zvlášť pro každý dětský proces.

Pro práci v rámci aktivního připojení (session) je klíčový pojem tzv. „session handle“, který dané připojení v rámci aplikace jednoznačně identifikuje. Pro identifikaci objektů a práci s objekty zavádíme jednoznačný identifikátor objektu, tzv. „object handle“.

Jednotlivé funkce Cryptoki

Funkce definované aplikačním rozhraním Cryptoki lze rozdělit do následujících skupin dle kategorií odpovídajícím účelu či oblasti, kam funkce patří. Výčet těchto funkcí dává zároveň stručný přehled toho, jaké služby Cryptoki poskytuje.

- *Obecné funkce.* Do této kategorie patří hlavně inicializační a finalizační funkce uvedené výše, dále pak funkce `C_GetInfo()`, která vrátí informace o aktuální instanci rozhraní Cryptoki.
- *Funkce pro správu slotů a tokenů.* Základní funkce z této kategorie `C_GetSlotInfo()` a `C_GetTokenInfo()` slouží k získání informací o slotu resp. tokenu. Je důležité zmínit funkce `C_InitToken()` sloužící k inicializaci tokenu, `C_InitPIN()` k inicializaci uživatelského PINu a `C_SetPIN()` k nastavení tohoto PINu. Dále do této kategorie patří funkce umožňující získání informací o mechanismech podporovaných určitým tokenem a informace o každém takovém mechanismu.
- *Funkce pro správu aktivních připojení.* Důležitými funkcemi této kategorie jsou funkce pro otevření a uzavření aktivního spojení, tj. funkce `C_OpenSession()` a `C_CloseSession()`. Uzavřít lze všechna otevřená spojení najednou: `C_CloseAllSessions()`. Funkcí `C_GetSessionInfo()` lze získat informace o spojení. Do této kategorie řadíme rovněž funkce sloužící k přihlášení/odhlášení se k/od tokenu, tj. funkce `C_Login()` a `C_Logout()`.
- *Funkce pro manipulaci s objekty.* Základní funkce pro manipulaci s objekty jsou funkce `C_CreateObject()`, `C_CopyObject()` a `C_DestroyObject()` sloužící k vytvoření, okopírování a zrušení objektu. Podporováno je vyhledávání objektů – `C_FindObjects()` a nastavování a získávání parametrů objektů pomocí dalších dvou funkcí: `C_SetAttributeValue()` a `C_GetAttributeValue()`.
- *Šifrovací funkce.* Funkce `C_Encrypt()` a `C_Decrypt()` slouží k zašifrování resp. dešifrování objektu. Obě tyto funkce mají ještě `Init`, `Update` a `Final` verze sloužící k šifrování po částech.
- *Funkce pro výpočet kontrolního řetězce zprávy.* Funkce `C_Digest()` slouží k výpočtu kontrolního řetězce předané řetězcové zprávy. Tato funkce má navíc také `Init`, `Update` a `Final` verze sloužící k počítání

po částech. Funkci `C_DigestKey()` použijeme v případě, že chceme spočítat kontrolní řetězec klíče.

- *Funkce pro elektronický podpis.* Základní funkcí této kategorie je funkce `C_Sign()`, opět se stejným způsobem podpory práce po částech. Navíc kategorie obsahuje funkci `C_SignRecover()`, která vytvoří takový podpis, že je možné jej použít k získání původních dat.
- *Funkce pro ověřování podpisů.* Stěžejní funkcí této kategorie je funkce `C_Verify()` sloužící k ověření elektronického podpisu. Verifikovat je možné standardním způsobem po částech. Stejně jako v předchozím případě existuje i verze `C_VerifyRecover()` podporující ověřování podpisu v případě, že něj lze odvodit vlastní data.
- *Funkce provádějící dvě kryptografické funkce najednou.* Představiteli této kategorie jsou 4 funkce: `C_DigestEncryptUpdate()`, `C_DecryptDigestUpdate()`, `C_SignEncryptUpdate()` a `C_DecryptVerifyUpdate()`. Tyto funkce kombinují šifrování a dešifrování s výpočtem kontrolního řetězce, případně podepsání se šifrování či dešifrování s ověřením podpisu. Všechny tyto funkce podporují práci po částech.
- *Funkce pro správu klíčů.* Klíče lze přímo v tokenu generovat. Tajný klíč vygenerujeme funkcí `C_GenerateKey()`. Pár klíčů veřejného a soukromého vygenerujeme pomocí `C_GenerateKeyPair()`. Klíč můžeme zašifrovat či dešifrovat funkcemi `C_WrapKey()` resp. `C_UnwrapKey()`. Od určitého základního klíče lze odvodit další klíč pomocí funkce `C_DeriveKey()`.
- *Funkce pro generování pseudonáhodných čísel.* Token může podporovat dvě klasické funkce pro generování pseudonáhodných čísel. Pomocí funkce `C_SeedRandom()` inicializujeme generátor. Ke generování vlastních pseudonáhodných čísel použijeme `C_GenerateRandom()`.
- *Funkce pro podporu paralelismu.* Do této kategorie bychom mohli započítat funkce pro zjišťování stavu ostatních funkcí případně přerušení jejich výpočtu, ale tyto již nejsou v nejnovější verzi PKCS #11 2.20 podporovány. Paralelismus si řeší každá funkce zvlášť. Dále do této kategorie můžeme zařadit tzv. „callback“ funkce sloužící ke zpětné notifikaci od Cryptoki.

Přehled Cryptoki funkcí nám dává základní představu o tom, co můžeme od zařízení podporujícího PKCS #11 očekávat. Detailnější popis jednotlivých funkčních volání jde již nad rámec tohoto článku. Každopádně je ale nutné upozornit, že nemůžeme stoprocentně spoléhat na to, že každé PKCS #11 zařízení podporuje všechny tyto funkce.

Možnosti využití Cryptoki pro účely projektu

Jak jsme již výše uvedli, pro účely projektu „Informační technologie pro rozvoj kontinuální sdílené péče o zdraví“ potřebujeme (mimo jiné) umět spolehlivě identifikovat a autentifikovat poskytovatele zdravotní péče, digitálně podepsat určitou část zdravotní dokumentace a zpětně bezpečně ověřit autora zdravotní dokumentace.

K těmto účelům můžeme vhodně využít funkčnost poskytovanou Cryptoki. To, že provádění kryptografických funkcí a přístup k soukromým objektům tokenu pomocí Cryptoki vyžaduje dodatečnou autorizaci pomocí PINu nebo jiné metody dodává další úroveň bezpečnosti zabraňující jednoduchému zneužití pouhým odcizením kryptografického zařízení.

Dalším prvkem bezpečnosti je možnost označení klíče jako „citlivého“ či „neexportovatelného“. Citlivý klíč není možné exportovat bez bezpečného zašifrování. Neexportovatelný klíč nelze vyjmout z tokenu žádným způsobem. Uvědomíme-li si, že klíče do tokenu standardně není možné vložit, nýbrž je přímo v tokenu generujeme, máme při splnění určitých podmínek jistotu, že daný klíč existuje pouze v konkrétním kryptografickém zařízení. Této vlastnosti můžeme vhodně využít právě k jednoznačné identifikaci autora zdravotní dokumentace.

Diskuse a závěr

Využití PKCS standardu pro komunikaci s kryptografickými zařízení pro účely projektu „Informační technologie pro rozvoj kontinuální sdílené péče o zdraví“ je možné, ale ne jediné řešení. Existují i jiné standardy jako například „Microsoft Crypto API“ (MS CAPI). Využití tohoto standardu má na rozdíl od PKCS jednu nevýhodu – je silně vázáno na platformu MS Windows. Na druhou stranu, prakticky všichni lékaři v České republice využívají PC s operačním systémem MS Windows. Standard MS CAPI je tomuto operačnímu systému bližší a umožňuje například nahradit klasický přihlašovací dialog Windows přihlášením pomocí kryptografického či biometrického zařízení.

Zvolení vhodného standardu není jedinou volbou, kterou je v rámci projektu nutné udělat. Je třeba též zvolit vhodné kryptografické zařízení. Jako vhodná se v první chvíli zdají zařízení dvou druhů: kryptografické procesorové čipové karty a kryptografické USB klíče. S některými těmito výrobky lze komunikovat jak pomocí PKCS tak MS CAPI rozhraní. Zařízení se liší cenou, mírou implementace standardu, velikostí paměti pro objekty, velikostí paměti pro uložení klíčů apod. K čipovým kartám je vždy třeba čtečka kdežto kryptografický USB klíč můžeme použít prakticky u každého počítače vybaveného USB rozhraním.

Poděkování

Práce je podporována grantem číslo 1ET200300413 Akademie věd České republiky.

Literatura

- [1] Zvárová Jana, Hanzlíček Petr, Špidlen Josef: Electronic Health Record in Cardiology: Pilot Application in the Czech Republic. In: Achieving a Knowledge-based Economy in Biomedicine through Informatics, Taiwan Association for Medical Informatics 2002, ss. 10–13.
- [2] Hanzlíček Petr, Špidlen Josef, Nagy Miroslav: Universal Electronic Health Record MUDR. In: Duplaga M., Zielinski K., Ingram D. (Eds.): Transformation of Healthcare with Information Technologies, IOS Press 2004, ss. 190-201.
- [3] Špidlen Josef, Říha Antonín: Elektronický zdravotní záznam a telemedicína. In: Hakl F. (Ed.): Doktorandský den 03, Praha, MATFYZPRESS 2003, ss. 133–143.
- [4] Špidlen Josef, Hanzlíček Petr, Říha Antonín, Zvárová Jana: Flexible Information Storage in MUDR² EHR. In: Zvárová J., Hanzlíček P., Peleška J., Přečková P., Svátek V., Valenta Z. (Eds.): International Joint Meeting EuroMISE 2004 Proceedings. Praha, EuroMISE 2004, s. 58.
- [5] RSA Laboratories, <http://www.rsasecurity.com/rsalabs/>
- [6] RSA Laboratories: PKCS #1: RSA Cryptography Standard, <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf>
- [7] RSA Laboratories: PKCS #3: Diffie-Hellman Key Agreement Standard, <ftp://ftp.rsasecurity.com/pub/pkcs/ps/pkcs-3.ps>
- [8] RSA Laboratories: PKCS #11: Cryptographic Token Interface Standard, <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v2-20/pkcs-11v2-20.pdf>
- [9] RSA Laboratories: PKCS #12: Personal Information Exchange Syntax Standard, <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-12/pkcs-12v1.pdf>
- [10] RSA Laboratories: PKCS #15: Cryptographic Token Information Format Standard, ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-15/pkcs-15v1_1.pdf

Adresa pro korespondenci

Josef Špidlen, EuroMISE centrum,
Ústav Informatiky AV ČR,
Pod Vodárenskou věží 2, 182 07 Praha 8, Česká republika
Email: spidlen@euomise.cz